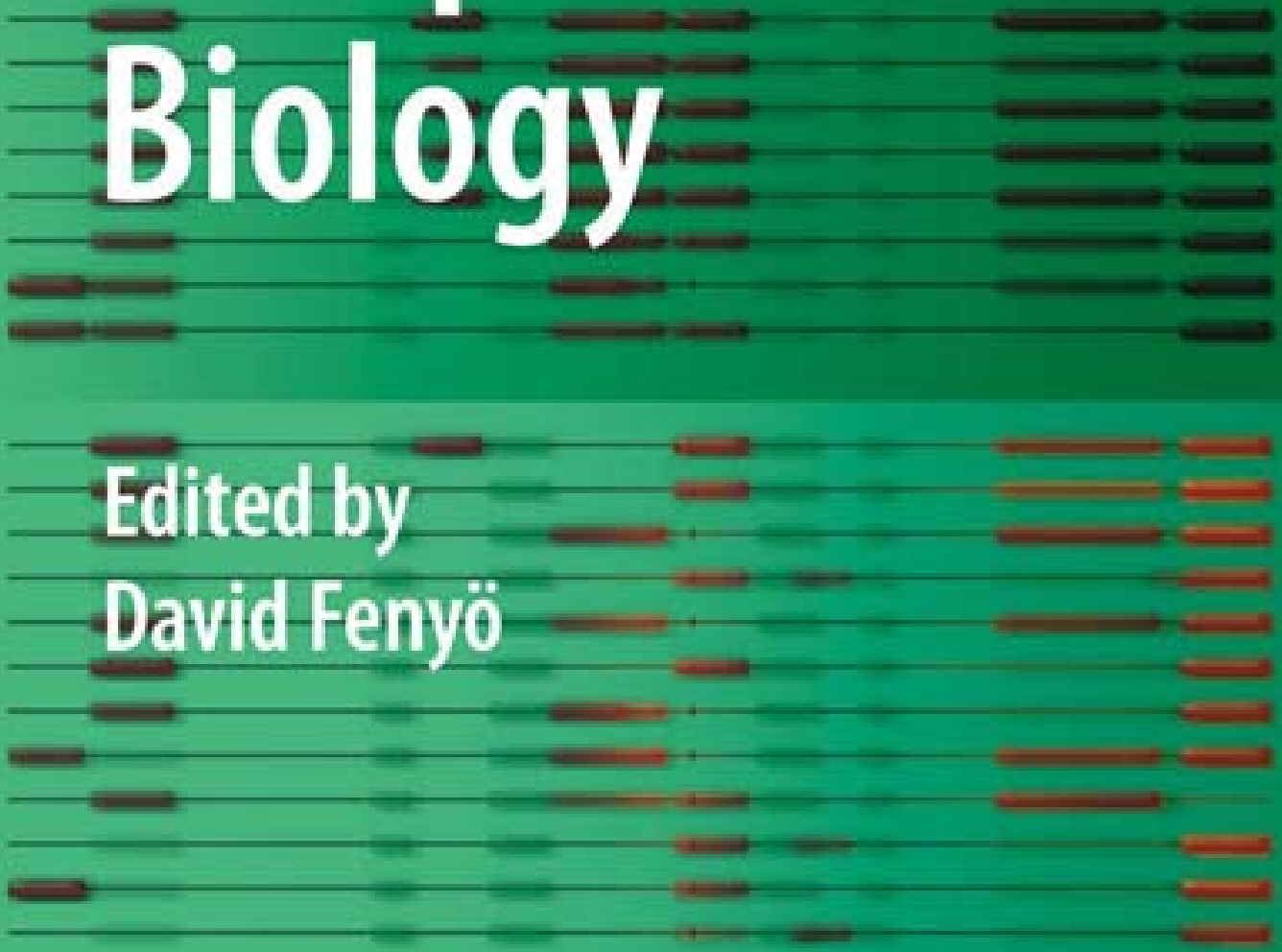
Springer Protocols

Methods in Molecular Biology 673

Computational Biology



Edited by
David Fenyö

Humana Press

METHODS IN MOLECULAR BIOLOGY™

Series Editor
John M. Walker
School of Life Sciences
University of Hertfordshire
Hatfield, Hertfordshire, AL10 9AB, UK

For other titles published in this series, go to
www.springer.com/series/7651

Computational Biology

Edited by

David Fenyö

The Rockefeller University, New York, NY, USA

Humana Press

Editor
David Fenyö
The Rockefeller University
1230 York Avenue
New York, NY 10065
USA
fenyo@rockefeller.edu

ISSN 1064-3745 e-ISSN 1940-6029
ISBN 978-1-60761-841-6 e-ISBN 978-1-60761-842-3
DOI 10.1007/978-1-60761-842-3
Springer New York Dordrecht Heidelberg London

Library of Congress Control Number: 2010934634

Cover Illustration Caption: Different experimental observation deposited in GPMDB of peptides from PSF2 a protein associated with the replication fork. Observed peptides are shown in red and regions of the protein that are difficult to observe in proteomics experiments are shown in green. In a majority of the 20 experiments shown, the same 5 proteotypic peptides are observed.

Printed on acid-free paper

Humana Press is part of Springer Science+Business Media (www.springer.com)

Preface

Computational biology is an interdisciplinary field that applies mathematical, statistical, and computer science methods to answer biological questions. The importance of computational biology has increased with the introduction of high-throughput techniques such as automatic DNA sequencing, comprehensive expression analysis with microarrays, and proteome analysis with modern mass spectrometry. Computational methods are not only critical for the effective analysis of the large amount of data from these experiments but also important to fully utilize this wealth of information to provide more realistic models of biological systems. The ultimate goal of modeling complex systems like an entire cell might be far in the future, but computational methods are essential in building the foundation that will allow this goal to be reached. The primary purpose of this book is to present a broad survey of computational biology methods by focusing on their applications, including primary sequence analysis, protein structure elucidation, transcriptomics and proteomics data analysis, and exploration of protein interaction networks.

New York, NY *David Fenyö*

Contents

Contributors

Réka Albert • *Department of Physics, Penn State University, University Park, PA, USA*

Ronald Beavis • *The Biomedical Research Centre, University of British Columbia, Vancouver, BC, Canada*

Paul R. J. Birch • *Scottish Crop Research Institute, Dundee, UK*

Stefano Calza • *Department of Medical Epidemiology and Biostatistics, Karolinska Institute, Stockholm, Sweden*

Brian T. Chait • *The Rockefeller University, New York, NY, USA*

Bhaskar DasGupta • *Department of Computer Science, University of Illinois at Chicago, Chicago, IL, USA*

Jeffrey A. DeGrasse • *U.S. Food and Drug Administration, College Park, MD, USA*

Damien Devos • *Structural Bioinformatics, European Molecular Biology Laboratory, Heidelberg, Germany*

Frank Dondelinger • *Biomathematics & Statistics Scotland and School of Informatics, University of Edinburgh, Edinburgh, UK*

Jan Eriksson • *Department of Chemistry, Swedish University of Agricultural Sciences, Uppsala, Sweden*

Maria Fälth Savitski • *Unit Cancer Genome Research, Division of Molecular Genetics, Heidelberg, Germany*

David Fenyö • *The Rockefeller University, 1230 York Avenue, New York, NY 10065, USA*

Andras Fiser • *Department of Systems and Computational Biology & Department for Biochemistry, Albert Einstein College of Medicine, Bronx, NY, USA*

David R. Goodlett • *Department of Medicinal Chemistry, University of Washington, Seattle, Washington, USA*

Wolfram Gronwald • *Institute for Biophysics and Physical Biochemistry, University of Regensburg, Regensburg, Germany*

Dirk Husmeier • *Biomathematics and Statistics Scotland, Edinburgh & Aberdeen, UK*

Deepti Jain • *National Centre for Biological Sciences, Bangalore, India*

Hans Robert Kalbitzer • *Institute for Biophysics and Physical Biochemistry, University of Regensburg, Regensburg, Germany*

Lukas Käll • *Department of Biochemistry and Biophysics, Center for Biomembrane Research and Stockholm Bioinformatics Center, Stockholm University, Stockholm, Sweden*

Johannes F. Knabe • *Biological and Neural Computation Laboratory and Adaptive Systems Research Group, STRI, University of Hertfordshire, Hatfield, Hertfordshire, UK*

Valerie Lamour • *Institute of Genetics and Molecular and Cellular Biology, Illkirch, France*

Kuang Lin • *Biomathematics and Statistics Scotland, Edinburgh & Aberdeen, UK*

Hui Liu • *Scottish Crop Research Institute, Dundee, UK*

Steven J. Ludtke • *Department of Biochemistry and Molecular Biology, Baylor College of Medicine, Houston, TX, USA*
Lars Malmström • *Institute for Molecular Systems Biology, ETH Zurich, Zurich, Switzerland*
Claus D. Mayer • *Biomathematics and Statistics Scotland, Edinburgh & Aberdeen, UK*
István Miklós • *Rényi Institute, Hungarian Academy of Sciences, Budapest, Hungary*
Kelly Molloy • *The Rockefeller University, New York, NY, USA*
Sunnie Myung • *The Rockefeller University, New York, NY, USA*
Niranjan Nagarajan • *Center for Bioinformatics and Computational Biology, Institute for Advanced Computer Studies and Department of Computer Science, University of Maryland, College Park, MD, USA*
Chrystopher L. Nehaniv • *Biological and Neural Computation Laboratory and Adaptive Systems Research Group, STRI, University of Hertfordshire, Hatfield, Hertfordshire, UK*
Thomas A. Neubert • *Kimmel Center for Biology and Medicine at the Skirball Institute and Department of Pharmacology, New York University School of Medicine, New York, NY, USA*
Peter Q. Nguyen • *Department of Biochemistry and Cell Biology, Rice University, Houston, TX, USA*
Yudi Pawitan • *Department of Medical Epidemiology and Biostatistics, Karolinska Institute, Stockholm, Sweden*
Mihai Pop • *Center for Bioinformatics and Computational Biology, Institute for Advanced Computer Studies and Department of Computer Science, University of Maryland, College Park, MD, USA*
Leighton Pritchard • *Scottish Crop Research Institute, Dundee, UK*
George P. C. Salmond • *Department of Biochemistry, University of Cambridge, Cambridge, UK*
Mikhail M. Savitski • *Cellzome AG, Heidelberg, Germany*
Maria J. Schilstra • *Biological and Neural Computation Laboratory, University of Hertfordshire, Hatfield, Hertfordshire, UK*
Jonathan J. Silberg • *Cellzome AG, Heidelberg, Germany*
Eduardo Sontag • *Department of Mathematics, Rutgers University, New Brunswick, NJ, USA*
Taylor Stevenson • *Department of Biochemistry and Cell Biology, Rice University, Houston, TX, USA*
Michael P. H. Stumpf • *Division of Molecular Biosciences, Centre for Bioinformatics, Imperial College London, London, UK*
Tina Toni • *Division of Molecular Biosciences, Centre for Bioinformatics, Imperial College London, London, UK*
Ian K. Toth • *Scottish Crop Research Institute, Dundee, UK*
Beatrix M. Ueberheide • *The Rockefeller University, New York, NY, USA*
Sofia Waldemarson • *Kimmel Center for Biology and Medicine at the Skirball Institute and Department of Pharmacology, New York University School of Medicine, New York, NY, USA*

Katja Wegner • *Albstadt-Sigmaringen University, Sigmaringen, Germany*
Guoan Zhang • *Kimmel Center for Biology and Medicine at the Skirball Institute and Department of Pharmacology, New York University School of Medicine, New York, NY, USA*

Chapter 1

Sequencing and Genome Assembly Using Next-Generation Technologies

Niranjan Nagarajan and Mihai Pop

Abstract

Several sequencing technologies have been introduced in recent years that dramatically outperform the traditional Sanger technology in terms of throughput and cost. The data generated by these technologies are characterized by generally shorter read lengths (as low as 35 bp) and different error characteristics than Sanger data. Existing software tools for assembly and analysis of sequencing data are, therefore, ill-suited to handle the new types of data generated. This paper surveys the recent software packages aimed specifically at analyzing new generation sequencing data.

Key words: Next-generation sequencing, Genome assembly, Sequence analysis

1. Introduction

Recent advances in sequencing technologies have resulted in a dramatic reduction of sequencing costs and a corresponding increase in throughput. As data produced by these technologies is rapidly becoming available, it is increasingly clear that software tools developed for the assembly and analysis of Sanger data are ill-suited to handle the specific characteristics of new generation sequencing data. In particular, these technologies generate much shorter read lengths (as low as 35 bp), complicating repeat resolution during both de novo assembly and while mapping the reads to a reference genome. Furthermore, the sheer size of the data produced by the new sequencing machines poses performance problems not previously encountered in Sanger data. This is further exacerbated by the fact that the new technologies make it possible for individual labs (rather than large sequencing centers) to perform high-throughput sequencing experiments, and these labs do not have the computational infrastructure commonly

David Fenyö (ed.), *Computational Biology*, Methods in Molecular Biology, vol. 673,
DOI 10.1007/978-1-60761-842-3_1,

available at large sequencing facilities. In this paper we survey software packages recently developed to specifically handle new generation sequencing data. We briefly overview the main characteristics of the new sequencing technologies and the computational challenges encountered in the assembly of such data; however, a full survey of these topics is beyond the scope of our paper. For more information, we refer the reader to other surveys on sequencing and assembly (1–3).

We hope the information provided here will provide a starting point for any researcher interested in applying the new technologies to either de novo sequencing applications or to resequencing projects. Due to the rapid pace of technological and software developments in this field we try to focus on more general concepts and urge the reader to follow the links provided in order to obtain up-to-date information about the software packages described.

2. Sequencing Technologies

Before discussing the software tools available for analyzing the new generation sequencing data we briefly summarize the specific characteristics of these technologies. For a more in-depth summary, the reader is referred to a recent review by Mardis (1).

2.1. Roche/454 Pyrosequencing

The first, and arguably most mature, of the new generation sequencing technologies is the pyrosequencing approach from Roche/454 Life Sciences. Current sequencing instruments (GS FLX Titanium) can generate in a single run ~500 Mbp of DNA in sequencing reads that are ~400 bp in length (approximately 1.2 million reads per run), while the previous generation instruments (GS FLX) generate ~100 Mbp of DNA in reads that are ~250 bp in length (approximately 400,000 reads per run). Initial versions of mate-pair protocols are also available that generate paired reads spaced by approximately 3 kbp.

The main challenge in analyzing 454 data is the high error-rate in homopolymer regions – sections of DNA comprised of a single repeated base. The 454 sequencing approach is based on a technique called pyrosequencing (4) wherein double-stranded DNA is synthesized from single-strand templates (DNA fragments being sequenced) through the iterative addition of individual nucleotides, and the incorporation of a nucleotide is detected by the emission of light. When encountering a run of multiple identical nucleotides in the template DNA, the amount of light emitted should be proportional to the length of this homopolymer run. This correspondence, however, is nonlinear due to limitations of the optical device used to detect the signal. As a result, the length

of homopolymer runs is frequently misestimated by the 454 instrument, in particular for long homopolymer runs.

A 454 sequencing instrument can output copious information, including raw images obtained during the sequencing process. For most purposes, however, it is sufficient to retain the 454 equivalent of sequence traces, information stored in .SFF files. These files contain information about the sequence of nucleotide additions during the sequencing experiment, the corresponding intensities (normalized) for every sequence produced by the instrument and the results of the base-calling algorithm for these sequences. Each called base is also associated with a phred-style quality value (log-probability of error at that base), providing the same information as available from the traditional Sanger sequencing instruments. Note, however, that homopolymer artifacts also affect the accuracy of the quality values – Huse et al. (5) show that the quality values decrease within a homopolymer run irrespective of the actual confidence in the base-calls.

Due to the long reads and availability of mate-pair protocols, the 454 technology can be viewed as a direct competitor to traditional Sanger sequencing and has been successfully applied in similar contexts such as de novo bacterial and eukaryotic sequencing (6, 7) and transcriptome sequencing (8).

2.2. Solexa/Illumina Sequencing

The Solexa/Illumina sequencing technology achieves much higher throughput than 454 sequencing (~1.5 Gbp/run) at the cost, however, of significantly smaller read lengths (currently ~35 bp). Initial mate-pair protocols are available for this technology that generate paired reads separated by ~200 bp and approaches to generate longer libraries are currently being introduced. While the reads are relatively short, the quality of the sequence generated is quite high, with error rates of less than 1%. The sequencing approach used by Solexa relies on reversible terminator chemistry and is, therefore, not affected by homopolymer runs to the same extent as the 454 technology. Note that homopolymers, especially long ones, cause problems in all sequencing technologies, including Sanger sequencing.

The analysis of Solexa/Illumina data poses several challenges. First of all, a single run of the machine produces hundreds of gigabytes of image files that must be transferred to a separate computer for processing. In addition to the sheer size of the data generated, a single Solexa run results in ~50 million reads leading to difficulties in analyzing the data, even after the images have been processed. Finally, the short length of the reads generated complicates de novo assembly of the data due to the inability to span repeats. The short reads also complicate alignment to a reference genome in resequencing applications, both in terms of efficiency and due to the increased number of spurious matches caused by short repeats.

Analogous to 454 sequencing, the output from an Illumina sequencing instrument contains a wealth of information, including raw image data that could be reprocessed to take advantage of new base-calling algorithms. In practice, however, these data are rarely retained due to the large memory requirements. For most applications it is sufficient to use the sequence trace information encoded in an SRF file – a newly developed format for encoding new generation sequencing data. When just the sequence and quality information are needed, these data are usually stored in a FASTQ file (an extension of the FASTA format that combines sequence and quality data) and represents quality values in a compressed (one character per base) format.

2.3. ABI/SOLiD Sequencing

The ABI/SOLiD technology generates data with characteristics similar to that generated by Solexa/Illumina instruments, albeit at higher throughput (~3 Gbp/run). Challenges in image storage and processing that are present with Solexa technology are therefore also there for the ABI/SOLiD instrument. The latter, however, integrates computer hardware with the sequencing machine, eliminating the need to transfer large image files for analysis purposes.

A major challenge in analyzing SOLiD data stems from the sequencing-by-ligation approach used in this technology. Specifically, the sequencing of a DNA template is performed by iteratively interrogating pairs of positions in the template with semi-degenerate oligomers of the form NNNACNNN, where N indicates a degenerate base. Each oligomer is tagged with one of four colors, allowing the instrument to "read" the sequence of the template. Note, however, that each color is associated with four distinct pairs of nucleotides, complicating the determination of the actual DNA sequence. In fact, the sequence of colors observed by the instrument during the sequencing process is not sufficient to decode the DNA sequence – rather it is necessary to also know the first base in the sequence (the last base within the sequencing adapter). The lack of a direct correspondence between the sequencing signal and the DNA sequence complicates the analysis of SOLiD data in the presence of errors. A single sequencing error (missing or incorrect color) can result in a "frame-shift" that affects the remainder of the DNA sequence decoded by the instrument. Note that this phenomenon is similar to that encountered during gene translation from three-letter codons. Due to this property of SOLiD data, most software tools attempt to operate in "color space" in order to avoid having to consider all possible frame-shift events during data analysis. This also makes it difficult to apply SOLiD data in de novo assembly applications.

File formats for representing SOLiD data are still being developed and a SOLiD-specific extension to the SRF format is expected in the near future.

2.4. Others

We presented in more detail the three technologies outlined above because they are the only technologies currently deployed on a large scale within the community. It is important to note, however, that new sequencing technologies are being actively developed and several will become available in the near future. For example, Helicos Biosciences have recently reported the sale of the first instruments of a high-throughput, single-molecule (requiring no amplification) sequencing technology (9). Also, recently, Pacific Biosciences have described a new technology characterized by substantially longer read lengths and higher throughputs than the technologies currently available (10). These advances underscore the dynamic nature of research on DNA sequencing technologies, and highlight the fact that the information we provide in this article is necessarily limited to the present and might become partly obsolete in the near future.

2.5. NCBI Short Read Archive

The large volumes of data generated by the new technologies as well as the rapidly evolving technological landscape are posing significant challenges to disseminating and storing this data. To address these challenges and provide a central repository for new generation data, the NCBI has established the Short Read Archive, an effort paralleling the successful Trace Archive – a repository of raw Sanger sequence information. The Short Read Archive (http://ncbi.nlm.nih.gov/Traces/sra) already contains a wealth of data generated through the 454 and Illumina technologies, including data from the 1,000 Genomes project – an effort to sequence the genomes of 1,000 human individuals. In addition to being a data repository, the Short Read Archive is actively involved in efforts to standardize data formats used to represent new generation data, efforts that resulted in the creation of the .SFF format (454) and the .SRF format meant to become a universal format for representing sequence information.

3. Assembly Programs

The assembly of sequences from a shotgun-sequencing project is typically a challenging computational task akin to solving a very large one-dimensional puzzle. Several assembly programs have been described in the literature (such as **Celera Assembler** (11), **ARACHNE** (12, 13), and **PHRAP** (14)) and have been successfully used to assemble the genomes of a variety of organisms – from viruses to humans. These programs were designed when Sanger sequencing was the only technology available and were therefore tailored to the characteristics of the data. With the advent of new technologies there has been a flurry of efforts to

cope with the characteristics of the new datasets. An important consideration is the reduced read length and the limited form of mated read libraries. These make the assembly problem even more difficult as we discuss in Subheading 3.2. What the new technologies do offer is the ability to sequence genomes to high redundancy (every base in the genome is represented in many reads) and in a relatively unbiased manner. Managing the corresponding flood of information effectively is an important challenge facing new computational tools.

3.1. Mapping and Comparative Assembly

In many sequencing projects, an assembled genome of a related organism is available and this can dramatically simplify the assembly task. The task of assembly is then often translated to one of matching sequences to the reference genome and de novo assembly of just the polymorphic regions from the unmatched reads. This strategy has been widely used for resequencing projects (15). It has also been used to assemble closely related bacterial strains (16). The strategy of sequencing and mapping to a reference genome has also been used in a variety of other applications – from discovering novel noncoding RNA elements (17) to profiling methylation patterns (18) (see Subheading 4 for more examples). The general pipeline for these applications is outlined in Fig. 1 with mapping of reads to a reference being an important common component.

In recent years, several programs have been developed to handle the challenge of mapping a large collection of reads onto a reference genome while accounting for sequencing errors and polymorphisms. These programs often trade-off flexibility in matching policy – how many mismatches and indels they can handle – in

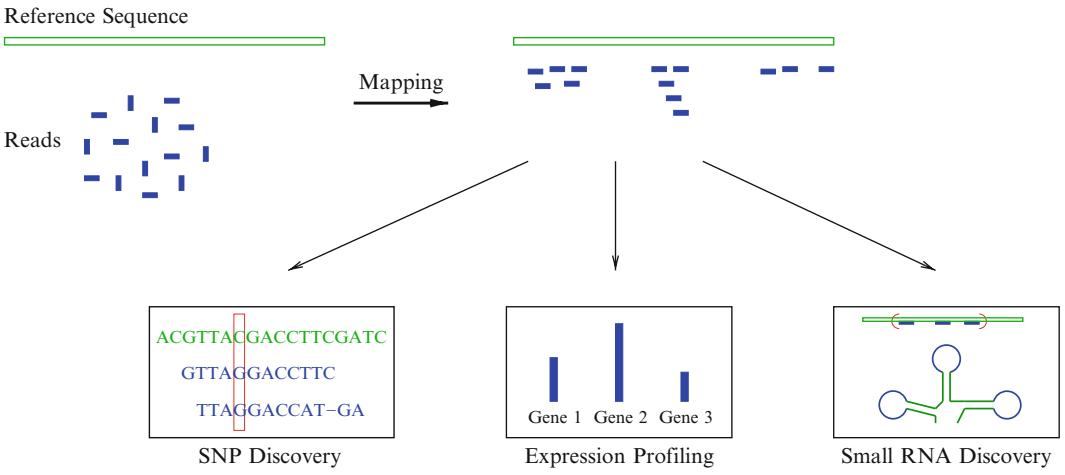


Fig. 1. Read mapping and its applications. Mapping programs are widely used to align reads to a reference while allowing some flexibility in terms of mismatches and indels and a policy for handling ambiguous matches. The matches are then processed in different ways depending on the application of interest.

order to improve computational efficiency and the size of their memory footprints. For the longer reads from Sanger and 454 sequencing, programs such as **MUMmer** (19) and **BLAT** (20) provide the right trade-off between efficiency and flexibility in matching policy; they allow many mismatches and indels and are correspondingly slower.

The large volume of reads from Illumina and SOLiD sequencing has spurred the development of a new set of tools. In order to efficiently handle large amounts of short-read data, these programs attempt to find the right balance between alignment sensitivity and performance. Performance is generally achieved by constructing efficient indexes of either the reference genome or the set of reads, allowing the rapid identification of putative matches which are then refined through more time-intensive algorithms. Further improvements in performance arise from the handling of reads that map within repeat regions – most programs only report a few (or even just one) of the possible mappings. Finally, these programs allow only a few differences between a read and the reference genome and frequently do not allow indels. The choice of alignment program and corresponding parameters ultimately depends on the specific application: for example, in SNP discovery it is important to allow for differences between the reads and the reference beyond those expected due to sequencing errors, while in CHIP-seq experiments (21), exact or almost-exact alignments are probably sufficient. We review some of the popular mapping programs below.

MAQ (22) (it stands for *Mapping and Assembly with Quality*) is designed to map millions of very short reads accurately to a reference genome by taking into account the quality values associated with bases. In addition, MAQ also assigns to every mapped read, an assessment of the quality of the mapping itself. This information allows MAQ to perform well in SNP-calling applications. MAQ constructs an index of the reads, therefore its memory footprint is proportional to the size of the input and the authors recommend performing the alignment in chunks of two million sequences. MAQ only allows for mismatches in the alignment (no indels) and randomly assigns a read to one of several equally good locations when multiple alignments are possible (though this behavior can be modified through command-line parameters). Furthermore, MAQ can utilize mate-pair information in order to disambiguate repetitive matches. MAQ was originally developed for Illumina data, though it can also handle SOLiD sequencing using a transformation of the reference sequence into color space.

The inputs to MAQ are provided in FASTA (reference) and FASTQ (reads) formats and the output consists of a list of matches with associated qualities. MAQ also includes modules for SNP

calling, as well as a viewer **Maqview** that provides a graphical representation of the alignments.

The source code is available for download at http://maq.sourceforge.net under the GNU Public License.

SOAP (23) which stands for *Short Oligonucleotide Alignment Program* indexes the reference instead of the reads and therefore its memory footprint should be constant irrespective of the number of reads processed. Its alignment strategy allows alignments with one short indel (1–3 bp) in addition to mismatches between the read and the reference. Its treatment of reads with multiple alignments can be tuned through command-line parameters. Like MAQ, SOAP also provides support for mate-pairs, and includes a module for SNP calling. In addition, SOAP provides an iterative trimming procedure aimed at removing low quality regions at the ends of reads, as well as specialized modules for small RNA discovery and for profiling of mRNA tags.

SOAP is available for download at http://soap.genomics.org.cn as a Linux executable.

SHRiMP (unpubl.) is one of the first alignment programs specifically targeted at SOLiD data, though Illumina data can also be processed. This program uses a spaced-seed index followed by Smith–Waterman alignment to provide full alignment accuracy and flexibility. Since SHRiMP uses a full dynamic programming approach for alignment instead of heuristics, it is considerably slower than MAQ or SOAP, even though the implementation of the Smith–Waterman algorithm is parallelized through vectored operations supported by Intel and AMD processors. In addition to SOLiD data, SHRiMP now also supports data generated by the Helicos technology.

SHRiMP is available from http://compbio.cs.toronto.edu/shrimp as both source code and precompiled binaries.

Bowtie (24) is the first of a new-breed of fast and memory-efficient short-read aligners based on the compact Burrows–Wheeler index (both MAQ and SOAP now offer BWT-based indices), used to index the reference sequence. While following the same alignment policies as MAQ and SOAP, Bowtie is typically more than an order of magnitude faster, aligning more than 20 million reads per hour to the human genome on a typical workstation. Unlike other aligners, Bowtie allows the index for a genome to be precomputed, reducing the overall alignment time and making it easier to parallelize the alignment process. Furthermore, the indexing structure used is space-efficient, requiring just over 1 GB for the entire human genome.

Bowtie is available at http://bowtie-bio.sourceforge.net as an open-source package together with an associated program called **TopHat** to map splice junctions from RNA-seq experiments.

Other programs. Several other programs are available for the alignment of short reads and more will likely become available in

the near future. Among the most widely used is **Eland**, the aligner provided by Illumina with their sequencing instruments. This program is proprietary and unpublished and we cannot provide any additional information on its performance. Another commercial offering is SX-OligoSearch from Synamatix, a program that is provided together with the specialized hardware necessary to run it. Finally, **SeqMap** (25), **RMAP** (http://rulai.cshl.edu/rmap), and **ZOOM** (26) are other aligners that have been recently reported in the literature. The latter is based on a spaced-seed index and appears to be very efficient; however, the code can currently only be obtained by direct request from the authors.

Postalignment analysis. Several of the alignment programs described above provide additional modules for postprocessing the set of alignments in order to identify SNPs, discover small RNAs or analyze transcriptome profiling data or splicing patterns. The resulting alignments can also be provided as input to a comparative assembler such as **AMOScmp** (27) to construct local assemblies of the set of reads in a "template-guided" fashion. In a recent work, Salzberg et al. (16) demonstrated the use of this tool with Solexa data in bacterial sequencing, and have also proposed an approach to leverage similarity at the amino acid level to construct gene-centric assemblies of the data.

3.2. De Novo Assembly

In the absence of a reference genome, researchers typically rely on de novo assembly programs to reconstruct the sequences represented in the shotgun sequencing reads. An overview of the assembly process is presented in Fig. 2. The de novo assembly of a genome relies on the assumption that two reads that overlap significantly in their sequence are likely to represent neighboring segments of a genome. This assumption is, however, violated when the overlapping sequence is part of a repetitive region in the genome and recognizing such regions is an important part of

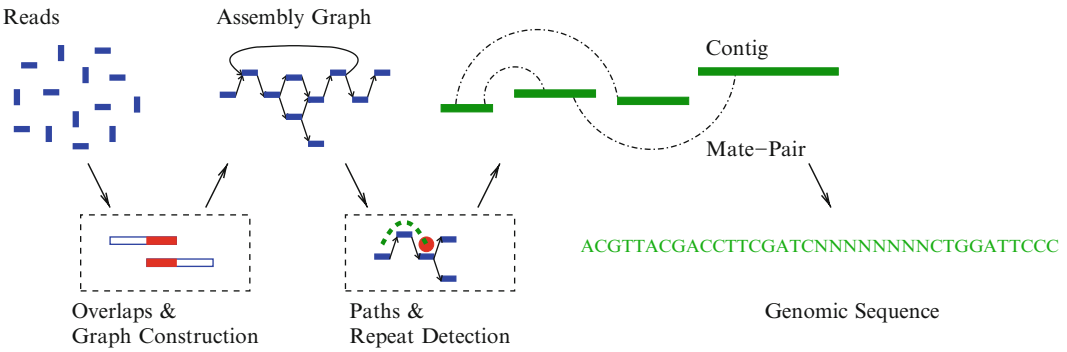


Fig. 2. Overview of de novo assembly. De novo assembly programs typically use the overlap between reads to construct a graph structure. After some simplifications of the graph, unambiguous paths in the graph are used to reconstruct contiguous sequences (contigs). Information such as the presence of mate-pair links between contigs may also allow the construction of gapped sequences (scaffolds).

genome assembly. The short read lengths of the new sequencing technologies entail that even short genomic repeats (that tend to be more frequent as well) can introduce ambiguities into the assembly process. As a result, the output from the assemblers is often a highly fragmented picture of the genome. Despite these limitations, several sequencing projects have successfully used short-read technologies.

Due to its longer read lengths 454 sequencing is a popular approach for de novo sequencing of bacterial genomes and increasingly for larger eukaryotic genomes as well. The **Newbler** assembler (http://www.roche-applied-science.com) that is distributed with 454 instruments has been used to assemble 454 data in several sequencing projects. The Newbler assembler supports mate-pairs and can do comparative as well as hybrid assembly (see Subheading 3.2.2). With sufficient read coverage (>20×) it generally produces accurate and conservative assemblies containing few misassemblies due to repeats. The consensus sequence of the resulting contigs is of high quality despite the relatively common sequencing errors in homopolymer regions within 454 sequence data.

The Celera Assembler (http://wgs-assembler.sourceforge.net) (11), originally developed for the assembly of large mammalian genomes from Sanger data, has recently been extended to allow assembly of 454 data as well as of mixtures of 454 and Sanger data. Both Celera Assembler and Newbler directly accept 454 data as input in .SFF format and produce outputs in both FASTA format and in several more detailed assembly formats, including the popular ACE format used by the phred-phrap-consed suite of programs.

For assembling the even shorter reads from Illumina and SOLiD, several assembly programs have recently been developed. In order to deal with the large volume of reads, early programs such as SSAKE (28), VCAKE (29), and SHARCGS (30) relied on a simple greedy approach to assembly. However, two new programs (Edena and Velvet) that are based on a graph-theoretic approach to assembly were shown to produce more accurate and larger assemblies and we describe them in more detail here. Note that even the best assemblers generate highly fragmented assemblies from short-read data (~35 bp), leading to contigs in the range of just a few to tens of thousands of basepairs instead of hundreds to millions of bases common in 454 and Sanger assemblies. These programs are, thus, better suited for the assembly of targeted regions, such as individual genes, or data generated in CHIP-seq experiments.

Edena (31), which stands for Exact DE Novo Assembler, was designed for assembling Illumina sequences based on a classic overlap-layout-consensus assembly framework. To avoid spurious

overlaps, Edena restricts itself to exact matches and this also allows it to compute overlaps efficiently. In addition, Edena incorporates some heuristic approaches to simplify the overlap graph and only linear sections of the graph are assembled into sequences. In addition to this conservative approach, Edena also allows for a non-strict mode which can create longer sequences but with an increased chance of incorrect assembly. Experiments with ~35 bp Illumina reads for a few bacterial genomes have shown that Edena can very accurately assemble them into sequences that are on average a few kilobases long. These assemblies were performed on a desktop computer with 4 GB of memory and in less than 20 min. The Edena program is available for download at http://www.genomic.ch/edena as a linux executable (an experimental windows executable is also available). The program takes a FASTA or FASTQ file of reads as input and produces a FASTA file of assembled sequences as output. It also allows the user flexibility in choosing an overlap size, trimming of reads and filtering of short assemblies. The current implementation of Edena does not handle mated reads.

Velvet (32) is an open-source program that uses a de Bruijn graph-based approach to assembly (33). Correspondingly, while the graph construction step is simplified, the program relies on several error-correction heuristics to improve the structure of the graph. The program also has a module to use mated reads to disambiguate some repeat structures and join contigs. Using simulated mated reads, this approach was shown to produce much longer contigs in prokaryotic genomes. Velvet is available for download at http://www.ebi.ac.uk/~zerbino/velvet and has been tested on Linux, MacOS, and Windows systems with Cygwin. It accepts reads in FASTA as well as FASTQ format and its output is a set of assembled sequences in a FASTA file as well as an AMOS compatible assembly file. Velvet also allows the user to choose the overlap size and can filter sequences that have a low read coverage.

ABySS (34) is a new parallelized sequence assembly program based on the de Bruijn graph approach that can efficiently do de novo assembly of relatively large datasets (billions of reads). It also allows for the use of paired-end information to produce longer contigs. ABySS can take in reads in FASTA format and produce contigs in FASTA format and is available as an open-source package at http://www.bcgsc.ca/platform/bioinfo/software/abyss.

Other software. The **Minimus** assembler (35) which is part of the AMOS package of open-source assembly tools (http://amos.sourceforge.net), like Edena, is based on an overlap-layout-consensus framework for assembly. Due to its modular structure, Minimus can easily be adapted for various sequencing technologies and a

version for Illumina sequences (http://amos.sourceforge.net/docs/pipeline/minimus.html) is also available. The **ALLPATHS** program (36) is a new short-read assembler, based on the Eulerian assembly strategy, that aims to explicitly present assembly ambiguity to the user in the form of a graph. The authors plan to release a production version of the program soon.

3.2.1. Scaffolding

While programs such as Edena and Minimus do not directly handle information about mated reads, a scaffolding program such as **Bambus** (37) can use this information to stitch together contigs into larger sections of the genome (aka scaffolds). Bambus is available at http://amos.sourceforge.net/docs/bambus. Note that Newbler, Celera Assembler, Velvet, and ALLPATHS can use mate-pair information directly to guide the assembly process and generate larger contigs and scaffolds (where some of the intervening regions can be ambiguous).

Another promising new approach to scaffold short-read sequences is based on Optical Mapping technology (38) (http://www.opgen.com). Optical Maps are a form of restriction maps where genomic DNA is fragmented using a restriction enzyme and the fragment sizes are measured. In optical mapping, both fragment sizes and the order in which they occur within the genome can be determined. This genome-wide map provides an ideal reference to determine the order of the sequences assembled from a shotgun sequencing project. For a typical prokarytoic sequencing project more than 90% of the genome can be scaffolded using these maps into a single genome-wide scaffold (39). The open-source **SOMA** package is specifically designed to map short-read assemblies onto one or more optical maps and scaffold them, and is available for download and as a webservice at http://www.cbcb.umd.edu/soma.

3.2.2. Hybrid Assembly

As discussed in Subheading 2 the various sequencing technologies have different advantages and disadvantages, some of which are complementary. Correspondingly there is an interest in constructing hybrid assemblies that, for example, can combine mated reads from one technology with high coverage reads from another. In recent work, Goldberg et al. (40) showed that high-quality assemblies of microbial genomes can be obtained in a cost-effective manner using a Sanger-454 hybrid approach. In order to assemble the data, they relied on an ad-hoc approach where sequences assembled from 454 reads using Newbler were shredded in-silico before assembly with other Sanger reads using the Celera Assembler. Recently, more carefully tuned versions of the Celera Assembler (41) and Newbler have been released that can perform true Sanger-454 hybrid assemblies. Assemblers that are fine-tuned to incorporate various other mixtures of sequence data are still an active area of research.

4. Applications

The dramatic reduction in the cost of sequencing using next-generation technologies has led to widespread adoption of sequencing as a routine research technique. On the one hand, the traditional use of sequencing, i.e., to reconstruct the genomes of a range of model organisms and pathogenic microbes has received a boost. Researchers are now looking to sequence several individuals and strains of the same species to understand within species variation. While in some cases these related genomes can be assembled based on the reference, in others, de novo assembly programs are required. The other popular use for sequencing has been as a substitute for common array based techniques for studying mRNA expression, transcription-factor-binding sites and methylation patterns, among others. These applications rely on read mapping programs followed by application-specific analysis as shown in Fig. 1. Here we highlight a few of the diverse collection of problems that are being impacted by the availability of new sequencing platforms and the computational tools to analyze the data.

4.1. Variant Discovery

High-throughput sequencing has enabled researchers to study the extent of variability in our genomes both in terms of single base mutations as well as larger structural changes that are much more common than we once believed. The current approach for these studies is to map reads to a reference genome to detect changes from the reference and is aided by the array of mapping programs available as detailed in Subheading 3.1. In addition to general-mapping programs such as MAQ that are well-suited for SNP calling and have a built-in procedure to do so, there are other programs that are specifically designed for SNP calling. The **PyroBayes** program is one such tool that was developed to specifically take into account the characteristics of 454 reads, given a set of read mappings (available at http://bioinformatics.bc.edu/marthlab/PyroBayes). The **ssahaSNP** program is another (http://www.sanger.ac.uk/Software/analysis/ssahaSNP), that performs both the mapping and SNP calling for Illumina sequences and also includes a module for indel discovery. Tools to detect larger structural variations based on mated reads are an active area of current research (42).

4.2. Metagenomics

Metagenomics studies where a collection of organisms are sequenced together are, in principle, the prime application for new sequencing technologies that enable cheap and relatively bias-free sequencing. The crucial impediment however is the ability to assemble and annotate the short reads from these technologies.

In recent years, several programs have been designed for classification and gene-finding in 454 reads. Programs such as **MEGAN** (43) and **CARMA** (44), in particular, have had some success using translated BLAST searches to classify and annotate 454 reads. Ideally, annotation and gene-finding of metagenomic sequences would be preceded or done in tandem with assembly of the short reads. Assembly algorithms tuned for metagenomics datasets, especially those based on short reads are, however, still being developed (45), and there is much work to be done in this direction. As read lengths for Illumina and SOLiD sequencing increase, metagenomics studies are likely to more widely use these technologies in the future.

4.3. Small RNA Discovery

Sequencing in combination with computational filters provides an ideal approach to discover various noncoding small RNAs whose regulatory importance is increasingly being apparent. The dramatic decrease in the cost of sequencing has enabled researchers to detect even rarely transcribed elements and fortunately the length of these elements is small enough for them to be profiled with short reads. Analyzing reads from such a project involves mapping them to a reference genome and using annotations on the genome and RNA structure prediction programs to filter out uninteresting loci. The analysis pipeline typically needs to be tailored to the sequencing platform used and the kinds of small RNA that the researchers are interested in. The **miRDeep** package (17), for example, was specifically designed to analyze sequences for microRNAs and more such packages are likely to be made available in the near future. In another recent work, Moxon et al. (46) describe a set of webservices to analyze large datasets of plant small RNA sequences to find various plant-specific RNA elements.

5. Conclusion

In this chapter, we provided an overview of the tools available for assembling and analyzing the new generation sequencing technologies that have emerged in recent years. As these technologies have only recently become available and research on new technologies is ongoing, the associated software tools are also continuously being adapted. Therefore, the information provided here is just a starting point, rather than a complete survey of the field. We hope this information provides the necessary background and we urge the reader to follow the links provided within the text in order to obtain up-to-date information about the software tools described here.

References

1. Mardis, E. R. (2008) The impact of next generation sequencing technology on genetics, *Trends Genet* **24**, 133–141.
2. Pop, M. (2004) Shotgun sequence assembly, *Adv Comput* **60**, 193–248.
3. Pop, M., and Salzberg, S. L. (2008) Bioinformatics challenges of new sequencing technology, *Trends Genet* **24**, 142–149.
4. Ronaghi, M., Uhlen, M., and Nyren, P. (1998) A sequencing method based on real-time pyrophosphate, *Science* **281**, 363–365.
5. Huse, S. M., Huber, J. A., Morrison, H. G., Sogin, M. L., and Welch, D. M. (2007) Accuracy and quality of massively parallel DNA pyrosequencing, *Genome Biol* **8**, R143.
6. Margulies, M., Egholm, M., Altman, W. E., Attiya, S., Bader, J. S., Bemben, L. A., Berka, J., Braverman, M. S., Chen, Y. J., Chen, Z., Dewell, S. B., Du, L., Fierro, J. M., Gomes, X. V., Godwin, B. C., He, W., Helgesen, S., Ho, C. H., Irzyk, G. P., Jando, S. C., Alenquer, M. L., Jarvie, T. P., Jirage, K. B., Kim, J. B., Knight, J. R., Lanza, J. R., Leamon, J. H., Lefkowitz, S. M., Lei, M., Li, J., Lohman, K. L., Lu, H., Makhijani, V. B., McDade, K. E., McKenna, M. P., Myers, E. W., Nickerson, E., Nobile, J. R., Plant, R., Puc, B. P., Ronan, M. T., Roth, G. T., Sarkis, G. J., Simons, J. F., Simpson, J. W., Srinivasan, M., Tartaro, K. R., Tomasz, A., Vogt, K. A., Volkmer, G. A., Wang, S. H., Wang, Y., Weiner, M. P., Yu, P., Begley, R. F., and Rothberg, J. M. (2005) Genome sequencing in microfabricated high-density picolitre reactors, *Nature* **437**, 376–380.
7. Wheeler, D. A., Srinivasan, M., Egholm, M., Shen, Y., Chen, L., McGuire, A., He, W., Chen, Y. J., Makhijani, V., Roth, G. T., Gomes, X., Tartaro, K., Niazi, F., Turcotte, C. L., Irzyk, G. P., Lupski, J. R., Chinault, C., Song, X. Z., Liu, Y., Yuan, Y., Nazareth, L., Qin, X., Muzny, D. M., Margulies, M., Weinstock, G. M., Gibbs, R. A., and Rothberg, J. M. (2008) The complete genome of an individual by massively parallel DNA sequencing, *Nature* **452**, 872–876.
8. Emrich, S. J., Barbazuk, W. B., Li, L., and Schnable, P. S. (2007) Gene discovery and annotation using LCM-454 transcriptome sequencing, *Genome Res* **17**, 69–73.
9. Harris, T. D., Buzby, P. R., Babcock, H., Beer, E., Bowers, J., Braslavsky, I., Causey, M., Colonell, J., Dimeo, J., Efcavitch, J. W., Giladi, E., Gill, J., Healy, J., Jarosz, M., Lapen, D., Moulton, K., Quake, S. R., Steinmann, K., Thayer, E., Tyurina, A., Ward, R., Weiss, H., and Xie, Z. (2008) Single-molecule DNA sequencing of a viral genome, *Science* **320**, 106–109.
10. Eid, J., Fehr, A., Gray, J., Luong, K., Lyle, J., Otto, G., Peluso, P., Rank, D., Baybayan, P., Bettman, B., Bibillo, A., Bjornson, K., Chaudhuri, B., Christians, F., Cicero, R., Clark, S., Dalal, R., Dewinter, A., Dixon, J., Foquet, M., Gaertner, A., Hardenbol, P., Heiner, C., Hester, K., Holden, D., Kearns, G., Kong, X., Kuse, R., Lacroix, Y., Lin, S., Lundquist, P., Ma, C., Marks, P., Maxham, M., Murphy, D., Park, I., Pham, T., Phillips, M., Roy, J., Sebra, R., Shen, G., Sorenson, J., Tomaney, A., Travers, K., Trulson, M., Vieceli, J., Wegener, J., Wu, D., Yang, A., Zaccarin, D., Zhao, P., Zhong, F., Korlach, J., and Turner, S. (2009) Real-time DNA sequencing from single polymerase molecules, *Science* **323**, 133–138.
11. Myers, E. W., Sutton, G. G., Delcher, A. L., Dew, I. M., Fasulo, D. P., Flanigan, M. J., Kravitz, S. A., Mobarry, C. M., Reinert, K. H., Remington, K. A., Anson, E. L., Bolanos, R. A., Chou, H. H., Jordan, C. M., Halpern, A. L., Lonardi, S., Beasley, E. M., Brandon, R. C., Chen, L., Dunn, P. J., Lai, Z., Liang, Y., Nusskern, D. R., Zhan, M., Zhang, Q., Zheng, X., Rubin, G. M., Adams, M. D., and Venter, J. C. (2000) A whole-genome assembly of Drosophila, *Science* **287**, 2196–2204.
12. Batzoglou, S., Jaffe, D. B., Stanley, K., Butler, J., Gnerre, S., Mauceli, E., Berger, B., Mesirov, J. P., and Lander, E. S. (2002) ARACHNE: a whole-genome shotgun assembler, *Genome Res* **12**, 177–189.
13. Jaffe, D. B., Butler, J., Gnerre, S., Mauceli, E., Lindblad-Toh, K., Mesirov, J. P., Zody, M. C., and Lander, E. S. (2003) Whole-genome sequence assembly for mammalian genomes: Arachne 2, *Genome Res* **13**, 91–96.
14. Green, P. (1994) Statistical aspects of imaging, *Stat Methods Med Res* **3**, 1–3.
15. Hillier, L. W., Marth, G. T., Quinlan, A. R., Dooling, D., Fewell, G., Barnett, D., Fox, P., Glasscock, J. I., Hickenbotham, M., Huang, W., Magrini, V. J., Richt, R. J., Sander, S. N., Stewart, D. A., Stromberg, M., Tsung, E. F., Wylie, T., Schedl, T., Wilson, R. K., and Mardis, E. R. (2008) Whole-genome sequencing and variant discovery in C. elegans, *Nat Methods* **5**, 183–188.
16. Salzberg, S. L., Sommer, D. D., Puiu, D., and Lee, V. T. (2008) Gene-boosted assembly of a novel bacterial genome from very short reads, *PLoS Comput Biol* **4**, e1000186.

17. Friedlander, M. R., Chen, W., Adamidi, C., Maaskola, J., Einspanier, R., Knespel, S., and Rajewsky, N. (2008) Discovering microRNAs from deep sequencing data using miRDeep, *Nat Biotechnol* **26**, 407–415.
18. Down, T. A., Rakyan, V. K., Turner, D. J., Flicek, P., Li, H., Kulesha, E., Graf, S., Johnson, N., Herrero, J., Tomazou, E. M., Thorne, N. P., Backdahl, L., Herberth, M., Howe, K. L., Jackson, D. K., Miretti, M. M., Marioni, J. C., Birney, E., Hubbard, T. J., Durbin, R., Tavare, S., and Beck, S. (2008) A Bayesian deconvolution strategy for immunoprecipitation-based DNA methylome analysis, *Nat Biotechnol* **26**, 779–785.
19. Kurtz, S., Phillippy, A., Delcher, A. L., Smoot, M., Shumway, M., Antonescu, C., and Salzberg, S. L. (2004) Versatile and open software for comparing large genomes, *Genome Biol* **5**, R12.
20. Kent, W. J. (2002) BLAT–the BLAST-like alignment tool, *Genome Res* **12**, 656–664.
21. Johnson, D. S., Mortazavi, A., Myers, R. M., and Wold, B. (2007) Genome-wide mapping of in vivo protein-DNA interactions, *Science* **316**, 1497–1502.
22. Li, H., Ruan, J., and Durbin, R. (2008) Mapping short DNA sequencing reads and calling variants using mapping quality scores, *Genome Res* **18**, 1851–1858.
23. Li, R., Li, Y., Kristiansen, K., and Wang, J. (2008) SOAP: short oligonucleotide alignment program, *Bioinformatics* **24**, 713–714.
24. Langmead, B., Trapnell, C., Pop, M., and Salzberg, S. L. (2009) Ultrafast and memory-efficient alignment of short DNA sequences to the human genome, *Genome Biol* **10**, R25.
25. Jiang, H., and Wong, W. H. (2008) SeqMap: mapping massive amount of oligonucleotides to the genome, *Bioinformatics* **24**, 2395–2396.
26. Lin, H., Zhang, Z., Zhang, M. Q., Ma, B., and Li, M. (2008) ZOOM! Zillions of oligos mapped, *Bioinformatics* **24**, 2431–2437.
27. Pop, M., Phillippy, A., Delcher, A. L., and Salzberg, S. L. (2004) Comparative genome assembly, *Brief Bioinform* **5**, 237–248.
28. Warren, R. L., Sutton, G. G., Jones, S. J., and Holt, R. A. (2007) Assembling millions of short DNA sequences using SSAKE, *Bioinformatics* **23**, 500–501.
29. Jeck, W. R., Reinhardt, J. A., Baltrus, D. A., Hickenbotham, M. T., Magrini, V., Mardis, E. R., Dangl, J. L., and Jones, C. D. (2007) Extending assembly of short DNA sequences to handle error, *Bioinformatics* **23**, 2942–2944.
30. Dohm, J. C., Lottaz, C., Borodina, T., and Himmelbauer, H. (2007) SHARCGS, a fast and highly accurate short-read assembly algorithm for de novo genomic sequencing, *Genome Res* **17**, 1697–1706.
31. Hernandez, D., Francois, P., Farinelli, L., Osteras, M., and Schrenzel, J. (2008) De novo bacterial genome sequencing: millions of very short reads assembled on a desktop computer, *Genome Res* **18**, 802–809.
32. Zerbino, D. R., and Birney, E. (2008) Velvet: algorithms for de novo short read assembly using de Bruijn graphs, *Genome Res* **18**, 821–829.
33. Pevzner, P. A., Tang, H., and Waterman, M. S. (2001) An Eulerian path approach to DNA fragment assembly, *Proc Natl Acad Sci U S A* **98**, 9748–9753.
34. Simpson, J. T., Wong, K., Jackman, S. D., Schein, J. E., Jones, S. J., and Birol, I. (2009) ABySS: a parallel assembler for short read sequence data, *Genome Res* **19**, 1117–1123.
35. Sommer, D. D., Delcher, A. L., Salzberg, S. L., and Pop, M. (2007) Minimus: a fast, lightweight genome assembler, *BMC Bioinformatics* **8**, 64.
36. Butler, J., MacCallum, I., Kleber, M., Shlyakhter, I. A., Belmonte, M. K., Lander, E. S., Nusbaum, C., and Jaffe, D. B. (2008) ALLPATHS: de novo assembly of whole-genome shotgun microreads, *Genome Res* **18**, 810–820.
37. Pop, M., Kosack, D. S., and Salzberg, S. L. (2004) Hierarchical scaffolding with Bambus, *Genome Res* **14**, 149–159.
38. Samad, A., Huff, E. F., Cai, W., and Schwartz, D. C. (1995) Optical mapping: a novel, single-molecule approach to genomic analysis, *Genome Res* **5**, 1–4.
39. Nagarajan, N., Read, T. D., and Pop, M. (2008) Scaffolding and validation of bacterial genome assemblies using optical restriction maps, *Bioinformatics* **24**, 1229–1235.
40. Goldberg, S. M., Johnson, J., Busam, D., Feldblyum, T., Ferriera, S., Friedman, R., Halpern, A., Khouri, H., Kravitz, S. A., Lauro, F. M., Li, K., Rogers, Y. H., Strausberg, R., Sutton, G., Tallon, L., Thomas, T., Venter, E., Frazier, M., and Venter, J. C. (2006) A Sanger/pyrosequencing hybrid approach for the generation of high-quality draft assemblies of marine microbial genomes, *Proc Natl Acad Sci U S A* **103**, 11240–11245.
41. Miller, J. R., Delcher, A. L., Koren, S., Venter, E., Walenz, B. P., Brownley, A., Johnson, J., Li, K., Mobarry, C., and Sutton, G. (2008)

Aggressive assembly of pyrosequencing reads with mates, *Bioinformatics* **24**, 2818–2824.

42. Lee, S., Cheran, E., and Brudno, M. (2008) A robust framework for detecting structural variations in a genome, *Bioinformatics* **24**, i59–i67.
43. Huson, D. H., Auch, A. F., Qi, J., and Schuster, S. C. (2007) MEGAN analysis of metagenomic data, *Genome Res* **17**, 377–386.
44. Krause, L., Diaz, N. N., Goesmann, A., Kelley, S., Nattkemper, T. W., Rohwer, F., Edwards, R. A., and Stoye, J. (2008) Phylogenetic classification of short environmental DNA fragments, *Nucleic Acids Res* **36**, 2230–2239.
45. Ye, Y., and Tang, X. (2008) in "Proceedings of the Seventh Annual International Conference on Computational Systems Bioinformatics", Stanford, CA.
46. Moxon, S., Schwach, F., Dalmay, T., Maclean, D., Studholme, D. J., and Moulton, V. (2008) A toolkit for analysing large-scale plant small RNA datasets, *Bioinformatics* **24**, 2252–2253.

Chapter 2

RNA Structure Prediction

István Miklós

Abstract

We give an overview of RNA structure predictions in this chapter. We discuss here the main approaches to RNA structure prediction: combinatorial approaches, comparative approaches, and kinetic approaches. The main algorithms and mathematical concepts such as transformational grammars will be briefly introduced.

Key words: RNA secondary structure, Pseudoknot, Stochastic context-free grammars, Dynamic programming, Comparative methods, Folding kinetics

1. Introduction

RNA sequences are single-stranded biopolymers that can fold themselves. For a long time, only three types of RNA sequences were known, tRNAs, mRNAs, and rRNAs. Although Woese, Crick, and Orgel already in 1967 and 1968 suggested that RNA could act as a catalyst (1–3), the first ribozyme (enzymatic RNA) has been found by Cech and his colleagues only in the 1980s (4). The first algorithm to predict RNA secondary structure was published in 1980 (5), and today, several different approaches have been published for RNA structure prediction.

We will start with some formal definitions, and then briefly overview the different approaches for RNA structure prediction.

1.1. RNA Secondary Structure

By a formal definition, an RNA secondary structure is a set of pairs of positions, $\{(i_k \cdot j_k), k = 1,2,\ldots n\}$. Each position of the RNA string can participate in, at most, one pair. We will assume that $i_k < j_k$ for all ks. The pairs show which nucleic acids of the RNA sequence form base pairs.

David Fenyö (ed.), *Computational Biology*, Methods in Molecular Biology, vol. 673,
DOI 10.1007/978-1-60761-842-3_2, © Springer Science+Business Media, LLC 2010

We say that a secondary structure is pseudoknot-free if for any two base pairs $i{\cdot}j$ and $i'{\cdot}j'$, $i<i'$ either $j<i'$ or $j'<j$. Namely, the two base pairs are separated or nested, see Fig. 1. Two base pairs in order $i<i'<j<j'$ form a pseudoknot. The simplest pseudoknot is shown in Fig. 2.

We say that a nucleic acid in position k separates the base pair $i{\cdot}j$ if $i<k<j$. A base pair $i'{\cdot}j'$ is nested into $i{\cdot}j$, if $i<i'<j'<j$.

A helix is a set of consecutive base pairs, namely, a set of pairs of positions $\{(i_k{\cdot}j_k), k=1,2,\ldots n\}$ in which for each k, $i_k+1=i_{k+1}$ and $j_{k+1}+1=j_k$.

If we denote the RNA sequence with a line and each base pair with an arc above the line, a pseudoknot-free secondary structure can be drawn without any crossing arcs. Some of the pseudoknotted structures can be drawn without crossing arcs if it is allowed to

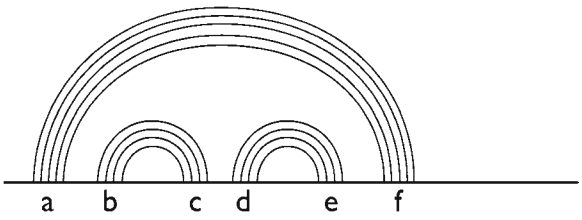


Fig. 1. Pseudoknot-free secondary structure. Each arc represents a base pair. Base pairs connecting regions **b** and **c** are nested into base pairs connecting regions **a** and **f**. Base pairs connecting regions **b** and **c** are separated from base pairs connecting regions **d** and **e**.

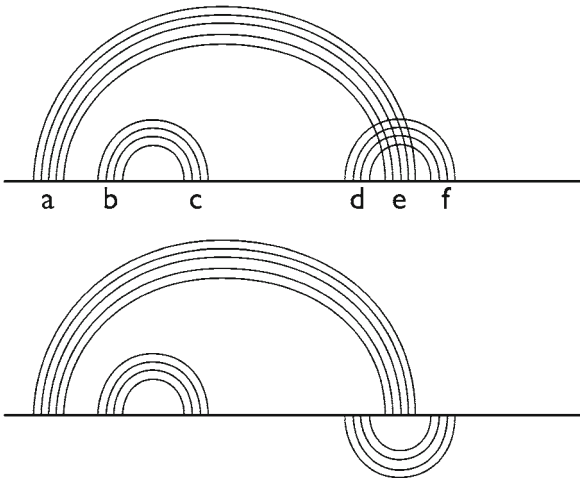


Fig. 2. Secondary structure with a planar pseudoknot. Base pairs connecting regions **a** and **e** are pseudoknotted with base pairs connecting regions **d** and **f**; however, the arcs can be drawn without crossing each other if both sides of the string representing the RNA sequence can be used.

use both sides of the line. These structures are called planar pseudoknotted secondary structures. There are pseudoknotted structures that are not planar. Such secondary structures appear in real life, too; for example, the *E. coli* alpha-operon ribosome possesses the simplest nonplanar pseudoknotted secondary structure. The topology of that structure is shown in Fig. 3: it contains three helices such that any two helices form a pseudoknot.

It is important to distinguish between pseudoknot-free, planar, and nonplanar pseudoknotted secondary structures from a computational point of view. Indeed, finding the best pseudoknot-free secondary structure is computationally easy. There are several ways to define what the "best" structure is, but in all cases, the running time of the algorithms that find the "best" structure takes $O(L^3)$ time, where L is the length of the RNA sequence. Though predicting planar pseudoknotted structures is still a theoretically easy computational problem, the running time of the optimization algorithm goes up to $O(L^6)$. Finding the best secondary structure when there are no limitations on the pseudoknotted structures is an NP-hard optimization problem even in very simple models.

1.2. Concepts of Predicting RNA Structures

Just like in other parts of bioinformatics, it is also true for RNA sequences that measuring the structure in lab is significantly more costly and time-consuming than obtaining the sequence itself. Therefore, a central task in structural RNA bioinformatics is to predict (secondary) structures from RNA sequences. There are several concepts how to choose a secondary structure as the prediction for the structure of an RNA sequence.

1.2.1. Combinatorial Approaches

Combinatorial approaches define a score function for each possible RNA secondary structure and try to find the structure that minimizes or maximizes this function. They use combinatorial optimization techniques, typically dynamic programming approaches that can find the optimal solution without investigating each particular solution.

The simplest approach associates a weight for each possible base pair and tries to maximize the sum of the weights of the base pairs in the secondary structure. The reason for this is that each

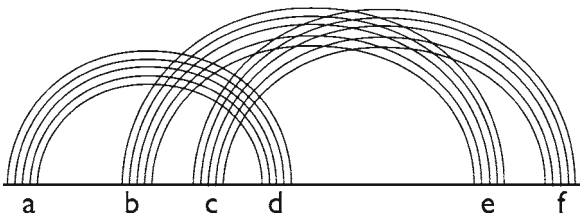


Fig. 3. A nonplanar pseudoknot. The three sets of arcs cannot be drawn without crossing each other even when using both sides of the string.

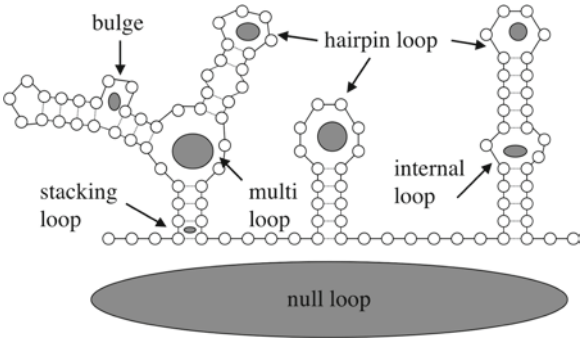


Fig. 4. Any pseudoknot-free RNA structure can be decomposed into cycles. See text for more details.

nucleic acid pair makes hydrogen bonds, which deepens the free energy.

Tinoco and his colleagues introduced an energy model (6, 7). They decomposed pseudoknot-free RNA structures into loops. They defined the following loops (see Fig. 4):

1. Null loop. This is the loop that is not a real loop. If we connect the 5′ end of an RNA sequence with its 3′ end, then we would get a loop, and this would correlate with the null loop. As per a precise mathematical definition, a null loop contains those single-stranded nucleic acids that do not separate any base pair and the nucleic acids which are base-paired but are not nested into other base pairs.
2. Stacking loop. This loop is formed by the hydrogen bonds of two consecutive base pairs in a helix and the sugar–phosphate backbone between the nucleic acids of the two pairs. The name of the loop is after the fact that there are stacking forces between two neighbor base pairs that stabilizes the secondary structure.
3. Internal loop. An internal loop is a loop inside a helix. A special internal loop is a bulge. A bulge contains single-stranded nucleotides only on one of the RNA strands.
4. Multiloop. A multiloop is a loop where a helix branches into several (at least two) helices.
5. Hairpin loop. A hairpin loop closes a helix.

The free energy of an RNA secondary structure is the sum of the free energies of the loops. The individual loop energies can be measured in lab. Zucker and Sankoff (8) gave the first polynomial running time algorithm that finds the pseudoknot-free secondary structure in $O(L^3)$ time, where L is the length of the RNA sequence.

1.2.2. Comparative Methods

Comparative methods assume that the structure is more conserved than the sequence itself, and homologous sequences have the same structure. To maintain the structure, base pairs coevolve hence keeping the secondary structure. This coevolutionary pattern provides the base of the comparative methods, which try to find a structure that all the sequences can take. Some of the methods need a multiple alignment as input, while other methods try to align and estimate the secondary structure in a common framework.

1.2.3. Folding Kinetics

There are evidences that the folding of an RNA sequence starts with its transcription. The secondary structure that an RNA sequence possesses might not necessarily be the minimum free energy (mfe). Indeed, as in silico, searching algorithms might not be able to find the mfe structure, since RNA sequences might not fold into the mfe structure in vivo. Therefore, it is a reasonable approach to try to simulate in silico the folding kinetics of RNA sequences and thereby predict their secondary structures.

2. Combinatorial Optimization

In this section, we give an overview of combinatorial approaches.

2.1. Nussinov Algorithm

We start with the simplest method that maximizes the number of base pairs in pseudoknot-free secondary structures. Below, we will talk about pseudoknot-free secondary structures, and until mentioned otherwise, secondary structure will mean pseudoknot-free secondary structure. The input of the Nussinov algorithm (5) is an RNA sequence A and a weight function $w:\Sigma\times\Sigma\to R$, where $\Sigma=\{A,C,G,U\}$, and for any two characters a and b, $w(a,b)$ defines the weight for making a base pair between a and b. The output of the algorithm is secondary structure $\{(i_k\cdot j_k), k=1,2,\ldots n\}$ that maximizes

$$\sum_{k=1}^{n} w\left(a_{ik}, a_{jk}\right)$$

where a_i is the character in the ith position. The algorithm is a dynamic programming algorithm. The dynamic programming algorithms try to find the solution for a problem using solutions of subproblems. The Nussinov algorithm finds the weight of the maximum weight secondary structure of each substring $a_i\ldots a_j$. The dynamic programming idea is that whatever the maximum

weight the secondary is for substring $a_i \ldots a_j$, at least one of the following holds:

1. a_i is not base-paired.
2. a_j is not base-paired.
3. a_i is base-paired with a_j.
4. Both a_i and a_j are base-paired, but not with each other.

Let $n(i,j)$ denote the weight of the maximum weight secondary structure that the substring $a_i \ldots a_j$ can possess. If case 1 holds for this structure, then

$$n(i,j) = n(i+1,j)$$

Indeed, if a_i is not base-paired, then all base pairs in $a_i \ldots a_j$ are also in the substring $a_{i+1} \ldots a_j$. Similarly, if a_j is not base-paired in the maximum weight secondary structure of substring $a_i \ldots a_j$, then all base pairs in $a_i \ldots a_j$ will be base pairs in the substring $a_i \ldots a_{j+1}$. If a_i is base-paired with a_j, then the maximum weight secondary structure of $a_i \ldots a_j$ will contain one more base pair than the maximum weight secondary structure of $a_{i+1} \ldots a_{j-1}$. Finally, if a_i is base-paired with some a_k, $k \neq j$, then there is no base pair $l \cdot l'$ for which $i < l < k < l'$, since it would be a pseudoknot. Therefore, the substring can be cut into two parts between the nucleotides in position k and $k+1$ without cutting any base pair.

Since we do not know which one from the above mentioned four cases holds for a substring $a_i \ldots a_j$, and for which k, a_i is base-paired with a_k if only case 4 holds in the above list, we have to consider all cases. Therefore, the recursion of the Nussinov algorithm is the following:

$$n(i,j) = \max \begin{cases} n(i+1,j); \\ n(i,j-1); \\ n(i+1,j-1) + w(i,j); \\ \max\limits_{i<k<j} \{n(i,k) + n(k+1,j)\} \end{cases}$$

The scores $n(i, j)$ must be calculated for each $1 \leq i < j \leq$, starting with short substrings and then longer ones. Once $n(1,L)$ is calculated, the maximum scoring secondary structure can be drawn by tracebacking the recursion.

2.2. Zuker–Sankoff Algorithm

The main problem with maximizing the score of base pairings is that the stacking energies between base pairs contribute significantly to the stabilization of the secondary structure. Moreover, the entropy of different loops also significantly contribute to the free energy of the secondary structure. Tinoco et al. introduced an energy model in which the free energy of a secondary structure is the sum of free energies of different loops, see Subheading 1.2.1.

Zuker and Sankoff (8) gave the first algorithm that finds the mfe secondary structure in the Tinoco energy model (6, 7).

Here, we give a simplified description of the Zuker–Sankoff algorithm, the readers are referred to refs. 9–11 for further details. The basic concept of the algorithm is that for each j long prefix, the free energy of the mfe secondary structure is calculated. The free energy of the null loop is simply the sum of the so-called dangling energies of base pairs in the null loop. The dangling energies are the free energies due to the interaction between the base pairs and the neighbor nucleic acids. If a_j is not base-paired, then the free energy of the mfe structure of the j long prefix is the free energy of the $j-1$ long prefix (neglecting dangling energies). If a_j is base-paired, then the prefix can be cut into two parts, a shorter prefix and a substring. Therefore, the recursion is:

$$F(j) = \max\left\{F(j-1), \max_{1<k<j}\left\{F(k-1) + C(k,j) + \text{dangling}\right\}\right\}$$

where $C(i, j)$ tells the free energy of the mfe secondary structure of the $a_i \ldots a_j$ substring in which a_i is base-paired with a_j. This base pair might close a

1. A hairpin
2. An internal loop
3. A multiloop
4. A helix

There is only one possible structure in which the base pair closes a hairpin-loop. The Zuker lab keeps refining the free energies associated to different hairpin-loops (12). The recent software packages (10, 13) implementing the Zuker algorithm score hairpin-loops according to the most up-to-date published values.

When $i \cdot j$ closes an internal loop, the dynamic programming recursion has to consider all $i < p < q < j$, for which $p \cdot q$ closes the other end of the loop. Since there are $O(L^2)$ possible $i \cdot j$ base pairs and for each $i \cdot j$ there are $O(L^2)$ possible $p \cdot q$ pairs, this part of the dynamic programming recursion would need $O(L^4)$ running time on its own. For the current scoring of internal loops, a speed-up to $O(L^3)$ is possible (14).

Since there is no theoretical upper bound on the number of helices appearing in a multiloop, dynamic programming is not possible for multiloops and arbitrary energy scores of multiloops. A simplified, linear model is applied for multiloops for which the free energy of a multiloop is defined as

$$a + b\#s + c\#d + \text{dangling}$$

where $\#s$ is the number of single-stranded nucleotides, and $\#d$ is the number of base pairs in the multiloop. The constants a, b, and

c are estimated with regression based on measured free-energies of different RNA sequences with known secondary structures.

The details of the dynamic programming for calculating $C(i, j)$ is quite involved and will not be introduced here. The readers are referred to the work of Wuchty et al. (9) and the references in it. The running time of the algorithm is $O(L^3)$.

2.3. The McCaskill Algorithm

The RNA sequences can dynamically change their secondary structures. The secondary structures of an ensemble of RNA sequences are in a Boltzmann distribution in which the probability of a particular structure S is

$$P(S) = \frac{1}{Z} e^{-\Delta G(S)/RT}$$

where $\Delta G(S)$ is the free energy of the structure, R is the universal gas constant, and Z is the partition function:

$$Z = \sum_{S'} e^{-\Delta G(S')/RT}$$

where the sum is over all the possible structures that the RNA sequence might have. McCaskill (15) gave the first algorithm that calculated this partition function. The algorithm uses similar dynamic programming ideas than the Zuker–Sankoff algorithm, but it uses additions and multiplications instead of maximization. The idea is that if we already calculated

$$\sum_{S} e^{-\Delta G(S)/RT}$$

where the sum is over all the possible secondary structures of a substring $a_i \ldots a_k$, and

$$\sum_{S'} e^{-\Delta G(S')/RT}$$

where the sum is over all the possible secondary structures of a substring $a_{k+1} \ldots a_j$, then

$$\left(\sum_{S} e^{-\Delta G(S)/RT} \right) \times \left(\sum_{S'} e^{-\Delta G(S')/RT} \right) = \sum_{S \cup S'} e^{-\Delta G(S \cup S')/RT}$$

is the partial partition function of substring $a_i \ldots a_j$, that consider such secondary structures of $a_i \ldots a_j$ that can be cut between position k and $k+1$.

The dynamic programming algorithm must be implemented carefully, since it is possible to cut a secondary structure into two parts at several positions without cutting any base pair. Hence, a noncareful implementation might consider the same secondary structure many times, which would yield an overcounting of the partition function. Fortunately, it is possible to decompose each

possible secondary structure into smaller components in a unique way, and this unequivocal decomposition is the base of the McCaskill algorithm. For details, see also refs.(9) and (11).

2.4. Predicting Pseudoknots

Rivas and Eddy published a dynamic programming algorithm for predicting any planar pseudoknotted structure (16, 17). The idea is that they calculate the best possible secondary structure for any pair of substrings. A planar pseudoknotted secondary structure can always be decomposed into two smaller parts such that the smaller parts also contain planar pseudoknotted structures, see Fig. 5. The two substrings can be described by the beginnings and ends of the two substrings, i, j, k, and l; therefore, it needs an $O(L^4)$ memory usage. For each pair of substrings, two cutting points, r and s, are needed to split the structure into two parts. Hence, the overall running time of the algorithm is $O(L^6)$.

There are special algorithms that run in only $O(L^5)$ running time; however, these algorithms can predict only some special pseudoknots and cannot consider all possible planar pseudoknotted secondary structures (18–20). The partition function can also be calculated for several pseudoknot model (21). Interested readers are referred to a review paper by Reeder and Giegerich (22).

2.5. The General Pseudoknot Problem

The general pseudoknot prediction problem is NP-hard. The first proof for NP-hardness was given by Lyngsoe and Pedersen in (19, 20). Lyngsoe (23) considered three very simple models. In these models, the best structure is that maximizes:

1. The number of base pairs
2. The number of base pair stackings
3. The number of stacking base pairs

The difference between the two later models is that the score of an m long helix is $m-1$ when counting the number of base pair stackings, while the score is m when counting the number of stacking base pairs, $m>1$.

The first approach that maximizes the number of base pairs is equivalent to the Maximum Weighted Matching problem.

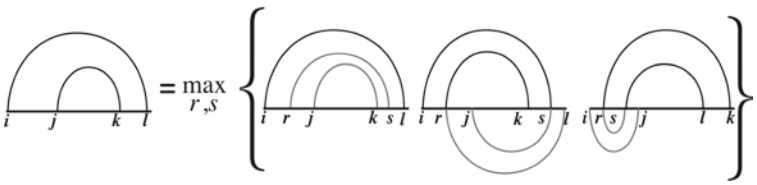


Fig. 5. The schematic representation of the Rivas–Eddy dynamic programming algorithm that can predict arbitrary planar pseudoknots. The algorithm obtains the best secondary structure for each pair of substrings. Due to limited space, the case when both r and s are in the interval $[l,k]$ is not indicated

Lyngsoe showed that it is NP-hard to determine if an RNA sequence can accommodate a secondary structure that contains a given number of base pair stackings. Finding the structure that maximizes the number of stacking base pairs is also NP-hard if the size of the alphabet is not limited. For a four-letter alphabet, the best algorithm he could give was an $O(L^{81})$ algorithm, which is obviously practically intractable, though theoretically it is a polynomial running time algorithm.

3. Comparative Methods

Comparative methods assume that the structure is more conserved than the sequences themselves. Hence, they aim at predicting the joint secondary structure of a set of sequences.

3.1. The Knudsen–Hein Grammar

Although this is not the historical order of works, we start this section with the Knudsen–Hein grammar (24, 25) for didactic reasons. The Knudsen–Hein grammar is a Stochastic Context-Free Grammar (SCFG) that describes the joint secondary structure of a set of aligned RNA structures. Context-Free grammars (CFGs) are special transformational grammars (26). A transformational grammar is a tuple $\{N, T, S, R\}$, where N is a finite set of nonterminal symbols, T is a finite set of terminal symbols, S, the starting nonterminal is an element of N, and R is a finite set of rewriting rules. The general form of a rewriting rule is

$$\alpha \rightarrow \beta$$

where α is a substring of terminal and nonterminal symbols and contains at least one nonterminal character, and β is an arbitrary substring of terminal and nonterminal symbols; it might contain no nonterminal symbols. A generation of a transformational grammar starts with rewriting the starting non-terminal, S to some substring, and then continues with rewriting any substring of the so-generated intermediate string. The generation stops when the string contains only terminal symbols. In a CFG, all rewriting rule is in the form

$$W \rightarrow \beta$$

where W is a single nonterminal symbol, and β is an arbitrary substring of terminal and non-terminal symbols; it might contain only terminal symbols. It is called context-free because rewriting the nonterminal W does not depend on its content. When the same nonterminal can be rewritten into several substrings, we write

$$W \rightarrow \beta_1 | \beta_2 | \ldots | \beta_k$$

which means that W can be rewritten into β_1 or $\beta_2 \ldots \beta_k$. A CFG becomes stochastic if for each W, there is a probability distribution over the possible substrings that can replace W.

Knudsen and Hein introduced a SCFG that can generate all possible pseudoknot-free secondary structure. The rewriting rules are:

$$S \rightarrow LS \mid S$$
$$L \rightarrow s \mid dFd$$
$$F \rightarrow dFd \mid S$$

where s is a single-stranded nucleic acid, and ds are double-stranded nucleic acids. An example generation is shown on Fig. 6.

Knudsen and Hein used this grammar to estimate the common secondary structure of aligned RNA sequences. First, they estimated the rewriting probabilities training the grammar on known secondary structures. They also estimated parameters for a continuous-time Markov model describing the evolution of nucleotide substitutions. They also estimated parameters for a continuous-time Markov model describing the dinucleotide substitutions in helices. In both the cases, they estimated the parameters from a priori data. This dataset contained aligned RNA sequences with known secondary structures, hence it was known which nucleic acids are single-stranded and which are double-stranded. The authors mixed the SCFG with these substitution models. The final model needs an evolutionary tree as input and in this joint model, the SCFG generates alignment columns instead of a single s character, and a pair of alignment columns replacing the two ds in the dFd substring. The probability of rewriting the nonterminal L symbol into a particular alignment column is the product of the $L \rightarrow s$ rewriting probability multiplied with the likelihood of the alignment column, given an evolutionary tree. This likelihood can be efficiently calculated using the Felsenstein's algorithm (27). Generating correlated alignment columns replacing the two ds in the dFd substring can be done in a similar way.

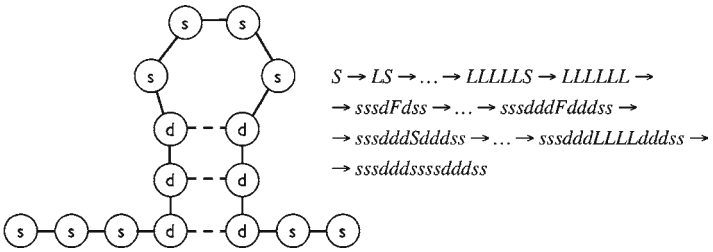


Fig. 6. An example generation of an RNA secondary structure in the Knudsen–Hein grammar.

The authors used numerical approaches to find the tree topology and edge lengths that maximize the probability of generating the multiple alignment. Once the Maximum Likelihood tree has been found, the most likely generation by the SCFG was calculated using the CYK algorithm. The CYK algorithm is also a dynamic programming algorithm that finds for each substring and nonterminal the most likely generation of the substring, starting with the nonterminal. The running time of the CYK algorithm is $O(L^3M^3)$, where L is the length of the RNA string and M is the number of nonterminal symbols. Since the number of nonterminal symbols is fixed, this algorithm – just like the Nussinov algorithm and the Zuker–Sankoff algorithm – runs in cubic time with the sequence length.

As also can be seen in Fig. 6, any generation of the Knudsen–Hein grammar defines a secondary structure in an unequivocal way, and this secondary structure is the prediction for the common secondary structure. It is also possible to calculate posterior probabilities that a nucleic acid is single stranded or base-paired with a particular partner nucleic acid. The posterior probability for being single stranded means the conditional probability that the character (or alignment column) was generated by the $L \rightarrow s$ rewriting rule, with the condition that the SCFG generated the sequence (or alignment). Similarly, the posterior probability that the nucleic acids in positions i and j are base-paired is the conditional probability that the nonterminal symbol F generated the substring $a_{i+1} \ldots a_{j-1}$, with the condition that the SCFG generated the sequence. Indeed, whenever an F appears on the right hand side of the rewriting rules, it comes together with two ds, which go to positions i and j if F generates the substring $a_{i+1} \ldots a_{j-1}$ Knudsen and Hein showed that this posterior probability correlates with the probability that the secondary structure prediction is correct for that particular position (25).

The Knudsen–Hein grammar is very simple. Though it can generate any pseudoknot-free secondary structure, the distribution of the structures it generates is far from the distribution that we can find in biological databases. Indeed, the grammar can generate a run of single-stranded nucleic acids with a geometric distribution, disregarding where these single-stranded nucleotides are, in a hairpin, an internal loop, a bulge, or multiloop. A better distribution can be achieved by increasing the number of nonterminals. The different nonterminals are used for generating different structural elements such as hairpin loops, internal loops, bulges, and multiloops. Nebel introduced such an extended grammar and showed that such an extended grammar indeed improves the goodness of predictions (28).

3.2. Covarion Models

Covarion Models can be considered as the CFG equivalent of profile Hidden Markov Models (29–31). While profile-HMMs

describe the profile of a multiple sequence alignment of protein sequences, Covarion Models describe the profile of a multiple sequence alignment of RNA sequences that belong to a fold family. They cannot generate arbitrary secondary structures, and they are specific to the common structure of the RNA sequences in the multiple alignment. The most likely generations (also called most likely parsings) align the RNA sequences together via their Covarion Model: if two characters in two RNA sequences are generated by the same nonterminal, then they are also aligned together in the multiple alignment. Similarly, if two pairs of nucleotides are generated by the same nonterminal, the two pairs are predicted to form base pairs, and both the left hand sides and the right hand sides are aligned together.

Since Covarion Models are specific to an RNA family, they are used to decide if a sequence belongs to a fold family. The difference between the energies of mfe secondary structures of a randomly drawn RNA sequence and a functional RNA sequence is not statistically sufficient to find novel RNA genes in genomic sequences (32, 33). On the other hand, Covarion Models are very successful in finding specific RNA genes. One of the most successful applications is the tRNAScan-SE (34), which can detect ~99% of eukaryotic nuclear or prokaryotic tRNA genes, with a false positive rate of less than one per 15 Gb, and with a search speed of about 30 kb/s. This high-throughput method first quickly select a small part of the genome that contains basically all tRNA genes using a simple Hidden Markov Model. Then, this smaller part is analyzed further with a Covarion Model specific for tRNA sequences.

3.3. Joint Prediction of Alignments and Secondary Structures

The main drawback of the Knudsen–Hein method is that it needs an accurate alignment to predict the common secondary structure of RNA sequences. Indeed, structure prediction for misaligned parts is very hard, and it is impossible if base-paired homologous nucleic acids are not aligned together. Covarion Models give both an alignment and a predicted structure for each sequence, however, they need a large set of homologous sequences for an accurate prediction.

The Knudsen–Hein grammar and the Covarion Models can predict only pseudoknot-free structures. Witwer introduced a method that can predict planar pseudoknotted structures for aligned RNA sequences (35). Her method is based on the Maximum Weighted Matching (MWM) algorithm (36). The MWM algorithm finds the set of base pairings that maximizes the sum of the weights of base pairings, without any constraint. Although it can predict arbitrary pseudoknots, the outcome might also contain a subset of base pairings that clearly cannot be formed due to steric constraints, see Fig. 7. Witwer suggested to start with an MWM algorithm, and then remove base pairs to get

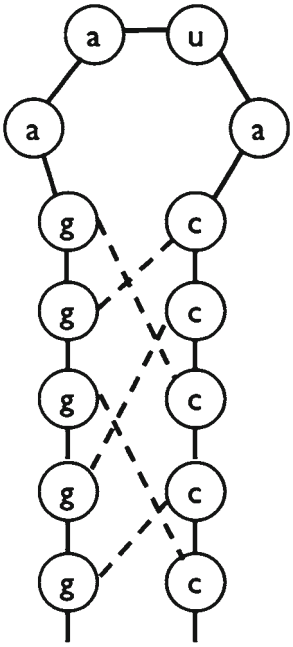


Fig. 7. A nonsense secondary structure that might be easily the outcome of a MWM algorithm. The MWM algorithm maximizes the sum of weights of base pairs, disregarding whether or not all base pairs can be formed. Clearly, the showed structure cannot exist in real life due to steric constraints.

a planar pseudoknot or pseudoknot-free structure; In a final step, the helices are extended, if possible. This algorithm also needs a good initial alignment.

When the number of homologous sequences is small and the sequences are hard to align without knowing their secondary structure, no good initial alignment is available. An ideal approach would coestimate the alignment and the secondary structure. Sankoff suggested the problem to align and estimate the common structure of two RNA sequences, and he gave an algorithm that runs in $O(L^6)$ running time, where L is the geometric mean of the sequence lengths (37). Unfortunately, this algorithm is too slow to be used in practice. Holmes and Rubin introduced the SCFG approach for pairwise RNA structure comparison (38), and Holmes introduced an evolutionary model for evolving RNA structures (39). Although this latter method has been accelerated (40), the acceleration is just a constant factor, and no theoretical breakthrough has been achieved.

Meyer and Miklós introduced a Markov chain Monte Carlo approach for the joint estimation of multiple alignments, evolutionary trees, and RNA secondary structures including pseudoknots (41). The drawback of the approach is that the convergence of the Markov chain might be quite slow, and it might take a lot of computational time to draw a sufficient number of samples

from the Markov chain. The authors also provide software for the analysis of the samples from the Markov chain. It is possible to highlight the consensus structure of the sampled structures and to give posterior probabilities for each base pair.

Finally, we mention the CARNAC method (42, 43) that tries to find a set of conserved helices in a family of homologous RNA sequences. The set of conserved helices are not allowed to form a pseudoknot. The approach is based on the k-clique problem, which is known to be NP-hard. CARNAC has an implementation that is reasonably fast on small datasets, and hence it provides a practical approach.

4. Folding Kinetics

There are both experimental (44–46) and statistical evidences (47) that RNA sequences start folding during their translation. Gultyaev published the first work on simulating RNA folding pathways (48, 49). Isambert and his colleagues implemented a software package called KineFold that also simulates RNA folding (50, 51). They also considered the possibility of cotranscriptional folding, namely, the RNA sequence is being folded during its transcription in the computer simulation. The authors also provide a graphical interface for visualizing the folding dynamics. KineFold is also available on a webserver. Pseudoknotted structures are allowed in Kinefold.

So far, all implemented methods simulate the folding of a single RNA sequence. Comparative approaches do not consider directly the folding dynamics, but they try to find common local structures, for example, in mRNA sequences (52, 53).

5. Discussion

In this paper, we have given an overview of approaches and techniques for predicting RNA secondary structures. There are three main approaches for predicting RNA secondary structures: combinatorial optimization methods, comparative methods, and folding simulations. The speed of algorithms based on combinatorial optimization depends on whether or not pseudoknots are allowed, and if so, what kind of pseudoknots might be considered. Although pseudoknots are common in biological RNA sequences (54], the general pseudoknot prediction problem is hard. Planar pseudoknots can be predicted in polynomial time; however, the $O(L^5)$ or $O(L^6)$ running time makes these algorithms impractical.

Comparative approaches are also common, and if there are more than one sequence within the same secondary structure, there is at least a hope that the common structure can be predicted with a better accuracy due to the increased amount of information represented in the set of homologous sequences.

Kinetic approaches try to simulate the folding dynamics of RNA sequences. Although these simulations are very impressive, so far, no large-scale analysis has been published about the accuracy of such methods. Nevertheless, the kinetic approach has not yet been combined with comparative approaches. We even do not know how well the folding dynamics is conserved during evolution.

Acknowledgements

The author thanks the following grants: Bolyai postdoctoral fellowship, OTKA grant F61730. Caddy and Kanazawa are thanked for their support.

References

1. Woese CR (1967) The Genetic Code. New York, Evanston and London: Harper and Row.
2. Crick FHC (1968) The origin of the genetic code. J Mol Biol 38: 367.
3. Orgel LE (1968) Evolution of the genetic apparatus. J Mol Biol 38: 381.
4. Kruger K, Grabowski PJ, Zaug AJ, Sands J, Gottschling DE, Cech TR (1982) Self-splicing RNA: Autoexcision and autocyclization of the ribosomal RNA intervening sequence of Tetrahymena. Cell 31: 147–157.
5. Nussinov R, Jacobson A (1980) Fast algorithm for predicting the secondary structure of single-stranded RNA. Proc Natl Acad Sci U S A 77: 6309–6313.
6. Tinoco I Jr, Uhlenbeck OC, Levine MD (1971) Estimation of secondary structure in ribonucleic acids. Nature 230: 362–367.
7. Tinoco I Jr, Borer PN, Dengler B, Levine MD, Uhlenbeck OC, et al (1973) Improved estimation of secondary structure in ribonucleic acids. Nature New Biol 246: 40–41.
8. Zuker M, Sankoff D (1984) RNA secondary structures and their prediction. Bull Math Biol 46: 591–621.
9. Wuchty S, Fontana W, Hofacker I, Schuster P (1999) Complete suboptimal folding of RNA and the stability of secondary structures. Biopolymers 49: 145–165.
10. Zuker M (2003) Mfold web server for nucleic acid folding and hybridization prediction. Nucleic Acids Res 31: 3406–3415.
11. Miklós I, Meyer IM, Nagy B (2005) Moments of the Boltzmann distribution for RNA secondary structures. Bull Math Biol 67(5): 1031–1047.
12. Mathews D, Sabina J, Zuker M, Turner D (1999) Expanded sequence dependence of thermodynamic parameters improves prediction of RNA secondary structure. J Mol Biol 288: 911–940.
13. Hofacker IL (2003) Vienna RNA secondary structure server. Nucleic Acids Res 31: 3429–3431.
14. Lyngsø R, Zuker M, Pedersen C (1999) Fast evaluation of internal loops in RNA secondary structure prediction. Bioinformatics 15: 440–445.
15. McCaskill JS (1990) The equilibrium partition function and base pair binding probabilities for RNA secondary structure. Biopolymers 29: 1105–1119.
16. Rivas E, Eddy SR (1999) A dynamic programming algorithm for RNA structure prediction including pseudoknots. J Mol Biol 285: 2053–2068.

17. Rivas E, Eddy SR (2000) The language of RNA: A formal grammar that includes pseudoknots. Bioinformatics 16: 334–340.
18. Akutsu T (2000) Dynamic programming algorithms for RNA secondary prediction with pseudoknots. Discrete Appl Math 104: 45–62.
19. Lyngsø R, Pedersen C (2000) RNA pseudoknot prediction in energy based models. J Comput Biol 7: 409–428.
20. Lyngsø R, Pedersen C (2000) Pseudoknots in RNA secondary structures. In: Shamir R, Miyano S, Istrail S, Pevzner P, Waterman M, editors. Proceedings of the Fourth Annual International Conference on Computational Molecular Virology. New York: ACM Press. pp. 201–209.
21. Dirks RM, Pierce NA (2003) A partition function algorithm for nucleic acid secondary structure including pseudoknots. J Comput Chem 24: 1664–1677.
22. Reeder J, Giegerich R (2004) Design, implementation and evaluation of a practical pseudoknot folding algorithm based on thermodynamics. BMC Bioinformatics 5: 104.
23. Lyngsø R (2004) Complexity of pseudoknot prediction in simple models. In: Diaz J, Karhumäki J, Lepistö A, Sannella D, editors. Proceedings of the 31st International Colloquium on Automata, Languages, and Programming (ICALP), 12–16 July 2004, Turku, Finland. pp. 919–931.
24. Knudsen B, Hein J (1999) RNA secondary structure prediction using stochastic context-free grammars and evolutionary history. Bioinformatics 15: 446–454.
25. Knudsen B, Hein J (2003) Pfold: RNA secondary structure prediction using stochastic context-free grammars. Nucleic Acids Res 31: 3423–3428.
26. Durbin R, Eddy S, Krogh A, Mitchison G (1998) Biological sequence analysis: Probabilistic models of proteins and nucleic acids. Cambridge: Cambridge University Press. p. 356.
27. Felsenstein J (1981) Evolutionary trees from DNA sequences: A maximum likelihood approach. J Mol Evol 17(6): 368–376.
28. Nebel M (2004) Identifying good predictions of RNA secondary structure. Proc Pac Symp Biocomput 9: 423–434.
29. Eddy SR, Durbin R (1994) RNA sequence analysis using covariance models. Nucleic Acids Res 22: 2079–2088.
30. Hofacker IL, Fontana W, Stadler PF, Bonhoeffer S, Tacker M, et al (1994) Fast folding and comparison of RNA secondary structures. Monatsh Chem 125: 167–188.
31. Sakakibara Y, Brown M, Underwood R, Mian IS, Haussler D (1994) Stochastic context-free grammars for modeling RNA. In: Proceedings of the 27th Hawaii International Conference on System Sciences. Honolulu: IEEE Computer Society Press. pp. 283–284.
32. Rivas E, Eddy SR (2000) Secondary structure alone is generally not statistically significant for the detection of noncoding RNAs. Bioinformatics 16(7): 583–605.
33. Workman C, Krogh A (1999) No evidence that mRNAs have lower folding free energies than random sequences with the same dinucleotide distribution. Nucleic Acids Res 27(24): 4816–4822.
34. Lowe T, Eddy S (1997) tRNAscan-SE: A program for improved detection of transfer RNA genes in genomic sequence. Nucleic Acids Res 25: 955–964.
35. Witwer C (2003) Prediction of conserved and consensus RNA structures [dissertation]. Vienna: Universität Wien. p. 187.
36. Tabaska J, Cary R, Gabow H, Stormo G (1998) An RNA folding method capable of identifying pseudoknots and base triples. Bioinformatics 14: 691–699.
37. Sankoff D (1985) Simultaneous solution of the RNA folding, alignment and protosequence problems. SIAM J Appl Math 45: 810–825.
38. Holmes I, Rubin G (2002) Pairwise RNA structure comparison with stochastic context-free grammars. Pac Symp Biocomput 2002: 163–174.
39. Holmes I (2004) A probabilistic model for the evolution of RNA structure. BMC Bioinformatics 5: 166.
40. Holmes I (2005) Accelerated probabilistic inference of RNA structure evolution. BMC Bioinformatics 6: 73.
41. Miklós I, Meyer IM (2007) SimulFold: Simultaneously inferring RNA structures including pseudoknots, alignments, and trees using a Bayesian MCMC framework. PLoS Comput Biol 3(8): e149.
42. Perriquet O, Touzet H, Dauchet M (2003) Finding the common structure shared by two homologous RNAs. Bioinformatics 19: 108–116.
43. Touzet H, Perriquet O (2004) CARNAC: Folding families of related RNAs. Nucleic Acids Res 32: W142–W145.
44. Boyle J, Robillard G, Kim S (1980) Sequential folding of transfer RNA. A nuclear magnetic resonance study of successively longer tRNA

fragments with a common 59 end. J Mol Biol 139: 601–625.

45. Morgan SR, Higgs PG (1996) Evidence for kinetic effects in the folding of large RNA molecules. J Chem Phys 105: 7152–7157.
46. Heilmann-Miller SL, Woodson SA (2003) Effect of transcription on folding of the Tetrahymena ribozyme. RNA 9: 722–733.
47. Meyer IM, Miklós I (2004) Co-transcriptional folding is encoded within RNA genes. BMC Mol Biol 5: 10.
48. Gultyaev A (1991) The computer-simulation of RNA folding involving pseudoknot formation. Nucleic Acids Res 19: 2489–2493.
49. Gultyaev A, von Batenburg F, Pleij C (1995) The computer-simulation of RNA folding pathways using a genetic algorithm. J Mol Biol 250: 37–51.
50. Isambert H, Siggia E (2000) Modeling RNA folding paths with pseudoknots: Application to hepatitis delta virus ribozyme. Proc Natl Acad Sci U S A 97: 6515–6520.
51. Xayaphoummine A, Bucher T, Thalmann F, Isambert H (2003) Prediction and statistics of pseudoknots in RNA structures using exactly clustered stochastic simulations. Proc Natl Acad Sci U S A 100: 15310–15315.
52. Pedersen JS, Forsberg R, Meyer IM, Hein J (2004) An evolutionary model for protein-coding regions with conserved RNA structure. Mol Biol Evol 21: 1913–1922.
53. Pedersen JS, Meyer IM, Forsberg R, Simmonds P, Hein J (2004) A comparative method for finding and folding RNA secondary structures within protein-coding regions. Nucleic Acids Res 32: 4925–4936.
54. Staple DW, Butcher SE (2005) Pseudoknots: RNA structures with diverse functions. PLoS Biol 3: e213. doi:10.1371/journal.pbio.0030213.

Chapter 3

Normalization of Gene-Expression Microarray Data

Stefano Calza and Yudi Pawitan

Abstract

Expression microarrays are designed to quantify the amount of mRNA in a specific sample. However, this can only be done indirectly through quantifying the color intensities returned by labeled mRNA molecules bound to the array surface. Translating pixel intensities into transcript expression requires a series of computations, generically known as preprocessing and normalization steps. In this chapter, we introduce the basic concepts and methods, and illustrate them using data from three commonly used commercial platforms.

Key words: Affymetrix, Agilent, Illumina, mRNA expression, Oligonucleotide arrays

1. Introduction

Along with the intrinsic biological variability, microarray data will show nonbiological or technical variability due to many potential sources: slide fabrication, biological material extraction, quantities of mRNA hybridized, sample labeling, dye affinity, scanner settings, image acquisition conditions, spatial anomalies, etc. The normalization step is the process that attempts to remove or reduce these systematic technical biases among the samples. This is a crucial step that can substantially affect the results of downstream statistical analyses.

In this chapter, we introduce some basic concepts, followed by several normalization methods. The list is not exhaustive, but rather represents a description of the most commonly-used procedures. For practical illustrations, we use real data from a comparison study (1) that employed the three currently most commonly-used commercial platforms, namely, Affymetrix (2), Agilent (3), and Illumina (4). Multiple use of technology platforms on the same samples is ideal for illustration, since each has

David Fenyö (ed.), *Computational Biology*, Methods in Molecular Biology, vol. 673,
DOI 10.1007/978-1-60761-842-3_3,

its own unique probe design and issues in the normalization step. The usage of software packages available under the R statistical environment (5) and the Bioconductor framework (6) will be demonstrated.

2. Preprocessing

2.1. Image Processing

The single slide or chip is first read by a scanner, typically followed by some image-analysis steps: gridding, segmentation, and intensity extraction. Gridding is the process of assigning coordinates to each of the probes/spots; segmentation classifies the pixels as foreground (belonging to a specific probe) or as background (outside the probe); during extraction, foreground and background intensities are summarized for each probe. There could be other image-analysis steps specific to each technology; we refer to the manufacturer's manual for details.

2.2. Background Correction

Background noise may derive from many sources such as target binding to slide substrate, processing effects, and cross-hybridization or nonspecific binding. Automated hybridization reduces background noise, thanks to tighter control of temperature and chemical conditions, as well as robot intervention. Nevertheless, a certain amount of baseline noise is inevitable, and it must be accounted for in the preprocessing.

When both background and foreground values are available (e.g., in Agilent arrays), background correction can be performed via subtraction of the background from the foreground values, based on a simplistic assumption of an additive background effect. As an undesired side effect, this method might generate a lot of negative values that are problematic since most downstream analyses are done in log scale. According to a recent comparison paper (7), the best method is the so-called *normexp*, which is a modification of the Affymetrix background correction implemented in the RMA method below.

Because of the tight probe arrangement on the chip, the Affymetrix oligonucleotide array cannot use the pixels surrounding the probe/spot to estimate background levels. Therefore, probe intensities have to be used to estimate both foreground and background signals. Several procedures are available, though there are two that are most commonly used. The first one is implemented in the so-called MAS5 algorithm (8). It works at single-array basis and uses both perfect-match (PM) and mismatch (MM) probe intensities. Two corrections are applied: (1) a location-specific background adjustment, aimed at removing overall background noise, (2) PM correction based on the MM

value. For (1), each array is divided in 16 equally sized regions arranged in 4-by-4 grids. Within region k, a background value b_k is estimated using the mean of the lowest 2% of probe values, and a noise level n_k based on their standard deviation. Each probe intensity $P(x, y)$ is then adjusted using a weighted average for the background values $B(x, y)$, as well as for the noise values $N(x, y)$, as follows

$$P(x,y) = \max\{P(x,y) - B(x,y); \alpha N(x,y)\}. \quad (1)$$

where α is a tuning parameter (defaults to 0.5), and $B(x,y) = \sum w_k(x,y) b_k$, and the weight w_k is a function of the Euclidean distance of the probe location (x, y) and the centroid of grid k.

Perfect-match correction based on MM value was originally designed by a simple subtraction of MM from the PM intensities. This turned out to be problematic as around 30% of MM values are bigger than the corresponding PM values (9), thus causing negative values. The remedy proposed by Affymetrix is based on the computation of the *ideal mismatch*, a quantity that is smaller than PM (10). It should be noted that the estimate of both the foreground intensity and its local background will inevitably be affected by measurement errors, so that subtracting one noisy estimate from another is going to increase the overall variability of the signal.

The second procedure is that used by the Robust Multichip Analysis (RMA) algorithm (11, 12). It is based on the assumption that the observed PM intensity X is the sum of a Gaussian component for the *real* probe signal S and an exponential component for the background B. In contrast to the MAS5 algorithm, the MM intensities are not used. First, the mode of the distribution of PM values is estimated. Then, points above the mode are used to estimate the parameters of the exponential, while a half-Gaussian is fitted to those below the mode to estimate the Gaussian parameters. The final background-corrected intensities are computed as the conditional mean $E(S|X)$, which is the best predicted value of the signal S given the observed data X.

3. Different Methods of Normalization

Most normalization procedures rely on rather strong assumptions: (1) the great majority of genes are unaltered between arrays, and (2) genes are expected to be roughly symmetrically distributed among the upregulated and the downregulated. These assumptions seem reasonable in many lab studies, where an intervention is not expected to have extensive changes in global

gene expression. However, they are questionable in many clinical studies, such as those with heterogeneous samples or custom-made chips (e.g., human cancer chips). When these assumptions are not met, most normalization methods would fail in removing the unwanted technical variation, and still worse, they might introduce some unpredictable biases that would lead to higher false discovery rates.

3.1. Plots

In addition to performing normalization, it is often useful to view the transcript intensities as they might immediately reveal various biases. For two-color arrays, a common practice is to plot the relationship between the red and green intensities within each array. (For single-color arrays, the plot can be constructed for a pair of arrays.) Yang et al. (13) proposed to plot the log-intensity ratio

$$M = \log X_r - \log X_g$$

against the average log intensity

$$A = \frac{1}{2}(\log X_r + \log X_g)$$

Assuming that the great majority of genes have similar intensities under the two conditions corresponding to the colors, the cloud of points in the resulting MA-plot should be concentrated along the x-axis; see Fig. 1. Deviations from this ideal indicate, for example, dye and intensity-dependent biases.

3.2. Global Normalization

The simplest method for equalizing the global intensity of different arrays is the *global (mean or median) normalization*. Several versions have been proposed in the literature both for two-color (13–16) and one-color arrays, and the method is applied in the Affymetrix MAS5 algorithm (8). The method is applied array by array separately. The basic idea is to adjust the array intensities to

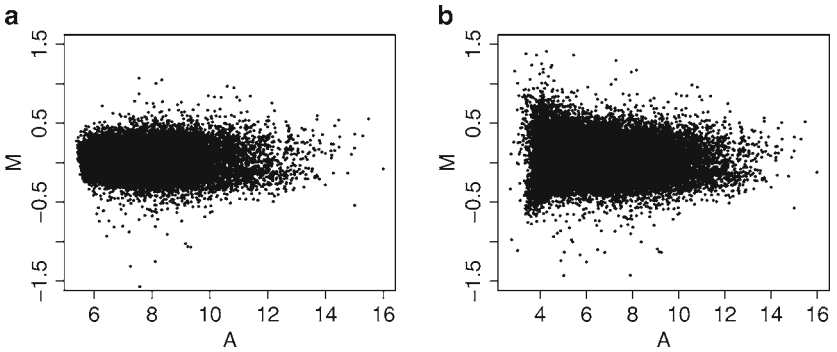


Fig. 1. MA-plot of a mouse microarray data from Agilent. (**a**) No background subtraction; (**b**) With background subtraction. Note the increased variability after background subtraction.

have mean (or median) equal to some arbitrary target value (e.g., 500 in MAS5). If we call y_{ig} the probe intensity for gene g in the array i, then the normalized value y^*_{ig} will be computed as

$$y^*_{ig} = \frac{y_{ig}}{\overline{y}_i} T$$

where $\overline{y}_i$ is the mean intensity of array i and T the target value.

This approach has the advantage of being easy to understand and simple to compute. The main drawback is that the assumption that intensity variation among arrays can be captured by multiplicative shift might be too simplistic. From graphical inspection (see Fig. 2), an intensity-dependent variation is often visible. This might be partially due to the fact that, during scanning, intensity values are constrained between 0 and $2^k - 1$, where k is the image resolution in bit (usually 16 bit). At the boundaries, the variation might not follow a linear trend. Similarly, a spatial trend might exist, especially for cDNA slides. All these effects would still persist after global normalization.

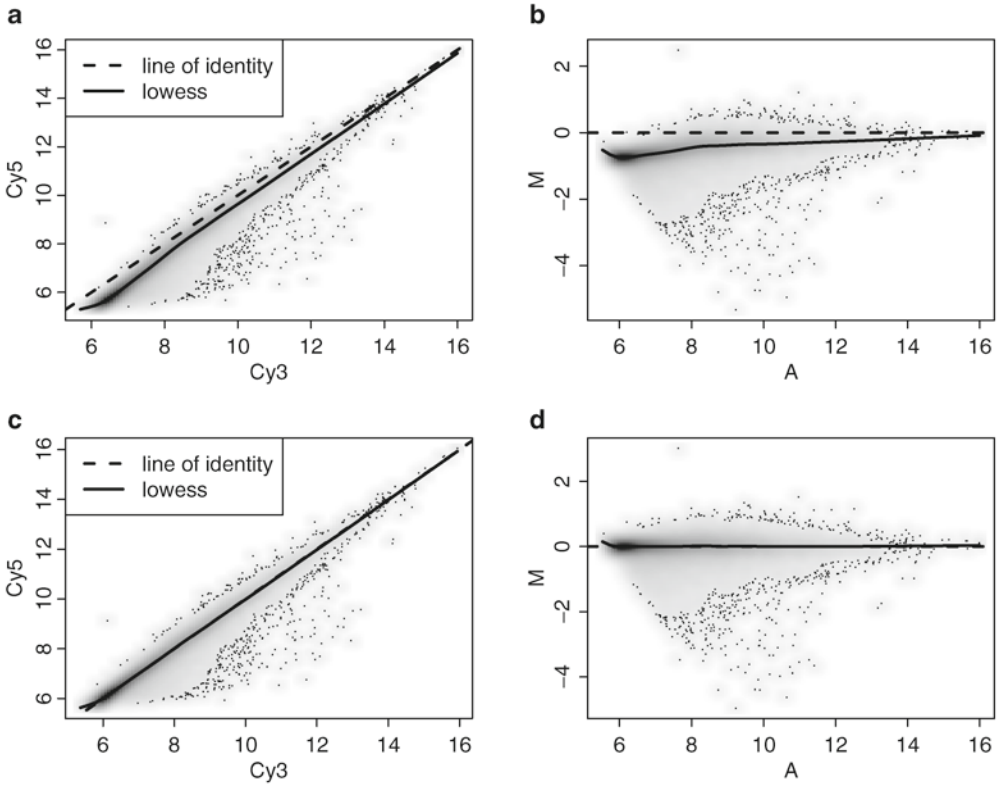


Fig. 2. Lowess normalization of Agilent array. (**a**) Scatter plot of Cy5 versus Cy3 for raw data (no background correction). (**b**) MA-plot for raw data. (**c**) Scatter plot after lowess normalization. (**d**) MA plot after lowess normalization.

3.3. Lowess Normalization

For two-color arrays, Yang et al. (17) proposed an intensity-dependent normalization procedure based on *lowess smoothing* of the MA-plot. Lowess smoothing, also known as locally weighted regression (18), is a technique for smoothing scatterplots, where a nonlinear function of a predictor variable is fitted to a continuous outcome variable using robust weighted least squares. Let $\hat{M}_g$ be the smoothed value from the MA-plot; then the normalized log ratio value is $M_g - \hat{M}_g$.

For single-color platforms, the lowess normalization deals with pairs of arrays that are normalized relative to each other. The procedure cycles through all pairwise combinations of arrays until convergence. The main drawback is that it is computationally intensive, especially for a large number of arrays, so it is rarely used.

3.4. Quantile Normalization

The idea behind the global normalization is that arrays measuring the same (large collection of) genes should deliver similar averages. It is clear, however, that simply equalizing the center of distribution of measured intensities, and likewise possibly the scale, might not be sufficient as the whole distribution may vary; see Fig. 3. Several authors (19, 20) suggested the *quantile*

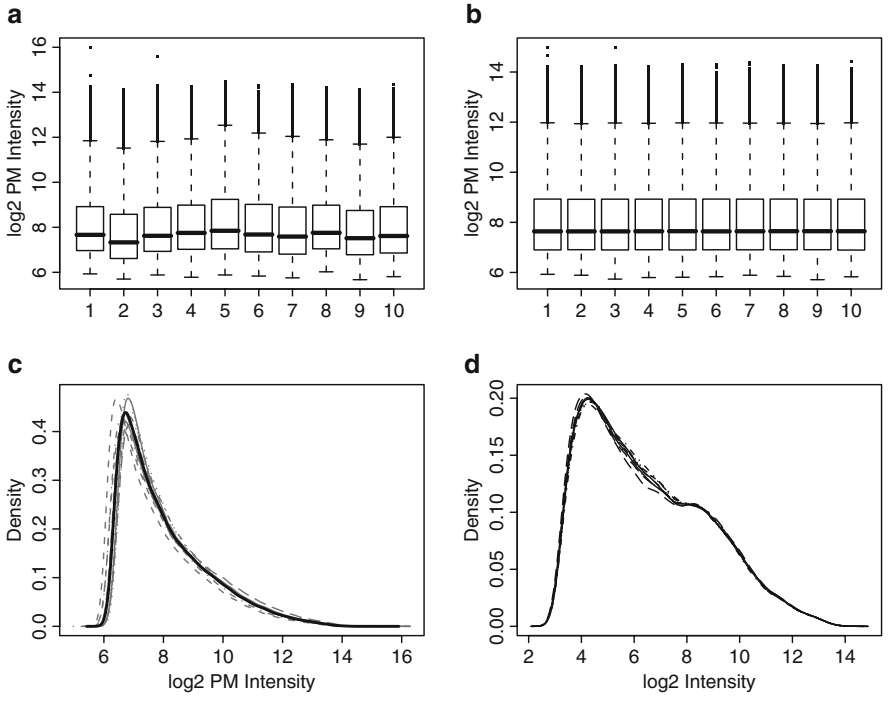


Fig. 3. (**a**) Boxplots of unnormalized PM intensities. (**b**) Boxplots of PM intensities normalized using quantile normalization (no background correction), (**c**) Within-array probe intensity densities for unnormalized data (*gray lines*) and after normalization (*bold black line*). (**d**) Within-array probeset intensity densities after background correction, normalization and summarization with RMA.

normalization, whose goal is to impose to each array the same empirical distribution of intensities. The distribution of within-gene averages is usually used as the target or the reference.

Mathematically, the procedure applies a transformation $F^{-1}(G_i(y))$, where G_i is the cumulative distribution of intensities in the array i, and F is the reference distribution. The algorithm itself is very simple; intensities in each array are first ranked in increasing order. Each quantile value in then substituted by the corresponding quantile in the reference distribution. Finally, values are brought back to the original order. Using only the observation ranks, the algorithm is able to deal with a nonlinear trend, and runs quite fast. Where several replicates of the same gene intensities are available (e.g., Illumina and Affymetrix), the algorithm is usually run before summarization, thus exploiting more information and possibly with a better estimation of the real underlying distribution of gene intensities.

3.5. Housekeeping-Gene Normalization

Most commonly-used normalization procedures use the whole set of genes, under the assumption that the great majority of genes are fairly invariant across arrays. Nevertheless, this assumption is often questionable, especially in experiments where a large variation in expression profiles is expected. To overcome this problem, the housekeeping-gene approach borrows the idea from standard laboratory procedures (e.g., Northern blot or quantitative RT-PCR), where an internal control is used for data normalization. It assumes that some (not all) genes are similarly expressed across arrays, so that they can be used as a reference for the relative expression levels of other genes. For example, Affymetrix platforms include a set of control probes of housekeeping genes (e.g., β-Actin, GAPDH and others).

However, there is a serious concern about the assumption of invariant expression of the so-called housekeeping genes as they are often affected by various factors that are not controlled in the experiment. Also, those genes are usually highly expressed, thus not representing genes of low intensities. Furthermore, they are usually a very small subset of the whole array chip, so fluctuations in their intensities are highly affected by random or systematic errors. Any normalization based on such a limited number of internal references would be unreliable. Therefore, normalization based on housekeeping genes selected a priori is not recommended.

A possible variation of the same framework is to use spiked-in control spots with genetic material from unrelated species. Again several problems arise with such an approach. First, spike-ins are added into the sample at a different stage of cDNA preparation, so that intensity levels of spike-ins are subject to less experimental variation than the naturally expressed transcripts of comparable abundance. Second, nonspecific hybridization cannot be excluded, though might be reduced with careful probe design. Finally, a

relatively large number of control spots, with a broad spectrum of abundance, would be needed, making the whole process highly elaborate.

3.6. Invariant Set Normalization

To overcome the drawbacks of a prespecified set of housekeeping genes as a reference for normalization, a data-driven procedure to select invariant genes has been proposed (21). Probes related to genes that are not differentially expressed among two or more biological conditions are expected to have similar intensity ranks. An iterative procedure is used to select the so-called *invariant set* of probes. First, the algorithm selects a reference array, for example, a mean or median array, or a pseudomean array (i.e., an array whose probe expressions are computed as the gene-wise averages). Then, each probe intensity is ranked within each array and compared with the corresponding value in the reference array. If the rank difference, divided by the total number of probes on the array, is smaller than a threshold, then the probe is selected for the invariant set and excluded from the whole list. The ranking and selection are then repeated on the reduced list. The iteration stops when the number of invariant probes in the reduced list is small enough. The resulting invariant set of genes is then used for an intensity-dependent normalization based on a lowess or smoothing spline.

One advantage of the rank-invariant method is that it does not require a symmetry in number of up- and downregulated genes. However, since the number of the rank-invariant genes selected by the algorithm is usually quite small (of the order of a few hundred to one thousand), they may not cover the entire range of intensity values of all genes on the array. This might lead to some instability in the normalization procedure (22).

3.7. LVS Normalization

A data-driven procedure for identifying a set of genes that are the least variant across samples, and therefore might be a good reference set for normalization, is the basis of an algorithm proposed by Calza et al. (23). At present, the algorithm is implemented for Affymetrix arrays only. The LVS (least variant set) algorithm follows the same idea of the invariant-set procedure, but instead of using pairwise comparison between arrays, it exploits the total information from all the arrays. The information is extracted from the probe-level data by partitioning the observed variability of probes intensities into array-to-array variation, within-probeset variation, and residual variation. Probesets whose array-to-array variability is below a given threshold provide the reference set for normalization.

The identification of the reference genes works via fitting the following linear model for the *unnormalized* PM values:

$$\log_2(\mathrm{PM}_{ij}) = \mu + \alpha_i + \beta_j + \varepsilon_{ij}.$$

The model is fitted by a robust estimation method (24). The array-to-array variability is captured by the χ^2 test statistic given by

$$\chi^2 = \hat{\alpha}' V^{-1} \hat{\alpha}$$

where $\hat{\alpha}$ is the vector of estimated α_i's, and V is its estimated covariance matrix. A quantile regression (25) is then fitted to χ^2 values as a function of the residual standard deviation. A parameter has to be chosen, namely the proportion of genes to be considered as reference (a good compromise in a general experimental setting is 40%). Genes below the values fitted by the quantile regression model are considered as the LVS genes.

Once the LVS genes are identified, the normalization algorithm works on the individual arrays by fitting a spline smoother between the arrays and an arbitrary reference array. The latter is, for example, a pseudomedian array or any user-specified array. The curve fitted through the least variant genes is then used to map intensities of all the genes in each array to be normalized.

4. Data Examples

For the purpose of illustration, we use three datasets recently produced on three different platforms (Agilent, Affymetrix, and Illumina), using the same biological material (1). The biological samples come from the hippocampus of five wild-type mice and five transgenic mice overexpressing DCLK short. These data are available from Genome Expression Omnibus (GEO) with series number GSE8349 and have been bundled in the **R** package `iNorm` (http://www.biostatistics.it/software/iNorm_1.0.0.tar.gz).

4.1. Software

All the examples described in the following sections are implemented using the **R** and Bioconductor platforms.

1. For Agilent data, as more generally for two-colors platforms, both `marray` and `limma` packages provide the basic functions for data processing. In this tutorial, we show how to use `limma` functions.
2. Affymetrix data preprocessing is covered in the `affy` package, while LVS normalization routines are provided by the `FLUSH.LVS.bundle` library.
3. Illumina BeadChip data can be processed using the set of functions provided by the beadarray package.
4. The general package `Biobase` as well as some more packages might be needed for dependencies issues (a complete list is available in the directory `doc` of the `iNorm` package).

All packages are free to download from the Bioconductor website (http://www.bioconductor.org). All the R commands we give below will run after you install all these packages, including the `iNorm` package for the data.

4.2. Agilent Platform

Agilent is a two-color platform, so usually two samples are hybridized to the same array on the red and green channels. (Sometimes the platform is used on a single-channel basis, and samples are hybridized to the red channel only.) The data presented here come from the Whole Mouse Genome G4122A chip (1), which contains 41,534 60-mer oligonucleotide probes representing over 41,000 mouse genes and transcripts. Every transgenic mouse was cohybridized with a wild-type one, based on a dye-swap design.

As for other platforms, Agilent chip preprocessing works through background correction and normalization steps; see Subheading 2.2. Figure 1 shows an MA-plot for the example data before and after the ordinary background subtraction. Note the increased variability after correction; this will lead to increased false-positive and false-negative rates in the detection of differential expression. Therefore, some authors and Agilent itself suggest avoiding any background correction (26).

Normalization in two-color platforms is usually performed array by array, aimed at removing any intensity-dependent dye and spatial effect within each array. Lowess normalization is the most commonly used, based on smoothing of M values as a function of the A values. In Fig. 2a, b, we can see that dyes have different efficiencies, where we expect on average the log ratio Red/Green to be zero (in the MA-plot).

Currently, the Agilent Feature Extraction software uses lowess and a custom error model to extract measurements. No real advantage seems to be derived from the complex error model though, as it probably captures mostly the variation introduced by the preprocessing steps rather than any real component of probe variability (26).

To perform normalization for two-color Agilent arrays, the functions provided in the **R** library limma can be used. First we load the library, and then read the data using the following R commands:

```
> library(limma)

> file.target <-system.file("data","Agilent_Targets.txt",

                package="iNorm")

> file.dir <- system.file("data","Agilent",package="iNorm")

> targets <- readTargets(file.target,row.names="Name")

> pedotti.AGL <- read.maimages(targets,

                 source="agilent",path=file.dir).
```

The first two commands simply retrieve the position of the *target* file and the directory where the Agilent files are located. To import the data, we use the function `read.maimages`. This function requires, as first argument, either a character vector listing all file names or a matrix (target) with column "`FileName`", specifying the names of files to be read. This usually also contains two column names "Cy3" and "Cy5" with information on samples hybridization, and possibly any other clinical information of interest. This file is read into R with the `readTargets` function. Finally, the source argument specifies that we are importing Agilent output file located in the file.dir directory in the system. For more details, see the help files by typing `?readTargets` and `?read.maimages`.

Then, we normalize within-array using `"loess"` (a more recent version of `"lowess"`) method with no background correction:

```
> pedotti.nobg.loess <- normalizeWithinArrays
                        (pedotti.AGL,method="loess",
                        bc.method="none").
```

Figure 2c, d show a sample array after normalization.

4.3. Affymetrix Platform

Affymetrix data were produced based on GeneChip Mouse Genome 430 2.0 Array, allowing for the measurement of 45,101 features. Procedures for reading and processing Affymetrix data are implemented in the affy package. The first step is to read CEL files from the scanner into R using the function ReadAffy. The easiest way is to simply supply the path to the directory containing the CEL files. The necessary R commands are

```
> library(affy)
> path2files <- system.file("data","CEL",package="iNorm")
> pedotti.AffyBatch <- ReadAffy(celfile.path=path2files).
```

The first command after `library()` defines the directory where the example CEL files are located, while the second line performs the actual reading. The result is an object with a specific S4 class (`AffyBatch`). Simply typing the object name will output some summary information on it.

Preprocessing and normalizing data according to the Affymetrix MAS5.0 algorithm can be done in a single step using the following code (note that the output data will not be log-transformed):

```
> pedotti.mas5 <- mas5(pedotti.AffyBatch)
```

One of the most commonly used procedures is the RMA, which performs background correction (Subheading 2.2), quantile normalization at probe level, and a robust multiarray summarization.

It works exclusively on the PM data and completely ignores the MM probes. The following command will return data preprocessed with the RMA algorithm (already in log2 scale):

```
> pedotti.rma <- rma(pedotti.AffyBatch)
```

Figure 3a–c show probe-intensity boxplots and density plots before and after quantile normalization, while Fig. 3d shows the densities for background corrected, normalized, and summarized (with RMA) data.

When a large number of arrays needs to be processed, there might be some memory issues. In this case, to reduce memory requirements, we might consider using the special function justRMA that performs RMA normalization while reading data:

```
> pedotti.rma <- justRMA(celfile.path=path2files)
```

The LVS algorithm (Subheading 7) is based on fitting a quantile regression on a scatter plot of the between-array variation versus the residual standard deviation from a probe-level robust linear model (Fig. 4). Points below a given threshold (gray points in the figure) are used as the reference for normalization. The current implementation of LVS allows one to normalize data *after* summarization, giving the user the choice of background correction and summarization procedures. For example, to perform LVS normalization using MAS5 preprocessing steps, the following commands are used:

```
> library(FLUSH.LVS.bundle)
> pedotti.lvs <- lvs(pedotti.AffyBatch,
                     bgcorrect.method = "mas",
                     pmcorrect.method = "mas",
                     summary.method = "mas"),
```

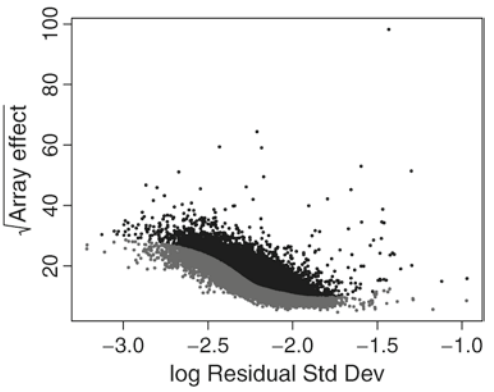


Fig. 4. Scatter plot of the between-array variation vs. the residual standard deviation from a probe-level robust linear model. The *gray points* are used for LVS normalization.

while to use RMA background correction and summarization, we use:

```
> pedotti.lvs2 <- lvs(pedotti.AffyBatch,
                      bgcorrect.method = "rma",
                      pmcorrect.method = "pmonly",
                      summary.method = "medianpolish")
```

4.4. Illumina Platform

The Illumina chip used in the experiment was the Sentrix Mouse-6 Expression BeadChip, containing 46,120 probes. The main package for processing the data is `beadarray`, which has functions for reading, normalizing, and plotting. Data from Illumina BeadChip are available in two different formats. The raw TIFF images and text files output by the BeadScan software are referred to as *bead-level data*. The second format is the output from Illumina's BeadStudio software, which performs a first set of preprocessing, like sharpening and summarization. This output is usually defined as *bead-summary data*. The example here will deal only with the summary data. The necessary commands are:

```
> library(beadarray)
> path2file <- system.file("data","Illumina",
                "illumina_raw_data.csv",
                package="iNorm")
> pedotti.eset <- readBeadSummaryData(path2file,
                   ProbeID="TargetID", sep=",",
                   columns = list(exprs = "AVG_Signal",
                   se.exprs="BEAD_STDEV",
                   NoBeads = "Avg_NBEADS",
                   Detection="Detection"))
```

In this example, the dataset, which is a summary data output from BeadStudio, is provided in a single text file. The function `readBeadSummaryData` requires, as arguments, the path to the data file (here provided by path2file) and target column names (but usually default ones are fine). The object holding the intensity values has a specific S4 class (`ExpressionSetIllumina`), which is an extension of that used to store Affymetrix expression data. This object class allows one to use many already available functions.

Boxplots of intensity levels are a good tool for quality assessment. Given the random nature of the number of beads probing each transcript on each array, we can produce a boxplot for the distribution of beads counts. In a normal situation, we expect

fairly similar distribution center approximately around 40. The R commands to produce the boxplots (Fig. 5) are:

```
> par(mfrow = c(1, 2))
> boxplot(as.data.frame(log2(exprs(pedotti.eset))),
          cex=.3,pch=16,las = 2, ylab = "log2(intensity)")
>
boxplot(as.data.frame(NoBeads(pedotti.eset)),cex=.3,pch=16,
         las = 2, ylab = "number of beads")
```

The commonly suggested normalization procedure for Illumina data is the quantile normalization. The MA-plot of a sample pair of arrays before and after normalization is given in Fig. 6. To perform the normalization, we use the following command:

```
> pedotti.ilm <- normaliseIllumina(pedotti.eset,
                    method="quantile",
                    transform="log2")
```

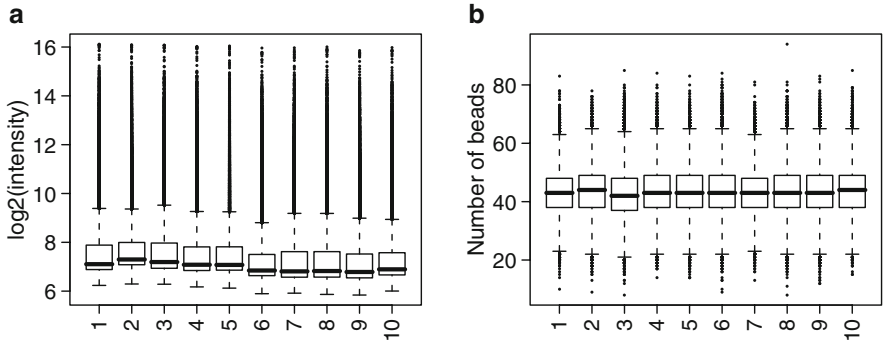


Fig. 5. (**a**) Boxplots of log2 intensities. As expected in BeadChip data there is a limited inter-array variation. (**b**) Boxplots of beads counts within each array. As expected counts have fairly similar distribution centered approximately around 40.

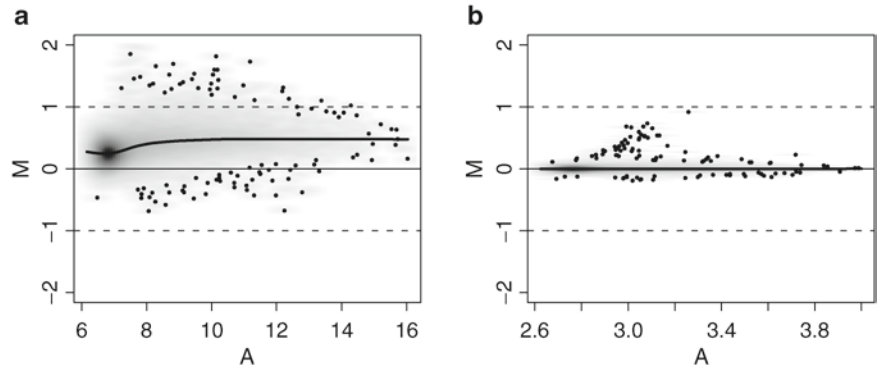


Fig. 6. MA-plot of two arrays (1510547074 A and 1510547074 F). (**a**) Raw data. (**b**) After quantile normalization.

5. Summary

Measured gene expressions from microarrays contain various technical noises that need to be removed before we can perform meaningful analyses of the data. We summarize here general recommendations that apply to all platforms:

- Since different normalization procedures can lead to different final results, it is important to know which procedure has been used to produce the analyzable dataset. This is not always easy to do, since different platforms need different methodology, but at least, there should be some indication that the normalization method used has been investigated carefully for the specific platform.
- All normalization procedures rely on rather strong assumptions, e.g., the great majority of genes are unaltered among arrays. Users need to assess whether these assumptions are sensible in their specific application. For example, microRNA or custom-made arrays, such as human cancer chips, are likely to violate these assumptions, so they require extra investigation.
- Simple-minded background corrections often introduce more noise and negative values, so it is not generally recommended. Sometimes, no background correction is preferable.
- Housekeeping-gene normalization based on an a priori set of small number of genes (e.g., 100) can produce poor normalization and is not recommended.

References

1. Pedotti, P., 't Hoen, P. A. C., Vreugdenhil, E., Schenk, G. J., Vossen, R. H., Ariyurek, Y., de Hollander, M., Kuiper, R., van Ommen, G. J. B., den Dunnen, J. T., Boer, J. M., and de Menezes, R. X. (2008) Can subtle changes in gene expression be consistently detected with different microarray platforms? *BMC Genomics* 9, 124.
2. Lipshutz, R. J., Fodor, S. P., Gingeras, T. R., and Lockhart, D. J. (1999) High density synthetic oligonucleotide arrays. *Nat Genet* 21 (1 Suppl), 20–24.
3. Hughes, T. R., Mao, M., Jones, A. R., Burchard, J., Marton, M. J., Shannon, K. W., Lefkowitz, S. M., Ziman, M., Schelter, J. M., Meyer, M. R., Kobayashi, S., Davis, C., Dai, H., He, Y. D., Stephaniants, S. B., Cavet, G., Walker, W. L., West, A., Coffey, E., Shoemaker, D. D., Stoughton, R., Blanchard, A. P., Friend, S. H., and Linsley, P. S. (2001) Expression profiling using microarrays fabricated by an ink-jet oligonucleotide synthesizer. *Nat Biotechnol* 19, 342–347.
4. Oliphant, A., Barker, D. L., Stuelpnagel, J. R., and Chee, M. S. (2002) BeadArray technology: enabling an accurate, cost-effective approach to high-throughput genotyping. *Biotechniques* 32(Suppl), 56–58, 60–61.
5. R Development Core Team (2008) *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.
6. Gentleman, R. C., Carey, V. J., Bates, D. M., Bolstad, B., Dettling, M., Dudoit, S., Ellis, B., Gautier, L., Ge, Y., Gentry, J., Hornik, K., Hothorn, T., Huber, W., Iacus, S., Irizarry, R., Li, F. L. C., Maechler, M., Rossini, A. J., Sawitzki, G., Smith, C., Smyth, G., Tierney,

L., Yang, J. Y. H., and Zhang, J. (2004) Bioconductor: open software development for computational biology and bioinformatics. *Genome Biol* 5, R80.

7. Ritchie, M. E., Silver, J., Oshlack, A., Holmes, M., Diyagama, D., Holloway, A., and Smyth, G. K. (2007) A comparison of background correction methods for two-colour microarrays. *Bioinformatics* 23, 2700–2707.
8. Affymetrix (2002) *Statistical Algorithms Description Document.*
9. Naef, F., Lim, D. A., Patil, N., and Magnasco, M. (2002) DNA hybridization to mismatched templates: a chip study. *Phys Rev E Stat Nonlin Soft Matter Phys* 65 (4 Pt 1), 040902.
10. Affymetrix (2001) *Affymetrix microarray suite users guide, Version 5.0.* Santa Clara, CA.
11. Irizarry, R. A., Hobbs, B., Collin, F., Beazer-Barclay, Y. D., Antonellis, K. J., Scherf, U., and Speed, T. P. (2003) Exploration, normalization, and summaries of high density oligonucleotide array probe level data. *Biostatistics* 4, 249–264.
12. Irizarry, R. A., Bolstad, B. M., Collin, F., Cope, L. M., Hobbs, B., and Speed, T. P. (2003) Summaries of Affymetrix GeneChip probe level data. *Nucleic Acids Res* 31, e15.
13. Yang, Y. H., Dudoit, S., Luu, P., and Speed, T. P. (2002) Normalization of cDNA Microarray Data.
14. Kerr, M. K. and Churchill, G. A. (2001) Experimental design for gene expression microarrays. *Biostatistics* 2, 183–201.
15. Zien, A., Aigner, T., Zimmer, R., and Lengauer, T. (2001) Centralization: a new method for the normalization of gene expression data. *Bioinformatics* 17(Suppl 1), S323–S331.
16. Kroll, T. C. and Wolfl, S. (2002) Ranking: a closer look on globalisation methods for normalisation of gene expression arrays. *Nucleic Acids Res* 30(11), e50.
17. Yang, Y. H., Dudoit, S., Luu, P., Lin, D. M., Peng, V., Ngai, J., and Speed, T. P. (2002) Normalization for cDNA microarray data: a robust composite method addressing single and multiple slide systematic variation. *Nucleic Acids Res* 30(4), e15.
18. Cleveland, W. S. (1979) Robust locally weighted regression and smoothing scatterplots. *J Am Stat Assoc* 74, 829–836.
19. Bolstad, B. M., Irizarry, R. A., Astrand, M., and Speed, T. P. (2003) A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. *Bioinformatics* 19(2), 185–193.
20. Wernisch, L., Kendall, S. L., Soneji, S., Wietzorrek, A., Parish, T., Hinds, J., Butcher, P. D., and Stoker, N. G. (2003) Analysis of whole-genome microarray replicates using mixed models. *Bioinformatics* 19(1), 53–61.
21. Li, C. and Wong, W. H. (2001) Model-based analysis of oligonucleotide arrays: model validation, design issues and standard error application. *Genome Biol* 2(8), 32.
22. Chua, S. W., Vijayakumar, P., Nissom, P. M., Yam, C. Y., Wong, V. V., and Yang, H. (2006) A novel normalization method for effective removal of systematic variation in microarray data. *Nucleic Acids Res* 34, e38.
23. Calza, S., Valentini, D., and Pawitan, Y. (2008) Normalization of oligonucleotide arrays based on the least-variant set of genes. *BMC Bioinformatics* 9(1), 140.
24. Huber, P. (1981) *Robust statistics.* John Wiley & Sons, Inc., New York.
25. Koenker, R. and Bassett, G. (1978) Regression quantiles. *Econometrica* 46, 33–50.
26. Zahurak, M., Parmigiani, G., Yu, W., Scharpf, R. B., Berman, D., Schaeffer, E., Shabbeer, S., and Cope, L. (2007) Pre-processing Agilent microarray data. *BMC Bioinformatics* 8, 142.

Chapter 4

Prediction of Transmembrane Topology and Signal Peptide Given a Protein's Amino Acid Sequence

Lukas Käll

Abstract

Here, we describe transmembrane topology and signal peptide predictors and highlight their advantages and shortcomings. We also discuss the relation between these two types of prediction.

Key words: Membrane protein, signal peptide, transmembrane topology, prediction, bioinformatics

1. Introduction

Transmembrane (TM) proteins make up about a fifth of all protein sequences known, yet there are less than 200 structures of membrane protein available, making up only a small fraction of all crystal structures available. This discrepancy is due to TM proteins being hard to over express and crystallize, and therefore difficult to examine with X-ray diffraction or NMR. Even though significant progress has been made in the last couple of years, our overall knowledge of membrane proteins lags far behind our knowledge of soluble proteins. This is despite a commercial interest in membrane proteins as they are appealing drug targets.

Here, we discuss different available methods to predict properties of membrane proteins and signal peptides, given their amino acid sequence (Table 1).

2. TM Topology Prediction

Instead of determining full structural information, one may instead determine the TM topology. That is localizing all TM segments

David Fenyö (ed.), *Computational Biology*, Methods in Molecular Biology, vol. 673,
DOI 10.1007/978-1-60761-842-3_4,

Table 1
Some publicly available transmembrane topology and signal peptide predictors and their URLs

Name	TM	SP	Homologs	URLs
MEMSAT3 (26)	X	X	X	http://bioinf.cs.ucl.ac.uk/psipred/psiform.html
SPOCTOPUS (46)	X	X	X	http://octopus.cbr.su.se
PHOBIUS (20)	X	X	X	http://phobius.cbr.su.se
PHILIUS (21)	X	X	–	http://www.yeastrc.org/philius
HMMTOP (16)	X	–	X	http://www.enzim.hu/hmmtop/
SIGNALP (40)	–	X	–	http://www.cbs.dtu.dk/services/SignalP

as well as determining as to which subcellular compartment the loops between the TM segments are exposed. We can determine TM topology by experimental means, using fused-reporter genes (1–3), glycosylation sites (4), or mass spectrometry (5). Significantly easier, and maybe as accurate (6), is to predict TM topology in silico from a protein's amino acid sequence.

TM topology prediction is one of the "classical" domains of bioinformatics, ranging back as far as to the eighties. The first TM helix prediction methods were based on theoretically or experimentally determined hydrophobicity indexes. Each amino acid was given a score based on its preference to water or lipids. For the examined protein, a hydrophobicity plot was calculated by summing the hydrophobicity indexes over a window of a fixed length. A heuristically determined cutoff value was then used to indicate possible TM segments (7, 8). An important improvement to this strategy came from the observation that positively charged amino acids (arginine and lysine) are over-represented near the TM helices, on the originating side loops of TM proteins (the positive inside rule) (9). This gives an indication about the orientation of the helices and leads to the development of the first automated full TM topology prediction method TOPPRED (10).

TOPPRED first scans a query sequence for certain and putative TM segments and then selects the putative segments that maximize the difference in charged amino acids in loops, summed over each side separately. Instead of only using a hydrophobicity index, some methods use a combination of this and indexes for amino acids known to be frequent near the end of membrane helix ends, e.g., SOSUI (11). Other methods are letting an artificial neural network (ANN), e.g., PHDHTM (12) or a support vector machine, e.g., SVMTOP (13), to detect potential TM segments.

A more integrated approach could be taken to the problem. Instead of first scanning the sequence for TM segments and sorting out the topology as a second step, the search for TM segments can be integrated with the evaluation of possible topologies in one step. The amino acid distribution of the investigated sequence is compared with precalculated amino acid distributions in each type of topologically distinct region (TM helices, originating side loops, and translocated side loops) of training sets of TM proteins. Given correlation measurements between the amino acid distributions of the examined protein and the expected amino acid distributions in different topological regions, the most likely topology can be predicted. A nice feature of this approach is the ability to model all parts of the protein so that all topogenic signals are properly weighted, which is preferable to giving priority to the hydrophobic signal. This was first done by a dynamic programing algorithm in the method MEMSAT (14). The parameters of MEMSAT were estimated by expectation maximization (15), so the method is highly related to subsequent statistical models.

Pure statistical approaches to the problem have ensued MEMSAT. Some popular Hidden Markov model-based predictors are HMMTOP (16, 17) and TMHMM (18, 19) and its sequel PHOBIUS (20). Recently, a method PHILIUS (21) made use of dynamic Bayesian networks (DBNs), an extension of HMMs enabling the inclusion of more complex relationships within the topology model. Much grace to the DBN, PHILIUS also outputs easily interpreted reliability figures to its predictions.

A recent and quite important step toward understanding the mechanism governing the insertion of membrane proteins was the development of the Hessa scale (22). The authors developed a model system making it possible to measure the propensity for the insertion of different systematically engineered hydrophobic amino acid stretches into the membrane. They managed to show that the probability of a potential membrane segments to insert is proportional to the difference in free energy between being and not being inserted. Furthermore, they demonstrated that if we ignore addition term stemming from helix amphipathicity and helix length terms, the free energy roughly is a linear combination of the free energy contribution of the different amino acids, given their depth in the membrane. This hypothesis gives support for statistical predictors such as HMM- or DBN-based predictors (21). However, the Hessa scales triggered the development of SCAMPI (23) that makes use of the scales determined by the Hessa et al. experiments, rather than properties of the training sets as for statistical methods.

2.1. Predictions Supported by Homologs

A common way to improve the performance of a predictor is to not only look at the examined sequence, but instead find homologs using homolog retrieval tools like BLAST (24), and then predicting

the topology of all the sequences simultaneously. A general performance increase is observed with this approach, as that topology is likely to be conserved within a family, and one can get a clearer view of the topology of a protein by integrating the topogenic features of the different family members. There are several available methods making use of this observation. We can divide the methods using information from homologs into three different groups.

First, we have profile-based predictors that take a sequence profile from BLAST and match the obtained amino acid distribution of each position against the expected amino acid distribution of the model using the positional information from the query protein. Examples of such predictors are TMAP (25), PHDHTM (12), MEMSAT3 (26), and OCTOPUS (27).

A second way to incorporate information from homologs is implemented in HMMTOP (16) that implements an adaptive training procedure to retrain its HMM on all the homologs, before finally decoding the query sequence.

A third way is to align the homologs with a multiple sequence alignment method, score each sequence separately, and then superimpose the scores for each position of the alignment. The advantage of this strategy is that we can make use of the positional information of all the examined sequence rather than just the positional information of the query protein. POLYPHOBIUS (28) is an example of this kind of prediction method.

2.2. Constrained Prediction

Lately, an interesting type of bioinformatics and experimental hybrid technique has been used to determine the topology of large sets of *Escherichia coli* TM proteins (29). By fusing a set of inner membrane proteins with LacZ and GFP, their C-terminus can be located as cytoplasmic or periplasmic. This piece of topogenic signal was used as an input to a constrained prediction by TMHMM (30). Full topological models of 601 *E. coli* (31) and 546 *Saccharomyces cerevisiae* (32) TM proteins were proposed.

2.3. Prediction of Irregular Structures

Recently elucidated 3D structures suggest that irregularities, such as reentrant membrane segments (membrane segments beginning and ending on the same side of the membrane), coiled TM segments, broken α-helices, and membrane segments displaced from their hydrophobic core, are not unexpected oddities but rather commonly occurring features. There is also support for structure irregularities being more frequent in membrane proteins carrying out more complex tasks such as transporters (33). However, currently very few methods are capable of predicting such anomalies.

One recently published method, OCTOPUS (27), models reentrant loops, with at least partial success. Information can also

be gained by using the ZPRED predictor (34, 35) that predicts the Z-coordinate, i.e., the coordinate orthogonal to the membrane layer. The authors demonstrated a capability to detect reentrant loops and broken α-helices.

In many bacteria and in the mitochondria, we do not only have α-helical membrane proteins but also have the so-called β-barrel TM proteins. This class of proteins is hard to predict with classical TM prediction methods, since their TM segments generally are shorter and with a different amino acid composition than α-helical TM segments. There are dedicated predictors available for these kinds of proteins, B2TMR-HMM (36) and BBF (37).

2.4. Prediction of Signal Peptides

When trying to determine the function of a protein, an important question to answer is where in the cell it is located. The location of a protein governs what other types of proteins or other molecules it will be able to interact with. A first step in this process is to determine if it has a signal peptide (SP) or not, since that will tell us if it is a cytosolic protein or not. An SP is a short N-terminal peptide, cleaved off during the export of the protein. It is also valuable to know where the mature protein starts, so there is an interest in localizing the cleavage site of an SP. About 15% of the proteins in the human proteome have an SP (20). As signal peptides and TM segments are imported by the same mechanism – the translocon – it is not surprising that they resemble each other a lot (See Fig. 1).

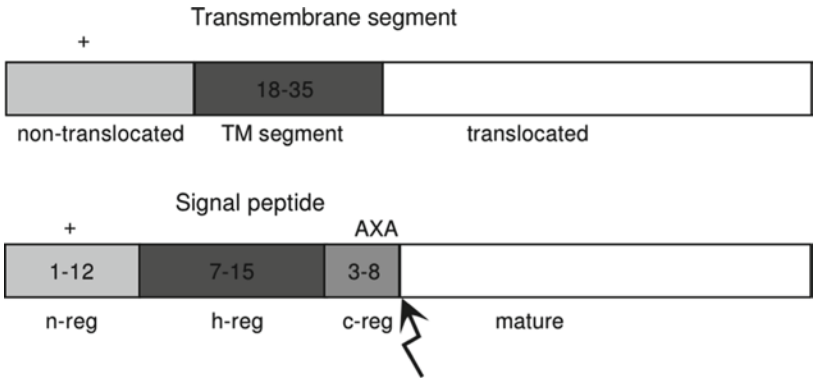


Fig. 1. A comparison of a transmembrane (TM) segment with N-terminus in the cytosol (*above*) and a signal peptide (*below*). Similar to the TM segment, one of the strongest indications of a signal peptide is a hydrophobic α-helical region (the h-region). However, the hydrophobic region is generally shorter for a signal peptide (approximately 7–15 residues) than for a TM helix. The h-region is near the N-terminal of the protein but it is preceded by a slightly positively charged n-region with high variability in length (approximately 1–12 amino acids). Between the h-region and the cleavage site, a somewhat polar and uncharged 3–8 amino acid long c-region is situated. Another clear motif of the SP is the presence of small, neutral residues at the –3 and –1 relative to the cleavage site (48, 49). We often see helix-breaking amino acids, i.e., proline, serine, or glycine, in between the h- and c-regions (48, 49).

Most available SP prediction methods use weight matrices (38), ANNs (e.g., SIGNALP-NN (39, 40)), HMMs, e.g., SIGNALP-HMM (41), or support vector machines (42–44). SIGNALP-NN has trained one ANN for the detection of cleavage site motifs (the C-score), and one ANN to detect the existence of an SP (the S-score). The prediction scores are calculated for each position in the sequence sequentially. Finally, cleavage sites are predicted by regarding a Y-score, a geometrical mean between the C-score of the position and the difference in S-score before and after the position. Existence of an SP is predicted by the value of the average S-score from the start of the sequence till the maximal Y-score (39). An additional criteria is introduced in SIGNALP-NN 3.0 where the average S-score is replaced by a D-score, that is defined as the average of the average S-score and the maximal Y-score (40). The HMMs have, thanks to their ability to model length distributions, the advantage of easily modeling all regions of an SP in a single model. Hence, the prediction of cleavage site is predicted at the same time as the existence of an SP, and we will get one single answer, as to whether an SP is present or not (20, 41).

3. Discriminating TM Helices and Signal Peptides

N-terminal TM helices and SPs tend to confuse predictors. Because they have similar composition, TM topology predictors often classify SPs as TM helices, and SP predictors often classify N-terminal TM helices as SPs, so called *cross prediction*. This occurs frequently. Applying TMHMM and SignalP to a proteomes results in overlaps between 30–65% of all predicted signal peptides and 25–35% of all the predicted TM topologies first TM segment (45). This impairs predictions of 5–10% of the proteome; hence, this is an important issue in protein annotation.

There are a couple of predictors trying to address this situation. First out was PHOBIUS (20), which in essence combines the hidden Markov models of TMHMM and SIGNALP-HMM. The authors demonstrated a drastic reduction in the frequency of cross predictions. Lately, we note three interesting successors: PHILIUS (21), SPOCTOPUS (46), and MEMSAT3 (26), all demonstrating higher accuracy than PHOBIUS in their publications.

Specially for the prediction of presence or absence of signal peptides, it is our recommendation to use any of these methods over methods such as SIGNALP that are not using membrane proteins in their negative training set. These methods have a much higher accuracy on a full proteome scale.

4. Recommendation

So, what method do we recommend you to use for predicting the TM topology or the existence of signal peptides? There is currently a lack of well-conducted benchmarks to guide you in your selection. Most modern methods are dependent on training data. Testing performance on training data will lead to inflated performance figures. Due to a lack of fresh topologies, benchmarked performance is often more indicative of the overlap between the selected benchmarking set and the tested method's training set, than of the actual performance of the benchmarked method.

Not so surprising, modern topology predictors are better than older ones. It is also safe to say that predictions supported by homologs hold higher quality than single-sequence predictions (6, 47). As long as we do not explicitly know that a sequence does not contain a signal peptide, the topology prediction accuracy is also greatly increased, when selecting a combined signal peptide and TM topology predictor (45). Hence, use POLYPHOBIUS (28), SPOCTOPUS (46), or MEMSAT3 (26).

As for the prediction of presence of a signal peptide – select a method that uses membrane proteins in its negative training set, i.e., PHOBIUS (20), PHILIUS (21), or SPOCTOPUS (46). If you are interested in localizing the cleavage site of a sequence already known to have a signal peptide – use SIGNALP 3.0 (40) as it is benchmarked to be accurate on this task.

For all the tasks mentioned above, it frequently pays off to compare predictions from different methods.

5. Future Directions

Finally, we would like to take the opportunity to speculate about interesting future directions and ideas concerning membrane topology predictors.

None of the TM predictors that we listed above make any assumptions about the lipid composition of the surrounding membrane. That is probably not a limitation as long as we are only interested in predicting the topology of membrane proteins inserted in the endoplasmic reticulum by the Sec machinery. However, there are other mechanisms governing the insertion of membrane proteins in other organelles. There is currently no predictors that we are aware of that are dedicated for predicting the topology of membrane proteins in specific organelles. Specially, a predictor of mitochondrial membrane proteins topology would be valuable, as most predictors get their topology wrong. It would

also be useful to be able to predict a membrane protein preference for lipid environment. If we could predict which membrane proteins that have a preference to, i.e., a cholesterol-rich environment, we would get important clues about the function of the protein.

References

1. Ehrmann, M., Boyd, D., and Beckwith, J. (1990) Genetic analysis of membrane protein topology by a sandwich gene fusion approach. *Proceedings of the National Academy of Sciences of the United States of America* **87**, 7574–7578.
2. Manoil, C. and Beckwith, J. (1986) A genetic approach to analyzing membrane protein topology, *Science* **233**, 1403–1408.
3. Feilmeier, B.J., Iseminger, G., Schroeder, D., Webber, H., and Phillips, G.J. (2000) Green fluorescent protein functions as a reporter for protein localization in *Escherichia coli*. *Journal of Bacteriology* **182**, 4068–4076.
4. Hart, G.W., Brew, K., Grant, G.A., Bradshaw, R.A., and Lennarz, W.J. (1979) Primary structural requirements for the enzymatic formation of the N-glycosidic bond in glycoproteins. Studies with natural and synthetic peptides. *The Journal of Biological Chemistry* **254**, 9747–9753.
5. Wu, C.C., MacCoss, M.J., Howell, K.E., and Yates, J.R., III (2003) A method for the comprehensive proteomic analysis of membrane proteins. *Nature Biotechnology* **21**, 532–538.
6. Chen, C.P., Kernytsky, A., and Rost, B. (2002) Transmembrane helix predictions revisited. *Protein Science* **11**, 2774–2791.
7. Argos, P., Rao, J.K., and Hargrave, P.A. (1982) Structural prediction of membrane-bound proteins. *European Journal of Biochemistry* **128**, 565–575.
8. Kyte, J. and Doolittle, R.F. (1982) A simple method for displaying the hydropathic character of a protein. *Journal of Molecular Biology* **157**, 105–132.
9. von Heijne, G. (1986) The distribution of positively charged residues in bacterial inner membrane proteins correlates with the transmembrane topology. *EMBO Journal* **5**, 3021–3027.
10. von Heijne, G. (1992) Membrane protein structure prediction: hydrophobicity analysis and the positive-inside rule. *Journal of Molecular Biology* **225**, 487–494.
11. Mitaku, S., Hirokawa, T., and Tsuji, T. (2002) Amphiphilicity index of polar amino acids as an aid in the characterization of amino acid preference at membrane-water interfaces. *Bioinformatics* **18**, 608–616.
12. Rost, B., Casadio, R., Fariselli, P., and Sander, C. (1995) Transmembrane helices predicted at 95% accuracy. *Protein Science* **4**, 521–533.
13. Lo, A., Chiu, H.S., Sung, T.Y., Lyu, P.C., and Hsu, W.L. (2008) Enhanced membrane protein topology prediction using a hierarchical classification method and a new scoring function. *Journal of Proteome Research* **7**, 487–496.
14. Jones, D.T., Taylor, W.R., and Thornton, J.M. (1994) A model recognition approach to the prediction of all-helical membrane protein structure and topology. *Biochemistry* **33**, 3038–3049.
15. Dempster, A.P., Laird, N.M., and Rubin, D.B. (1977) Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society* **39**, 1–22.
16. Tusnady, G.E. and Simon, I. (1998) Principles governing amino acid composition of integral membrane proteins: application to topology prediction. *Journal of Molecular Biology* **283**, 489–506.
17. Tusnady, G.E. and Simon, I. (2001) The HMMTOP transmembrane topology prediction server. *Bioinformatics* **17**, 849–850.
18. Sonnhammer, E.L., von Heijne, G., and Krogh, A. (1998) A hidden Markov model for predicting transmembrane helices in protein sequences. *Proceedings of the Sixth International Conference on Intelligent Systems for Molecular Biology* **6**, 175–182.
19. Krogh, A., Larsson, B., von Heijne, G., and Sonnhammer, E.L. (2001) Predicting transmembrane protein topology with a hidden Markov model: application to complete genomes. *Journal of Molecular Biology* **305**, 567–580.
20. Käll, L., Krogh, A., and Sonnhammer, E.L.L. (2004) A combined transmembrane topology and signal peptide prediction method. *Journal of Molecular Biology* **338**, 1027–1036.
21. Reynolds, S.M., Käll, L., Riffle, M.E., Bilmes, J.A., and Noble, W.S. (2008) Transmembrane topology and signal peptide prediction using dynamic bayesian networks. *PLoS Computational Biology* **4**, e1000213.

22. Hessa, T., Kim, H., Bihlmaier, K., Lundin, C., Boekel, J., Andersson, H., Nilsson, I., White, S.H., and von Heijne, G. (2005) Recognition of transmembrane helices by the endoplasmic reticulum translocon. *Nature* **433**, 377–381.
23. Bernsel, A., Viklund, H., Falk, J., Lindahl, E., von Heijne, G., and Elofsson, A. (2008) Prediction of membrane-protein topology from first principles. *Proceedings of the National Academy of Sciences of the United States of America* **105**, 7177–7181.
24. Altschul, S.F., Gish, W., Miller, W., Myers, E.W., and Lipman, D.J. (1990) A basic local alignment search tool. *Journal of Molecular Biology* **215**, 403–410.
25. Persson, B. and Argos, P. (1997) Prediction of membrane protein topology utilizing multiple sequence alignments. *Journal of Protein Chemistry* **16**, 453–457.
26. Jones, D.T. (2007) Improving the accuracy of transmembrane protein topology prediction using evolutionary information. *Bioinformatics* **23**, 538–544.
27. Viklund, H. and Elofsson, A. (2008) OCTOPUS: improving topology prediction by two-track ANN-based preference scores and an extended topological grammar. *Bioinformatics* **24**, 1662–1668.
28. Käll, L., Krogh, A., and Sonnhammer, E.L.L. (2005) An HMM posterior decoder for sequence feature prediction that includes homology information. *Bioinformatics* **21** (S1), i251–i257.
29. Drew, D., Sjostrand, D., Nilsson, J., Urbig, T., Chin, C., de Gier, J.W., and von Heijne, G. (2002) Rapid topology mapping of *Escherichia coli* inner-membrane proteins by prediction and PhoA/GFP fusion analysis. *Proceedings of the National Academy of Sciences* of the United States of America **99**, 2690–2695.
30. Melén, K., Krogh, A., and von Heijne, G. (2003) Reliability measures for membrane protein topology prediction algorithms. *Journal of Molecular Biology* **327**, 735–744.
31. Daley, D.O., Rapp, M., Granseth, E., Melen, K., Drew, D., and von Heijne, G. (2005) Global topology analysis of the *Escherichia coli* inner membrane proteome. *Science* **308**, 1321–1323.
32. Kim, H., Melen, K., Osterberg, M., and von Heijne, G. (2006) A global topology map of the *Saccharomyces cerevisiae* membrane proteome. *Proceedings of the National Academy of Sciences* of the United States of America **103**, 11142.
33. Kauko, A., Illergård, K., and Elofsson, A. (2008) Coils in the membrane core are conserved and functionally important. *Journal of Molecular Biology* **380**, 170–180.
34. Granseth, E., Viklund, H., and Elofsson, A. (2006) ZPRED: predicting the distance to the membrane center for residues in alpha-helical membrane proteins. *Bioinformatics* **22**, e191–e196.
35. Papaloukas, C., Granseth, E., Viklund, H., and Elofsson, A. (2008) Estimating the length of transmembrane helices using Z-coordinate predictions. *Protein Science* **17**, 271–278.
36. Martelli, P.L., Fariselli, P., Krogh, A., and Casadio, R. (2002) A sequence-profile-based HMM for predicting and discriminating beta barrel membrane proteins. *Bioinformatics* **18** Suppl 1, S46–S53.
37. Zhai, Y. and Saier, M.H., Jr. (2002) The beta-barrel finder (BBF) program, allowing identification of outer membrane beta-barrel proteins encoded within prokaryotic genomes. *Protein Science* **11**, 2196–2207.
38. von Heijne, G. (1986) A new method for predicting signal sequence cleavage sites. *Nucleic Acids Research* **14**, 4683–4690.
39. Nielsen, H., Engelbrecht, J., Brunak, S., and von Heijne, G. (1997) Identification of prokaryotic and eukaryotic signal peptides and prediction of their cleavage sites. *Protein Engineering* **10**, 1–6.
40. Bendtsen, J.D., Nielsen, H., von Heijne, G., and Brunak, S. (2004) Improved prediction of signal peptides: SignalP 3.0. *Journal of Molecular Biology* **340**, 783–795.
41. Nielsen, H. and Krogh, A. (1998) Prediction of signal peptides and signal anchors by a hidden Markov model. *Proceedings of the International Conference on Intelligent Systems for Molecular Biology* **6**, 122–130.
42. Chou, K.C. (2001) Prediction of protein signal sequences and their cleavage sites. *Proteins* **42**, 136–139.
43. Vert, J.P. (2002) Support vector machine prediction of signal peptide cleavage site using a new class of kernels for strings. In R.B. Altman, A.K. Dunker, L. Hunter, K. Lauerdale, and T.E. Klein, editors, *Proceedings of the Pacific Symposium on Biocomputing*, pages 649–660. World Scientific, Singapore
44. Kahsay, R.Y., Gao, G., and Liao, L. (2005) Discriminating transmembrane proteins from signal peptides using svm-Fisher approach. In *ICMLA '05: Proceedings of the Fourth International Conference on Machine Learning and Applications*, pages 151–155. IEEE Computer Society, Washington, DC, USA, ISBN 0-7695-2495-8.

45. Käll, L., Krogh, A., and Sonnhammer, E.L.L. (2007) Advantages of combined transmembrane topology and signal peptide prediction – the Phobius web server. *Nucleic Acids Research* **35**, W429.
46. Viklund, H., Bernsel, A., Skwark, M., and Elofsson, A. (2008) SPOCTOPUS: a combined predictor of signal peptides and membrane protein topology. *Bioinformatics* **24**, 2928.
47. Viklund, H. and Elofsson, A. (2004) Best α-helical transmembrane protein topology predictions are achieved using hidden Markov models and evolutionary information. *Protein Science* **3**, 1908–1917.
48. von Heijne, G. (1983) Patterns of amino acids near signal-sequence cleavage sites. *European Journal of Biochemistry* **133**, 17–21.
49. Perlman, D. and Halvorson, H.O. (1983) A putative signal peptidase recognition site and sequence in eukaryotic and prokaryotic signal peptides. *Journal of Molecular Biology* **167**, 391–409.

Chapter 5

Protein Structure Modeling

Lars Malmström and David R. Goodlett

Abstract

The tertiary structure of proteins can reveal information that is hard to detect in a linear sequence. Knowing the tertiary structure is valuable when generating hypothesis and interpreting data. Unfortunately, the gap between the number of known protein sequences and their associated structures is widening. One way to bridge this gap is to use computer-generated structure models of proteins. Here we present concepts and online resources that can be used to identify structural domains in proteins and to create structure models of those domains.

Key words: Tertiary structure, Protein structure modeling, Protein folding

1. Introduction

Knowing the 3D, or tertiary, structure of a protein provides a wealth of information that provides a valuable resource for hypothesis generation and analysis of various types of biochemical and biological data [1] (see Note 1). For example, knowledge of amino acids spatial juxtaposition, separated in primary sequence space that are close in higher-order space to create an active site, may be obtained by structure predictions where it is possible to discriminate amino acids that are conserved for structural reasons from those conserved for biochemical reasons or for protein–protein interactions. Structures are traditionally determined through experimental means where X-ray crystallography and nuclear magnetic resonance (NMR) dominate (2). Common between the two is that they require highly trained personnel, expensive equipment, long analysis times, and relatively large amounts of protein. The gap between the number of known protein sequences and the number of protein structures is constantly widening as DNA sequencing technologies develop faster than

David Fenyö (ed.), *Computational Biology*, Methods in Molecular Biology, vol. 673,
DOI 10.1007/978-1-60761-842-3_5,

high-throughput structure determination approaches. To bridge this gap, computational scientists have spent the better part of four decades coming up with ways to determine the structure of a protein in silico. This is referred to protein structure modeling or protein structure prediction.

Given space constraints within this chapter, it is not possible to write a comprehensive guide covering all modeling approaches. Instead, we will focus on a limited number of techniques and demonstrate them with a specific example that hopefully will shed some light on this powerful technique. Notably, we will omit discussion of high-resolution approaches in favor for low-resolution technologies where the primer goal is to roughly estimate how the amino acids relate to each other in space.

Modeling a protein is equivalent to finding an energy minimum (which is context dependent) and can be divided up to two separate problems: (1) an accurate energy function and (2) a strategy to sample conformational space [3]. A conformation is generated by moving one of many parts of the protein structure (such as atoms) and then estimating the energy using an energy function. This is then repeated until the lowest energy-conformation is found. It might seem simple at first but, when considering the enormous number of possible conformations that exceed the total number of atoms in the universe even for small proteins in conjunction with an incomplete understanding of what governs the energy of a protein chain, the problem is daunting. In addition, the energy difference between the folded and unfolded states is quite small, which for biological reasons allows organisms to more easily regulate and degrade proteins, but complicates locating the native energy minimum of a protein.

Since we cannot explore the entire conformational space even for small proteins, nor can we distinguish correct from incorrect with absolute precision, the problem must be constrained as much as possible, thereby reducing the search space. Reducing the search space leads to a more manageable problem and it also minimizes the number of possible low-energy conformations. However, the types of constraints available are highly dependent on the protein one wishes to model. The best type of constraint often comes from a close sequence homolog for which an experimental structure is known. This works because we know that in general structure is a highly conserved component across species allowing us to constrain the protein of unknown structure with the structure of the homolog. This approach provides a reasonably confident template of the most-likely correct topology where most amino acids are close to their correct positions for related biological function to be carried out. This type of modeling goes under the designation *homology modeling*. We will briefly discuss Modeller (http://salilab.org/modweb) [4] in this tutorial, but we want to make clear that

we are not making any claims that this choice reflects a more accurate outcome in all cases. Modeller is, however, one of the more popular algorithms and it is free for academics. In addition, the originator, Andrej Sali and his colleagues have modeled millions of proteins and made them available to the public through ModBase [5], which we will also discuss.

If no homolog for which the structure is known can be detected, one has to resort to different types of constraints and simplifications. This template-less approach is sometimes referred to as de novo modeling and again, there are numerous approaches to this problem. Here, we have chosen to cover use of the software Rosetta [6]. Rosetta is one of the more successful approaches judging from repeated success in the double-blind assessment of protein structure prediction technologies organized by the protein folding community every other year [7]. Rosetta constrains the search space by assuming that each segment of three amino acids and nine amino acids has a limited number of likely conformations; possible conformations are extracted from the protein data bank, PDB [2] and hence are determined by experimental means. These local conformations are then assembled by a Monte Carlo approach until an energy minimum is reached. This type of approach is sometimes referred to as fragment insertion method since the local conformations extracted from the PDB are called fragments. Rosetta starts with a highly simplified model where amino acids are represented as a single point, or centroid, and the energy function considers only radius of gyration and centroids overlapping. As the simulation proceeds, the energy function gets more complex adding more terms such as environment and beta-strand pairing. Later, the centroids are replaced with a full atom representation and the energy function includes terms such as hydrogen-bond energy. The further the simulation proceeds, the more realistic the model and the more unlikely it is that large conformational changes will be accepted. The result of this outcome is that many simulations end in a local energy minima failing to find the native topology. To circumvent this problem, Rosetta is run thousands of times and the resulting models are clustered based on structural similarity [8]. This works because the native energy basin is normally large compared to the local minima basins. The larger the basin, the more models end up in that minima and hence large clusters are more likely to be correct. In general, more than one model is reported from a Rosetta simulation. There is a large web-based resource that provides premodeled proteins found under the address http://yeastrc.org/pdr [9] in which structural domains (see below) have been predicted using Ginzu [10] and domains lacking homologs of known structure have been modeled using Rosetta with the top five predictions selected using MCM [11].

It is quite common that the protein of interest lacks close homologs of known structure, in which case more distant homologs of known structure can often be detected [12]. These cases are sometimes modeled by hybrid approaches where parts of the protein chain that can be aligned to one or more templates are modeled by the homology approach whereas the rest are modeled by de novo approaches. This category of modeling is called fold recognition modeling, FR for short and both Modeller and Rosetta [13] and numerous other approaches can be used in this approach.

There is an additional layer of complexity that needs to be considered when modeling proteins, and that is structural domains [14]. The above-mentioned technologies are all optimized for a single structural domain analysis. A structural domain is loosely defined as an autonomously folding unit located in the primary protein sequence. From a simplified point of view, domains are modules that perform a specific biological task that species tend to reuse in different contexts. Sometimes, these domains are expressed as single peptide chains, but more often they are part of a bigger peptide chain, which contains more than one structural domain.

2. Materials

This tutorial covers the basic types of modeling where a personal computer with a modern Internet browser such as Firefox, Opera, Safari, or Internet Explorer is the only requirements (see Note 2). We will be demonstrating these techniques using the human protein "Protein kinase-like protein SgK071," SwissProt AC Q8NE28, see Fig. 1. Please note that the "results" presented here have not been verified and cannot be trusted. This information is presented simply as an example.

3. Methods

3.1. Find the Protein of Interest in a Sequence Database

1. Search for the protein of interest in Expasy (http://www.expasy.org) (15), one of the many online resources that collect information about proteins. In this case, since we know the SwissProt AC (Q8NE28), we'll simply type it into the search box.
2. At the bottom of the page, the primary sequence (see Fig. 1) is displayed.

```
>sp|Q8NE28|SGK71_HUMAN Protein kinase-like protein SgK071 OS=Homo
sapiens GN=SGK071 PE=2 SV=4
MLGPGSNRRRPTQGERGPGSPGEPMEKYQVLYQLNPGALGVNLVVEEMETKVKHVIKQVE
CMDDHYASQALEELMPLLKLRHAHISVYQELFITWNGEISSLYLCLVMEFNELSFQEVIE
DKRKAKKIIDSEWMQNVLGQVLDALEYLHHLDIIHRNLKPSNIILISSDHCKLQDLSSNV
LMTDKAKWNIRAEEDPFRKSWMAPEALNFSFSQKSDIWSLGCIILDMTSCSFMDGTEAMH
LRKSLRQSPGSLKAVLKTMEEKQIPDVETFRNLLPLMLQIDPSDRITIKDVVHITFLRGS
FKSSCVSLTLHRQMVPASITDMLLEGNVASILEVMQKFSGWPEVQLRAMKRLLKMPADQL
GLPWPPELVEVVVTTMELHDRVLDVQLCACSLLLHLLGQALVHHPEAKAPCNQAITSTLL
SALQSHPEEEPLLVMVYSLLAITTTQESESLSEELQNAGLLEHILEHLNSSLKSRDVCAS
GLGLLWALLLDGIIVNKAPLEKVPDLISQVLATYPADGEMAEASCGVFWLLSLLGCIKEQ
QFEQVVALLLQSIRLCQDRALLVNNAYRGLASLVKVSELAAFKVVVQEEGGSGLSLIKET
YQLHRDDPEVVENVGMLLVHLASYEEILPELVSSSMKALLQEIKERFTSSLVSDSSAFSK
PGLPPGGSPQLGCTTSGGLE
```

Fig. 1. The primary sequence of Protein kinase-like protein SgK071.

3. Under the section cross-reference, subsection 3D structure databases, there might be links to the Protein DataBank (PDB). If these links exist, the native structure of this protein is experimentally determined making the modeling unnecessary. Only a small fraction of proteins are of known structure (51,757 as of July 2008) and it is likely that the protein will not be found in the RCSB database.

3.2. Identify Structural Domains

1. Identifying structural domains is non-trivial and is mostly based on methods that either find similarities to proteins where the domain boundaries are known through experimental means or by identifying conserved stretches of amino acids that are present in many different proteins (see Note 3).
2. Check to see if the protein is present in InterPro (http://www.ebi.ac.uk/interpro) [16]. InterPro has run several domain prediction algorithms for most proteins in SwissProt and the InterPro entries are linked from the Expasy page under section Cross-references.
3. In this case, two domains are identified; the first domain, a protein kinase-like domain is detected by three different methods, Pfam [17], Prosite [18], and Superfamily [19]. Superfamily also detects the second domain, and ARM-repeat

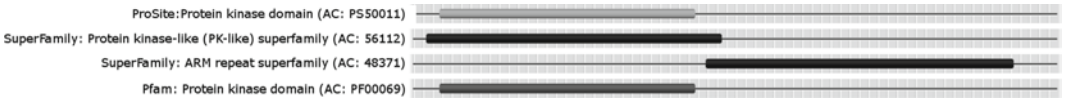


Fig. 2. Sequence-based domain predictions, from the InterPro database (original URL: http://www.ebi.ac.uk/interpro/ISpy?mode=single&ac=Q8NE28). InterPro integrates a number of domain prediction tools. In this image, only the integrated tools are displayed. This protein is a two domain protein where the first part is a kinase-like domain and the second domain is an ARM-repeat domain.

domain, see Fig. 2. The two domains are roughly equal in size.

4. If the protein is not present in InterPro, it is of course possible to run these tools on their respective websites and infer domains and domain boundaries from the results (see Note 4).

3.3. Homology Model Database, ModBase

1. ModBase (http://modbase.compbio.ucsf.edu) is an online resource published by Andrej Sali and his colleagues at University of San Francisco (see Note 5). ModBase is an attempt to make homology models of every protein using Modeller. Structures of 1.34 million proteins are available through the web interface making this a valuable resource.
2. ModBase is linked by Expasy from the section cross-references, subsection 3D structure databases, but can also be searched in various ways, including BLAST which only requires the primary sequence.
3. In our example, we find that two models exist for our protein, both covering the first domain, the kinase-like domain. The first model covers amino acids 9–342 and was created using the protein structure 1ywrA (PDB AC 1ywr, polypeptide chain A), with a sequence identity of 20%. The second model is covering amino acids 9–382, created using 2ozaA as template, with a sequence identify of 17%. Both these models are statistically significant and either one can be used.
4. The models can be downloaded and viewed in any protein structure viewer, such as rasmol (free download from http://www.openrasmol.org) [20]. Visual inspection of the two models reveals that the second model has two segments that seem to be sticking out of the domain, the first segment is a loop between amino acid 184 and 207, and the second segment is the C-terminal part from 359. This observation, together with the higher sequence identity for model 1, makes model 1 more preferable, see Fig. 3a.
5. If the protein of interest is missing from ModBase, it's possible to make models with Modeller using ModWeb,

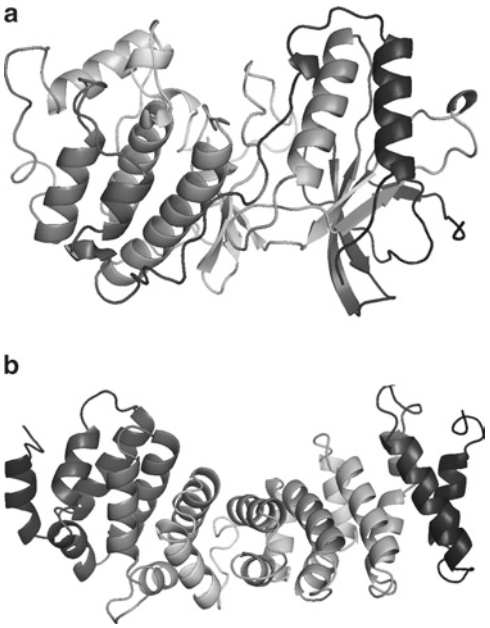


Fig. 3. Cartoon representation of domain 1 (**a**) from ModBase and domain 2 (**b**) generated from an alignment from the Meta server using Modeller.

(http://www.salilab.org/modweb/) which lets you enter a protein sequence of interest.

3.4. De Novo Model Database: The YRC Public Data Repository

1. In this example, we have a putative structure for the first domain but for the second domain we have no structure. The Yeast Resource Center (YRC) at the University of Washington has a public database, the public data repository (http://www.yeastrc.org/pdr) which contains de novo protein structure modeling from large-scale protein structure modeling projects (see Note 5). It also contains domain predictions from the domain prediction tool Ginzu [10].
2. Searching the database with the accession number returns a single protein and at the bottom of the protein overview page is the structure prediction information. Ginzu predicts five types of domains, PSI-BLAST, FFAS03, Pfam, MSA, and deduced. The first type of domain is for domains which have a homolog of known structure detected using blast or psi-blast. A FFAS03 domain indicates that a homolog of known structure can be detected using FFAS03, a fold recognition technology. The three remaining domains are domains that indicate that no homolog of known structure can be detected. Only the last three will have de novo models and only in the case where the domains are less than 150 amino acids. If de novo models are present, they are presented with a possible Structural Classification of Proteins (SCOP) classification.

The models can be downloaded by following the link to the domain.

3. In this case, the two domains of our protein are identified, the first as a PSI-BLAST domain and the second a FFAS03 domain. The FFAS03 domains are not selected for modeling and hence, we will use the Meta server (http://www.bioinfo.pl/meta) to detect a template and to generate models.
4. The Meta server uses multiple other algorithms to detect a potential template and to create an optimal alignment and then combines information from all the algorithms using a neural net (12). To submit a sequence to the Meta server is quite self-explanatory. The Meta server returns numerous templates and displays the alignment ranked by the J-score. J-scores over 50 are considered significant. The second domain returned with a J-score of 215 for template 2z6hA, which belongs to the ARM Superfamily (SCOP AC a.118.1). It is possible to create a model from this alignment by clicking the [model] link to the right of the alignment. This service is free to academic users who must register. The resulting model is displayed in Fig. 3b.
5. If no significant result was returned by the Meta server, Rosetta is available online at http://robetta.bakerlab.org (21) (see Note 6). While this portal will run a domain prediction software and then predict the structure of each individual domain with a template based method where a template is available and a de novo method (fragment insertion) where no template is available, it is quite resource intensive and the turn-around time is long.

4. Notes

1. For an example of the application of these techniques, please see ref. 1.
2. Most of these tools are available online. There are advantages to this, but there are also disadvantages. One obvious disadvantage is that it is quite difficult to "scale" the modeling effort to large number of proteins and that the turn-around time is sometimes long. Also possible is that most of the tools are available as a download for local use. This requires more computer skills than running them online, and hence we do not cover that here as it will be self-explanatory to the skilled computer specialist exploring the use of protein modeling.
3. The average length of structural a domain is less than 200 (based on the SCOP definition of domains) and it is closer to

400 for SwissProt and hence, it is expected that the average protein will have two structural domains that must be examined.

4. If no domains can be detected, one can resort to identifying "block structures" in a multiple sequence alignment. The multiple sequence alignment can be generated using blast or PSI-BLAST from NCBI webpage, http://blast.ncbi.nlm.nih.gov/Blast.cgi. Viewing the alignment of longer proteins sometimes has a "blocky" appearance where one part of the sequence has numerous homologs that do not cover the other parts. These blocks are indicative of domains and thus putative domains can be identified by the block boundaries.
5. The online databases are quite comprehensive, but newly sequenced proteins are, for obvious reasons, not present. However, because all the tools presented here are available via web services, it is possible to model these proteins too.
6. There are also proteins that belong to protein families that are less studied for which most of these techniques fail. Note that the tools presented herein are dependent on knowing something about homologs to the protein of interest.

References

1. Pacheco, B., Maccarana, M., Goodlett, DR., Malmström, A., Malmström, L. (2008), *Identification of the active site of DS-epimerase 1 and requirement of N-glycosylation for enzyme function. J Biol Chem 2009 Jan 16; 284(3)*: 1741–7.
2. Berman, H., Henrick, K., Nakamura, H., Markley, JL. (2007), *The worldwide Protein Data Bank (wwPDB): ensuring a single, uniform archive of PDB data. Nucleic Acids Res 35*: D301–3 (pmid: 17142228).
3. Rohl, CA., Strauss, CE., Misura, KM., Baker, D. (2004), *Protein structure prediction using Rosetta. Methods Enzymol 383*: 66–93. (pmid: 15063647).
4. Eswar, N., Eramian, D., Webb, B., Shen, MY., Sali, A. (2008), *Protein structure modeling with Modeller. Methods Mol Biol 426*: 145–59. (pmid: 18542861).
5. Pieper, U., Eswar, N., Davis, FP., Braberg, H., Madhusudhan, MS., Rossi, A., Marti-Renom, M., Karchin, R., Webb, BM., Eramian, D., Shen, MY., Kelly, L., Melo, F., Sali, A. (2006), *MODBASE: a database of annotated comparative protein structure models and associated resources. Nucleic Acids Res 34*: D291–5. (pmid: 16381869).
6. Simons, KT., Kooperberg, C., Huang, E., Baker, D. (1997), *Assembly of protein tertiary structures from fragments with similar local sequences using simulated annealing and Bayesian scoring functions. J Mol Biol 268*: 209–25. (pmid: 9149153).
7. Das, R., Qian, B., Raman, S., Vernon, R., Thompson, J., Bradley, P., Khare, S., Tyka, MD., Bhat, D., Chivian, D., Kim, DE., Sheffler, WH., Malmström, L., Wollacott, AM., Wang, C., Andre, I., Baker, D. (2007), *Structure prediction for CASP7 targets using extensive all-atom refinement with Rosetta@home. Proteins 1*: 118–28. (pmid: 17894356).
8. Shortle, D., Simons, KT., Baker, D. (1998), *Clustering of low-energy conformations near the native structures of small proteins. Proc Natl Acad Sci USA 95*: 11158–62. (pmid: 9736706).
9. Riffle, M., Malmström, L., Davis, TN. *The yeast resource center public data repository.* (2005), *Nucleic Acids Res 33*: D378–82. (pmid: 15608220).
10. Kim, DE., Chivian, D., Malmström, L., Baker, D. (2005), *Automated prediction of domain boundaries in CASP6 targets using Ginzu and*

RosettaDOM. Proteins Suppl 7: 193–200. (pmid: 16187362).

11. Malmström, L., Riffle, M., Strauss, CE., Chivian, D., Davis, TN., Bonneau, R., Baker, D. (2007), *Superfamily assignments for the yeast proteome through integration of structure prediction with the gene ontology. PLoS Biol 5*: e76. (pmid: 17373854).
12. Ginalski, K., Elofsson, A., Fischer, D., Rychlewski, L. (2003), *3D-Jury: a simple approach to improve protein structure predictions. Bioinformatics 19*: 1015–8. (pmid: 12761065).
13. Misura, KM., Chivian, D., Rohl, CA., Kim, DE., Baker, D. (2006), *Physically realistic homology models built with ROSETTA can be more accurate than their templates. Proc Natl Acad Sci USA 103*: 5361–6. (pmid: 16567638).
14. Wetlaufer, DB. (1973), *Nucleation, rapid folding, and globular intrachain regions in proteins. Proc Natl Acad Sci USA 70*: 697–701. (pmid: 4351801).
15. The UniProt Consortium (2008), *The Universal Protein Resource (UniProt) 2009. Nucleic Acids Res 2009 Jan; 37*(Database issue): D169–74.
16. Hunter, S., Apweiler, R., Attwood, TK., Bairoch, A., Bateman, A., Binns, D., Bork, P., Das, U., Daugherty, L., Duquenne, L., Finn, RD., Gough, J., Haft, D., Hulo, N., Kahn, D., Kelly, E., Laugraud, A., Letunic, I., Lonsdale, D., Lopez, R., Madera, M., Mas (2008) InterPro: the integrative protein signature database. *Nucleic Acids Res* 2009 *Jan; 37*(Database issue): D211–5.
17. Bateman, A., Birney, E., Cerruti, L., Durbin, R., Etwiller, L., Eddy, SR., Griffiths-Jones, S., Howe, KL., Marshall, M., Sonnhammer, EL. (2002), *The Pfam protein families database. Nucleic Acids Res 30*: 276–80. (pmid: 11752314).
18. Falquet, L., Pagni, M., Bucher, P., Hulo, N., Sigrist, CJ., Hofmann, K., Bairoch, A. (2002), *The PROSITE database, its status in 2002. Nucleic Acids Res 30*: 235–8. (pmid: 11752303).
19. Gough, J., Chothia, C. (2002), *SUPERFAMILY: HMMs representing all proteins of known structure. SCOP sequence searches, alignments and genome assignments. Nucleic Acids Res 30*: 268–72. (pmid: 11752312)
20. Sayle, RA., Milner-White, EJ. (1995), *RASMOL: biomolecular graphics for all. Trends Biochem Sci 20:* 374. (pmid: 7482707).
21. Kim, DE., Chivian, D., Baker, D. (2004), *Protein structure prediction and analysis using the Robetta server. Nucleic Acids Res 32*: W526–31. (pmid: 15215442).

Chapter 6

Template-Based Protein Structure Modeling

Andras Fiser

Abstract

Functional characterization of a protein is often facilitated by its 3D structure. However, the fraction of experimentally known 3D models is currently less than 1% due to the inherently time-consuming and complicated nature of structure determination techniques. Computational approaches are employed to bridge the gap between the number of known sequences and that of 3D models. Template-based protein structure modeling techniques rely on the study of principles that dictate the 3D structure of natural proteins from the theory of evolution viewpoint. Strategies for template-based structure modeling will be discussed with a focus on comparative modeling, by reviewing techniques available for all the major steps involved in the comparative modeling pipeline.

Key words: Homology modeling, Comparative protein structure modeling, Template-based modeling, Loop modeling, Side chain modeling, Sequence-to-structure alignment

1. Introduction

The class of methods referred to as template-based modeling includes both the threading techniques that return a full 3D description for the target and comparative modeling (1). This class of protein structure modeling relies on detectable similarity spanning most of the modeled sequence and at least one known structure. Comparative modeling refers to those template-based modeling cases where not only the fold is determined from a possible set of available templates, but a full atom model is also built (2). In practice, it means that if the structure of at least one protein in the family has been determined by experimentation, the other members of the family can be modeled based on their alignment to the known structure. It is possible because a small change in the protein sequence usually results in a small change in its 3D structure (3). It is also facilitated by the fact that 3D structure of

David Fenyö (ed.), *Computational Biology*, Methods in Molecular Biology, vol. 673,
DOI 10.1007/978-1-60761-842-3_6,

proteins from the same family is more conserved than their amino-acid sequences (4). Therefore, if similarity between two proteins is detectable at the sequence level, then structural similarity can usually be assumed. The increasing applicability of template-based modeling is owing to the observation that the number of different folds that proteins adopt is rather limited and because worldwide Structural Genomics projects are aggressively mapping out the universe of possible folds (5–7).

Template-based approaches to structure prediction have their advantages and limitations. Comparative protein structure modeling usually provides high-quality models that are comparable with low-resolution X-ray crystallography or medium-resolution NMR solution structures. However, the applicability of these approaches is limited to those sequences that can be confidently mapped to known structures. Currently, the probability of finding related proteins of known structure for a sequence picked randomly from a genome ranges approximately from 30 to 80%, depending on the genome. Approximately 70% of all known sequences have at least one domain that is detectably related to at least one protein of known structure (8). This fraction is more than an order of magnitude larger than the number of experimentally determined protein structures deposited in the Protein Data Bank (PDB) (9). As we will see, in practice, template-based modeling always includes information that is independent from the template, in the form of various force restraints from general statistical observations or molecular mechanical force fields. As a consequence of improving force fields and search algorithms, the most successful approaches often explore more and more template-independent conformational space (10, 11).

2. Methods

All current comparative modeling methods consist of five sequential steps: (1) to search for proteins with known 3D structures that are related to the target sequence, (2) to pick those structures that will be used as templates, (3) to align their sequences with the target sequence, (4) to build the model for the target sequence given its alignment with the template structures, and (5) to evaluate the model, using a variety of criteria.

There are several computer programs and web servers that automate the comparative modeling process (Table 1). While the web servers are convenient and useful (10, 12–14), the best results are still obtained by nonautomated, expert use of the various modeling tools (15). Complex decisions for selecting the structurally and biologically most relevant templates, optimally

Table 1
Names and www addresses of some online tools useful for various aspects of comparative modeling

Template search and alignments	
BLAST/PSI-BLAST	http://www.ncbi.nlm.nih.gov/BLAST/
FastA/SSEARCH	http://www.ebi.ac.uk/fasta33
FASS03	http://www.ffas.ljcrf.edu/ffas-cgi/cgi/ffas.pl
PSIPRED	http://www.bioinf.cs.ucl.ac.uk/psipred/
123D	http://www.123d.ncifcrf.gov
UCLA-DOE	http://www.doe-mbi.ucla.edu/Services/FOLD/
PHYRE/3D-PSSM	http://www.sbg.bio.ic.ac.uk/~3dpssm
FUGUE	http://www.cryst.bioc.cam.ac.uk/~fugue
LOOPP	http://www.cbsuapps.tc.cornell.edu/
MUSTER	http://www.zhang.bioinformatics.ku.edu/MUSTER/
SAM-T06	http://www.soe.ucsc.edu/research/compbio/SAM_T06/T06-query.html
Prospect	http://www.compbio.ornl.gov/structure/prospect
Smith–Waterman	http://www.jaligner.sourceforge.net/
ClustalW	http://www.ebi.ac.uk/clustalw/
MUSCLE	http://www.drive5.com/lobster/
T-COFFEE	http://www.tcoffee.vital-it.ch/
PROMALS	http://www.prodata.swmed.edu/promals/promals.php
PROBCONS	http://www.probcons.stanford.edu
Homology modeling, loop and side-chain modeling	
MMM	http://www.fiserlab.org/servers/MMM
M4T	http://www.fiserlab.org/servers/M4T
MODELLER	http://www.salilab.org/modeller/modeller.html
MODWEB	http://www.modbase.compbio.ucsf.edu/ModWeb20-html/modweb.html
I-TASSER	http://www.zhang.bioinformatics.ku.edu/I-TASSER/
HHPRED	http://www.toolkit.tuebingen.mpg.de/hhpred
3D-JIGSAW	http://www.bmm.icnet.uk/servers/3djigsaw/
CPH-MODELS	http://www.cbs.dtu.dk/services/CPHmodels/
COMPOSER	http://www.cryst.bioc.cam.ac.uk
SWISSMODEL	http://swissmodel.expasy.org/workspace/
FAMS	http://www.pharm.kitasato-u.ac.jp/fams/

(continued)

Table 1 (continued)

WHATIF	http://www.cmbi.kun.nl/whatif/
PUDGE	http://www.wiki.c2b2.columbia.edu/honiglab_public/index.php/Software
3D-JURY	http://www.meta.bioinfo.pl
RAPPER	http://www.mordred.bioc.cam.ac.uk/~rapper
ESYPRED3D	http://www.fundp.ac.be/sciences/biologie/urbm/bioinfo/esypred/
CONSENSUS	http://www.structure.bu.edu/cgi-bin/consensus/consensus.cgi
PCONS	http://www.pcons.net
SCWRL	http://www.dunbrack.fccc.edu/SCWRL3.php
WLOOP	http://www.bioserv.rpbs.jussieu.fr/cgi-bin/WLoop
ARCHPRED	http://www.fiserlab.org/servers/archpred
MODLOOP	http://www.salilab.org/modloop
Model evaluation	
PROCHECK	http://www.biochem.ucl.ac.uk/~roman/procheck/procheck.html
WHATCHECK	http://www.swift.cmbi.ru.nl/gv/whatcheck/
Prosa-web	http://www.prosa.services.came.sbg.ac.at/prosa.php
VERIFY3D	http://www.nihserver.mbi.ucla.edu/Verify_3D
ANOLEA	http://www.protein.bio.puc.cl/cardex/servers/anolea/
AQUA	http://www.urchin.bmrb.wisc.edu/~jurgen/Aqua/server/
PROQ	http://www.sbc.su.se/~bjornw/ProQ/ProQ.cgi

combining multiple template information, refining alignments in nontrivial cases, selecting segments for loop modeling, including cofactors and ligands in the model, or specifying external restraints require an expert knowledge that is difficult to fully automate (16), although more and more efforts on automation point to this direction (17, 18).

2.1. Searching for Structures Related to the Target Sequence

Comparative modeling usually starts by searching the PDB (9) for known protein structures using the target sequence as the query. This search is generally done by comparing the target sequence with the sequence of each of the structures in the database.

There are two main classes of protein comparison methods that are useful in fold identification. The first class compares the sequences of the target with each of the database templates by using pairwise sequence–sequence comparisons (such as FASTA

and BLAST (19)) (20–22) and fold assignments (23). To improve the sensitivity of the sequence-based searches, evolutionary information can be incorporated in the form of multiple sequence alignment (24–28). These approaches begin by finding all sequences in a sequence database that are clearly related to the target and easily aligned with it (29, 30). The multiple alignment of these sequences is the target sequence profile, which implicitly carries additional information about the location and pattern of evolutionarily conserved positions of the protein. The most well-known program in this class is PSI-BLAST (27), which implements a heuristic search algorithm for short motifs. A further step to increase the sensitivity of this approach is to precalculate sequence profiles for all the known structures and then use pairwise dynamic programming algorithm to compare the two profiles. This has been implemented, among other programs, in COACH (31) and FFAS03 (32, 33). The construction of profile-based Hidden Markov Models (HMM) is another sensitive way to locate universally conserved motifs among sequences (34). A substantial improvement in HMM approaches was achieved by incorporating information about predicted secondary structural elements (35, 36). Another development in this group of methods is the phylogenetic tree-driven HMM, which selects a different subset of sequences for profile HMM analysis at each node in the evolutionary tree (37). Locating sequence intermediates that are homologous to both sequences may also enhance the template searches (22, 38). These more sensitive fold identification techniques are especially useful for finding significant structural relationships when sequence identity between the target and the template drops below 25%. More accurate sequence profiles and structural alignments can be constructed with consistency-based approaches such as T-Coffee (39), PROMAL (and PROMAL3D for structures) (40, 41), and ProbCons (42).

The second class of methods relies on pairwise comparison of a protein sequence and a protein structure; the target sequence is matched against a library of 3D profiles or threaded through a library of 3D folds. These methods are also called fold assignment, threading, or 3D template matching (32, 43–47). These methods are especially useful when sequence profiles are not possible to construct because there are not enough known sequences that are clearly related to the target or potential templates.

Template search methods "outperform" the needs of comparative modeling in the sense that they are able to locate sequences that are so remotely related as to render construction of a reliable comparative model impossible. The reason for this is that sequence relationships are often established on short conserved segments, while a successful comparative modeling exercise requires an overall correct alignment for the entire modeled part of the protein.

2.2. Selecting Templates

Once a list of potential templates is obtained using searching methods, it is necessary to select one or more templates that are appropriate for the particular modeling problem. Several factors need to be taken into account when selecting a template.

2.2.1. Considerations in Template Selection

The simplest template selection rule is to select the structure with the highest sequence similarity to the modeled sequence. The construction of a multiple alignment and a phylogenetic tree (48) can help in selecting the template from the subfamily that is closest to the target sequence. The similarity between the "environment" of the template and the environment in which the target needs to be modeled should also be considered. The term "environment" is used here in a broad sense, including everything that is not the protein itself (e.g., solvent, pH, ligands, quaternary interactions). If possible, a template bound to the same or similar ligands as the modeled sequence should generally be used. The quality of the experimentally determined structure is another important factor in template selection. Resolution and R-factor of a crystal structure and the number of restraints per residue for an NMR structure are indicative of their accuracy. The criteria for selecting templates also depend on the purpose of a comparative model. For example, if a protein–ligand model is to be constructed, the choice of the template that contains a similar ligand is probably more important than the resolution of the template.

2.2.2. Advantage of Using Multiple Templates

It is not necessary to select only one template. In fact, the optimal use of several templates increases the model accuracy (13, 17, 49, 50); however, not all modeling programs are designed to accept more than one template. The benefit of combining multiple template structures can be twofold. First, multiple template structures may be aligned with different domains of the target, with little overlap between them, in which case, the modeling procedure can construct a homology-based model of the whole target sequence. Second, the template structures may be aligned with the same part of the target and build the model on the locally best template.

An elaborate way to select suitable templates is to generate and evaluate models for each candidate template structure and/or their combinations. The optimized all-atom models can then be evaluated by an energy or scoring function, such as the Z-score of PROSA (46) or VERIFY3D (51). These scoring methods are often sufficiently accurate to allow selection of the most accurate of the generated models (52). This trial-and-error approach can be viewed as limited threading (i.e., the target sequence is threaded through similar template structures). However, these approaches are good only at selecting various templates on a global level.

A recently developed method M4T (Multiple Mapping Method with Multiple Templates) selects and combines multiple

template structures through an iterative clustering approach that takes into account the "unique" contribution of each template, their sequence similarity among themselves and to the target sequence, and their experimental resolution (13, 17). The resulting models systematically outperformed models that were based on the single best template.

Another important observation from the same study was that below 40% sequence identity, models built using multiple templates are more accurate than those built using a single template only, and this trend is accentuated as one moves into more remote target–template pair cases. Meanwhile, the advantage of using multiple templates gradually disappears above 40% target–template sequence identity cases. This suggests that in this range, the average differences between the template and target structures are smaller than the average differences among alternative template structures that are all highly similar to the target (17).

2.3. Sequence-to-Structure Alignment

To build a model, all comparative modeling programs depend on a list of assumed structural equivalences between the target and template residues. This list is defined by the alignment of the target and template sequences. Many template search methods will produce such an alignment, and these sometimes can directly be used as the input for modeling. Often, however, especially in the difficult cases, this initial alignment is not the optimal target–template alignment. This is because search methods may be tuned for detection of remote relationships, which is often realized on a local motif and not on a full-length, optimal alignment. Therefore, once the templates are selected, an alignment method should be used to align them with the target sequence. When the target–template sequence identity is lower than 40%, the alignment accuracy becomes the most important factor affecting the quality of the resulting model. A misalignment by only one residue position will result in an error of approximately 4 Å in the model.

2.3.1. Taking Advantage of Structural Information in Alignments

Alignments in comparative modeling represent a unique class because on one side of the alignment there is always a 3D structure, the template. Therefore, alignments can be improved by including structural information from the template. For example, gaps should be avoided in secondary structure elements, in buried regions, or between two residues that are far in space. Some alignment methods take such criteria into account (47, 53, 54).

When multiple template structures are available, a good strategy is to superpose them with each other first, to obtain a multiple structure-based alignment highlighting structurally conserved residues (55–57). In the next step, the target sequence is aligned with this multiple structure-based alignment. The benefits of using multiple structures and multiple sequences are that they provide evolutionary and structural information

about the templates, as well as evolutionary information about the target sequence, and they often produce a better alignment for modeling than the pairwise sequence alignment methods (22, 58).

Multiple Mapping Method (MMM) directly relies on information from the 3D structure (14, 59). MMM minimizes alignment errors by selecting and optimally splicing differently aligned fragments from a set of alternative input alignments. This selection is guided by a scoring function that determines the preference of each alternatively aligned fragment of the target sequence in the structural environment of the template. The scoring function has four terms, which are used to assess the compatibility of alternative variable segments in the protein environment:(a) environment specific substitution matrices from FUGUE (47), (b) residue substitution matrix, Blosum (60), (c) A 3D–1D substitution matrix, H3P2, that scores the matches of predicted secondary structure of the target sequence to the observed secondary structures and accessibility types of the template residues (61), and (d) a statistically derived residue–residue contact energy term (62). MMM essentially performs a limited and inverse threading of short fragments: in this exercise the actual question is not the identification of a right fold, but identification of the correct alignment mapping, among many alternatives, for sequence segments that are threaded on the same fold. These local mappings are evaluated in the context of the rest of the model, where alignments provide a consistent solution and framework for the evaluation.

2.4. Model Building

When discussing the model building step within comparative protein structure modeling, it is useful to distinguish two parts: *template-dependent* and *template-independent* modeling. This distinction is necessary because certain parts of the target must be built without the aid of any template. These parts correspond to gaps in the template sequence within the target–template alignment. Modeling of these regions is commonly referred to as loop modeling problem. It is evident that these loops are responsible for the most characteristic differences between the template and target, and therefore are chiefly responsible for structural and consequently functional differences. In contrast to these loops, the rest of the target, and in particular the conserved core of the fold of the target, is built using information from the template structure.

2.4.1. Template-Dependent Modeling

A comparative model can be assembled from a framework of small number of rigid bodies obtained from the aligned template protein structures (63–65). The approach is based on the natural dissection of the protein structure into conserved core regions, variable loops that connect them, and side chains that decorate the backbone (66). A widely used program in this class is

2.4.1.1. Modeling by Assembly of Rigid Bodies

COMPOSER (67). The accuracy of a model can be somewhat increased when more than one template structure is used to construct the framework (68).

2.4.1.2. Modeling by Segment Matching or Coordinate Reconstruction

Comparative models can be constructed by using a subset of atomic positions from template structures as "guiding" positions, such as the Cα atoms, and by identifying and assembling short, all-atom segments that fit these guiding positions. The all-atom segments that fit the guiding positions can be obtained either by scanning all the known protein structures (69, 70) or by a conformational search restrained by an energy function (71, 72) or by a general method for modeling by segment matching (SEGMOD) (73). Even some side-chain modeling methods (74) and the class of loop construction methods based on finding suitable fragments in the database of known structures (75) can be seen as segment matching or coordinate reconstruction methods.

2.4.1.3. Modeling by Satisfaction of Spatial Restraints

The methods in this class begin by generating many constraints or restraints on the structure of the target sequence, using its alignment to related protein structures as a guide in a procedure that is conceptually similar to that used in determination of protein structures from NMR-derived restraints. The restraints are generally obtained by assuming that the corresponding distances between aligned residues in the template and the target structures are similar. These homology-derived restraints are usually supplemented by stereochemical restraints on bond lengths, bond angles, dihedral angles, and nonbonded atom–atom contacts that are obtained from a molecular mechanics force field (76). The model is then derived by minimizing the violations of all the restraints. Comparative modeling by satisfaction of spatial restraints is implemented in the computer program MODELLER (16, 77), currently the most popular comparative protein modeling program. In MODELLER, the various spatial relationships of distances, angles are expressed as conditional probability density functions (pdfs) and can be used directly as spatial restraints. For example, probabilities for different values of the main chain dihedral angles are calculated from the type of residue considered, from the main chain conformation of an equivalent template residue, and from sequence similarity between the two proteins. An important feature of the method is that the forms of spatial restraints were obtained empirically, from a database of protein structure alignments, without any user imposed subjective assumption. Finally, the model is obtained by optimizing the objective function in Cartesian space by the use of the variable target function method (78), employing methods of conjugate gradients and molecular dynamics with simulated annealing (79).

A similar comprehensive package is NEST that can build a homology model based on single sequence–template alignment

or from multiple templates. It can also consider different structures for different parts of the target (55).

2.4.1.4. Combining Alignments, Combining Structures

It is frequently difficult to select the best templates or calculate a good alignment. One way of improving a comparative model in such cases is to proceed with an iteration of template selection, alignment, and model building, guided by model assessment, until no improvement in the model is detected (80, 81). Some of these approaches are automated (55, 82). In one example, this task was achieved by a genetic algorithm protocol that starts with a set of initial alignments and then iterates through realignment, model building, and model assessment to optimize a model assessment score. Comparative models corresponding to various evolving alignments are built and assessed by a variety of criteria, partly depending on an atomic statistical potential. In another approach, a genetic algorithm was applied to automatically combine templates and alignments. A relatively simple structure-dependent scoring function was used to evaluate the sampled combinations (18).

Other attempts to optimize target–template alignments include the Robetta server, where alignments are generated by dynamic programming using a scoring function that combines information on many protein features, including a novel measure of how obligate a sequence region is to the protein fold. By systematically varying the weights on the different features that contribute to the alignment score, very large ensembles of diverse alignments are generated. A variety of approaches to select the best models from the ensemble, including consensus of the alignments, a hydrophobic burial measure, low- and high-resolution energy functions, and combinations of these evaluation methods were explored (83).

Those metaserver approaches that do not simply score and rank alternative models obtained from a variety of methods but further combine them could also be perceived as approaches that explore the alignment and conformational space for a given target sequence (84).

Another alternative for combined servers is provided by M4T. The M4T program automatically identifies the best templates and explores and optimally splices alternative alignments according to its internal scoring function that focuses on the features of the structural environment of each template (17).

2.4.1.5. Metaservers

Metaserver approaches have been developed to take advantage of the variety of other existing programs. Metaservers collect models from alternative methods and either use them for inputs to make new models or look for consensus solutions within them. For instance, FAMS-ACE (85) takes inputs from other servers as starting points for refinement and remodeling after which Verify3D

(51) is used to select the most accurate solution. Other consensus approaches include PCONS, a neural network approach that identifies a consensus model by combining information on reliability scores and structural similarity of models obtained from other techniques (86). 3D-JURY operates along the same idea; its selection is mainly based on the consensus of model structure similarity (87).

2.4.2. Template Independent Modeling: Modeling Loops, Insertions

In comparative modeling, target sequences often have inserted residues relative to the template structures or have regions that are structurally different from the corresponding regions in the templates. Therefore, no structural information about these inserted segments can be extracted from the template structures. These regions frequently correspond to surface loops. Loops often play an important role in defining the functional specificity of a given protein framework, forming the functional, ligand-binding active sites. The accuracy of loop modeling is a major factor determining the usefulness of comparative models in applications such as ligand docking or functional annotation. Loops are generally too short to provide sufficient information about their local fold, and the environment of each loop is uniquely defined by the solvent and the protein that cradles it. In a few rare cases, it was shown that even identical decapeptides in different proteins do not always have the same conformation (88, 89).

There are two main classes of loop modeling methods: (1) the database search approaches and (2) the conformational search approaches (90–92). There are also methods that combine these two approaches (93–95).

2.4.2.1. Fragment-Based Approach to Loop Modeling

Earlier, it was predicted that it is unlikely that structure databanks will ever reach a point when fragment-based approaches become efficient to model loops (96), which resulted in a boost in the development of conformational search approaches from around 2000. However, many details of the fold universe have been explored during the last decade due to the large number of new folds solved experimentally, which had a profound effect on the extent of known structural fragments. Recent analyses showed that loop fragments are not only well represented in current structure databanks, but shorter segments are also possibly completely explored already (97). It was reported that sequence segments up to 10–12 residues had a related (i.e. at least 50% identical) segment in PDB with a known conformation, and despite the six-fold increase in the sequence databank size and the doubling of PDB since 2002, there was not a single unique loop conformation or sequence segment entered in the PDB ever since. Consequently, more recent efforts have been taken to classify loop conformations into more general categories, thus extending the applicability of the database search approach for more cases

(98, 99). A recent work described the advantage of using HMM sequence profiles in classifying and predicting loops (100). An another recently published loop prediction approach first predicts conformation for a query loop sequence and then structurally aligns the predicted structural fragments to a set of nonredundant loop structural templates. These sequence–template loop alignments are then quantitatively evaluated with an artificial neural network model trained on a set of predictions with known outcomes (101).

ArchPred (98, 102), currently perhaps the most accurate database loop modeling approach, exploits a hierarchical and multidimensional database that has been set up to classify about 300,000 loop fragments and loop flanking secondary structures. Besides the length of the loops and types of bracing secondary structures, the database is organized along four internal coordinates, a distance and three types of angles characterizing the geometry of stem regions (103). Candidate fragments are selected from this library by matching the length, the types of bracing secondary structures of the query and by satisfying the geometrical restraints of the stems and subsequently inserted in the query protein framework where their fit is assessed by the root mean squared deviation (RMSD) of stem regions and by the number of rigid body clashes with the environment. In the final step, remaining candidate loops are ranked by a Z-score that combines information on sequence similarity and fit of predicted and observed ϕ/ψ main chain dihedral angle propensities. Confidence Z-score cutoffs are determined for each loop length. A web server implements the method. Predicted segments are returned, or optionally, these can be completed with side-chain reconstruction and subsequently annealed in the environment of the query protein by conjugate gradient minimization.

In summary, the recent reports about the more favorable coverage of loop conformations in the PDB suggest that database approaches are now rather limited by their ability to recognize suitable fragments, and not by the lack of these segments (i.e., sampling), as thought earlier .

2.4.2.2. Ab Initio Modeling of Loops

To overcome the limitations of the database search methods, conformational search methods were developed. There are many such methods, exploiting different protein representations, objective function terms, and optimization or enumeration algorithms. The search strategies include the minimum perturbation method (104), molecular dynamics simulations (92), genetic algorithms (105), Monte Carlo and simulated annealing (106, 107), multiple-copy simultaneous search (108), self-consistent field optimization (109), and an enumeration based on the graph theory (110). Loop prediction by optimization is applicable to both simultaneous modeling of several loops and those loops interacting with ligands,

neither of which is straightforward for the database search approaches, where fragments are collected from unrelated structures with different environments.

The MODLOOP module in MODELLER implements the optimization-based approach (111, 112). Loop optimization in MODLOOP relies on conjugate gradients and molecular dynamics with simulated annealing. The pseudoenergy function is a sum of many terms, including some terms from the CHARMM-22 molecular mechanics force field (76) and spatial restraints based on distributions of distances (113, 114) and dihedral angles in known protein structures. The performance of the approach later was further improved by using CHARMM molecular mechanic force field with Generalized Born (GB) solvation potential to rank final conformations (115). Incorporation of solvation terms in the scoring function was a central theme in several other subsequent studies (95, 116–118). Improved loop prediction accuracy resulted from the incorporation of an entropy like term to the scoring function, the "colony energy," derived from geometrical comparisons and clustering of sampled loop conformations (119, 120). The continuous improvement of scoring functions delivers improved loop modeling methods. Two recent loop modeling procedures have been introduced that are utilizing the effective statistical pair potential that is encoded in DFIRE (121–123). Another method is developed to predict very long loops using the Rosetta approach, essentially performing a mini folding exercise for the loop segments (124). In the Prime program, large numbers of loops are generated by using a dihedral angle-based building procedure followed by iterative cycles of clustering, side-chain optimization, and complete energy minimization of selected loop structures using a full-atom molecular mechanic force field (OPLS) with implicit solvation model (125).

2.5. Model Evaluation

After a model is built, it is important to check it for possible errors (see Note 1). The quality of a model can be approximately predicted from the sequence similarity between the target and the template and by performing internal and external evaluations.

Sequence identity above 30% is a relatively good predictor of the expected accuracy of a model. If the target–template sequence identity falls below 30%, the sequence identity becomes significantly less reliable as a measure of the expected accuracy of a single model (see Note 2). It is in such cases that model evaluation methods are most informative.

"Internal" evaluation of self-consistency checks whether or not a model satisfies the restraints used to calculate it, including restraints that originate from the template structure or obtained from statistical observations. Assessment of the stereochemistry of a model (e.g., bonds, bond angles, dihedral angles, and nonbonded atom–atom distances) with programs such as PROCHECK (126)

and WHATCHECK (127) is an example of internal evaluation. Although errors in stereochemistry are rare and less informative than errors detected by methods for external evaluation, a cluster of stereochemical errors may indicate that the corresponding region also contains other larger errors (e.g., alignment errors).

"External" evaluation relies on information that was not used in the calculation of the model and as a minimum test whether or not a correct template was used. A wrong template can be detected relatively easily with the currently available scoring functions. A more challenging task for the scoring functions is the prediction of unreliable regions in the model. One way to approach this problem is to calculate a "pseudoenergy" profile of a model, such as that produced by PROSA (128) or Verify3D (51). The profile reports the energy for each position in the model. Peaks in the profile frequently correspond to errors in the model. Other recent approaches usually combine a variety of inputs to assess the models, either wholly (129) or locally (130). In benchmarks, the best quality assessor techniques use a simple consensus approach, where reliability of a model is assessed by the agreement among alternative models that are sometimes obtained from a variety of methods (131, 132).

3. Accuracy of Modeling Methods and Typical Errors in Template Based Models

3.1. Accuracy of Methods

An informative way to test protein structure modeling methods, including comparative modeling, is provided by the biannual meetings on Critical Assessment of Techniques for Protein Structure Prediction (CASP) (133). Protein modelers are challenged to model sequences with unknown 3D structure and to submit their models to the organizers before the meeting. At the same time, the 3D structures of the prediction targets are being determined by X-ray crystallography or NMR methods. They only become available after the models are calculated and submitted. Thus, a bona fide evaluation of protein structure modeling methods is possible, although in these exercises it is not trivial to separate the contributions from programs and human expert knowledge. Alternatively a large-scale, continuous, and automated prediction benchmarking experiment is implemented in the program EVA – EValuation of Automatic protein structure prediction (134). Every week EVA submits prereleased PDB sequences to participating modeling servers, collects the results, and provides detailed statistics on secondary structure prediction, fold recognition, comparative modeling, and prediction on 3D contacts. The LiveBench program has implemented its evaluations

in a similar spirit (135). After many years of operations, these benchmark platforms are not kept up to date lately, although their service would be essential to keep the user community well informed about latest developments and the best-performing techniques available. A rigorous statistical evaluation (136) of a blind prediction experiment illustrated that the accuracies of the various model-building methods, using segment matching, rigid body assembly, satisfaction of spatial restraints, or any combinations of these are relatively similar when used optimally (137, 138). This also reflects on the fact that such major factors as template selection and alignment accuracy have a large impact on the overall model accuracy, and that the core of protein structures is highly conserved.

3.2. Errors in Comparative Models

The overall accuracy of comparative models spans a wide range. At the low end of the spectrum are the low resolution models whose only essentially correct feature is their fold. At the high end of the spectrum are the models with an accuracy comparable to medium-resolution crystallographic structures (139). Even low-resolution models are often useful to address biological questions because function can many times be predicted from only coarse structural features of a model. The errors in comparative models can be divided into five categories: (1) Errors in side-chain packing, (2) Distortions or shifts of a region that is aligned correctly with the template structures, (3) Distortions or shifts of a region that does not have an equivalent segment in any of the template structures, (4) Distortions or shifts of a region that is aligned incorrectly with the template structures, and (5) A misfolded structure resulting from using an incorrect template. Approximately 90% of the main-chain atoms are likely to be modeled with an RMS error of about 1 Å when the overall sequence identity is above 40% (140). When sequence identity is between 30 and 40%, the structural differences become larger, and the gaps in the alignment are more frequent and longer; misalignments and insertions in the target sequence become the major problems. As a result, the main-chain RMS error rises to about 1.5 Å for about 80% of residues. When sequence identity drops below 30%, the main problem becomes the identification of related templates and their alignment with the sequence to be modeled. In general, it can be expected that about 20% of residues will be misaligned and consequently incorrectly modeled with an error larger than 3 Å, at this level of sequence similarity. To put the errors in comparative models into perspective, we list the differences among structures of the same protein that have been determined experimentally. A 1 Å accuracy of main-chain atom positions corresponds to X-ray structures defined at a low resolution of about 2.5 Å and with an R-factor of about 25% (141), as well as to medium-resolution NMR structures determined from ten interproton distance

restraints per residue. Similarly, differences between the highly refined X-ray and NMR structures of the same protein also tend to be about 1 Å (142). Changes in the environment (e.g., oligomeric state, crystal packing, solvent, ligands) can also have a significant effect on the structure (143). The performance of comparative modeling may sometimes appear overstated because what is usually discussed in the literature are the mean values of backbone deviations. However, individual errors in certain residues essential for the protein function, even in the context of an overall backbone RMSD of less than 1 Å, can still be large enough to prevent reliable conclusions to be drawn regarding mechanism, protein function, or drug design.

Acknowledgments

This review is partially based on our previous publications (1, 144).

References

1. Fiser, A. (2004) Protein structure modeling in the proteomics era. *Expert Rev Proteomics*, **1**, 97–110.
2. Marti-Renom, M.A., Stuart, A.C., Fiser, A., Sanchez, R., Melo, F., and Sali, A. (2000) Comparative protein structure modeling of genes and genomes. *Annu Rev Biophys Biomol Struct*, **29**, 291.
3. Chothia, C. and Lesk, A.M. (1986) The relation between the divergence of sequence and structure in proteins. *EMBO J*, **5**, 823.
4. Lesk, A.M. and Chothia, C. (1980) How different amino acid sequences determine similar protein structures: the structure and evolutionary dynamics of the globins. *J Mol Biol*, **136**, 225.
5. Andreeva, A., Howorth, D., Chandonia, J.M., Brenner, S.E., Hubbard, T.J., Chothia, C., and Murzin, A.G. (2008) Data growth and its impact on the SCOP database: new developments. *Nucleic Acids Res*, **36**, D419–D425.
6. Chothia, C., Gough, J., Vogel, C., and Teichmann, S.A. (2003) Evolution of the protein repertoire. *Science*, **300**, 1701.
7. Greene, L.H., Lewis, T.E., Addou, S., Cuff, A., Dallman, T., Dibley, M., Redfern, O., Pearl, F., Nambudiry, R., Reid, A., et al. (2007) The CATH domain structure database: new protocols and classification levels give a more comprehensive resource for exploring evolution. *Nucleic Acids Res*, **35**, D291–D297.
8. Pieper, U., Eswar, N., Davis, F.P., Braberg, H., Madhusudhan, M.S., Rossi, A., Marti-Renom, M., Karchin, R., Webb, B.M., Eramian, D., et al. (2006) MODBASE: a database of annotated comparative protein structure models and associated resources. *Nucleic Acids Res*, **34**, D291–D295.
9. Berman, H., Henrick, K., Nakamura, H., and Markley, J.L. (2007) The worldwide Protein Data Bank (wwPDB): ensuring a single, uniform archive of PDB data. *Nucleic Acids Res*, **35**, D301–D303.
10. Zhang, Y. (2007) Template-based modeling and free modeling by I-TASSER in CASP7. *Proteins*, **69 Suppl 8**, 108–117.
11. Das, R., Qian, B., Raman, S., Vernon, R., Thompson, J., Bradley, P., Khare, S., Tyka, M.D., Bhat, D., Chivian, D., et al. (2007) Structure prediction for CASP7 targets using extensive all-atom refinement with Rosetta@home. *Proteins*, **69 Suppl 8**, 118–128.
12. Battey, J.N., Kopp, J., Bordoli, L., Read, R.J., Clarke, N.D., and Schwede, T. (2007) Automated server predictions in CASP7. *Proteins*, **69 Suppl 8**, 68–82.
13. Fernandez-Fuentes, N., Madrid-Aliste, C.J., Rai, B.K., Fajardo, J.E., and Fiser, A. (2007) M4T: a comparative protein structure

modeling server. *Nucleic Acids Res*, **35**, W363–W368.

14. Rai, B.K., Madrid-Aliste, C.J., Fajardo, J.E., and Fiser, A. (2006) MMM: a sequence-to-structure alignment protocol. *Bioinformatics*, **22**, 2691–2692.
15. Kopp, J., Bordoli, L., Battey, J.N., Kiefer, F., and Schwede, T. (2007) Assessment of CASP7 predictions for template-based modeling targets. *Proteins*, **69 Suppl 8**, 38–56.
16. Fiser, A. and Sali, A. (2003) Modeller: generation and refinement of homology-based protein structure models. *Methods Enzymol*, **374**, 461.
17. Fernandez-Fuentes, N., Rai, B.K., Madrid-Aliste, C.J., Fajardo, J.E., and Fiser, A. (2007) Comparative protein structure modeling by combining multiple templates and optimizing sequence-to-structure alignments. *Bioinformatics*, **23**, 2558–2565.
18. Contreras-Moreira, B., Fitzjohn, P.W., Offman, M., Smith, G.R., and Bates, P.A. (2003) Novel use of a genetic algorithm for protein structure prediction: searching template and sequence alignment space. *Proteins*, **53 Suppl 6**, 424.
19. Schaffer, A.A., Aravind, L., Madden, T.L., Shavirin, S., Spouge, J.L., Wolf, Y.I., Koonin, E.V., and Altschul, S.F. (2001) Improving the accuracy of PSI-BLAST protein database searches with composition-based statistics and other refinements. *Nucleic Acids Res*, **29**, 2994.
20. Apostolico, A. and Giancarlo, R. (1998) Sequence alignment in molecular biology. *J Comput Biol*, **5**, 173.
21. Pearson, W.R. (2000) Flexible sequence similarity searching with the FASTA3 program package. *Methods Mol Biol*, **132**, 185.
22. Sauder, J.M., Arthur, J.W., and Dunbrack, R.L., Jr. (2000) Large-scale comparison of protein sequence alignment algorithms with structure alignments. *Proteins*, **40**, 6.
23. Brenner, S.E., Chothia, C., and Hubbard, T.J. (1998) Assessing sequence comparison methods with reliable structurally identified distant evolutionary relationships. *Proc Natl Acad Sci U S A*, **95**, 6073.
24. Rychlewski, L., Jaroszewski, L., Li, W., and Godzik, A. (2000) Comparison of sequence profiles. Strategies for structural predictions using sequence information. *Protein Sci*, **9**, 232.
25. Krogh, A., Brown, M., Mian, I.S., Sjolander, K., and Haussler, D. (1994) Hidden Markov models in computational biology. Applications to protein modeling. *J Mol Biol*, **235**, 1501.
26. Henikoff, J.G., Pietrokovski, S., McCallum, C.M., and Henikoff, S. (2000) Blocks-based methods for detecting protein homology. *Electrophoresis*, **21**, 1700.
27. Altschul, S.F., Madden, T.L., Schaffer, A.A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D.J. (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res*, **25**, 3389.
28. Marti-Renom, M.A., Madhusudhan, M.S., and Sali, A. (2004) Alignment of protein sequences by their profiles. *Protein Sci*, **13**, 1071.
29. Notredame, C. (2007) Recent evolutions of multiple sequence alignment algorithms. *PLoS Comput Biol*, **3**, e123.
30. Edgar, R.C. and Batzoglou, S. (2006) Multiple sequence alignment. *Curr Opin Struct Biol*, **16**, 368–373.
31. Edgar, R.C. and Sjolander, K. (2004) COACH: profile–profile alignment of protein families using hidden Markov models. *Bioinformatics*, **20**, 1309.
32. Jaroszewski, L., Rychlewski, L., Zhang, B., and Godzik, A. (1998) Fold prediction by a hierarchy of sequence, threading, and modeling methods. *Protein Sci*, **7**, 1431.
33. Jaroszewski, L., Rychlewski, L., Li, Z., Li, W., and Godzik, A. (2005) FFAS03: a server for profile–profile sequence alignments. *Nucleic Acids Res*, **33**, W284–W288.
34. Karplus, K., Barrett, C., and Hughey, R. (1998) Hidden Markov models for detecting remote protein homologies. *Bioinformatics*, **14**, 846.
35. Karchin, R., Cline, M., Mandel-Gutfreund, Y., and Karplus, K. (2003) Hidden Markov models that use predicted local structure for fold recognition: alphabets of backbone geometry. *Proteins*, **51**, 504.
36. Karplus, K., Katzman, S., Shackleford, G., Koeva, M., Draper, J., Barnes, B., Soriano, M., and Hughey, R. (2005) SAM-T04: what is new in protein-structure prediction for CASP6. *Proteins*, **61 Suppl 7**, 135–142.
37. Edgar, R.C. and Sjolander, K. (2003) SATCHMO: sequence alignment and tree construction using hidden Markov models. *Bioinformatics*, **19**, 1404.
38. John, B. and Sali, A. (2004) Detection of homologous proteins by an intermediate sequence search. *Protein Sci*, **13**, 54.

39. Moretti, S., Armougom, F., Wallace, I.M., Higgins, D.G., Jongeneel, C.V., and Notredame, C. (2007) The M-Coffee web server: a meta-method for computing multiple sequence alignments by combining alternative alignment methods. *Nucleic Acids Res*, **35**, W645–W648.
40. Pei, J., Kim, B.H., and Grishin, N.V. (2008) PROMALS3D: a tool for multiple protein sequence and structure alignments. *Nucleic Acids Res*, **36**, 2295–2300.
41. Pei, J. and Grishin, N.V. (2007) PROMALS: towards accurate multiple sequence alignments of distantly related proteins. *Bioinformatics*, **23**, 802–808.
42. Do, C.B., Mahabhashyam, M.S., Brudno, M., and Batzoglou, S. (2005) ProbCons: probabilistic consistency-based multiple sequence alignment. *Genome Res*, **15**, 330.
43. Jones, D.T. (1999) GenTHREADER: an efficient and reliable protein fold recognition method for genomic sequences. *J Mol Biol*, **287**, 797.
44. Finkelstein, A.V. and Reva, B.A. (1991) A search for the most stable folds of protein chains. *Nature*, **351**, 497.
45. Bowie, J.U., Luthy, R., and Eisenberg, D. (1991) A method to identify protein sequences that fold into a known three-dimensional structure. *Science*, **253**, 164.
46. Sippl, M.J. (1995) Knowledge-based potentials for proteins. *Curr Opin Struct Biol*, **5**, 229.
47. Shi, J., Blundell, T.L., and Mizuguchi, K. (2001) FUGUE: sequence–structure homology recognition using environment-specific substitution tables and structure-dependent gap penalties. *J Mol Biol*, **310**, 243.
48. Felsenstein, J. (1981) Evolutionary trees from DNA sequences: a maximum likelihood approach. *J Mol Evol*, **17**, 368.
49. Venclovas, C. and Margelevicius, M. (2005) Comparative modeling in CASP6 using consensus approach to template selection, sequence–structure alignment, and structure assessment. *Proteins*, **61**, 99–105.
50. Sanchez, R. and Sali, A. (1997) Evaluation of comparative protein structure modeling by MODELLER-3. *Proteins*, **1 Suppl**, 50.
51. Eisenberg, D., Luthy, R., and Bowie, J.U. (1997) VERIFY3D: assessment of protein models with three-dimensional profiles. *Methods Enzymol*, **277**, 396.
52. Wu, G., McArthur, A.G., Fiser, A., Sali, A., Sogin, M.L., and Mllerm, M. (2000) Core histones of the amitochondriate protist, Giardia lamblia. *Mol Biol Evol*, **17**, 1156.
53. Jennings, A.J., Edge, C.M., and Sternberg, M.J. (2001) An approach to improving multiple alignments of protein sequences using predicted secondary structure. *Protein Eng*, **14**, 227.
54. Blake, J.D. and Cohen, F.E. (2001) Pairwise sequence alignment below the twilight zone. *J Mol Biol*, **307**, 721.
55. Petrey, D., Xiang, Z., Tang, C.L., Xie, L., Gimpelev, M., Mitros, T., Soto, C.S., Goldsmith-Fischman, S., Kernytsky, A., Schlessinger, A., et al. (2003) Using multiple structure alignments, fast model building, and energetic analysis in fold recognition and homology modeling. *Proteins*, **53 Suppl 6**, 430.
56. Al Lazikani, B., Sheinerman, F.B., and Honig, B. (2001) Combining multiple structure and sequence alignments to improve sequence detection and alignment: application to the SH2 domains of Janus kinases. *Proc Natl Acad Sci U S A*, **98**, 14796.
57. Reddy, B.V., Li, W.W., Shindyalov, I.N., and Bourne, P.E. (2001) Conserved key amino acid positions (CKAAPs) derived from the analysis of common substructures in proteins. *Proteins*, **42**, 148.
58. Jaroszewski, L., Rychlewski, L., and Godzik, A. (2000) Improving the quality of twilight-zone alignments. *Protein Sci*, **9**, 1487.
59. Rai, B.K. and Fiser, A. (2006) Multiple mapping method: a novel approach to the sequence-to-structure alignment problem in comparative protein structure modeling. *Proteins*, **63**, 644–661.
60. Henikoff, S. and Henikoff, J.G. (1992) Amino acid substitution matrices from protein blocks. *Proc Natl Acad Sci U S A*, **89**, 10915–10919.
61. Luthy, R., McLachlan, A.D., and Eisenberg, D. (1991) Secondary structure-based profiles: use of structure-conserving scoring tables in searching protein sequence databases for structural similarities. *Proteins*, **10**, 229–239.
62. Rykunov, D. and Fiser, A. (2007) Effects of amino acid composition, finite size of proteins, and sparse statistics on distance-dependent statistical pair potentials. *Proteins*, **67**, 559–568.
63. Blundell, T.L., Sibanda, B.L., Sternberg, M.J., and Thornton, J.M. (1987) Knowledge-based prediction of protein structures and the design of novel molecules. *Nature*, **326**, 347.
64. Browne, W.J., North, A.C.T., Phillips, D.C., Brew, K., Vanaman, T.C., and Hill, R.C.

(1969) A possible three-dimensional structure of bovine lactalbumin based on that of hen's egg-white lysosyme. *J Mol Biol*, **42**, 65.

65. Greer, J. (1990) Comparative modeling methods: application to the family of the mammalian serine proteases. *Proteins*, **7**, 317.
66. Topham, C.M., McLeod, A., Eisenmenger, F., Overington, J.P., Johnson, M.S., and Blundell, T.L. (1993) Fragment ranking in modelling of protein structure. Conformationally constrained environmental amino acid substitution tables. *J Mol Biol*, **229**, 194.
67. Sutcliffe, M.J., Haneef, I., Carney, D., and Blundell, T.L. (1987) Knowledge based modelling of homologous proteins, part I: three-dimensional frameworks derived from the simultaneous superposition of multiple structures. *Protein Eng*, **1**, 377.
68. Srinivasan, N. and Blundell, T.L. (1993) An evaluation of the performance of an automated procedure for comparative modelling of protein tertiary structure. *Protein Eng*, **6**, 501.
69. Claessens, M., Van Cutsem, E., Lasters, I., and Wodak, S. (1989) Modelling the polypeptide backbone with 'spare parts' from known protein structures. *Protein Eng*, **2**, 335.
70. Holm, L. and Sander, C. (1991) Database algorithm for generating protein backbone and side-chain co-ordinates from a C alpha trace application to model building and detection of co-ordinate errors. *J Mol Biol*, **218**, 183.
71. Bruccoleri, R.E. and Karplus, M. (1990) Conformational sampling using high-temperature molecular dynamics. *Biopolymers*, **29**, 1847.
72. van Gelder, C.W., Leusen, F.J., Leunissen, J.A., and Noordik, J.H. (1994) A molecular dynamics approach for the generation of complete protein structures from limited coordinate data. *Proteins*, **18**, 174.
73. Levitt, M. (1992) Accurate modeling of protein conformation by automatic segment matching. *J Mol Biol*, **226**, 507.
74. Chinea, G., Padron, G., Hooft, R.W., Sander, C., and Vriend, G. (1995) The use of position-specific rotamers in model building by homology. *Proteins*, **23**, 415.
75. Jones, T.A. and Thirup, S. (1986) Using known substructures in protein model building and crystallography. *EMBO J*, **5**, 819.
76. Brooks, C.L., III, Bruccoleri, R.E., Olafson, B.D., States, D.J., Swaminathan, S., and Karplus, M. (1983) CHARMM: a program for macromolecular energy minimization and dynamics calculations. *J Comput Chem*, **4**, 187.
77. Sali, A. and Blundell, T.L. (1993) Comparative protein modeling by satisfaction of spatial restraints. *J Mol Biol*, **234**, 779–815.
78. Braun, W. and Go, N. (1985) Calculation of protein conformations by proton–proton distance constraints. A new efficient algorithm. *J Mol Biol*, **186**, 611.
79. Clore, G.M., Brunger, A.T., Karplus, M., and Gronenborn, A.M. (1986) Application of molecular dynamics with interproton distance restraints to three-dimensional protein structure determination. A model study of crambin. *J Mol Biol*, **191**, 523.
80. Guenther, B., Onrust, R., Sali, A., O'Donnell, M., and Kuriyan, J. (1997) Crystal structure of the ë-subunit of the clamp-loader complex of *E. coli* DNA polymerase III. *Cell*, **91**, 335.
81. Fiser, A., Filipe, S.R., and Tomasz, A. (2003) Cell wall branches, penicillin resistance and the secrets of the MurM protein. *Trends Microbiol*, **11**, 547.
82. John, B. and Sali, A. (2003) Comparative protein structure modeling by iterative alignment, model building and model assessment. *Nucleic Acids Res*, **31**, 3982.
83. Chivian, D. and Baker, D. (2006) Homology modeling using parametric alignment ensemble generation with consensus and energy-based model selection. *Nucleic Acids Res*, **34**, e112.
84. Kolinski, A. and Bujnicki, J.M. (2005) Generalized protein structure prediction based on combination of fold-recognition with de novo folding and evaluation of models. *Proteins*, **61 Suppl 7**, 84–90.
85. Terashi, G., Takeda-Shitaka, M., Kanou, K., Iwadate, M., Takaya, D., Hosoi, A., Ohta, K., and Umeyama, H. (2007) Fams-ace: a combined method to select the best model after remodeling all server models. *Proteins*, **69 Suppl 8**, 98–107.
86. Wallner, B., Larsson, P., and Elofsson, A. (2007) Pcons.net: protein structure prediction meta server. *Nucleic Acids Res*, **35**, W369–W374.
87. Ginalski, K., Elofsson, A., Fischer, D., and Rychlewski, L. (2003) 3D-Jury: a simple approach to improve protein structure predictions. *Bioinformatics*, **19**, 1015–1018.

88. Mezei, M. (1998) Chameleon sequences in the PDB. *Protein Eng*, **11**, 411.
89. Fernandez-Fuentes, N. and Fiser, A. (2006) Saturating representation of loop conformational fragments in structure databanks. *BMC Struct Biol*, **6**, 15.
90. Shenkin, P.S., Yarmush, D.L., Fine, R.M., Wang, H.J., and Levinthal, C. (1987) Predicting antibody hypervariable loop conformation. I. Ensembles of random conformations for ringlike structures. *Biopolymers*, **26**, 2053.
91. Moult, J. and James, M.N. (1986) An algorithm for determining the conformation of polypeptide segments in proteins by systematic search. *Proteins*, **1**, 146.
92. Bruccoleri, R.E. and Karplus, M. (1987) Prediction of the folding of short polypeptide segments by uniform conformational sampling. *Biopolymers*, **26**, 137.
93. Deane, C.M. and Blundell, T.L. (2001) CODA: a combined algorithm for predicting the structurally variable regions of protein models. *Protein Sci*, **10**, 599.
94. van Vlijmen, H.W. and Karplus, M. (1997) PDB-based protein loop prediction: parameters for selection and methods for optimization. *J Mol Biol*, **267**, 975.
95. de Bakker, P.I., DePristo, M.A., Burke, D.F., and Blundell, T.L. (2003) Ab initio construction of polypeptide fragments: accuracy of loop decoy discrimination by an all-atom statistical potential and the AMBER force field with the Generalized Born solvation model. *Proteins*, **51**, 21.
96. Fidelis, K., Stern, P.S., Bacon, D., and Moult, J. (1994) Comparison of systematic search and database methods for constructing segments of protein structure. *Protein Eng*, **7**, 953.
97. Du, P., Andrec, M., and Levy, R.M. (2003) Have we seen all structures corresponding to short protein fragments in the Protein Data Bank? An update. *Protein Eng*, **16**, 407.
98. Fernandez-Fuentes, N., Oliva, B., and Fiser, A. (2006) A supersecondary structure library and search algorithm for modeling loops in protein structures. *Nucleic Acids Res*, **34**, 2085–2097.
99. Michalsky, E., Goede, A., and Preissner, R. (2003) Loops in proteins (LIP) – a comprehensive loop database for homology modelling. *Protein Eng*, **16**, 979.
100. Espadaler, J., Fernandez-Fuentes, N., Hermoso, A., Querol, E., Aviles, F.X., Sternberg, M.J., and Oliva, B. (2004) ArchDB: automated protein loop classification as a tool for structural genomics. *Nucleic Acids Res*, **32 Database issue**, D185.
101. Peng, H.P. and Yang, A.S. (2007) Modeling protein loops with knowledge-based prediction of sequence–structure alignment. *Bioinformatics*, **23**, 2836–2842.
102. Fernandez-Fuentes, N., Zhai, J., and Fiser, A. (2006) ArchPRED: a template based loop structure prediction server. *Nucleic Acids Res*, **34**, W173–W176.
103. Oliva, B., Bates, P.A., Querol, E., Aviles, F.X., and Sternberg, M.J. (1997) An automated classification of the structure of protein loops. *J Mol Biol*, **266**, 814.
104. Fine, R.M., Wang, H., Shenkin, P.S., Yarmush, D.L., and Levinthal, C. (1986) Predicting antibody hypervariable loop conformations. II: minimization and molecular dynamics studies of MCPC603 from many randomly generated loop conformations. *Proteins*, **1**, 342.
105. Ring, C.S. and Cohen, F.E. (1993) Modeling protein structures: construction and their applications. *FASEB J*, **7**, 783.
106. Abagyan, R. and Totrov, M. (1994) Biased probability Monte Carlo conformational searches and electrostatic calculations for peptides and proteins. *J Mol Biol*, **235**, 983.
107. Collura, V., Higo, J., and Garnier, J. (1993) Modeling of protein loops by simulated annealing. *Protein Sci*, **2**, 1502.
108. Zheng, Q., Rosenfeld, R., Vajda, S., and DeLisi, C. (1993) Determining protein loop conformation using scaling-relaxation techniques. *Protein Sci*, **2**, 1242.
109. Koehl, P. and Delarue, M. (1995) A self consistent mean field approach to simultaneous gap closure and side-chain positioning in homology modelling. *Nat Struct Biol*, **2**, 163.
110. Samudrala, R. and Moult, J. (1998) A graph-theoretic algorithm for comparative modeling of protein structure. *J Mol Biol*, **279**, 287.
111. Fiser, A. and Sali, A. (2003) ModLoop: automated modeling of loops in protein structures. *Bioinformatics*, **19**, 2500.
112. Fiser, A., Do, R.K., and Sali, A. (2000) Modeling of loops in protein structures. *Protein Sci*, **9**, 1753.
113. Sippl, M.J. (1990) Calculation of conformational ensembles from potentials of mean force. An approach to the knowledge-based

prediction of local structures in globular proteins. *J Mol Biol*, **213**, 859.

114. Melo, F. and Feytmans, E. (1997) Novel knowledge-based mean force potential at atomic level. *J Mol Biol*, **267**, 207.
115. Fiser, A., Feig, M., Brooks, C.L., III, and Sali, A. (2002) Evolution and physics in comparative protein structure modeling. *Acc Chem Res*, **35**, 413.
116. Das, B. and Meirovitch, H. (2003) Solvation parameters for predicting the structure of surface loops in proteins: transferability and entropic effects. *Proteins*, **51**, 470.
117. Forrest, L.R. and Woolf, T.B. (2003) Discrimination of native loop conformations in membrane proteins: decoy library design and evaluation of effective energy scoring functions. *Proteins*, **52**, 492.
118. DePristo, M.A., de Bakker, P.I., Lovell, S.C., and Blundell, T.L. (2003) Ab initio construction of polypeptide fragments: efficient generation of accurate, representative ensembles. *Proteins*, **51**, 41.
119. Xiang, Z., Soto, C.S., and Honig, B. (2002) Evaluating conformational free energies: the colony energy and its application to the problem of loop prediction. *Proc Natl Acad Sci U S A*, **99**, 7432–7437.
120. Fogolari, F. and Tosatto, S.C. (2005) Application of MM/PBSA colony free energy to loop decoy discrimination: toward correlation between energy and root mean square deviation. *Protein Sci*, **14**, 889–901.
121. Soto, C.S., Fasnacht, M., Zhu, J., Forrest, L., and Honig, B. (2007) Loop modeling: sampling, filtering, and scoring. *Proteins*, **70**, 834–843.
122. Zhang, C., Liu, S., and Zhou, Y. (2004) Accurate and efficient loop selections by the DFIRE-based all-atom statistical potential. *Protein Sci*, **13**, 391–399.
123. Soto, C.S., Fasnacht, M., Zhu, J., Forrest, L., and Honig, B. (2008) Loop modeling: Sampling, filtering, and scoring. *Proteins*, **70**, 834–843.
124. Rohl, C.A., Strauss, C.E., Chivian, D., and Baker, D. (2004) Modeling structurally variable regions in homologous proteins with rosetta. *Proteins*, **55**, 656–677.
125. Jacobson, M.P., Pincus, D.L., Rapp, C.S., Day, T.J., Honig, B., Shaw, D.E., and Friesner, R.A. (2004) A hierarchical approach to all-atom protein loop prediction. *Proteins*, **55**, 351.
126. Laskowski, R.A., Moss, D.S., and Thornton, J.M. (1993) Main-chain bond lengths and bond angles in protein structures. *J Mol Biol*, **231**, 1049.
127. Hooft, R.W., Vriend, G., Sander, C., and Abola, E.E. (1996) Errors in protein structures. *Nature*, **381**, 272.
128. Sippl, M.J. (1993) Recognition of errors in three-dimensional structures of proteins. *Proteins*, **17**, 355.
129. Eramian, D., Shen, M.Y., Devos, D., Melo, F., Sali, A., and Marti-Renom, M.A. (2006) A composite score for predicting errors in protein structure models. *Protein Sci*, **15**, 1653–1666.
130. Fasnacht, M., Zhu, J., and Honig, B. (2007) Local quality assessment in homology models using statistical potentials and support vector machines. *Protein Sci*, **16**, 1557–1568.
131. Wallner, B. and Elofsson, A. (2007) Prediction of global and local model quality in CASP7 using Pcons and ProQ. *Proteins*, **69 Suppl 8**, 184–193.
132. Wallner, B. and Elofsson, A. (2005) Pcons5: combining consensus, structural evaluation and fold recognition scores. *Bioinformatics*, **21**, 4248–4254.
133. Moult, J. (2005) A decade of CASP: progress, bottlenecks and prognosis in protein structure prediction. *Curr Opin Struct Biol*, **15**, 285–289.
134. Eyrich, V.A., Marti-Renom, M.A., Przybylski, D., Madhusudhan, M.S., Fiser, A., Pazos, F., Valencia, A., Sali, A., and Rost, B. (2001) EVA: continuous automatic evaluation of protein structure prediction servers. *Bioinformatics*, **17**, 1242.
135. Bujnicki, J.M., Elofsson, A., Fischer, D., and Rychlewski, L. (2001) LiveBench-1: continuous benchmarking of protein structure prediction servers. *Protein Sci*, **10**, 352.
136. Marti-Renom, M.A., Madhusudhan, M.S., Fiser, A., Rost, B., and Sali, A. (2002) Reliability of assessment of protein structure prediction methods. *Structure (Camb.)*, **10**, 435.
137. Wallner, B. and Elofsson, A. (2005) All are not equal: a benchmark of different homology modeling programs. *Protein Sci*, **14**, 1315–1327.
138. Dalton, J.A. and Jackson, R.M. (2007) An evaluation of automated homology modelling methods at low target template sequence similarity. *Bioinformatics*, **23**, 1901–1908.
139. Baker, D. and Sali, A. (2001) Protein structure prediction and structural genomics. *Science*, **294**, 93–96.

140. Sanchez, R. and Sali, A. (1998) Large-scale protein structure modeling of the *Saccharomyces cerevisiae* genome. *Proc Natl Acad Sci U S A*, **95**, 13597.
141. Ohlendorf, D.H. (1994) Accuracy of refined protein structures. Comparison of four independently refined models of human interleukin 1 beta. *Acta Crystallogr D Biol Crystallogr*, **D50**, 808.
142. Clore, G.M., Robien, M.A., and Gronenborn, A.M. (1993) Exploring the limits of precision and accuracy of protein structures determined by nuclear magnetic resonance spectroscopy. *J Mol Biol*, **231**, 82.
143. Faber, H.R. and Matthews, B.W. (1990) A mutant T4 lysozyme displays five different crystal conformations. *Nature*, **348**, 263.
144. Fiser, A. (2008) In Ridgen, D.J. (ed.), From Protein Structure to Function with Bioinformatics. Springer, pp. 57–81.

Chapter 7

Automated Protein NMR Structure Determination in Solution

Wolfram Gronwald and Hans Robert Kalbitzer

Abstract

The main drawback of protein NMR spectroscopy today is still the extensive amount of time required for solving a single structure. The main bottleneck in this respect is the manual evaluation of the experimental spectra. A clear solution to this challenge is the development of automated methods for this purpose. At the current stage of development, this goal has been almost or in a few cases fully reached for favorable cases such as well-behaved, stably folding smaller proteins below the 25 kDa range. For larger and/or more difficult molecules, the input of a human expert is still required. However, even here, automated routines will substantially speed up the structure determination process. In this report, we will summarize recent developments in this field and especially emphasize practical aspects important for a successful automated protein structure determination in solution. An important aspect closely related to structure determination is structure validation. Therefore, we devote a section to automated approaches for this topic.

Key words: NMR, Automated structure determination, Resonance assignment, Computational

1. Introduction

During the last few years, rapid progress has been made in obtaining genomic information with the decoding of the human genome being the most prominent example. However, to fully use the decoded DNA and hence protein sequence information, it is necessary to know the spatial structures of the encoded proteins. Detailed structural information allows understanding of biological processes on an atomic level, to establish previously unknown evolutionary relationships between large protein sequence families, and to investigate intermolecular interactions such as protein–protein and protein–ligand complexes on an atomic scale. This last point is of particular importance to pharmaceutical research. In contrast to the large amount of available

David Fenyö (ed.), *Computational Biology*, Methods in Molecular Biology, vol. 673,
DOI 10.1007/978-1-60761-842-3_7,

sequence information, considerably fewer protein structures have been solved so far. Currently, about 61,000 protein structures are deposited in the protein data bank (1) (date 13.7.2010). However, a significant number of these structures stem from identical or highly homologous proteins.

The two main methods for structure determination of biological macromolecules are X-ray crystallography and NMR spectroscopy. The major advantage of X-ray crystallography is that virtually no size limit exists for the investigated molecular systems. Examples include the structures of complete ribosomes or virus particles. On the downside, only well crystallizable systems can be analyzed preventing the investigation of, for example, transient complexes. NMR spectroscopy has the benefit that analysis can be performed in solution under nearly physiological conditions and dynamic properties can be studied in detail. As long as the computational methods are not sufficiently well developed to predict an unknown structure for a particular protein sequence with high accuracy and reliability at atomic resolution, experimental methods for structure determination will play a dominant role in structural biology. These methods need to be optimized for higher efficiency to keep pace with the rapid increase of genetic information available. The only practical solution to this problem is a complete or almost complete automation of the experimental structure determination processes (for recent reviews see, e.g., Gronwald and Kalbitzer (2); Huang et al. (3); Güntert (4); Williamson and Craven (5)). In the following section, we focus on new developments related to automated protein NMR structure determination in solution. A detailed stepwise description of the mandatory steps to reach this goal will be given. It should be noted that currently several different avenues related to NMR structure determination are discussed in the literature, and it will not be possible to consider all of them in detail. Therefore, in this chapter, we will mainly concentrate on one possible solution. In this context, we will also discuss more specifically the project AUREMOL (2) developed at the University of Regensburg in cooperation with a major manufacturer of NMR instruments which is aimed to solve the problem of automated NMR structure determination of biological macromolecules.

1.1. Automated Structure Determination by Solution NMR Spectroscopy

A truly automated NMR structure determination would start with the sample preparation; all other steps including the introduction of the sample into the spectrometer, the recording and processing of spectra, the data evaluation, and the structure calculation would proceed without human interference. Such automation is still far out of reach in NMR spectroscopy. However, a semiautomated NMR structural determination of small well-behaved proteins (well soluble, globular, and uniquely folded) is nowadays, in most cases, a manageable scientific problem which

leads at the end to a safe solution. Data collection is easy to be automated for NMR, and the total recording time of a minimal NMR dataset probably can be further reduced by new developments such as projection-reconstruction techniques (6, 7). Data evaluation is the true bottleneck in NMR spectroscopy, and therefore, automation procedures should mainly concentrate on this task. Structure calculation is, in general, automatically performed. The last step is structure validation. A difficulty in NMR spectroscopy, in this respect, is that the spectra cannot be satisfactorily simulated from the structure alone, which complicates the comparison of simulated and experimental data for structure validation purposes.

2. Materials

2.1. Target Selection for NMR-Spectroscopy

A critical point determining the success of a structure determination project is suitable target selection. Usually, the first step in this regard is to screen possible candidates for properties allowing a reliable automated structure elucidation such as good solubility (preferably >1 mM but with cold probes at very high fields 200 µM can be sufficient for structure determination), sufficient stability under conditions typically used for NMR spectroscopy (at least 1 week at 298 K), negligible unspecific aggregation, a well-defined stable fold, low to moderate flexibility, and limited size (<~25 kDa). This screening also includes the definition of optimal domain boundaries in the case of large proteins. These procedures still need to be done mainly experimentally, since safe prediction of these properties is often not yet possible. One reliable avenue to screen for foldness and optimal domain boundaries of a certain protein is based on the easy-to-perform visual inspection of 2D ^{1}H–^{15}N HSQC spectra. For the investigation of a certain class of proteins independent of the species they originate from, a screen of selected proteins from several different species increased the output from typical ~50% soluble proteins to more than 90% for nonmembrane proteins (8). In summary, in the field of target selection, it can be expected that additional experimental and bioinformatic methods will be developed in the future. It is advised that this step should be performed as accurately as possible. For example, missing resonances caused by insufficient signal to noise ratios due to low solubility of the target cannot be regained in subsequent steps and will lead to poor structures.

2.2. High-Throughput Protein Production

Establishing the automated production of proteins is mainly necessary for two different reasons: (1) experimental optimization of protein properties such as solubility, foldness, domain boundaries, and minimum aggregation tendency require the simple and

fast production of small amounts of different protein varieties. (2) Automated NMR structure determination methods are dependent on mass production of proteins in mg quantities. Additionally, the proteins usually have to be ^{15}N, ^{13}C, and, for larger proteins, ^{2}H-enriched. A recent review is published by Gräslund et al. (9). As a consequence, the protein of interest in most cases will not be purified from the organism it stems from but will be obtained by other means such as by overexpression in *Escherichia coli* cells. Automated production of expression constructs for genes without introns should be straightforward, while for expression constructs of intron-containing genes, full-length cDNA clones are required. Libraries of full-length cDNA clones have been currently developed, and also, tools are available for finding a suitable cDNA library for a specific task, for example, (http://cgap.nci.nih.gov/Tissues/Tissues/LibraryFinder). For automation purposes, in most cases, it will be necessary to attach an affinity tag to the protein and to isotopically enrich by growing the bacteria in isotope-enriched minimal media or special commercial full media. Proteins with disulfide bonds or proteins that require glycosylation or other posttranslational modifications often cannot be obtained from expression in *E. coli*. In these cases, yeast expression systems such as *Pichia pastoris* (10) may be used. Another avenue for protein production is the use of cell-free expression systems (11) for in vitro translation. They have the principal advantage that interference of a toxic target protein with the cell metabolism cannot occur and that the environment during the protein expression can easily be manipulated by addition of molecular components such as protease inhibitors or chaperons (12). When isotope labeling is required, in vitro translation is extremely efficient since virtually all labeled compounds are introduced in the target protein. Also, site-specific labeling, that is often necessary for larger proteins, is possible (13). Here also, the so-called SAIL method should be mentioned where the amount of NMR active atoms is reduced to the absolute minimum by using stereospecific amino acid labeling. This method is especially interesting for larger proteins in and above the 25 kDa range (14) (see also Notes 1–3).

2.3. Optimized Strategies for Automated Spectra Recording

A complete automation of NMR structure determination is much easier when a minimal set of NMR experiments (and possibly the detailed experimental setup) is predefined by the program used. However, actually it is not known which set of experiments is optimally suited for which method, and since methodological development still continues, it may change with time.

A good overview of pulse sequences commonly used for the structure determination of biological macromolecules in solution is given by Sattler et al. and Cavanagh et al. (15, 16). When defining a minimal set of NMR experiments, it is obvious that the experiments that contain the necessary structural information are

indispensable. That is actually at least one experiment relying on dipolar couplings (NOEs or residual dipolar couplings), although in the long term, chemical shift information together with molecular modeling techniques may be sufficient (17). An additional important parameter is the complexity of the problem which mainly depends on the size of the protein under consideration and the spectral dispersion it gives rise to. Both experiments and programs have to be selected simultaneously with respect to the problem encountered. As an example, isotope enrichment with ^{2}H seems not to be necessary for small proteins but is often required for larger proteins.

Higher dimensional experiments are, in principle, useful for automated data evaluation since the main problem in automation is ambiguity that can be resolved in higher dimensions. However, going to higher dimensions substantially increases the minimal spectrometer time required. Here, projection-reconstruction experiments and sparse sampling methods may be useful. In these experiments of up to seven dimensions, the multidimensional information is reconstructed from a set of suitable 2D projections, which allows a significant reduction in measurement time (18–20).

Another avenue to resolve ambiguities in the assignment process is based on the use of amino-acid type selective experiments. A set of two-dimensional triple resonance ^{1}H–^{15}N correlation experiments is presented to achieve this goal (21–23). They are based on MUSIC pulse sequence elements that, in principle, accomplish an in-phase magnetization transfer for either XH_2 or XH_3 groups, while for other multiplicities this transfer is suppressed (X can be either ^{13}C or ^{15}N) (24) (see also Note 4).

3. Automated Top-Down NMR-Structure Determination

In the next section, we will concentrate on automated NMR data evaluation. This is a very diverse topic, and during the last years, many different strategies have been described in the literature that are summarized in several recent publications (2, 25, 26). For this book chapter, we will focus on the approach we have chosen for the program AUREMOL (2).

Bearing in mind that in many applications of multidimensional NMR-spectroscopy, the main aim is not a completely correct spectral assignment but a correct three-dimensional structure we have to ask for an optimal strategy to obtain this goal with a minimum of experiments in an automated fashion. It is apparent that we have to mainly concentrate on experiments that contain strong structural information since these are indispensable. At the most extreme (and with the rapid evolution of structure prediction in bioinformatics not unlikely) case, one could reduce the role of

NMR spectroscopy to the validation of a predicted structure without using NMR directly for structure determination. However, with the present state of the art, this seems only to be possible when structures of close homologues are readily available.

The validation of structures has two important aspects: the proof that (1) the obtained structure represents a solution consistent with all experimental data and (2) that the experimental data are sufficient to define the obtained structure as a unique solution within the limits of a predefined accuracy. For the first condition, a number of methods have already been reported, the most important one (but still far from optimal) is probably the calculation of quantities such as NMR R-factors. For practical purposes, the required quality of a structure is dependent on the specific problem to be solved. The amount of time and resources needed usually increases rapidly with the demand on quality (resolution). In addition, one would demand that the structures obtained from automated procedures should be at least as accurate as those obtained from manual data evaluation. In the so-called top-down approach that is described here, one starts from a trial structure and uses the structure information contained in the spectra to obtain iteratively improved structures and resonance assignments (Fig. 1). The trial structure may consist in the two extreme cases either of an arbitrary random structure or of the well-defined structure of a close homologue.

In the following section, we will use the AUREMOL example shown in Fig. 1 to describe the necessary steps of automated structure determination in more detail. Note that many of the details given below are also applicable to other programs for automated protein NMR structure determination.

3.1. Molecule Definition and Local and Global Databases

All relevant information about the considered biomolecule, for example, protein should be collected in a molecule-specific local database such as primary sequence information, composition of the used buffer, and physical parameters, for example, pH and temperature. A general local database provides additional information. It contains data such as the chemical structure of the amino acids, chemical shifts and their distributions, J-couplings, Karplus parameters, and temperature-dependent viscosities. Structures and sequences of homologous proteins can be loaded from nonlocal databases. Based on this information, a trial assignment and a trial structure are generated. It should be noted that for the basic algorithm one has to allow that these starting values can be far removed from the final results. For example, it must be possible to start with an extended strand as a starting structure.

3.2. Homology Modeling

Although it is possible to start with an extended strand structure, it is clear that it is advantageous to use the best structural model available (Fig. 1). In case that homologous structural information

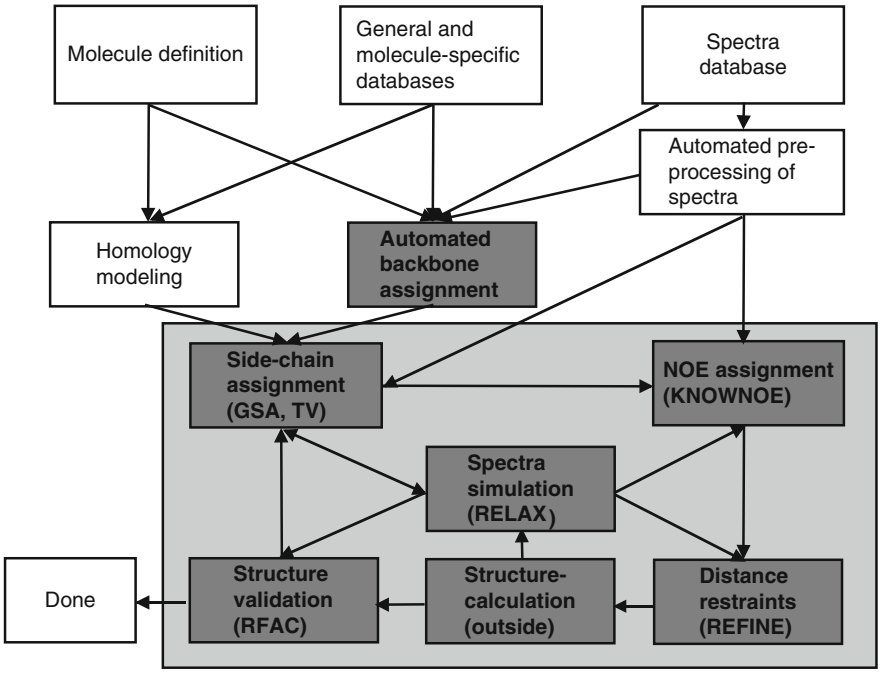


Fig. 1. Overview of the general top-down strategy used within the program AUREMOL. The most central modules are highlighted in grey.

is available, homology modeling approaches can be successfully applied for obtaining a starting model. AUREMOL contains for this purpose the PERMOL module. PERMOL (27, 28) is a restraint-based program for homology modeling of peptides and proteins. Restraints are generated from the information contained in structures of homologous template proteins. Employing the restraints generated by PERMOL, three-dimensional structures are obtained using MD programs such as CYANA (29), CNS (30), or XPLOR-NIH (31). AUREMOL provides a direct interface to these programs. In contrast to other programs, PERMOL is mainly based on the use of dihedral angle and hydrogen bond information which is optimally suited to preserve the local secondary structure, and on long-range distance restraints for the representation of the general fold.

3.3. Spectra Database

The following tasks concerning the spectra database are mostly performed outside of AUREMOL. Besides the optimization of the protein production, automated NMR data evaluation has the highest potential for substantially reducing the total time needed in automated NMR structure determination. Independent of the specific strategy used in the automated NMR structure determination, the analysis of the multidimensional NMR data comprises the following steps: (1) data processing including improvement

of spectra quality and signal enhancement, (2) pattern recognition and classification of objects, and (3) interpretation of objects and classes of objects. After recording a set of multidimensional spectra, the proper processing of the data is the first critical step, since information lost in this step cannot be regained in subsequent steps. Optimal processing of the data is especially important in automated data analysis since computer algorithms are usually not as good as human experts in distinguishing artifacts from true signals. Appropriate time domain filtering of the data is one of the most important steps performed prior to Fourier transformation. The key assumption used in these filtering methods is that resonance signals, noise, and artifacts have different time constants so that their contribution to the total detection signal varies during the acquisition period. Accordingly, a reduction in the intensity of the initial part of the time-domain signal decreases contributions from component signals, which slowly vary in the frequency domain, such as baseline rolls and tails of resonance signals. A reduction in the intensity of the final segments of time-domain signal decreases the intensity of rapidly varying components such as instrumental noise and as a consequence enhances not only the signal-to-noise ratio but also increases the line width (line broadening). In typical spectra used for automated protein structure determination such as 3D triple resonance and 3D NOESY spectra, a maximum signal-to-noise ratio is usually required to detect even weak signals, so in general, it is recommended to adjust the window (filter) functions for an optimal signal-to-noise ratio and to accept a slight increase in line width. We recommend using the 90° shifted squared sine bell or Lorentzian-to-Gaussian transformations for such applications (see also Notes 5–7 and 9).

3.3.1. Base Plane Correction in the Frequency Domain

A flat base plane is not only important for the correct integration of multidimensional NMR spectra, where base plane variation can dominate the integral, but also for peak recognition, where a threshold must be defined to sort resonance peaks from noise spikes. The fundamental assumption is that the base plane is flat in the absence of signals and that the slopes of resonance peaks are greater than those of base plane artifacts. Those regions which contain no cross peaks can either be defined by the user (32–34) or identified automatically by the program (35–37). However, the methods that are more convenient for the user are those which automatically identify base plane points. At least for spectra with similar signal-to-noise ratios, line widths, and spectral resolution, these automated routines work well and can be recommended. The simplest base plane correction method fits the baseline of each row to a cubic Lagrange polynomial where only three reference columns, which contain no signals, are defined (33). After correction of all the rows, the same method is applied to the corresponding columns. A similar method is implemented in the program

TOPSPIN, where the baseline points are automatically identified and the baseline is fitted to a polynomial of up to sixth order. Better results are obtained using the spline method (32), where an arbitrary number of cross-peak free rows and columns can be defined. The spline function then approximates the base plane between two neighboring points using a cubic polynomial function (see also Notes 8–10).

3.3.2. Removal of Spectral Artifacts

The signal of the physiological solvent, H_2O, is by far the most intense feature in ^1H NMR spectroscopy of biological macromolecules and causes spectral artifacts even when strongly attenuated by presaturation or selective excitation. Independent component analysis (ICA) or singular spectrum analysis (SSA) appear to be promising new approaches in this regard since no spectrum-dependent parameters have to be adjusted (38, 39).

3.4. Automated Pre-processing of Spectra for Information Extraction

The next critical step in any manual or automated structure determination project concerns the proper preprocessing of the various spectra for the optimal extraction of information.

3.4.1. Peak and Multiplet Recognition

Since, in general, a set of spectra is used in any automated structure determination process, it is important that all spectra have been referenced properly; this is usually achieved by using an internal standard such as DSS or TSP. For heteronuclei, it is advisable to use an indirect referencing scheme (40–42). The first and most important step in the automated spectra analysis is the recognition of resonance peaks which must be separated from the background. In principle, this step called peak picking is a straightforward procedure in multidimensional spectra since a relative maximum (or minimum) is defined by the property that all adjacent data points have a lower (or higher) intensity. However, since resonance peaks must be distinguished from the large number of noise and artifact signals, additional criteria must be defined. Approaches to automated peak picking can usually be divided into three types: (1) threshold-based methods, (2) peak shape-based methods, and (3) Bayesian approaches. (1) The simplest and most widely used criterion is the intensity threshold criterion, that is, only peaks with absolute intensities above a specific threshold are recognized as true resonance peaks (43–48). In many cases, a global threshold is not applicable for the whole spectrum; therefore, programs such as AUTOPSY (49) and AUREMOL allow the automatic calculation of local thresholds. Since the reliability of automatic assignment procedures critically depends on a high ratio of true peaks versus noise and artifact signals, optimal reduction of the number of noise and artifact contributions is mandatory. However, true signals lost in this filtering step can never be regained in

subsequent stages. A simple method for significantly reducing the number of noise and artifact peaks is the exclusion of areas from the peak search where no meaningful resonances can be expected. Such spectral areas include regions outside the spectral range of the molecule under investigation and spectral regions where resonance peaks cannot be separated from artifact peaks (e.g., near the water t_1-ridge). In programs such as AURELIA (50) and AUREMOL, these spectral regions can be defined interactively by the user. (2) Additional information can be derived from the line shape itself. With a segmentation procedure, the *n*-dimensional line widths can be determined and peaks with very small line widths (i.e., noise spikes) or very large line widths (ridges and baseline rolls) can be automatically removed (51). (3) A Bayesian approach coupled to a multivariate linear discriminant analysis of the data (52) can be used as a generally applicable method for the automated classification of multidimensional NMR peaks. The analysis relies on the assumption that different signal classes have different distributions of specific properties such as line shapes, line widths, and intensities. In addition, a nonlocal feature is included that takes into account the similarities of peak shapes in symmetry-related positions. The calculated probabilities for the different signal class memberships are realistic and reliable with a high efficiency of discriminating between peaks that are true signals and those that are not (53) (see also Notes 11–13).

3.4.2. Signal Integration

The basis for macromolecular structure determination in solution is still given by distance information from multidimensional NOE data. As a consequence, automated routines for NOE integration are required. Accurate integration of spectral cross-peaks demands a reliable definition of the cross-peak area. However, such a definition is always a compromise between requirements that the integration area be as large as possible so that a complete integration is obtained, and also, as small as possible to reduce the influence from artifacts associated with baseline rolls and tails of other peaks. A similar approach defines the peak integration area using an iterative "region-growing" algorithm (44, 51, 54), which recognizes all data points that are part of a given cross-peak; the integration can be performed based on a user-defined threshold level. In AUREMOL, this threshold is defined relative to the maximum value of the peak to ensure that the relative volumes are directly proportional to the strength of interaction. This automatic integration procedure works surprisingly well even for overlapping peaks as long as the peak maxima are separately visible and therefore recognizable by the peak picking procedure. In a different approach, peaks are fitted by a set of reference peaks defined by the user (48, 55). This approach is probably best suited in cases where peaks strongly overlap; however, it demands a careful selection of the reference peaks by the user and is therefore not applicable for fully automated applications (see also Notes 14–16).

3.5. Automated Backbone Assignment of Resonance Lines

Very different approaches have been published in the literature for this stage of the automated structure determination process. In this section, we will summarize the methods in use for spin system recognition and sequential resonance assignments that are necessary steps in most schemes proposed for automated structure determination in solution. For the methods described in this section, usually four separate steps are necessary, which can vary in the order they are applied and sometimes several steps are performed simultaneously (2). These steps are (a) grouping of resonances from one or more spectra to spin systems, (b) association of spin systems with amino acid types, (c) linking of spin systems to smaller or longer fragments, and (d) mapping of fragments obtained from step (c) to the primary sequence. The routine implemented in AUREMOL calculates sequential assignments based on information obtained from triple resonance spectra such as HNCA and CBCANH spectra. As input, a list of so called pseudo residues is used where signals from the various spectra are grouped into spin systems. Here, the user should check that the number of obtained spin systems corresponds to the number of residues in the primary sequence. A too high number indicates the presence of extensive noise and/or artifacts, whereas a too low number hints either poor spectra quality and/or insufficient preprocessing of the spectra. Next, individual spin systems are simultaneously connected to longer fragments and mapped to the primary sequence. For this purpose, a simulated annealing like algorithm that minimizes a pseudo energy is used. The used energy function contains a term that describes the matching of the individual fragments to each other and a term that facilitates the mapping to the primary sequence based on the comparison of expected and observed chemical shifts (also see Note 17).

3.6. Side Chain Assignment (GSA)

The other steps in the AUREMOL structure determination process are performed in an iterative fashion.

The basic idea for the assignment of side-chain resonances is the iterative comparison of simulated and experimental NOESY spectra to drive the assignment process. Using the preliminary structural model, NOESY spectra are simulated. Each signal is simulated with its proper line shape and volume. The shifts for the simulated signals are randomly assigned to positions where signals are present in the corresponding experimental spectra. In case that a start assignment is provided as input, the shifts are assigned according to the start assignment. As partial start assignment the resonance line assignment of the backbone atoms obtained in the previous step may be used. In the next step, the resulting simulated spectrum is compared with the experimental one with respect to line shapes and signal volumes. The degree of accordance is expressed as a probability. In the following section, a quenching protocol is applied to the simulation procedure to improve the agreement of the spectra.

A random perturbation swaps the shifts of two simulated signals and the probability of accordance is recalculated. If the new signal assignment leads to an improved agreement with the experimental data, it is accepted, otherwise declined. This method is repeated until the agreement between experiment and simulation does not improve any further. As a result, a sequential chemical shift assignment is obtained that can explain the experimental spectra with the final probability of accordance. Note that the output is a list of chemical shifts containing in an ideal case the complete side- and main-chain resonance line assignment, but it is not a fully assigned NOESY spectrum. Other well-established methods for side-chain assignments are based, for example, on the use of 3D ^{1}H–^{13}C HCCH-TOCSY spectra. However, while main-chain assignments can be obtained in a fully automated fashion in many cases, for side-chain assignments often manual intervention is still required (56). These difficulties stem on one hand from missing peaks due to incomplete TOCSY transfer and on the other hand from overlapping signals. To our knowledge, the FLYA package (25) is currently the only published software suite that has successfully fully automated this step by combining side-chain assignments, NOE-assignment, and structure calculations (also see Notes 18 and 19).

3.7. NOE Assignment (KNOWNOE)

In the previous two steps, the resonance line assignment of the backbone and side-chain resonances has been obtained. The next step is the complete NOE assignment including all structurally relevant long-range signals. For this purpose, several strategies have been published in the literature, of which two widely used methods are the r^{-6} averaging of ambiguous restraints in ARIA (57) and the network anchoring algorithm (58) implemented in CYANA (29). We will discuss here the probabilistic method KNOWNOE (59) implemented in AUREMOL. It is applicable when the sequential resonance line assignment has been fully or almost completely achieved. Structural information, when available is helpful in the assignment process especially for larger systems, but it is not required. KNOWNOE contains as a central part a knowledge driven Bayesian algorithm for solving ambiguities in the NOE assignments. These ambiguities arise mainly from chemical shift degenerancies, which allow multiple assignments of cross peaks. Statistical tables in the form of atom-pairwise volume probability distributions (VPDs) were derived from a set of 1,000 protein NMR structures. VPDs for all assignment possibilities relevant to the assignments of inter proton NOEs were calculated. Two examples are shown in Fig. 2. They basically contain expected volume distributions for a given assignment possibility. It is evident that, for example, for an intraresidual assignment another volume distribution is expected than for a long-range assignment. Therefore, they can be used together with the known cross-peak volume to solve ambiguous NOE assignments.

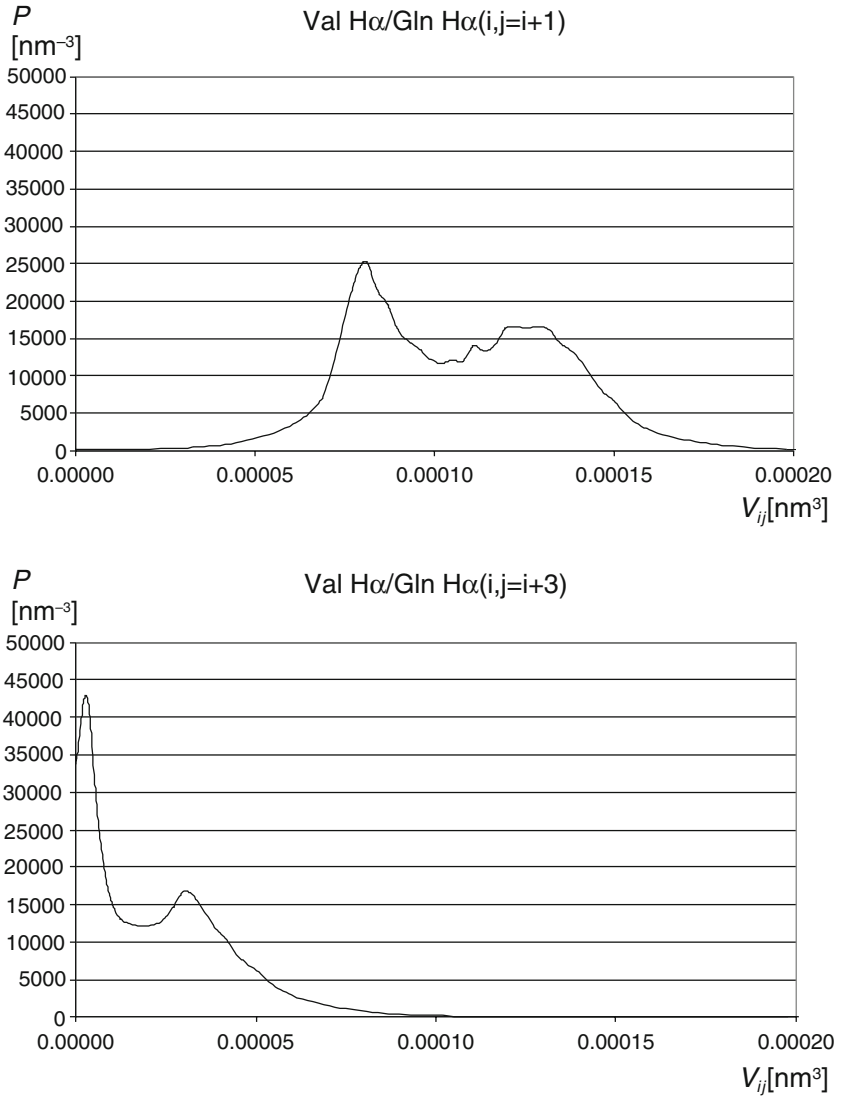


Fig. 2. Examples of volume probability distributions derived from 1,000 non homologous protein structures.

With these data for a given cross peak, with N possible assignments A_i $(i=1,\ldots, N)$, the conditional probabilities $P(A_i, a|V_0)$ that the assignment A_i determines essentially all (a-times) of the experimental cross-peak volume V_0 can be calculated. An assignment A_k with a probability $P(A_k, a|V_0)$ higher than, for example, 0.95 is transiently considered as unambiguously assigned. Note that usually not all of the signals are unambiguously assigned in the first round. With the list of unambiguously assigned peaks, a set of structures is calculated. These structures are used as input for the next cycle of iteration where a distance threshold D_{max} is dynamically reduced. Starting with a trial structure (e.g., an extended strand), all assignments of a cross

peak possible within the dimension specific chemical shift limits are considered where the corresponding atoms are separated in the trial structure by a distance $r_{ij} < D_{max}$. The aim of the distance threshold D_{max} is to reduce the number of assignment possibilities for the individual cross peaks, which in turn leads to a higher number of signals that could be unambiguously assigned. Again assignment probabilities are calculated, and a new round of structure calculations is performed. This procedure is iterated until D_{max} reaches its lower limit. The lower limit of D_{max} is usually set to 0.75 nm, in general, the maximum detection range of a NOESY spectrum plus a margin to allow for only partially fixed side-chain positions. Note that in each iteration the original unassigned peak list is used, and all previous assignments are discarded. This is done to ensure that the structure determination process does not get trapped in preliminary conformations.

In addition, KNOWNOE considers mutual information in a similar way as introduced by Herrmann et al. (2002) (58). That means putative NOE assignments that are supported by a network of neighboring assignments are treated as more probable than assignments that are isolated. Within KNOWNOE, the use of mutual information also termed network-anchoring is especially important during the first few cycles of NOE assignments and structure calculations. When no additional 3D structural information is available as it is often the case in the beginning of the structure determination process usually many NOE signals possess a high number of possible assignments within the ranges of the sequential chemical shift assignment. Using mutual information a prescreening step is performed for each ambiguous assignment and only assignment possibilities that are supported by a network of neighboring assignments are passed to the next step of KNOWNOE. Only from these preselected assignment possibilities the most probable assignment is calculated as described above. Compared with the original KNOWNOE method (59), this combination of methods usually allows for a considerably higher number of reliably assigned NOEs.

In the following, we will give an example for the iterative structure determination of the Ras-binding domain of RalGDS a small protein of 88 residues in size (60, 61). As input, the resonance line assignment for the main- and side-chain atoms was used together with a 2D ^{1}H and a 3D ^{15}N edited NOESY spectrum. In total, six iterations of assignments and structure calculations were performed, of which the first two and the last are shown below (also see Notes 20 and 21).

3.7.1 Iteration 1

See Fig. 3 and Table 1.

3.7.2 Iteration 2

See Fig. 4 and Table 2.

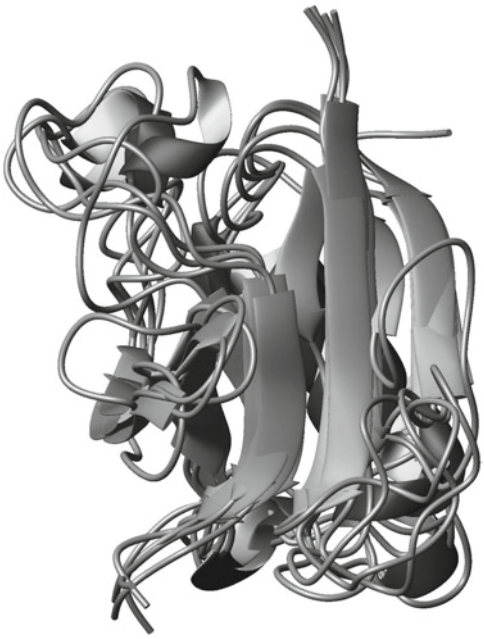


Fig. 3. Bundle of five selected structures of RalGDS after iteration 1.

Table 1
Iteration 1: Parameters used within AUREMOL/KNOWNOE (see Notes 22 and 23)

Parameter	Value	Description
Mixing time 2D	0.08 s	NOESY mixing time
Relaxation delay	1.56 s	D1 plus acquisition time
Used structure	Extended strand	Determines used 3D structure
Assign limit F1 (2D)	0.02 ppm	Allowed divergence between assignment and actual spectrum
Assign limit F2 (2D)	0.02 ppm	Allowed divergence between assignment and actual spectrum
Lower probability limit	0.95	Minimal accepted assignment probability
Distance limit	100 nm	Allowed atom separation for possible assignments in current structure
Mutual information prob limit	0.2	Minimal accepted mutual information probability
Use mutual information	Yes	Switches use of mutual information on/off

(continued)

Table 1
(continued)

Parameter	Value	Description
Assign all peaks	No	Switches additional assignment of signals on where "mutual information probability limit" and "lower probability limit" values are below thresholds
Assignments to master list	Yes	Transfers assignments to spectrum
Specify error bounds	Yes	Manual definition of error bounds
Upper bound (see Note 25)	$dist^2/8$	Calculation of upper bounds in CNS restraint file
Lower bound (see Note 25)	dist-0.165 nm	Calculation of lower bounds in CNS restraint file
Additional information used for the MD-calculation		
Parameter	**Value**	**Description**
Dihedral angle restraints	104	Number of backbone dihedral angle restraints from TALOS
h-bond restraints	52	Number of h-bond restraints (2 for each h-bond)
Results		
Parameter	**Value**	**Description**
Assigned signals (2D)	483 of 1,614	Number of signals assigned by KNOWNOE
RMSD	0.22 nm	Average RMSD of selected structures to mean structure (CA)

Fig. 4. Bundle of five selected structures of RalGDS after iteration 2.

Table 2
Iteration 2: Parameters used within AUREMOL/KNOWNOE (see Notes 22 and 23)

Parameter	Value	Description
Mixing time 2D	0.08 s	NOESY mixing time
Relaxation delay	1.56 s	D1 plus acquisition time
Used structure	Best of previous iteration	Determines used 3D structure
Assign limit F1 (2D)	0.01 ppm	Allowed divergence between assignment and actual spectrum
Assign limit F2 (2D)	0.01 ppm	Allowed divergence between assignment and actual spectrum
Lower probability limit	0.95	Minimal accepted assignment probability
Distance limit	1.5 nm	Allowed atom separation for possible assignments in current structure
Mutual information prob limit	0.1	Minimal accepted mutual information probability
Use mutual information	Yes	Switches use of mutual information on/off
Assign all peaks	No	Switches additional assignment of signals on where "mutual information prob limit" and "lower probability limit" values are below thresholds
Assignments to master list	Yes	Transfers assignments to spectrum
Specify error bounds	Yes	Manual definition of error bounds
Upper bound	$dist^2/8$	Calculation of upper bounds in CNS restraint file
Lower bound	dist-0.165 nm	Calculation of lower bounds in CNS restraint file
Additional information used for the MD-calculation		
As before		
Results		
Parameter	**Value**	**Description**
Assigned signals (2D)	964 of 1,614	Number of signals assigned by KNOWNOE
RMSD	0.12 nm	Average RMSD of selected structures to mean structure (CA)

3.7.3. Iterations 3–5 See Table 3.

3.7.4. Iteration 6 See Fig. 5 and Table 4.

Table 3
Iterations 3 to 5: Parameters used within AUREMOL/ KNOWNOE (see Notes 22 and 23)

Same parameters as before but upper distance limits of 1.25, 1.00, and 0.75 nm were used, respectively.
Additional information used for the MD-calculation
As before

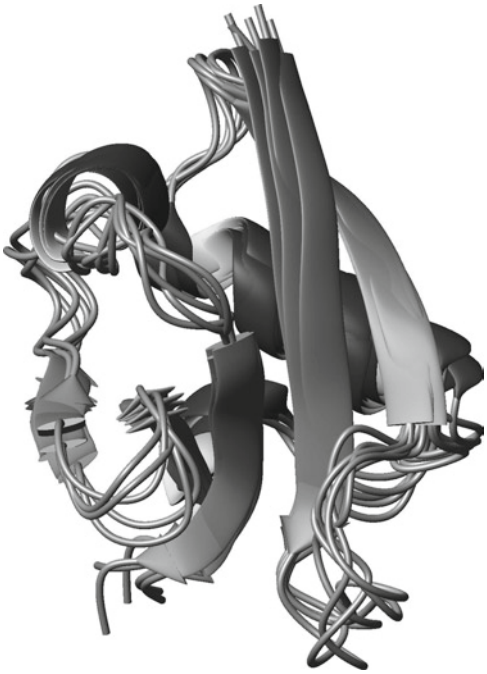


Fig. 5. Bundle of five selected structures of RalGDS after iteration 6.

A detailed description of the various parameters is given in the AUREMOL manual (see also Notes 22–26).

3.8. NOESY Spectra Simulation Using the Full Relaxation Matrix Algorithm

As can be seen in Fig. 1, the simulation of spectra is a central part of AUREMOL that is used in many of its various functions such as side-chain resonance line assignment, restraint generation, and structure validation. For this purpose, the module RELAX was incorporated in AUREMOL. RELAX (62–64), a program for the back-calculation of NOESY spectra is based on the complete relaxation matrix formalism. It differs from similar programs (65–71) in features such as the availability of a large number of motional models that, in principle, can be applied individually for all pairs

Table 4
Iteration 6: Parameters used within AUREMOL/KNOWNOE (see Notes 22 and 23)

Parameter	Value	Description
Mixing time 2D	0.08 s	NOESY mixing time
Relaxation delay	1.56 s	D1 plus acquisition time
Used structure	Best five of previous iteration	Determines used 3D structure
Assign limit F1 (2D)	0.015 ppm	Allowed divergence between assignment and actual spectrum
Assign limit F2 (2D)	0.015 ppm	Allowed divergence between assignment and actual spectrum
Assign limit F1 (3D)	0.10 ppm	Allowed divergence between assignment and actual spectrum
Assign limit F2 (3D)	0.50 ppm	Allowed divergence between assignment and actual spectrum
Assign limit F3 (3D)	0.02 ppm	Allowed divergence between assignment and actual spectrum
Lower probability limit	0.95	Minimal accepted assignment probability
Distance limit	0.75 nm	Allowed atom separation for possible assignments in current structure
Mutual information prob limit	0.01	Minimal accepted mutual information probability
Use mutual information	Yes	Switches use of mutual information on/off
Assign all peaks	Yes	Switches additional assignment of signals on where "mutual information prob limit" and "lower probability limit" values are below thresholds
Assignments to master list	Yes	Transfers assignments to spectrum
Specify error bounds	Yes	Manual definition of error bounds
Upper bound	Minimal error + 0.05 nm	Calculation of upper bounds in CNS restraint file
Lower bound	Minimal error + 0.05 nm	Calculation of lower bounds in CNS restraint file

(continued)

Table 4 (continued)

Parameter	Value	Description
Additional information used for the MD-calculation		
As before		
Results		
Parameter	Value	Description
Assigned signals (2D)	1,442 of 1,614	Number of signals assigned by KNOWNOE
Assigned signals (3D ^{15}N edited)	392 of 607	Number of signals assigned by KNOWNOE
RMSD	0.06 nm	Average RMSD of selected structures to mean structure (CA)

of spins, and that it also includes relaxation by chemical shift anisotropy, calculates individual T_2 values, and it allows the inclusion of *J*-coupling patterns in the simulation. It facilitates the simulation of ^{1}H 2D NOESY and ^{15}N or ^{13}C-edited 3D NOESY-HSQC spectra. The 3D NOESY–HSQC experiment is basically a concatenation of a homonuclear ^{1}H-NOESY and a heteronuclear HSQC-experiment. In a NOESY experiment, the evolution of the deviation of longitudinal magnetization from thermal equilibrium ΔM_z is described by the generalized Solomon equation

$$\frac{\mathrm{d}}{\mathrm{d}t}\Delta M_z(t) = -\mathbf{D}\Delta M_z(t) \tag{1}$$

The dynamics matrix **D** that governs the time evolution of the cross-peak intensities in a 2D-NOESY experiment is given by

$$\mathbf{D} = \mathbf{R} + \mathbf{K} \tag{2}$$

K is the kinetic matrix that describes chemical and/or conformational exchange (72), while **R** is the relaxation matrix (73–75). In the current version of RELAX, the effects of chemical exchange are neglected and the solution of Eq. 3 simplifies to

$$\Delta M_z(t) = \Delta M_z(0)\exp(-t\cdot\mathbf{R}) \tag{3}$$

wherein $\Delta M_z(0)$ is the deviation of the longitudinal magnetization from thermal equilibrium at time zero, i.e., directly after the last pulse of the NOESY experiment. For dipolar homo- or heteronuclear relaxation and spin $I = 1/2$, the rates of autorelaxation R_{ii} and the cross-relaxation R_{ij} between two spins *i* and *j* are given by

$$R_{ii} = \sum_{j\neq i} q_{ij}\left[J_{ij}^{0}(\omega_i - \omega_j) + 3J_{ij}^{1}(\omega_i) + 6J_{ij}^{2}(\omega_i + \omega_j)\right] \tag{4}$$

and

$$R_{ij} = q_{ij}\left[6J^2_{ij}(\omega_i + \omega_j) - J^0_{ij}(\omega_i - \omega_j)\right] \quad (5)$$

respectively. With J^n_{ij} $(n = 0, 1, 2)$ being the spectral densities for n-quantum transitions characterizing the motion of a vector connecting spin i and j relative to the $\mathbf{B}_0$-field, the dipolar coupling constants q_{ij} are given by

$$q_{ij} = (1/10)\gamma_i^2\gamma_j^2h^2(\mu_0/4\pi)^2 \quad (6)$$

where γ_i and γ_j are the gyromagnetic ratios of spin i and j, respectively. Within RELAX, different motional models can be used to describe internal and external motions of the molecule. Examples include a slow jump model to describe slow aromatic ring flips, a fast jump model to describe fast rotating methyl groups, a rigid model in cases where isotropic overall tumbling is assumed, and the Lipari model free approach for internal motions not easily described by a simple motional model.

Besides dipolar interactions RELAX also takes contributions from chemical shift anisotropy in the relaxation matrix into account. In addition to the calculation of accurate volumes/intensities also individual line shapes and line widths are calculated for each signal. For the calculation of line shapes, appropriate coupling constants are estimated from the trial structure. Figure 6 shows an example for a simulated 2D NOESY spectrum.

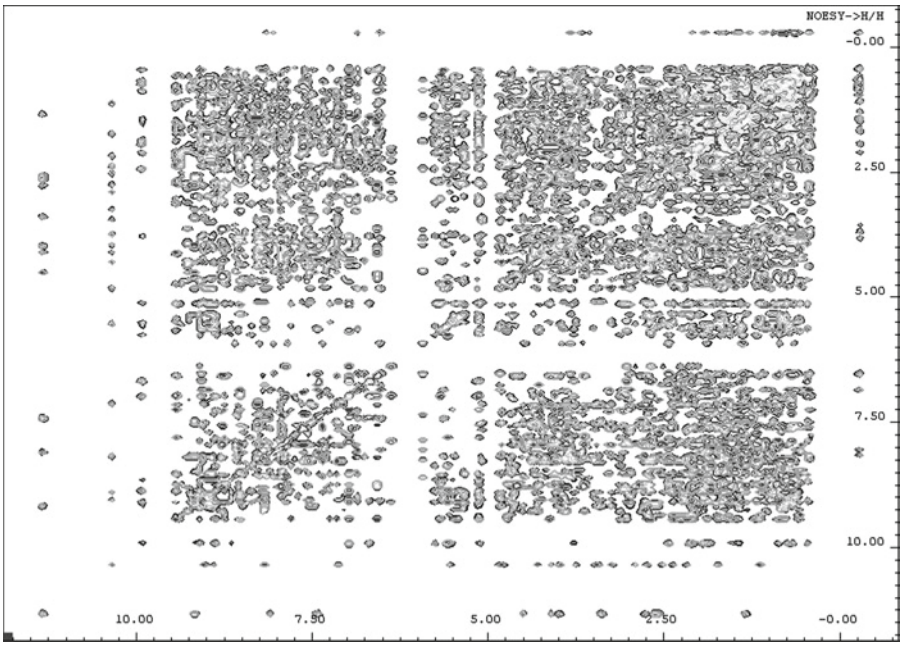


Fig. 6. Example for a simulated 2D 1H NOESY spectrum of the medium-sized protein HPr.

Simulation was performed with proper calculation of line-widths and line-shapes.

The application of RELAX requires the following settings (see also Note 26):

1. Define the pdb-file for a three-dimensional structure or a set of three-dimensional structures.
2. Define the type of spectrum, that is, 2D-NOESY or 3D-NOESY-HSQC spectrum.
3. Define the spectral parameters, that is, frequencies, spectral ranges, digital resolution, repetition time, mixing time.
4. Define the relaxation parameters, that is, the global rotational correlation time τ_c, local correlation times τ_e, type of interaction (chemical shift anisotropy on/off), motional models, order parameters S^2. The rotational correlation time can also be predicted by AUREMOL from the structure.
5. Define the spectrum simulation parameters, mainly J-couplings (on/off) and additional line broadening.

3.9. Calculation of Distance Restraints and Distance Errors with the Full Relaxation Matrix Formalism

One important task in the structure determination process is the conversion of the NOE volume information into distance restraints with appropriate error bounds. In the simplest case, one can classify the NOEs into distance classes such as short, medium, and long. A bit more realistic is the use of the so-called two spin approximation with $r = \alpha \cdot V^{-1/6}$ where α is a user-defined scaling factor to take varying instrumental and experimental factors into account. Here, often a fixed percentage of the distance is given as lower and upper error bound. More precise is the full relaxation matrix analysis for this purpose (67, 76) as it is, for example, implemented in the AUREMOL module RELAX/REFINE (to be published). Here also, spin diffusion effects are taken into account. Based on the input structure and motional models, the relaxation matrix is set up for the molecule of interest. In the next step, the relaxation matrix is iteratively refined. The rates σ_{ij} of step $n+1$ are calculated from the rates of the previous one, $\sigma_{ij}(n)$ by

$$\sigma_{ij}(n+1) = \sigma_{ij}(n) \frac{\ln cA_{ij}(\exp)}{\ln A_{ij}(n, \mathrm{sim})} \tag{7}$$

with an arbitrary scaling factor c to take into account unknown experimental and instrumental factors, the experimental cross-peak volumes $A_{ij}(\exp)$, and the corresponding simulated volumes $A_{ij}(n, \mathrm{sim})$ of step n. After the autorelaxation rates have been adjusted, new NOEs are calculated from the refined relaxation matrix and the next iteration step is performed. After convergence, accurate distances are obtained from the refined relaxation matrix. Minimal error bounds are obtained from an analysis of the experimental volume error that was determined in the peak integration step.

For using the routine REFINE the following parameters have to be set:

1. Set parameters as for RELAX.
2. Set parameters for the error calculation. Here, one can choose whether minimal error bounds plus an optional user error should be selected (recommended for final cycles of automated structure calculation, or fully user defined error bounds, such as a certain percentage of the restraint distance, should be used).

3.10. Structure Calculation

AUREMOL provides an interface for performing structure calculations. Necessary input files are automatically created by AUREMOL. The structure calculations itself are done by external programs such as CYANA or CNS, whereas the analysis of the resulting structures is again performed within AUREMOL.

3.11. Structure Validation by NMR-R-Factor Calculations

One of the most important steps in any structure determination project is the validation of the final and/or intermediate structures. Often the quality of an NMR structure is mainly judged by factors such as RMSD values or the quality of the Ramachandran plot. However, these methods do not provide a direct measure of how well the calculated structures fit the experimental data. Therefore, we have implemented the program RFAC (61, 77) in AUREMOL, which automatically calculates R-factors for protein NMR structures to provide such a measure. The automated R-factor analysis envisaged here consists, in principle, of two separate parts: (1) a comparison of the experimental NOESY spectrum with the NOESY spectrum back-calculated from a given structure, and (2) the calculation of the R-factor(s) from the data. In the first part, the NOESY spectrum has to be calculated from the trial structure or a bundle of trial structures using the resonance line assignments of the side- and main-chain atoms. For the algorithm to work properly, these assignments have to be complete or almost complete. In our implementation, we use the full relaxation matrix approach of the AUREMOL module RELAX to obtain accurate simulated peaks defined by their positions, intensities, and line shapes. The corresponding experimental NOESY spectrum is as described above automatically peak picked and integrated in the preprocessing stage of AUREMOL. In addition, the probabilities p_i of the peaks i to be true NMR signals and not noise or artifact peaks are also calculated according to Bayes' theorem and are used as weighting factors during the calculation of the R-factors. For the purpose of R-factor calculation, the experimental data are automatically assigned based on the corresponding simulated spectrum and the sequential resonance line assignment. Note that in difference to KNOWNOE only assignments are made that could be expected from the trial structure. The AUREMOL routine SHIFTOPT (78) is used in this process

to optimally adapt the chemical shift values obtained from the general sequential resonance assignment to the actual experimental data. The assigned experimental and simulated spectra are fed into the programs RFAC or in the case of 3D spectra RFAC-3D. The presence of noise and artifact signals in the experimental spectrum can be further reduced by applying a lattice algorithm. In this algorithm, only peaks are taken into account where at least one back-calculated peak in each dimension can be found within user-defined search radii, for example, 0.01 ppm for 2D spectra. For resonances that are not stereospecifically assigned, the solution that fits best (gives the smallest R-factor) is selected. In this context, one should note that for each atom at least the diagonal peak is back calculated. Where more than one back-calculated peak is assigned to a single experimental peak, the mean volume of the corresponding back-calculated peaks is estimated before the comparison is done, while the volume of the experimental peak is divided by the number of corresponding back-calculated peaks. For the calculation of R-factors, several equations have been developed. RFAC and RFAC-3D use mainly the one shown below.

$$R_{\mathrm{PWAUR}}(\alpha) = \sqrt{\frac{\sum_{i \in A} (sf_\alpha V^\alpha_{\mathrm{exp},i} - V^\alpha_{\mathrm{calc},i})^2 p^2_{\mathrm{exp},i} + \sum_{i \in u} (sf_\alpha V^\alpha_{\mathrm{exp},i} - V^\alpha_{\mathrm{noise}})^2 p^2_{\mathrm{exp},i}}{\sum_{i \in A} sf^2_\alpha V^{2\alpha}_{\mathrm{exp},i} p^2_{\mathrm{exp},i} + \sum_{i \in u} (sf_\alpha V^\alpha_{\mathrm{exp},i} - V^\alpha_{\mathrm{noise}})^2 p^2_{\mathrm{exp},i}}} \tag{8}$$

The PWAUR R-factor (probability weighted assigned and unassigned resonances based R-factor) takes both the assigned (A-list) and unassigned (U-list) experimental signals into account. Here, sf_α is a global scaling factor to facilitate the proper scaling between experimental and simulated spectra. $V^\alpha{}_{\mathrm{noise}}$ is a standard noise volume to substitute for missing simulated signals, which are not available for the unassigned experimental signals. As mentioned above, the p_i values describe the probability that an experimental peak is a true signal. The parameter α is usually set to $-1/6$ to ensure that the R-factor is not dominated by the largest signals. For an ideal structure and a perfect spectrum, the R-factor approaches a value of zero, while for erroneous structures increased values are expected.

The following steps have to be performed:

1. Carefully prepare the peak list. Be sure that the peak picking threshold is low enough to retain the weak long-range peaks. Set the probability cutoff at a value that the number of peaks is smaller than twice the number of expected cross peaks. Exclude areas where severe overlap of resonances can be expected, that is, in homonuclear 2D-NOESY spectra usually the upfield range between 0.3 and 4 ppm.

2. Before starting the calculation be sure that the chemical shift table corresponds closely to the conditions used for the recording of the NOESY spectrum. Apply the chemical shift optimization routine AUREMOL-SHIFTOPT (78) that adapts the shift table to the actual spectrum.
3. Set the parameters of the NOESY-back calculation routine correctly. Especially give the correct mixing time, repetition time, and ^{1}H-frequency. Improved results are obtained when the correct motional models and motional parameters like external (global) and internal correlation times, order parameters etc., are specified. Since usually experimentally derived order parameters are not available for side-chain atoms, RELAX provides for these average values obtained from the literature.
4. Select an NMR R-factor most suitable for your actual problem.
5. In case that several different experimental NOESY spectra are available, it is advised to calculate an average R-factor (61).

4. Notes

The following notes summarize important points that should be considered in the automated structure determination process. The sections where a detailed description of these subjects can be found are given in parentheses.

1. Carefully optimize your protein sample and the experimental conditions (also see Subheading 2.2).
2. Do not start acquisition of multidimensional spectra and data evaluation before the system is really optimized. Experience shows that after a necessary improvement of the sample conditions, the results obtained earlier are discarded (also see Subheading 2.2).
3. For automated procedures, >90% of all expected resonances should be visible (also see Subheading 2.2).
4. When automated methods are to be used, perform the set of experiments required by the specific software (also see Subheading 2.3).
5. For the processing of the experimental spectra, the use of an appropriate filter is important; as a rule of thumb, a filter that induces an additional line broadening of approximately 30% gives a clear improvement of the signal-to-noise ratio without significantly deteriorating the resolution (also see Subheading 3.3).
6. Be aware when filtering FIDs in the processing step that the filter characteristics of the used filter depend often on the digital

resolution (number of data points). This is the case with the cosine filter as well as the Lorentzian-to-Gaussian filter (here as an artifact resulting from the definitions provided by TOPSPIN) (also see Subheading 3.3).

7. Application of an unshifted sine filter removes the volume information (the volume of all cross peaks is zero) (also see Subheading 3.3).
8. Do a careful baseline correction of all spectra before starting with the data evaluation (also see Subheading 3.3.1).
9. Apply the different processing methods that your software provides sequentially, e. g., a back prediction of the first data points in the time domain, followed by appropriate filtering and Fourier transform of the time domain data and a polynomial baseline correction in the frequency domain (also see Subheadings 3.3 and 3.3.1).
10. Be careful that some of the different processing functions do not work properly with oversampled data that have a different time domain data structure (also see Subheadings 3.3 and 3.3.1).
11. Most carefully prepare your peak list since the quality of the peak list is the most important single factor influencing the outcome of any automated procedure (also see Subheadings 3.3.2, 3.4, and 3.4.1).
12. Do not remove significant weak peaks by a too high peak picking threshold (in AUREMOL the threshold is set automatically) (also see Subheadings 3.4 and 3.4.1).
13. Bayesian methods are rather powerful to discriminate between true signals and noise and artifacts, but they can only decide on the basis of the information available. Important information is the number of expected cross peaks that depends on the pulse sequence and the sample composition. Choose the final probability cutoff such that the number of experimental cross-peaks corresponds to the expectation. A factor of two is still acceptable by many routines (also see Subheadings 3.4 and 3.4.1).
14. Try to understand the principles of the integration software used (also see Subheading 3.4.2).
15. The iterative segmentation software that facilitates the signal integration of AUREMOL requires the definition of a maximum integration area. This has to be defined large enough, otherwise the integrals will be erroneous (also see Subheading 3.4.2).
16. For an efficient computing such a parameter is also (inherently) present in many other routines (read the manual!) (also see Subheading 3.4.2).

17. For all sequential assignment methods, the obtained result strongly depends on the quality of the peak lists (see above), since artifact peaks and missing peaks lead to wrong or ambiguous connectivities. In most cases, connectivity information solely based on Cα's is not sufficient for obtaining unambiguous assignments. Therefore, it is highly recommended to additionally provide information for the Cβ's as well. In case that a nonexhaustive routine like simulated annealing is used, it is generally recommended to perform the routines several times and to analyze the results for discrepancies between the different runs (also see Subheading 3.5).
18. The method that has been implemented in AUREMOL for side-chain assignment works directly on the NOESY spectra; therefore, the preparation of peak lists is not critical (also see Subheading 3.6).
19. Since for the side-chain assignments the backbone assignments are a strong source of information (when available), be careful that the chemical shifts given fit optimally to the NOESY spectra used. A typical effect that should be corrected for is the shift introduced by using TROSY methods (also see Subheading 3.6).
20. For automated NOE assignment, carefully prepare the peak list. Be sure that the peak picking threshold is low enough to retain the weak long range peaks. Set the probability cutoff at a value that the number of peaks is smaller than twice the number of expected cross peaks (also see Subheading 3.7).
21. Before starting the automated NOE assignment, be sure that the chemical shift table corresponds closely to the conditions used for the recording of the NOESY-spectrum. Apply the chemical shift optimization routine AUREMOL-SHIFTOPT (78) that adapts the shift table to the actual spectrum (also see Subheading 3.7).
22. Adapt the other parameters (especially the search distance range) to the iteration cycle (see also Subheadings 3.7.1–3.7.4).
23. For many of the parameters, default values are recommended by the program. For an additional detailed description of the various parameters, please refer to the AUREMOL manual (see also Subheadings 3.7.1–3.7.4).
24. Note that upper and lower bounds are added to and subtracted from the distance *dist* that was obtained from the individual cross-peak volume. Relatively large upper bounds were selected for the first cycles of structure calculations to ensure that the influence of erroneous assignments is limited. Upper bound definition was taken from Linge et al., 2001 (79).

Lower bounds were selected in a way that the minimal distance between two protons corresponds to the sum of their van der Waals radii (see also Subheadings 3.7.1–3.7.4).

25. Please note that for the last iteration (iteration 6) the option "assign all peaks" has been switched on, which leads to an increased number of NOE assignments (should only be done in the last iteration), since still ambiguous NOEs are assigned to the solution that correspond to the smallest distance in the trial structure, since this contribution will explain most of the crosspeak volume. It is clear that for truly overlapping signals the minor signal component will be neglected, which, in turn, will lead to an underestimation of the resulting restraint distance of the major component. However, in most cases, this deviation is rather small and within the measurement error. For the relaxation matrix-based restraint generation (see below), the error bounds were now calculated based on the local noise levels plus an additional error of 0.05 nm. This additional error was obtained by analyzing the obtained distance variations due to an assumed uncertainty 0.15 in the main and side-chain order parameters. The obtained structures were further refined in explicit solvent following the protocol by Linge et al., 2003 (80) (see also Subheadings 3.7.1–3.7.4).
26. For the simulation of NOESY spectra, make sure that the corresponding parameters are set correctly. Especially give the correct mixing time, repetition time and ^{1}H-frequency (see also Subheading 3.8).

5. Conclusion

In the last sections, we have seen that automated protein structure determination in solution at least for smaller well-behaved proteins is a feasible task. As a consequence, computer-automated determination of these structures will continue to grow in importance. However, for difficult test cases such as large biopolymers, aid from human experts is still required. Typically, in macromolecular NMR, the information content of the NMR spectra alone is often not sufficient for a complete three-dimensional structure determination. Signal-to-noise ratios are necessarily limited due to the limited solubility of most biopolymers. Furthermore, superpositions of resonance lines often lead to interpretational ambiguities. Therefore, it is still of importance that any computer program used for automated structure determination permits intervention at any step in the analysis to allow the inclusion of structural and spectroscopic information from other sources and

to supervise various aspects of the validation process. Especially for the structure improvement by the inclusion of data from other sources, we have recently developed the AUREMOL-ISIC (81) algorithm that allows the reliable combination of NMR and X-ray data.

In addition, the constant development of new experimental multidimensional NMR techniques requires that new strategies for automated analysis need to be implemented in existing computer programs. Within this chapter, we have summarized some of the main points required for automated protein structure determination in solution. It is clear, however, that due to space limitations not all possible strategies could be discussed.

Acknowledgments

This work was supported by the Bavarian Genomic Network BayGene (W. G.) and the European Union (H. R. K.).

References

1. Westbrook, J., Feng, Z., Chen, L., Yang, H., Berman, H. M. (2003) The Protein Data Bank and Structural Genomics. *Nucleic Acids Res. 31*, 489–491.
2. Gronwald, W., Kalbitzer, H. R. (2004) Automated Structure Determination of Proteins by NMR Spectroscopy. *Prog. NMR Spectrosc. 44*, 33–96.
3. Huang, Y. P., Moseley, H. N., Baran, M. C., Arrowsmith, C., Powers, R., Tejero, R., Szyperski, T., Montelione, G. T. (2005) An Integrated Platform for Automated Analysis of Protein NMR Structures. *Methods Enzymol. 394*, 111–141.
4. Güntert, P. (2009) Automated Structure Determination from NMR Spectra. *Eur. Biophys. J. 38*, 129–143.
5. Williamson, M. P., Craven, C. J. (2009) Automated Protein Structure Calculation from NMR Data. *J. Biomol. NMR 43*, 131–143.
6. Kupce, E., Freeman, R. (2004) Projection-Reconstruction Technique for Speeding Up Multidimensional NMR Spectroscopy. *J. Am. Chem. Soc. 126*, 6429–6440.
7. Hiller, S., Wasmer, S., Wider, G., Wüthrich, K. (2007) Sequence-Specific Resonance Assignment of Soluble Nonglobular Proteins by 7D APSY-NMR Spectroscopy. *J. Am. Chem. Soc. 129*, 10823–10828.
8. Savchenko, A., Yee, A., Khachatryan, A., Skarina, T., Evdokimova, E., Pavolva, M., Semesi, A., Northey, J., Beasley, S., Lan, N., Das, R., Gerstein, M., Arrowsmith, C. H., Edwards, A. M. (2003) Strategies for Structural Proteonomics of Prokaryotes: Quantifying the Advantages of Studying Orthologous Proteins and of Using Both NMR and X-Ray Crystallography Approaches. *Proteins 50*, 392–399.
9. Graslund, S., Nordlund, P., Weigelt, J., Hallberg, B. M., Bray, J., Gileadi, O., Knapp, S., Oppermann, U., Arrowsmith, C., Hui, R., Ming, J., dhe-Paganon, S., Park, H. W., Savchenko, A., Yee, A., Edwards, A., Vincentelli, R., Cambillau, C., Kim, R., Kim, S. H., Rao, Z., Shi, Y., Terwilliger, T. C., Kim, C. Y., Hung, L. W., Waldo, G. S., Peleg, Y., Albeck, S., Unger, T., Dym, O., Prilusky, J., Sussman, J. L., Stevens, R. C., Lesley, S. A., Wilson, I. A., Joachimiak, A., Collart, F., Dementieva, I., Donnelly, M. I., Eschenfeldt, W. H., Kim, Y., Stols, L., Wu, R., Zhou, M., Burley, S. K., Emtage, J. S., Sauder, J. M., Thompson, D., Bain, K., Luz, J., Gheyi, T., Zhang, F., Atwell, S., Almo, S. C., Bonanno, J. B., Fiser, A., Swaminathan, S., Studier, F. W., Chance, M. R., Sali, A., Acton, T. B., Xiao, R., Zhao, L., Ma, L. C., Hunt, J. F., Tong, L., Cunningham, K., Inouye, M., Anderson, S., Janjua, H., Shastry, R., Ho, C. K., Wang, D.,

Wang, H., Jiang, M., Montelione, G. T., Stuart, D. I., Owens, R. J., Daenke, S., Schutz, A., Heinemann, U., Yokoyama, S., Bussow, K., Gunsalus, K. C. (2008) Protein Production and Purification. *Nat. Methods 5*, 135–146.

10. Wood, M. J., Komives, E. A. (1999) Production of Large Quantities of Isotopically Labeled Protein in Pichia pastoris by Fermentation. *J. Biomol. NMR 13*, 149–159.
11. Nathans, D., Notani, G., Schwartz, J. H., Zinder, N. D. (1962) Biosynthesis of the Coat Protein of Coliphage f2 by E. coli Extracts. *Proc. Natl. Acad. Sci. U. S. A. 48*, 1424–1431.
12. Edwards, A. M., Arrowsmith, C. H., Christendat, D., Dharamsi, A., Friesen, J. D., Greenblatt, J. F., Vedadi, M. (2000) Protein Production: Feeding the Crystallographers and NMR spectroscopists. *Nat. Struc. Biol. 7*, 970–972.
13. Noren, C. J., Anthony-Cahill, S. J., Griffith, M. C., Schultz, P. G. (1989) A General Method for Site-Specific Incorporation of Unnatural Amino Acids into Proteins. *Science 244*, 182–189.
14. Kainosho, M., Torizawa, T., Iwashita, Y., Terauchi, T., Mei, O. A., Guntert, P. (2006) Optimal Isotope Labelling for NMR Protein Structure Determinations. *Nature 440* (7080), 52–57.
15. Sattler, M., Schleucher, J., Griesinger, C. (1999) Heteronuclear Multidimensional NMR Experiments for the Structure Determination of Proteins in Solution Employing Pulsed Field Gradients. *Prog. NMR Spectrosc. 34*, 93–158.
16. Cavanagh, J., Fairbrother, W. J., Palmer III, A. G., Rance, M., Skelton, N. J. (2007) *Protein NMR Spectroscopy.* Academic Press: New York.
17. Wishart, D. S., Case, D. A. (2003) Use of Chemical Shifts in Macromolecular Structure Determination. *Methods Enzymol. 338*, 3–34.
18. Szyperski, T., Banecki, B., Glaser, R. W. (1998) Sequential Resonance Assignment of Medium-Sized $^{15}N/^{13}C$-Labeled Proteins with Projected 4D Triple Resonance Experiments. *J. Biomol. NMR 11*, 387–405.
19. Xia, Y., Arrowsmith, C. H., Szyperski, T. (2002) Novel Projected 4D Triple Resonance Experiments for Polypeptide Backbone Chemical Shift Assignment. *J. Biomol. NMR 24*, 41–50.
20. Szyperski, T., Yeh, D. C., Sukumaran, D. K., Moseley, H. N., Montelione, G. T. (2002) Reduced-Dimensionality NMR Spectroscopy for High-Throughput Protein Resonance Assignment. *Proc. Natl. Acad. Sci. U. S. A 99*, 8009–8014.
21. Schubert, M., Smalla, M., Schmieder, P., Oschkinat, H. (1999) MUSIC in Triple-Resonance Experiments: Amino Acid Type-Selective ^{1}H-^{15}N Correlations. *J. Magn. Reson. 141*, 34–43.
22. Schubert, M., Oschkinat, H., Schmieder, P. (2001) MUSIC, Selective Pulses, and Tuned Delays: Amino Acid Type-Selective (1)H-(15) N Correlations, II. *J. Magn Reson. 148*, 61–72.
23. Schubert, M., Oschkinat, H., Schmieder, P. (2001) MUSIC and Aromatic Residues: Amino Acid Type-Selective (1)H-(15)N Correlations, III. *J. Magn. Reson. 153*, 186–192.
24. Schmieder, P., Leidert, M., Kelly, M., Oschkinat, H. (1998) Multiplicity-Selective Coherence Transfer Steps for the Design of Amino Acid-Selective Experiments-A Triple-Resonance Experiment Selective for Asn and Gln. *J. Magn. Reson. 131*, 199–202.
25. Lopez-Mendez, B., Güntert, P. (2006) Automated Protein Structure Determination from NMR Spectra. *J. Am. Chem. Soc. 128*, 13112–13122.
26. Liu, G., Shen, Y., Atreya, H. S., Parish, D., Shao, Y., Sukumaran, D. K., Xiao, R., Yee, A., Lemak, A., Bhattacharya, A., Acton, T. A., Arrowsmith, C. H., Montelione, G. T., Szyperski, T. (2005) NMR Data Collection and Analysis Protocol for High-Throughput Protein Structure Determination. *Proc. Natl. Acad. Sci. U. S. A. 102*, 10487–10492.
27. Möglich, A., Weinfurtner, D., Maurer, T., Gronwald, W., Kalbitzer, H. R. (2005) A Restraint Molecular Dynamics and Simulated Annealing Approach for Protein Homology Modeling Utilizing Mean Angles. *BMC Bioinformatics 6*, 91.
28. Möglich, A., Weinfurtner, D., Gronwald, W., Maurer, T., Kalbitzer, H. R. (2005) PERMOL: Restraint-Based Protein Homology Modeling Using DYANA or CNS. *Bioinformatics 21*, 2110–2111.
29. Güntert, P. (2004) Automated NMR Structure Calculation with CYANA. *Methods Mol. Biol. 278*, 353–378.
30. Brünger, A. T., Adams, P. D., Clore, G. M., DeLano, W. L., Gros, P., Grossekunstleve, R. W., Jiang, J.-S., Kuszewski, J., Nilges, M., Pannu, N. S., Read, R. J., Rice, L. M., Simonson, T., Warren, G. L. (1998) Crystallography & NMR System: A New Software Suite for Macromolecular Structure Determination. *Acta Cryst. D 54*, 905–921.
31. Schwieters, C. D., Kuszewski, J., Tjandra, N. L., Clore, G. M. (2003) The Xplor-NIH NMR Molecular Structure Determination Package. *J. Magn. Reson. 160*, 65–73.

32. Zolani, Z., Macura, S., Markley, J. L. (1989) Spline Method for Correcting Baseline Distortions in Two-Dimensional NMR Spectra. *J. Magn. Reson. 82*, 496–504.
33. Barsukov, I. L., Arseniev, A. S. (1987) Base-Plane Correction in 2D NMR. *J. Magn. Reson. 73*, 148–149.
34. Saffrich, R., Beneicke, W., Neidig, K.-P., Kalbitzer, H. R. (1993) Baseline Correction in n-Dimensional NMR Spectra by Sectionally Linear Interpolation. *J. Magn. Reson. B 101*, 304–308.
35. Dietrich, W., Rüdel, C. H., Neumann, M. (1991) Fast and Precise Automatic Baseline Correction of One- and Two-Dimensional NMR Spectra. *J. Magn. Reson. 91*, 1–11.
36. Golotvin, S., Williams, A. (2000) Improved Baseline Recognition and Modeling of FT NMR Spectra. *J. Magn. Reson. 146*, 122–125.
37. Güntert, P., Wüthrich, K. (1992) FLATT-A New Procedure for High-Quality Baseline Correction of Multidimensional NMR Spectra. *J. Magn. Reson. 96*, 403–407.
38. Stadlthanner, K., Tome, A. M., Theis, F. J., Gronwald, W., Kalbitzer, H. R., Lang, E. W. (2003) Blind Source Separation of Water Artefacts in NMR Spectra Using a Matrix Pencil. *Proc. ICA 2003*, 167–172.
39. Malloni, W.M., De Sanctis, S., Tome, A.M., Lang, E.W., Munte, C.E., Neidig, K.P., Kalbitzer, H. R. (2010) Automated solvent artifact removal and base plane correction of multidimensional NMR protein spectra by AUREMOL-SSA. J. Biomol. NMR, 47, 101–111.
40. Live, D. H., Davis, D. G., Agosta, W. C., Cowburn, D. (1984) Long Range Hydrogen Bond Mediated Effects in Peptides: 15N NMR Study of Gramicidin S in Water and Organic Solvents. *J. Am. Chem. Soc. 106*, 1939–1941.
41. Maurer, T., Kalbitzer, H. R. (1996) Indirect Referencing of 31P and 19 F NMR Spectra. *J. Magn. Reson. B 113*, 177–178.
42. Markley, J. L., Bax, A., Arata, Y., Hilbers, C. W., Kaptein, R., Sykes, B. D., Wright, P. E., Wüthrich, K. (1998) Recommendations for the Presentation of NMR Structures of Proteins and Nucleic Acids. *Pure & Appl. Chem. 70*, 117–142.
43. Neidig, K.-P., Bodenmüller, H., Kalbitzer, H. R. (1984) Computer Aided Evaluation of Two-Dimensional NMR Spectra of Proteins. *Biochem. Biophys. Res. Comm. 125*, 1143–1150.
44. Glaser, S., Kalbitzer, H. R. (1987) Automated Recognition and Assessment of Cross Peaks in Two-Dimensional NMR Spectra of Macromolecules. *J. Magn. Reson. 74*, 450–463.
45. Pfändler, P., Bodenhausen, G. (1988) Analysis of Multiplets in Two-Dimensional NMR Spectra by Topological Classification: Applications to Vinblastine and Cyclosporin A. *Magn. Reson. Chem. 26*, 888–894.
46. Novic, M., Eggenberger, U., Bodenhausen, G. (1988) Similarities Between Self-Convolution and Symmetry Mapping of Multiplets in Two-Dimensional NMR Spectra. *J. Magn. Reson. 77*, 394–400.
47. Stoven, V., Mikou, A., Piveteau, D., Guitettet, E., Lallemand, J.-Y. (1989) PARIS, a Program for Automatic Recognition and Integration of 2D NMR Signals. *J. Magn. Reson. 82*, 163–168.
48. Eccles, C., Guntert, P., Billeter, M., Wuthrich, K. (1991) Efficient analysis of protein 2D NMR spectra using the software package EASY. *J. Biomol. NMR 1*, 111–130.
49. Koradi, R., Billeter, M., Engeli, M., Guntert, P., Wuthrich, K. (1998) Automated Peak Picking and Peak Integration in Macromolecular NMR Spectra Using AUTOPSY. *J. Magn. Reson. 135*, 288–297.
50. Neidig, K.-P., Geyer, M., Görler, A., Antz, C., Saffrich, R., Beneicke, W., Kalbitzer, H. R. (1995) AURELIA, a Program for Computer-Aided Analysis of Multidimensional NMR Spectra. *J. Biomol. NMR 6*, 255–270.
51. Neidig, K.-P., Kalbitzer, H. R. (1990) Improved Representation of Two-Dimensional NMR Spectra by Local Rescaling. *J. Magn. Reson. 88*, 155–160.
52. Antz, C., Neidig, K.-P., Kalbitzer, H. R. (1995) A General Bayesian Method for an Automated Signal Class Recognition in 2D NMR Spectra Combined with a Multivariate Discriminant Analysis. *J. Biomol. NMR 5*, 287–296.
53. Schulte, A. C., Gorler, A., Antz, C., Neidig, K. P., Kalbitzer, H. R. (1997) Use of Global Symmetries in Automated Signal Class Recognition by a Bayesian Method. *J. Magn. Reson. 129*, 165–172.
54. Geyer, M., Neidig, K.-P., Kalbitzer, H. R. (1995) Automated Peak Integration in Multidimensional NMR Spectra by an Optimized Iterative Segmentation Procedure. *J. Magn. Reson. B 109*, 31–38.
55. Denk, W., Baumann, R., Wagner, G. (1986) Quantitative Evaluation of Cross-Peak Intensities by Projection of Two-Dimensional NOE Spectra on a Linear Space Spanned by a Set of Reference Resonance Lines. *J. Magn. Reson. 67*, 386–390.
56. Kobayashi, N., Iwahara, J., Koshiba, S., Tomizawa, T., Tochio, N., Güntert, P., Kigawa, T., Yokoyama, S. (2007) KUJARA, a

Package of Integrated Modules for Systematic and Interactive Analysis of NMR Data Directed to High-Throughput NMR Structure Studies. *J. Biomol. NMR 39*, 31–52.

57. Linge, J. P.; Habeck, M.; Rieping, W.; Nilges, M. (2003) ARIA: Automated NOE Assignment and NMR Structure Calculation. *Bioinformatics 19*, 315–316.
58. Herrmann, T., Güntert, P., Wüthrich, K. (2002) Protein NMR Structure Determination with Automated NOE-Identification in the NOESY Spectra Using the New Software ATNOS. *J. Biomol. NMR 24*, 171–189.
59. Gronwald, W., Moussa, S., Elsner, R., Jung, A., Ganslmeier, B., Trenner, J., Kremer, W., Neidig, K. P., Kalbitzer, H. R. (2002) Automated Assignment of NOESY NMR Spectra Using a Knowledge Based Method (KNOWNOE). *J. Biomol. NMR 23*, 271–287.
60. Geyer, M., Herrmann, C., Wohlgemuth, S., Wittinghofer, A., Kalbitzer, H. R. (1997) Structure of the Ras-Binding Domain of RalGEF and Implications for Ras Binding and Signalling. *Nat. Struc. Biol. 4*, 694–699.
61. Gronwald, W., Brunner, K., Kirchhöfer, R., Trenner, J., Neidig, K.-P., Kalbitzer, H. R. (2007) AUREMOL-RFAC-3D, Combination of R-Factors and Their Use for Automated Quality Assessment of Protein Structures. *J. Biomol. NMR 37*, 15–30.
62. Gorler, A., Gronwald, W., Neidig, K. P., Kalbitzer, H. R. (1999) Computer Assisted Assignment of 13C or 15N Edited 3D-NOESY-HSQC Spectra Using Back Calculated and Experimental Spectra. *J. Magn. Reson. 137*, 39–45.
63. Görler, A., Kalbitzer, H. R. (1997) Relax, a Flexible Program for the Back Calculation of NOESY Spectra Based on Complete-Relaxation-Matrix Formalism. *J. Magn. Reson. 124*, 177–188.
64. Ried, A., Gronwald, W., Trenner, J. M., Brunner, K., Neidig, K.-P., Kalbitzer, H. R. (2004) Improved Simulation of NOESY Spectra by RELAX-JT2 Including Effects of J-Coupling, Transverse Relaxation and Chemical Shift Anisotropy. *J. Biomol. NMR 30*, 121–131.
65. Keepers, J. W., James, T. L. (1984) A Theoretical Study of Distance Determination from NMR. Two-Dimensional Nuclear Overhauser Effect Spectra. *J. Magn. Reson. 57*, 404–426.
66. Boelens, R., Koning, M. G., van der Marel, G. A., van Boom, J. H., Kaptein, R. (1989) Iterative Procedure for Structure Determination from Proton-Proton NOEs Using a Full Relaxation Matrix Approach. Application to a DNA Octamer. *J. Magn. Reson. 82*, 290–380.
67. Borgias, B. A., James, T. L. (1990) MARDIGRAS-A Procedure for Matrix Analysis of Relaxation for Discerning Geometry of an Aqueous Structure. *J. Magn. Reson. 87*, 475–487.
68. Post, C. B., Meadows, R. P., Gorenstein, D. G. (1990) On the Evaluation of Interproton Distances for Three-Dimensional Structure Determination by NMR Using a Relaxation Rate Matrix Analysis. *J. Am. Chem. Soc. 112*, 6796–6803.
69. van de Ven, F. J. M., Blommers, M. J. J., Schouten, R. E., Hilbers, C. W. (1991) Calculation of Interproton Distances from NOE Intensities. A Relaxation Matrix Approach Without Requirement of a Molecular Model. *J. Magn. Reson. 94*, 140–151.
70. Madrid, M., Llinas, E., Llinas, M. (1991) Model-Independent Refinement of Interproton Distances Generated from ^{1}H NMR Overhauser Intensities. *J. Magn. Reson. 93*, 329–346.
71. Kim, S.-G., Reid, B. R. (1992) Automated NMR Structure Refinement via NOE Peak Volumes. Application to a Dodecamer DNA Duplex. *J. Magn. Reson. 100*, 382–390.
72. Moseley, H. N. B., Curto, E. V., Krishna, N. R. (1995) Complete Relaxation and Conformational Exchange Matrix (CORCEMA) Analysis of NOESY Spectra of Interacting Systems; Two-Dimensional Transferred NOESY. *J. Magn. Reson. B 108*, 243–261.
73. Ernst, R. R., Bodenhausen, G., Wokaun, A. (1987) *Principles of Nuclear Magnetic Resonance in One and Two Dimensions.* Clarendon Press: Oxford.
74. Jeener, J., Meier, B. H., Bachmann, P., Ernst, R. R. (1979) Investigation of Exchange Processes by Two-Dimensional NMR Spectroscopy. *J. Chem. Phys. 71*, 4546–4553.
75. Macura, S., Ernst, R. R. (1980) Elucidation of Cross Relaxation in Liquids by Two-Dimensional N.M.R. Spectroscopy. *Mol. Phys. 41*, 95–117.
76. Boelens, R., Koning, M. G., Kaptein, R. (1988) Determination of Biomolecular Structures from Protein-Protein NOE's Using a Relaxation Matrix Approach. *J. Mol. Struct. 173*, 299–311.
77. Gronwald, W., Kirchhofer, R., Gorler, A., Kremer, W., Ganslmeier, B., Neidig, K. P., Kalbitzer, H. R. (2000) RFAC, a Program for Automated NMR R-Factor Estimation. *J. Biomol. NMR 17*, 137–151.
78. Baskaran, K., Kirchhofer, R., Huber, F., Trenner, J., Brunner, K., Gronwald, W., Neidig, K. P., Kalbitzer, H. R. (2009) Chemical Shift Optimization in Multidimensional NMR

Spectra by AUREMOL-SHIFTOPT. *J. Biomol. NMR 43*, 197–210.

79. Linge, J. P., O'Donoghue, S. I., Nilges, M. (2001) Automated Assignment of Ambiguous Nuclear Overhauser Effects with ARIA. *Methods Enzymol. 339*, 71–90.
80. Linge, J. P., Williams, M. A., Spronk, C. A. E. M., Bonvin, A. M. J. J., Nilges, M. (2003) Refinement of Protein Structures in Explicit Solvent. *Proteins 50*, 496–506.
81. Brunner, K., Gronwald, W., Trenner, J. M., Neidig, K.-P., Kalbitzer, H. R. (2006) A general Method for the Unbiased Improvement of Solution NMR Structures by the Use of Related X-Ray Data, the AUREMOL-ISIC Algorithm. *BMC Struc. Biol. 6*, 14.

Chapter 8

Computational Tools in Protein Crystallography

Deepti Jain and Valerie Lamour

Abstract

Protein crystallography emerged in the early 1970s and is, to this day, one of the most powerful techniques for the analysis of enzyme mechanisms and macromolecular interactions at the atomic level. It is also an extremely powerful tool for drug design. This field has evolved together with developments in computer science and molecular biology, allowing faster three-dimensional structure determination of complex biological assemblies. In recent times, structural genomics initiatives have pushed the development of methods to further speed up this process. The algorithms initially defined in the last decade for structure determination are now more and more elaborate, but the computational tools have evolved toward simpler and more user-friendly packages and web interfaces. We present here a modest overview of the popular software packages that have been developed for solving protein structures, and give a few guidelines and examples for structure determination using the two most popular methods, molecular replacement and multiple anomalous dispersion.

Key words: X-ray crystallography, Protein structure determination, Crystallography software, Molecular replacement, Multi-wavelength anomalous dispersion

1. Introduction

The correct three-dimensional (3D) arrangement of constituent atoms is central to the proper functioning of a majority of biological molecules. This fact is especially true of proteins, and hence, it is important to determine the structures of macromolecules and their assemblies to understand their function in great detail. X-ray crystallography is an experimental science that has benefited tremendously from all the software and hardware developments during the past three decades. Structure determination through X-ray crystallography is becoming more and more automated through the development of user-friendly interfaces and new crystallography packages. These developments have made the method more accessible to the biochemist with little theoretical background in

David Fenyö (ed.), *Computational Biology*, Methods in Molecular Biology, vol. 673,
DOI 10.1007/978-1-60761-842-3_8,

crystallography. Often behind every structure lies months or more commonly years of bench work to clone, express, and purify milligram quantities of protein required to produce diffracting crystals, a prerequisite for any structural work using X-ray crystallography. Useful information regarding this aspect of the method can be found in numerous reviews that summarize decades of research in structural biology, spanning simple proteins to high-molecular weight biological macromolecular assemblies (1–3). The present chapter will focus on the major methods and tools available to determine the 3D structure of a macromolecule once diffracting crystals have been prepared. It provides a concise reference to a beginner in protein crystallography.

Computational tools in the form of program packages are constantly evolving to feed the need for faster and more convenient structure determination, especially driven by structural genomics initiatives. We provide here a brief overview of the most common and most widely used crystallographic packages (see Fig. 1) and guidelines for structure determination using the favored phasing

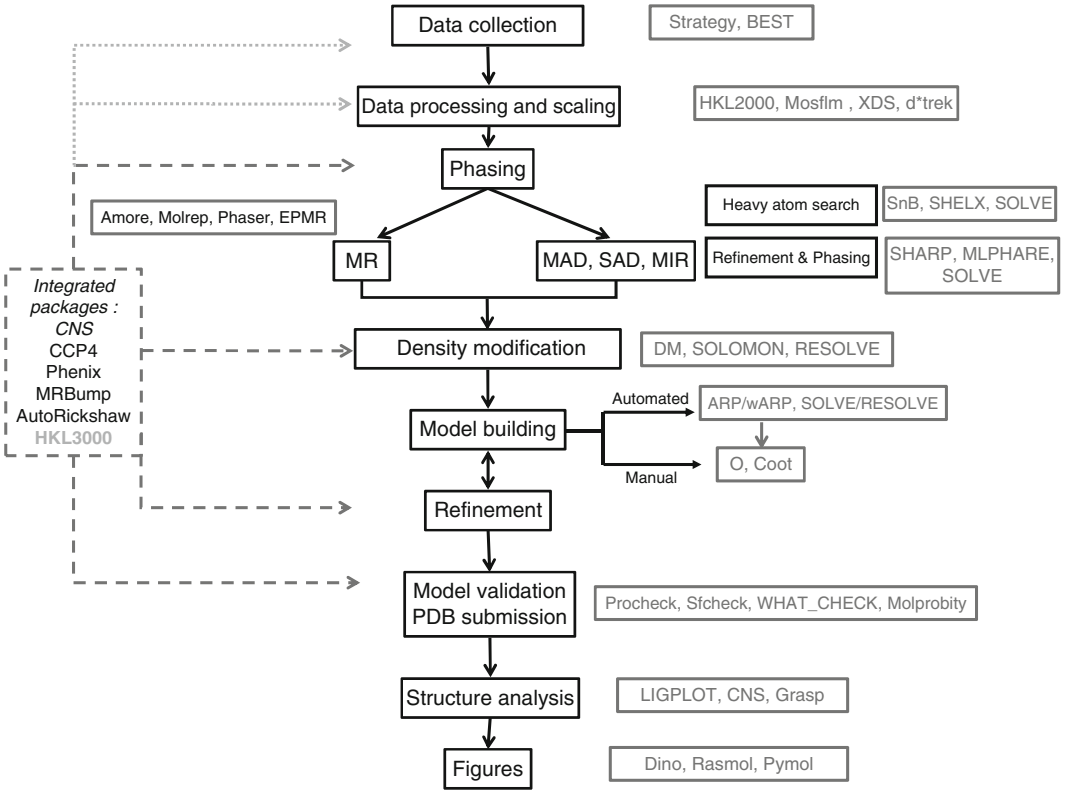


Fig. 1. General scheme representing the different steps to solve a crystallographic structure. A selection of programs is reported next to each step of X-ray structure determination. Program packages or pipelines (dash line box) cover several steps of structure determination. HKL3000 also includes data processing in the pipeline.

methods: molecular replacement (MR) and multi-wavelength anomalous dispersion (MAD). The programs involved in structure determination were first developed for Unix platforms. These programs have been redeveloped to run on Linux, and most of them can now run on Windows and Mac OSX platforms. There are different program packages available for different stages of structure determination: data collection and processing, structure solution, crystallographic refinement, structure validation, and documentation.

1.1. Phasing Methods

The diffraction data collected from exposure of crystals to X-rays provide the amplitudes of structure factors. The phases cannot be recorded by diffraction experiment, and hence, it is not possible to generate an electron density map, which requires both amplitudes and phases. Any strategy to determine the structure of a molecule has to find a solution for this "phase problem." Once protein crystals and X-ray diffraction data are obtained, several methods can be used to address the phase problem (4).

The direct methods are mathematical tools which use the probability theory and the assumption of approximate equal, resolved atoms to estimate phases from the measured intensities (5). Direct methods have been used to solve structures of small molecules (typically a few hundred atoms). It is also possible to solve structures of small proteins (250–1,000 non-H-atoms) using these methods; however, it requires the resolution of X-ray diffraction data to be better than 1.2 Å.

The experimental phasing of macromolecular structures is achieved by the multiple (or single) isomorphous replacement (MIR or SIR) method, which involves introducing heavy atoms (like platinum, samarium, mercury) into the macromolecular crystal. Heavy-atom derivatives are prepared by soaking the native crystals in a buffer containing heavy-atom compound or by cocrystallization. The requirement for this technique to work is that the crystals of the native protein and that of the heavy-atom derivative should be isomorphous, i.e., they should have the same space group and cell dimensions. The heavy atom substructure is then determined by difference Patterson, which is used to calculate the phases. At least, two isomorphous derivatives are necessary to unambiguously determine the phases. When one derivative dataset is combined with the native protein dataset, the phasing process is referred to as SIR. MIR is sometimes combined with anomalous scattering of some of the heavy atoms, and the technique is then referred to as multiple isomorphous replacement with anomalous scattering (MIRAS) or single isomorphous replacement with anomalous scattering (SIRAS).

Another experimental phasing method is MAD (6, 7), which uses anomalous scattering information from the datasets collected at different wavelengths near the absorption edges of

heavy metals. The protein structure to be determined should contain anomalous scatterers like selenomethionine or metallic centers. Different datasets on the same crystal are collected at wavelengths where both the anomalous and the dispersive signals are maximized. Also, this method requires tunable wavelength available at synchrotron source. In some cases, a single dataset collected at the single wavelength is sufficient for structure solution by single wavelength anomalous dispersion (SAD) (8, 9). SAD is attempted in cases where there is significant radiation damage preventing collection of several datasets on the same crystal. Some elements such as Fe give significant anomalous signal around CuKα wavelength; hence, a SAD dataset for a protein containing Fe can also be collected on a rotating anode X-ray generator. Most of the programs that work for MAD can also be used with a SAD dataset.

Radiation damage induced by high-energy X-rays is usually perceived as detrimental to the quality of diffraction data and structure determination. Techniques are being developed to minimize this effect during data collection (10). However, a technique called radiation-induced phasing (RIP) is emerging which takes advantage of site-specific radiation damage and the chemical changes they induce in the protein structure (11) for phasing purposes (12).

Molecular replacement (MR) uses the structural information from a similar known structure. The calculated phases from the available structure can be used as initial estimates for the target dataset. This process is facilitated by thousands of crystal structures deposited in the Protein Data Bank (www.rcsb.org), a number that has seen a significant rise in recent times owing to structural genomics consortiums depositing coordinates of many proteins.

Currently, MR and MAD represent the two most common methods to determine the 3D structure of biological macromolecules.

1.2. Data Collection and Processing Packages

X-ray data can be collected on home sources on a rotating anode or using the X-ray beam generated by a synchrotron radiation. The quality of the data collected on synchrotron beam lines is much superior to that collected on a traditional home source due to the high flux, better polarization, and tunable X-ray optics of the beam resulting in a substantial increase in the signal-over-noise ratio. However, the flux available from home-source generators has also improved significantly over the years allowing routine crystal structure determination from good diffracting crystals. Several generations of synchrotrons have been built in the past decades around the world, and information on synchrotron X-ray sources can be found in the following link: http://biosync.rcsb.org. Currently, the entire data processing can usually be done at the beam line during or immediately after data collection.

There are a number of software packages available for X-ray data processing. These programs usually attempt to index a few initial frames and provide the correct Bravais lattice and cell constants of the crystal. Once the data collection is complete, they can merge, reduce, and scale the data to provide a set of unique indices, their measured intensities, and corresponding standard deviations. The HKL suite is the most popular of them, and it consists of three modules: XdisplayF for visualization of the diffraction pattern, Denzo for data reduction and integration, and Scalepack for merging and scaling of the intensities obtained by Denzo or other programs (13). In the current package HKL2000, all these three modules have been integrated into a single window. The program provides graphs of mosaicity, *B* factors, completeness, and other statistics during data processing to assess the quality of the dataset.

MOSFLM (14) (also available in GUI format called iMosflm) is also a data processing program that is part of the CCP4 suite (15). It is coupled with the scaling program SCALA and can process data from a wide range of detectors and writes out the merged and scaled data in the CCP4 binary format – with the extension .mtz and can be directly read by other modules of the CCP4 suite for further analysis and structure determination. Mosflm/iMosflm is available free of charge whereas HKL2000 is a commercial package.

The program XDS, X-ray detector software (16), is available free of charge and contains the following components: XDS for processing single datasets, XSCALE for scaling datasets, XDSCONV that converts the output file into desirable formats, and VIEW for visualizing. d*TREK (available with Rigaku detectors) is another software suite for data collection and processing (17). It contains three components: dtcollect to set up parameters for data collection, dtdisplay to display X-ray diffraction images, and dtprocess, an interface for integration and processing of images.

The starting point and the total range of data collection depend on the orientation of the crystal and its symmetry. Several programs can be used to determine the best oscillation range to get a complete dataset using information about the crystal parameters (Bravais lattice, cell constant) and the geometrical relationships between the crystal and the detector. For this purpose, one can use the program Strategy included in HKL and Mosflm after collecting and indexing one or a couple of sequential images. The program BEST takes into account the diffraction anisotropy and evaluates the optimal oscillation width to avoid overlapping reflections after collecting one or two images with a 90° angle (18, 19). Most synchrotron beam lines offer some useful tips and guidelines for data collection (example: CHESS synchrotron website: http://www.macchess.cornell.edu/MacCHESS/collect_strategy.html).

1.3. Estimation of the Cell Contents

The number of molecules in the asymmetric unit (AU) is usually estimated through the calculation of the Matthews coefficient, which calculates the solvent content of protein crystals using the molecular weight of protein as a parameter. It has been observed that the value of the Matthews coefficient usually lies within the range 1.6–4.5, and this value can be used to calculate the number of molecules in the AU (20, 21). Several websites offer an online calculation (see Table 1). The Matthews coefficient can also be calculated directly in the MR module of CCP4 (cell content analysis) or CNS (matthews_coef input file).

Table 1
List of crystallography programs and related websites cited in this chapter

Program name	Link
3dSS	http://cluster.physics.iisc.ernet.in/3dss/
ARP/wARP	http://www.arp-warp.org
Auto-RIckshaw	http://www.embl-hamburg.de/Auto-Rickshaw/
BALBES	http://www.ysbl.york.ac.uk/~fei/balbes/
BEST	http://www.embl-hamburg.de/BEST/
CASTp	http://sts-fw.bioengr.uic.edu/castp/
CATH	http://www.cathdb.info/
CCP4i	http://www.ccp4.ac.uk
CNS	http://cns.csb.yale.edu
Coot	http://www.biop.ox.ac.uk/coot/
CRYSTAL	http://crystal.uvm.edu/ (Updated link to crystallographic softwares)
d*TREK	http://www.rigaku.com/software/dtrek_news.html
DALI	http://ekhidna.biocenter.helsinki.fi/dali_server/
Dino	http://www.dino3d.orghttp://bioserv.rpbs.jussieu.fr/cgi-bin/PPG
EPMR	http://www.epmr.info/
HIC-Up	http://xray.bmc.uu.se/hicup/
HKL suite	http://www.hkl-xray.com/
HKL2MAP	http://schneider.group.ifom-ieo-campus.it/hkl2map/index.html
JCSG validation suite	http://www.jcsg.org/scripts/prod/validation/sv_final.cgi
LIGPLOT	http://www.biochem.ucl.ac.uk/bsm/ligplot/ligplot.html

(continued)

Table 1
(continued)

Program name	Link
Matthews constant	http://www.ruppweb.org/Mattprob/http://pldserver1.biochem.queensu.ca/~rlc/pfd/links/calcs/vm_calc.shtml http://csb.wfu.edu/tools/vmcalc/vm.html
Modeler	http://salilab.org/modeller
Molprobity	http://molprobity.biochem.duke.edu/
MolScript	http://www.avatar.se/molscript/doc/index.html
MOSFLM and iMosflm	http://www.mrc-lmb.cam.ac.uk/harry/imosflm/
MrBUMP	http://www.ccp4.ac.uk/MrBUMP/
MSMS	http://mgltools.scripps.edu/packages/MSMS/
O	http://xray.bmc.uu.se/alwyn/A-Z_of_O/A-Z_frameset.html
PDB data deposit AUDIT	http://deposit.rcsb.org/
PHENIX	http://www.phenix-online.org
POVRAY	http://www.povray.org/
PredictProtein	http://www.predictprotein.org/
PRODRG	http://davapc1.bioch.dundee.ac.uk/prodrg/
PyMOL	http://www.pymol.org/http://www.pymolwiki.org/index.php/Main_Page
RAPIDO	http://webapps.embl-hamburg.de/rapido/
RASMOL	http://www.bernstein-plus-sons.com/software/rasmol/
REFMAC	http://www.ccp4.ac.uk/dist/html/refmac5.html
RIBBONS	http://www.cbse.uab.edu/carson/papers/index.html#Ribbons
SHARP/autoSHARP	http://www.globalphasing.com/sharp/
SHELX	http://shelx.uni-ac.gwdg.de/SHELX/
SnB	http://www.hwi.buffalo.edu/SnB/
SOLVE	http://solve.lanl.gov/
Strategy	http://www.crystal.chem.uu.nl/distr/strategy.html
Superpose	http://wishart.biology.ualberta.ca/SuperPose/
WHATIF	http://swift.cmbi.kun.nl/whatif/
XDS	http://xds.mpimf-heidelberg.mpg.de/html_doc/downloading.html

2. Molecular Replacement

2.1. Choice and Preparation of the Search Model

The correct choice of the search model is the main criterion to ensure success in solving a structure using MR. The structural homology of the search model with the target protein (i.e., the root mean square deviation between the expected structure and the known structure) and the primary sequence homology are two of the main parameters considered for defining the right search model. Generally, the initial choice is based on sequence homology, and the closer the target primary sequence is to the search model (i.e., >25%), the higher the chance of finding a MR solution. When the protein sequence is divergent from the model and it is known that the fold of the model and target protein is similar, then the coordinate file can be transformed into a polyalanine chain to reduce the errors introduced by side chains. Most MR modules in crystallographic program packages offer this option. The polyalanine backbone can also be generated using programs such as Moleman which can manipulate coordinate files (G.J. Kleywegt – Uppsala Software Factory: http://xray.bmc.uu.se/usf/moleman_man.html). To reduce the model bias, it is also advisable to set the temperature factors (*B* factor) to the average value determined from the Wilson plot (22) from the experimental data. Water molecules and other small ligands should be removed from the search model.

If the structural homology of candidate search models is not obvious, secondary structure prediction programs can be used to predict the structure of the target protein to make a more informed decision regarding the choice of the model. The PHD secondary structure prediction program (23) uses evolutionary information from multiple sequence alignments and is now available through the PredictProtein website (24).

In addition, modeling the target structure using structures of homologous proteins can be done to generate a search model. One of the most popular program for this purpose is Modeler (25). Large conformational changes in a structure might be an obstacle for solving the structure by MR. Flexible and/or disordered loops and small regions that could adopt a different conformation in the target structure can be removed from the initial model before running a MR program. One must keep in mind that it will decrease the completeness of the search model and might be detrimental to MR, but every effort must be made to bring the structure of the search model as close as possible to that of the target protein.

2.2. Choice of the Molecular Replacement Program

Most crystallographic packages offer one or several MR programs. Traditional programs are based on the Patterson function. An MR search is typically divided into the rotation and the

translation functions, which are correlation functions between observed and calculated model Pattersons. The self-rotation function can be used to determine the orientation and nature of noncrystallographic symmetry elements without the need of a search model (26).

The CCP4 package includes a MR module with four different programs: Phaser, BEAST, MOLREP, and AMoRe. BEAST (27) and the improved version of Phaser (28) use a maximum-likelihood objective function. AMoRe (29) carries out translation and rotation search using the fast rotation function as well as a rigid body. In case of a protein–protein or protein–DNA complex, traditional MR programs such as AMoRe only allow searching for one molecule at a time. Once the first one is found, the second molecule of a complex can be searched. MOLREP (30) is an all-automated program that takes into account the level of completeness and identity of the search model. When looking for an oligomer, for example, a dimer, either the monomer or the entire dimer can be tried as a search model. Phaser allows the user to search for several different molecules or protein domains in the AU at the same time. The REPLACE suite of programs written by Liang Tong also provides tools for MR: a rotation program GLRF and a translation program TF (31). Unlike the traditional MR programs, EPMR is a program that does not separate the translation and rotation steps but uses an evolutionary search algorithm simultaneously optimizing the orientation and position of a search model (32). It is also referred to as a 6D MR. This program can be successful for rather incomplete search models but requires more computing time.

MR "pipelines" now provide an all-automated tool for faster and user-friendly procedures integrating all steps, including the choice of the search model. One such package, PHENIX, can run Phaser using the AutoMR module and provides tools for automated model building (RESOLVE). MrBUMP includes a FASTA search using the target primary sequence to look for a homologous structure (33). It is available through the CCP4i interface and can run MR with Phaser or MOLREP. BALBES also requires little to no user intervention during the entire MR procedure (34) and can also be run through a CCP4i window. A web interface has been developed for CaspR also starting from a primary sequence and generating a homology model for the MR search (35).

One has to remember that more than using a particular program, a key step for MR is the choice of the search model, along with careful selection of the resolution range for the search as the quality of the data will greatly influence the outcome. The correct estimation of the number of molecules in the AU is helpful, since that allows the MR programs to know how many molecules of the search model to look for in the AU.

The MR solutions output from the search can be evaluated with the help of three parameters: the *R*-factor, the correlation coefficient (CC), and the resulting packing in the target unit cell. The possible solutions in the form of transformation matrices or transformed coordinates and their corresponding *R*-factor and correlation coefficients are listed in the MR output files. The presence of a solution with significantly higher correlation coefficient than the rest is an indication that the MR search has succeeded. A few issues of Acta Crystallographica have been dedicated to MR and give a more detailed review about all aspects of this method (36, 37). A special chapter in Methods in Molecular Biology Series also describes different strategies and programs for MR (38).

Modeler or other current modeling programs can be used after running MR to substitute the correct primary sequence in the MR model. Automated building programs such as ARP/wARP (see Subheading 5) can be used for model building. When only a few residues are different between the search model and the query, amino acid substitution can be done directly in most model-building programs (see Subheading 5). After a MR solution is obtained, electron-density maps can be calculated, and the structure has to go through several rounds of model building and refinement to reduce the bias introduced by the starting model. The procedures for crystallographic structure refinement are basically the same for different phasing methods and are detailed in Subheading 5 of this chapter. We provide in Table 2 an example of structure determined by MR with the list of programs used at each step.

3. Multi-wavelength Anomalous Dispersion

The MAD method exploits the observation that when the heavy atoms absorb X-rays, there is a violation of Friedels law and the Bijovet pairs (h,k,l) and (-h-k-l) will have different intensities. The difference in intensities is referred to as anomalous difference (41). For most atoms, anomalous dispersion is negligible, only select elements (Se, Hg, Au, Pt, Zn, W, Os, Br, lanthanides, etc.) have their X-ray absorption edges in the energy range that are accessible by synchrotron radiation. In addition, the presence of natural metal centers containing iron, copper, and zinc can be exploited to conduct a MAD experiment. Nucleic acids with bromouracil residues can be used for determining their structures and that of their complexes with proteins. Bound phosphate ions have been replaced by tungstate (42, 43), which can provide good anomalous signal. More recently covalent modification of nucleic acids has been done by solid-phase synthesis using selenium for crystallographic phasing (44). In proteins, sulfur atoms show

Table 2
Steps leading to structure determination of Tth Gyrase ATPase domain/novobiocin complex by molecular replacement

Programs and packages used	Steps for solving the structure	Result
HKL2000	Data processing and scaling	Data collected on a synchrotron beamline (ID14, ESRF) P21 space group with a = 44.9 Å, b = 125.5 Å, c = 79.8 Å, and γ = 96.4° Data scaled between 15 and 2.3 Å, Rsym = 6.4% Solvent content (Matthews coefficient) 51.5% indicating two molecules/AU
AMoRe CCP4 Moleman	Molecular replacement Translation and rotation search	Scalepack output file transformed into CCP4 format for molecular replacement (MR) using AMoRe Search Model: one monomer from the structure of *Escherichia coli* 43 kDa ATPase domain (PDB ID 1EI1); PDB transformed into a polyalanine chain with Moleman due to low primary sequence homology with the target protein; water molecules, and ligand removed for the search (search using the whole dimer failed) Result: Two peaks corresponding to the two molecules in the asymmetric unit; correlation factor 21.1
DM CCP4	Density modification	DM density modification with non-crystallographic symmetry (NCS) averaging (self-rotation function gave position of dimer twofold axis)
CNS	Reflection file formatting for refinement	Mtz reflection file transformed into CNS format (CCP4) 7.5% reflections put aside for free R calculation for refinement in CNS
O	Model building	Major conformational change between the two domains of each monomer had to be adjusted manually and the active site ATP lid had to be rebuilt
CNS minimization and simulated annealing	Refinement, R and R_{free} values monitoring	Iterative steps of model building and CNS refinement Map calculation after each major building step and refinement ($2F_{obs} - F_{calc}$, $F_{obs} - F_{calc}$) B factors included after most of the model was corrected Novobiocin ligand fit last and included in refinement Addition of water molecules and final round of minimization
CNS Ramachandran Plot and PDB submission formatting	Model validation PDB submission	Manual adjustment of residues in disallowed regions for final PDB submission followed by a few cycles of minimization Final R = 20%; R_{free} = 26%

An example of molecular replacement: Tth Gyrase 43 kDa ATPase domain in complex with the antibiotic novobiocin (PDB ID 1KIJ)
Source: Lamour et al. (39, 40)

weak anomalous signal, and more and more structures are being determined using this property (45). Some other choices of heavy atoms for structure solution have been described in Boggon and Shapiro (46).

However, the most preferred method for MAD is the use of Selenomethionine-substituted protein. SeMet protein is prepared by standard protocols involving suppression of methionine biosynthesis (47) or through the use of an *Escherichia coli* strain that is auxotrophic for methionine. Most often, the crystals of the derivatized protein grow from conditions nearly similar to the native crystals. In case there are problems getting crystals of the SeMet-substituted protein, microseeding using seeds from native crystals can be attempted. Amino acid or mass spectrometry analysis can be done to confirm the presence as well as to determine the percentage of incorporation of SeMet in the protein.

3.1. MAD Data Collection

MAD relies on good-quality X-ray diffraction data collected at three different wavelengths preferably from a single crystal. Initially, a fluorescence scan of the SeMet crystal is done to verify the presence and location of the absorption edge for the crystal. The fluorescence scan can be analyzed with a program called Chooch (48) to give the recommended wavelengths and estimates of f′ and f″. Usually the peak wavelength is the maximum in the fluorescent scan. High energy remote is usually 150–250 eV higher than the peak. The inflection value is halfway down the f′ absorption edge, usually about 3 eV less than the peak. It is advisable to collect the peak data first so that the dataset can be used to attempt SAD in case excessive radiation damage does not permit data collection at other wavelengths. A good MAD dataset should be highly redundant and complete to accurately determine the anomalous differences. It is advisable to collect a low-resolution (~3 Å), high-redundant dataset initially to solve the structure as compared to higher resolution dataset that is not redundant. Data are usually collected in wedges of 10–30° at two values of phi, which are 180° away from each other (inverse beam method). This minimizes the effect of radiation damage on the intensities of (h,k,l) and (-h,-k,-l) by collecting them close in time. Since this method takes twice as long to collect the data, for higher symmetry space groups, data collection by the inverse beam method can be avoided. While the data are being collected and processed, scale factors and *B* factors should be monitored in the log file. These values should change very gradually during entire data collection reflecting how much the crystal is affected by radiation. Data for each of the three wavelengths should be integrated and scaled separately. Any of the packages mentioned above can be used for data reduction and scaling.

3.2. Heavy Atom Search and Phase Calculation

As stated earlier, the major problem encountered in determining structures using crystallography is the phase problem. Locating the positions of the heavy atoms, i.e., determining the substructure of heavy atoms (MIR) or anomalous scatterers (MAD) is central to the success of the experimental methods. Currently, the crystallographic programs available for this purpose are based either on the direct methods or on the Patterson-based methods.

Crystallographic programs that use direct methods to locate heavy atom sites are Shake and Bake (SnB) (49) and SHELX (5, 50). These programs have been routinely used to solve structures with a large number of heavy-atom sites. SnB performs reciprocal space refinement of the heavy-atom sites combined with real space refinement using constraints in an alternate fashion. SnB finds the sites and outputs a figure of merit, which allows the user to judge the presence of probable solution. SnB is also part of the protein structure determination package BnP (The Buffalo and Pittsburgh Interface). This package combines SnB with the PHASES suite which can read intensity data and output protein electron density map (51).

The SHELXD module of SHELX is also widely used for locating the heavy-atom sites. The main difference between SnB and SHELXD is the use of Patterson function in SHELXD to provide better starting phases and figure of merit, whereas SnB locates the sites starting from a set of random coordinates. The SHELXE module of SHELX reads in the heavy-atom sites and can estimate the phases and weights and performs crude form of density modification (52). The SHELXC module of SHELX prepares the files necessary for running SHELXD and SHELXE for determination of heavy-atom structure factors and phase calculation. It also suggests the resolution cutoff that should be used to truncate the data for heavy atom search. These modules can be used independently and also through a graphical user interface (HKL2MAP) (53).

The other program which is used for locating the heavy-atom sites is SOLVE (54). It is an automated crystallographic structure solution program that can carry out all the steps of macromolecular structure determination from scaling the data to locating the heavy atom sites and determining the phases. It is a Patterson-based method; however, it interprets Patterson function automatically and combines it with repeated analysis of isomorphous and difference Fourier techniques.

Besides these, there are two major software packages used in crystallography: the crystallography and NMR system (CNS) (55) and CCP4, which can also locate heavy-atom sites (more information about these packages in the refinement section). CCP4 includes programs that employ both direct methods and Patterson-based methods for determination of heavy-atom

positions, RANTAN (56), and ACORN (57). CNS employs a combination of Patterson searches and difference Fourier to locate the heavy-atom sites or anomalous scatterers. It can also perform heavy-atom refinement, phase determination, density modification, and map calculations.

Once the sites have been located, their positions can be used to phase the MAD dataset using programs that refine the positions and occupancies of the heavy-atom sites. These include MLPHARE, CNS, SHARP, and SOLVE. These programs use maximum-likelihood approaches for refinement of sites and phase calculation. MLPHARE (58), which is part of the CCP4 suite, and CNS require the user to assign one of the datasets as a reference (usually the peak), and all the other datasets are treated as the derivatives. Statistical heavy-atom refinement and phasing (SHARP) program (59), refines coordinates, occupancies, and temperature factors for the heavy-atom sites together with scaling parameters and the nonisomorphism parameters for each dataset. SHARP operates on reduced, merged, and scaled data. The newer version of SHARP called autoSHARP includes a fully automated structure solution system from merged data to automatic model building. It also provides the interface to ARP/wARP for model building. The current version of SHARP/autoSHARP requires that the CCP4 suite be installed on the computer since SHARP uses a subset of its programs. An example of structure determination using MAD method is described in Table 3 along with the programs used at each steps.

4. Density Modification

The initial phase estimates obtained from experimental techniques are improved through density modification. One of the methods for density modification is through solvent flattening. If the phases are good, it is possible to make a mask around the molecule, and the rest of the density outside the mask is set to a low constant value. The new structure factors are then calculated and combined with the starting phases to yield improved phases. This process is done iteratively and results in improved electron-density map. The other methods of density modification are through solvent flipping, noncrystallographic symmetry averaging (NCS), histogram matching, skeletonization, Sayre's equation, and multiresolution modification (which combines solvent flattening and histogram matching over different resolution ranges) (61).

Table 3
Steps leading to structure determination of cII/DNA complex by MAD

Programs and packages used	Steps for solving the structure by MAD	Result
HKL2000 Scalepack	Data processing and scaling	Data collected at three different wavelengths. Peak (0.97903 Å), inflection (0.97922 Å), and remote (0.96384 Å) for SeMEt crystals and at 0.984 Å for native crystals at National Synchrotron Light Source beamline X9A (Brookhaven National Laboratory). P21 space group with a =44.77 Å, b =97.13 Å, c = 59.87 Å, and β = 105.36° Data scaled between 50 and 1.7 Å for native crystals and between 20 and 2.25 Å for SeMet crystals at three wavelengths
SnB	Locating the heavy atom sites	Using anomalous signal from data collected at peak, 10 of the possible 12 selenium sites were located using SnB
MLPHARE	Phase calculation	Heavy atom sites were refined and phases were calculated using peak wavelength data as reference. Figure of merit = 0.35. Anomalous signal from the three wavelengths resulted in the map at 2.3 Å
DM (CCP4)	Density modification	Density modification was done using both hands of the Se sites to identify the correct hand. Phases were extended up to 1.7 Å
Arp/Warp	Automatic model building	Built the protein backbone into the map
CNS	Reflection file formatting for refinement	Mtz reflection file transformed into CNS format 5% random reflections put aside for free R calculation for refinement in CNS
O	Model building	Side chains were built manually, DNA was also added using O
CNS	Refinement, R and R_{free} values monitoring	Maps were improved through iterative cycles of refinement against native amplitudes
CNS Ramachandran Plot and PDB submission formatting	Model validation-PDB submission	Manual adjustment of residues in disallowed regions for final PDB submission followed by a few cycles of minimization Final R = 20.9%; R_{free} = 22.8%

An example of structure solution by multi-wavelength anomalous dispersion (MAD): Crystal structure of bacteriophage lambda cII and its DNA complex (PDB ID 1ZS4)
Source: Jain et al. (60)

SOLOMON (62) and DM (63), which are part of the CCP4 suite, are popular programs used for density modification. SOLOMON performs solvent flattening/flipping, whereas DM performs solvent flattening, NCS averaging, histogram matching, and phase extension to improve the electron-density maps. DM also applies real-space constraints to the experimental map to improve the phases. SQUASH (64) is another density modification program that incorporates Sayre's equation, solvent flattening, and histogram matching simultaneously to improve the maps. More recently, a statistical approach to density modification has been implemented in the program called RESOLVE (65). Density modification protocols generally include a provision to extend the phases to native data resolution and thus provide experimental electron-density maps of high quality.

5. Interpretation of the Map and Model Building

With high-resolution data and good phases, one can also use automated model building procedures such as SOLVE/RESOLVE or ARP/wARP (66, 67). Initially, ARP/wARP could only be used for model building of structure where the data were available to high resolution. However, subsequent improvements in the algorithm have allowed it to be used in cases where the resolution extends up to 2.8 Å. The program can build the model starting from either experimental phases or from existing model. The protocol *warpNtrace* puts free atoms in positive peaks in the electron-density map iteratively refining the structure and recalculating the maps phased from the structure. If one gets enough atoms in correct places, the program converts these into polypeptide chain backbone. If the sequence is available, it docks it into the built backbone. It is now possible to identify ligands, cofactors, and solvent molecules in the difference electron-density maps in ARP/wARP after the protein structure is completed (68). The program is installed as a part of the CCP4 package (which implements the GUI) since it uses utilities from REFMAC.

RESOLVE can carry out automated model building by fragment identification where it can recognize the presence of fragments of structures like helices and strands. It then uses a tripeptide fragment library to extend these fragments in either direction. It then adds side chains and aligns the built model with the sequence provided by the user (69). The software PHENIX contains an Autobuild wizard that uses RESOLVE for automated model building (70). It also refines the built model and performs density

modification. Manual building might be required to adjust the parts of the structure that were not built accurately by the automated procedure

If the native dataset is medium/low resolution, then the entire chain tracing exercise has to be done manually. The secondary structure elements are built first. Coordinates of a typical α-helix or a beta sheet can be downloaded from known structures and can be dragged and fitted into the map. For a protein, i.e., the primary structure, the positions of the selenium atoms (and therefore the methionines) are used to manually position the amino acids in the map. Also, large residues such as Trp, Tyr, Phe, and Arg have their characteristic density and that along with the known positions of the methionines help in placing the correct amino acids in the map. Building the polypeptide chain in the correct direction can be difficult and the fact that side chains of an α-helix invariably point toward its N-terminus is a useful tip to remember. Information about the basics of electron-density fitting can be found on the Protein Crystallography Course website http://www-structmed.cimr.cam.ac.uk/Course/Fitting/fittingtalk.html.

O (71), crystallographic object-oriented toolkit (Coot) (72), and Xtalview (73) are the three most popular programs for building the model into the electron-density maps. O allows the user to build the structure in accordance with the known geometries. Alwyn's home page provides introduction, manual, and the tutorial for the program (link in Table 1). It can build models into the electron density maps from scratch and also provides to the user the ability to define macros. Coot is a recent model building tool and is part of the CCP4 suite. Both of these programs display maps and models and can also perform real-space refinement, manual rotation, and translation of the model, rigid body fitting, rotamer search, mutations, and display Ramachandran plots. They can also perform superimposition and can be used for model validation as well. Xfit (part of Xtalview) can also be used for fitting models into electron-density maps. The program has a built-in fft routine to calculate omit maps.

Small molecules and ligand coordinates to be included in the model can be retrieved from existing structures in the ligand database of the PDB (http://ligand-depot.rutgers.edu/). New compounds can be drawn using molecule editor programs such as ChemDraw (74, 75). Once coordinates are generated for a ligand, refinement programs (see next subheading) will require files describing its stereochemical constraints and energetic parameters. These files can be generated through HIC-Up (76) or the PRODRG server (77). The crystallographic package PHENIX also contains a module called eLBOW for optimization of known or novel ligand parameters.

6. Structure Refinement

During refinement, the parameters of the model are optimized to fit the observations using a refinement function. The classic parameters that are optimized during refinement are the positions of atoms, their occupancies, and their temperature factors. There are computational strategies to achieve this, but they might not be enough, and hence, refinement is usually performed simultaneously with model building. Iterative rounds of model building followed by refinement should ultimately yield a statistically validated model.

Several functions have been developed for crystal structure refinement. The function that was originally introduced for refinement of macromolecules is the least-square method, a simple statistical function (78). Most programs such as REFMAC (CCP4 suite) still use this method together with other minimization functions. In case of CNS, the primary idea behind structure refinement is that the correct model of the protein must be that of the lowest energy. Hence, energy minimization routines are used to reduce the mean difference between observed and calculated structure factors. The maximum-likelihood refinement is now one of the most common refinement methods. It is implemented with different variants in CNS, REFMAC (79), and BUSTER/TNT (80, 81). The simulated annealing refinement (82) is a search method generating a random set of models through molecular dynamic simulation to find models outside of local energy minima. This refinement is available in CNS (83) and requires a large amount of computational time. More information about models and parameters used for refinement can be found in the literature (84).

All the refinement steps and file formatting can be done using the data conversion utilities of different programs provided by crystallographic packages (CNS, CCP4, and PHENIX). The CNS website provides online command files that can be modified and saved. PHENIX (85), Python-based Hierarchical EnviroNment for Integrated Xtallography, has been designed to provide an all-automated structure solving procedure for either MR (running Phaser) or heavy atom search (SOLVE) with automated building (RESOLVE).

6.1. A Few Guidelines for Refinement

Two statistical values must be followed during refinement, the R and the R_{free} value. Briefly, the R value refers to the traditional crystallographic R-factor, which reflects the mean difference between the structure factors F_{calc} calculated from the PDB coordinates of the model and the measured structure factors F_{obs} (see formula below).

As the model gets closer to the correct structure, the R value will drop as the F_{calc} values get close to the F_{obs}.

$$R = \frac{\sum \left| F_{obs} - F_{calc} \right|}{\sum \left| F_{obs} \right|} \quad (1)$$

The R_{free} value can help monitor the progress of refinement and to check that the R value is not arbitrarily lowered by increasing the number of model parameters. The R_{free} factor was introduced by Axel Brünger (86) and is calculated during refinement as above but with a small set of reflections that are not used in the refinement of the structural model. R_{free} correlates with phase accuracy of the atomic model. Before refinement, part of the dataset (between 5 and 10%) is set aside as the "test set," and the refinement is only done with the remaining reflections. As the refinement converges, the R and R_{free} factors both approach stable values.

Practically, in case an experimental electron-density map is available, one builds as much of the model as is possible and then starts the refinement. In case the structure has been determined by MR, the search model is used to initiate refinement. Initially, a rigid-body refinement can be carried out where individual domains or different molecules in the AU are defined as separate groups. The NCS elements when present can be included especially at the early stages of the refinement as it can greatly improve the electron-density map. CNS and REFMAC calculate the position of NCS elements and include them in the refinement process. This should be removed later on since symmetry-related molecules, even theoretically identical, can display conformational differences in flexible secondary structure elements that have to be built separately. After rigid-body refinement, the individual positions of the atoms are optimized using either the least-square method (REFMAC) or through energy minimization (CNS). It has been seen that the use of simulated annealing protocols (CNS) is especially useful in the refinement of manually built models.

At every round, the refined structure is used to calculate a new electron-density map using various difference Fourier syntheses 2 m$|F_{obs}|$ – D$|F_{calc}|$(i.e., $2F_{obs} - F_{calc}$ with $m = 2$ and $D = 1$). If the map shows some improvement compared with the original map, whether it is the initial map calculated after MR or the density map from MAD data, the model can be rebuilt in appropriate regions. Difference map ($F_{obs} - F_{calc}$) or Omit maps can be used to build missing or wrongly built parts of the structure. For most crystallographic packages, electron-density map calculation can be included right after every refinement step.

The B factors or "temperature factors" for each atom may be introduced toward the end of the refinement when most of the

peptide chain is correctly fitted in the density. Water molecules and any other small molecule or ions that might be bound in protein cavities, or at catalytic sites, can then be included in the refinement. Together with monitoring of R and R_{free} values, improvement of the electron-density map is a good indicator that the refinement is proceeding in the right direction. Refinement is generally carried out until the R and R_{free} values converge, and can be followed by translation, libration, and screw (TLS) refinement (REFMAC/PHENIX) to get a better estimate of the mean square displacements of domains within a macromolecule. The TLS takes into account the anisotropy of the B factors usually modeled as isotropic (87).

7. Model Validation and PDB Submission

Several levels of quality control should be performed to assess the completion and quality of the model. The diffraction resolution limit and the resulting quality of the electron-density map have to be considered before addition of water molecules, ions, and small molecules in the model. At 2.0 Å, one can add to the model one water molecule per amino acid (88). Catalytic or structural ions can be located in a difference map, thanks to strong residual peaks that would only disappear after refinement if the correct ions are positioned in the model. The protein chemical environment, i.e., the coordination geometry of water molecules around the suspected atom, can help identify its chemical nature. One has to keep in mind that below 3.5 Å, the identification of ions or presence of water molecules is highly hypothetical and should be carefully checked.

Model validation on the protein polypeptide chain can be performed with several programs that provide a statistical evaluation of the geometrical parameters of the structure. The most popular validation is the calculation of the Ramachandran plot or Ramachandran diagram displaying the values of the psi and phi angles of the peptide bond as well as deviations from standard bond lengths and bond angles for amino acids of a structure (89).

PROCHECK (90) and Sfcheck (91) are structure validation programs that are available through the CCP4 package as well as the JCSG websites. PROCHECK produces PostScript plots analyzing the stereochemical quality of a protein structure. CNS has a PDB submission program that calculates the Ramachandran plot and extracts information from the coordinate file and from the PDB header (refinement and data processing statistics) to format the files for PDB deposition. Formatting and validation tools are also available directly on the Protein Data Bank website.

WHAT_CHECK part of WHAT_IF program provides a number of tools for structure validation and coordinate deposition (92). More and more packages and model building programs now include model validation together in a single interface allowing immediate visualization and correction of stereochemical errors. The Molprobity program belongs to the new generation of "all-in-one" interface and allows the user to correct geometrical errors in a user-friendly interface (93).

8. Structure Analysis and Figures

8.1. Structure Analysis

X-ray crystallography is a powerful technique as it allows researchers to visualize molecules close to atomic level. Areas of interactions between protein–protein, protein–ligand, or protein–nucleic acid complexes can be systematically analyzed to extract any information that can be cross-correlated with relevant biological data. We are providing here links to only a few of the most popular programs as many software packages, databases, and web servers with new options are constantly developed for this purpose.

Detailed amino-acid side chain interactions within a protein or contacts with a DNA molecule or a small ligand can be listed using LIGPLOT, a program for automatically plotting macromolecular interactions (94). This program generates a list of interactions and distances between interacting atoms as well as a schematic representation of the interaction network. Program CONTACT from CCP4 (and contact.inp in CNS) gives a list of residues in contact and the distance between the corresponding atoms. It can be used to list protein/protein interactions including oligomer interfaces, protein–nucleic acid interaction, and protein–ligand interactions.

GRASP (Grasp2 for windows) is a program that can calculate and visualize the surface and electrostatic potential of macromolecule (95, 96). Structural pockets and cavities can be identified using the CASTp server and visualized with PyMOL (see next subheading). Similarities with other known structures can be found using the DALI server (97). The refined PDB or any individual domain of the protein can be submitted through the DALI web interface to compare with other protein 3D structures. It can reveal similarities with other proteins with none or little primary sequence homology and provide an output of the regions of the proteins that superimpose with a good scoring (Z factor).

Superposition of the refined model with other 3D structures in the databases can reveal interesting similarities or conformational differences that may be correlated with functional data. Superimpositions can be done directly in some manual building

programs such as O (lsq_exp). Equivalent and rigid regions of the two molecules that match have to be carefully selected to get a correct superposition with good RMS deviation. One can search for similar structures using the SCOP structural classification database (98) or the CATH Protein structure classification (99) in order to analyze whole structures or individual domains for structural similarities with other families or superfamilies.

Many web servers have been developed for superposition. SuperPose from CCP4 (100) generates sequence and structure alignments with RMSD statistics through a web interface (101). 3dSS is also a web-based interactive computing server and is coupled to the visualization program RASMOL (102). The algorithm RAPIDO (103) for 3D alignment of protein structures is accessible through the Hamburg EMBL synchrotron website. Providing the PDB files, this program outputs in a single web interface the primary sequence alignment of the two proteins with the superposed regions highlighted, the listing of superposed residues in the PDB files, and displays the superposition through a Jmol window (104).

8.2. Figure Preparation

Several programs are available to generate figures of structures, and more and more user-friendly programs are being developed for this purpose. RASMOL is an interactive program for visualization of 3D structures that was developed in the 1990s (105). In the same decade appeared RIBBONS that creates images of solid-shaded ribbon models of macromolecules (106). It used to run on Silicon Graphics workstations, and newer versions can now run under Windows. It can check the quality of the models through Ramachandran plots and generate publication quality images. The program MolScript, running under Unix, can also produce different types of protein representations (107) and outputs images in many image formats such as jpeg, tiff, postscript.

Dino can display 3D structures and several types of surfaces (molecular surfaces, electron microscopy surfaces, topography surfaces). Surface parameters for a crystallographic structure have to be calculated using MSMS (108) to be read in Dino and produce a png, tiff, or postcript image. Dino can be coupled with persistance of vision raytracer (POVRAY), a powerful raytracing package for a high-quality image rendering. Stereoviews and animations can also be generated. A simple web interface based on Dino has been developed where the coordinate files can be directly input for generating a simple image with different chosen parameters (colors, different types of secondary structures, and atoms' representation). However, more sophisticated images have to be generated using command scripts with Dino running under X-Windows and using OpenGL. Versions for Linux and OSX are also available. Examples based on the crystal structure described in Table 2 are shown in Fig. 2.

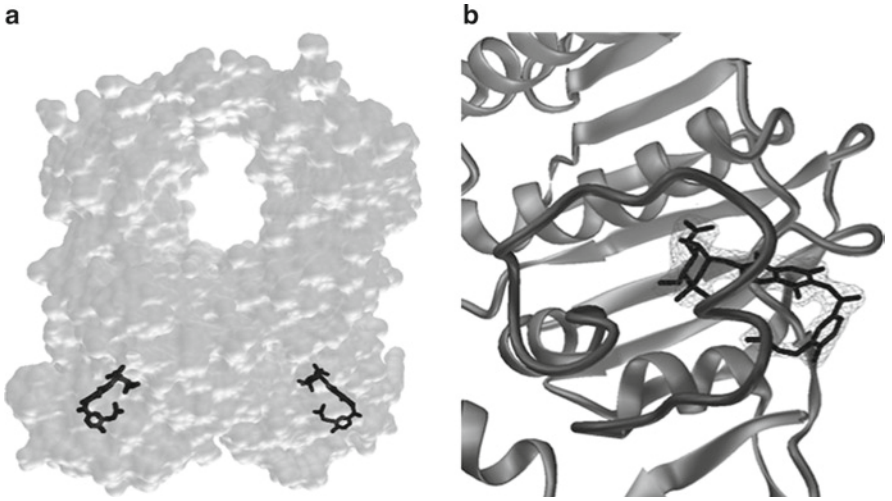


Fig. 2. Structure of Tth Gyrase 43 kDa ATPase domain in complex with the antibiotic novobiocin (39, 40). (**a**) Surface representation of the Gyr 43 kDa dimer in light gray with the novobiocin molecules positioned in the active site in black. The molecular surface was calculated using MSMS; this image has been written out of Dino directly in the .png format. (**b**) View of one of the two ATPase active sites with the protein helices and beta strands in the ribbon representation. The ATP lid closing the active site is displayed as a darker worm. The final $2F_{obs} - F_{calc}$ electron-density map contoured at 2.6σ after refinement appears as a light gray mesh around the novobiocin molecule. Figure prepared with Dino.

PyMOL is a very popular molecular visualization program that runs on a variety of platforms (Windows, Mac, and Linux/Unix) and is open source. It has a user-friendly interface with interactive menus. The user can operate the program in point and click mode (for rotating or zooming the molecule) or in command line mode for making more sophisticated images. The program can be used for viewing and analyzing 3D structures. High-resolution images showing interaction details, surfaces, and movies can be generated for presentations and publication. PyMOLwiki (link in Table 1) contains many scripts that extend the features of the program. Figure 3 displays an image generated using PyMOL of the structure presented in Table 3.

9. Conclusion

At present, there are numerous programs available for every step of crystal structure determination and representation of 3D structures, and they are constantly evolving, making it a difficult task to keep up to date with all the developments in this field. A comprehensive list of available programs can be found on the CRYSTAL website (109). The latest trend in computational tools in protein crystallography is the development of all-integrated pipelines. The principle of the new pipelines goes beyond the

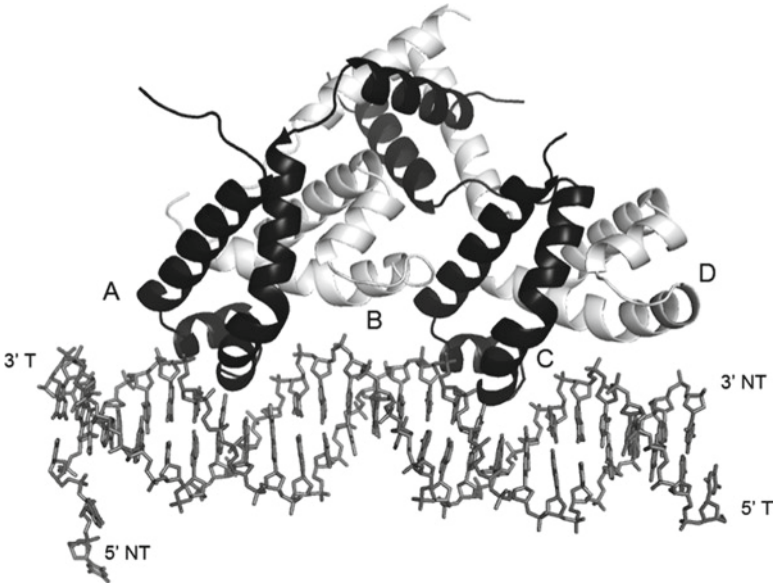


Fig. 3. Crystal structure of cII transcription factor from bacteriophage lambda in complex with DNA. The cII tetramer (each monomer being 11 kDa) is displayed bound to 27 bp double-stranded oligonucleotide (TACCTCGTTGCGTTTGTTTGCACGAAT) with TT overhang. The double-stranded DNA is represented in sticks. The 3′- and 5′-end of the template (T) and nontemplate (NT) have been denoted. The alpha carbon backbone of cII tetramer is represented in ribbon with monomer A and C in black and monomer B and D in gray. Figure prepared with PyMOL.

simple software package by guiding the user toward the right chain of programs. PHENIX, SOLVE/RESOLVE, MrBUMP, and BALBES are examples of all-in-one packages, providing tools from diffraction data analysis and phasing to structure validation. 3D structure determination platforms are now implemented on synchrotron beam lines such as the EMBL-Hamburg platform, Auto-Rickshaw (110). A new version of HKL, HKL3000, has been developed that includes all the steps from data collection processing and structure determination in a single interface with the traditional graphical features of HKL (111). These platforms can help solve structures right after data collection to help with decision-making for a better use of allocated beam time. Structural genomic initiatives with the support of synchrotron facilities and software developments have thus considerably accelerated the speed at which structures can be determined.

Acknowledgments

The authors would like to acknowledge Dr. Deepak Nair and Dr. Lasse Jenner for critical reading of the manuscript. We would like to thank Prof. Jean Cavarelli for useful discussions and suggestions about this review.

References

1. Laskowski, R. A. and Thornton, J. M. (2008) Understanding the molecular machinery of genetics through 3D structures. Nat Rev Genet 9, 141–151.
2. Chayen, N. E. and Saridakis, E. (2008) Protein crystallization: from purified protein to diffraction-quality crystal. Nat Methods 5, 147–153.
3. Doublié, S. (2007) Macromolecular crystallography protocols, vol. 1: preparation and crystallization of macromolecules, Methods in molecular biology series, vol. 363.
4. Taylor, G. (2003) The phase problem. Acta Crystallogr D Biol Crystallogr 59(Pt 11), 1881–1890.
5. Usón, I. and Sheldrick, G. M. (1999) Advances in direct methods for protein crystallography. Curr Opin Struct Biol 9, 643–648.
6. Hendrickson, W. A. (1985) Analysis of protein structure from diffraction measurement at multiple wavelengths. Trans Am Crystallogr Assoc 21, 11–21.
7. Hendrickson, W. A. (1991) Determination of macromolecular structure from anomalous diffraction of synchrotron radiation. Science 254, 51–58.
8. Hendrickson, W. A. and Teeter, M. M. (1981) Structure of the hydrophobic protein crambin determined directly from the anomalous scattering of sulfur. Nature (London) 290, 107–113.
9. Wang, B.-C. (1985) Resolution of phase ambiguity in macromolecular crystallography. Methods Enzymol 115, 90–112.
10. Borek, D., Ginell, S. L., Cymborowski, M., Minor, W., and Otwinowski, Z. (2007) The many faces of radiation-induced changes. J Synchrotron Radiat 14, 24–33.
11. Ravelli, R. B. and McSweeney, S. M. (2000) The "fingerprint" that X-rays can leave on structures. Structure 8, 315–328.
12. Banumathi, S., Zwart, P. H., Ramagopal, U. A., Dauter, M., and Dauter, Z. (2004) Structural effects of radiation damage and its potential for phasing. Acta Crystallogr D Biol Crystallogr 60, 1085–1093.
13. Otwinowski, Z. and Minor, W. (1997) Processing of X-ray diffraction data collected in oscillation mode. Methods Enzymol 276, 307–326.
14. Leslie, A. G. (2006) The integration of macromolecular diffraction data. Acta Crystallogr D Biol Crystallogr 62, 48–57.
15. Collaborative Computational Project Number 4. (1994) Acta Crystallogr D Biol Crystallogr 50, 760–763.
16. Kabsch, W. J. (1993) Automatic processing of rotation diffraction data from crystals of initially unknown symmetry and cell constants. J Appl Crystallogr 26, 795–800.
17. Pflugrath, J. W. (1999) The finer things in X-ray diffraction data collection. Acta Crystallogr D Biol Crystallogr 55, 1718–1725.
18. Popov, A. N. and Bourenkov, G. P. (2003) Choice of data-collection parameters based on statistic modeling. Acta Crystallogr D 59, 1145–1153.
19. Bourenkov, G. P. and Popov, A. N. (2006) A quantitative approach to data-collection strategies. Acta Crystallogr D 62, 58–64.
20. Matthews, B. W. (1968). Solvent content of protein crystals. J Mol Biol 33, 491–497.
21. Kantardjieff, A. A. and Rupp, B. (2003) Matthews coefficient probabilities: improved estimates for unit cell contents of proteins, DNA, and protein–nucleic acid complex crystals. Protein Sci 12, 1865–1871.
22. Wilson, A. J. C. (1942) Determination of absolute from relative X-ray intensity data. Nature 150, 151–152.
23. Rost, B., Sander, C., and Schneider, R. (1994) PHD – an automatic mail server for protein secondary structure prediction. Bioinformatics 10, 53–60.
24. Rost, B., Yachdav, G., and Liu, J. (2004) The PredictProtein server. Nucleic Acids Res 32(Web server issue), W321–W326.
25. Sali, A. and Blundell, T. L. (1993) Comparative protein modelling by satisfaction of spatial restraints. J Mol Biol 234, 779–815.
26. Rossmann, M. G. and Blow, D. M. (1962) The detection of sub-units within the crystallographic asymmetric unit. Acta Crystallogr 15, 24–31.
27. Read, R. J. (2001) Pushing the boundaries of molecular replacement with maximum likelihood. Acta Crystallogr D 57, 1373–1382.
28. Storoni, L. C., McCoy, A. J., and Read, R. J. (2004) Likelihood-enhanced fast rotation functions. Acta Crystallogr D Biol Crystallogr 60, 432–438.
29. Navaza, J. (1994) AMoRe: an automated package for molecular replacement. Acta Crystallogr A 50, 157–163.
30. Vagin, A. and Teplyakov, A. (1997) MOLREP: an automated program for molecular replacement. J Appl Crystallogr 30, 1022–1025.
31. Tong, L. and Rossmann, M. G. (1997) Rotation function calculations with GLRF program. Methods Enzymol 276, 594–611.

32. Kissinger, C. R., Smith, B. A., Gehlhaar, D. K., and Bouzida, D. (2001) Molecular replacement by evolutionary search. Acta Crystallogr D 57, 1474–1479.
33. Keegan, R. M. and Winn, M. D. (2008) MrBUMP: an automated pipeline for molecular replacement. Acta Crystallogr D 64, 119–124.
34. Long, F., Vagin, A., Young, P., and Murshudov, G. N. (2008) BALBES: a molecular replacement pipeline. Acta Crystallogr D 64, 125–132.
35. Claude, J.-P., Suhre, K., Notredame, C., Claverie, J.-M., and Abergel, C. (2004) CaspR: a web-server for automated molecular replacement using homology modelling. Nucleic Acids Res 32: W606–W609.
36. Naismith, N., Cowtan, K., Ashton, A. (2001) Molecular replacement and its relatives. Acta Crystallogr D Biol Crystallogr 57(Pt 10), 1355–1490.
37. Murshudov, G., von Delft, F., Ballard, C. (2008) Molecular replacement. Acta Crystallogr D 64(Pt 1), 1–140.
38. Toth, A. (2007) Molecular replacement, macromolecular crystallography protocols volume 2: structure determination. Methods Mol Biol 364, 121–147.
39. Lamour, V., Hoermann, L., Jeltsch, J.-M., Oudet, P., and Moras, D. (2002) An open conformation of the *Thermus thermophilus* Gyrase B ATP-binding domain. J Biol Chem 277, 18947–18953.
40. Lamour, V., Hoermann, L., Jeltsch, J.-M., Oudet, P., and Moras, D. (2002) Crystallization of the 43 KATPase domain of *Thermus thermophilus* Gyrase B in complex with novobiocin. Acta Crystallogr D 58, 1376–1378.
41. Blow, D. M. (2003) How Bijvoet made the difference: the growing power of anomalous scattering. Methods Enzymol 374, 3–22.
42. Egloff, M. P., Cohen, P. T., Reinemer, P., and Barford, D. (1995) Crystal structure of the catalytic subunit of human protein phosphatase 1 and its complex with tungstate. J Mol Biol 254, 942–959.
43. Lima, C. D., Klein, M. G., and Hendrickson, W. A. (1997) Structure-based analysis of catalysis and substrate definition in the HIT protein family. Science 278, 286–290.
44. Egli, M. and Pallan, P. S. (2007) Selenium modification of nucleic acids: preparation of phosphoroselenoate derivatives for crystallographic phasing of nucleic acid structures. Nat Protoc 2, 640–646.
45. Ramagopal, U. A., Dauter, M., Dauter, Z. (2003) Phasing on anomalous signal of sulphurs: what is the limit. Acta Crystallogr D 59, 1020–1027.
46. Boggon, T. J. and Shapiro, L. (2000) Screening for phasing atoms in protein crystallography. Structure 8, 143–149.
47. Doublie, S. (1997) Preparation of selenomethionyl proteins for phase determination. Methods Enzymol 276, 523–530.
48. Evans, G. and Pettifer, R. F. (2001) *CHOOCH*: a program for deriving anomalous-scattering factors from X-ray fluorescence spectra. J Appl Crystallogr 34, 82–86.
49. Weeks, C. M. and Miller, R. (1999) The design and implementation of SnB v2.0. J Appl Crystallogr 32, 120–124.
50. Schneider, T. R. and Sheldrick, G. M. (2002) Substructure solution with SHELXD. Acta Crystallogr D Biol Crystallogr 58, 1772–1779.
51. Xu, H. and Weeks, C. M. (2008) Rapid and automated substructure solution by *Shake-and-Bake*. Acta Crystallogr D 64, 172–177.
52. Sheldrick, G. M. (2002) Macromolecular phasing with SHELXE. Z Kristallogr 217, 644–650.
53. Pape, T. and Schneider, T. R. (2004) HKL2MAP: a graphical user interface for macromolecular phasing with SHELX programs. J Appl Crystallogr 37, 843–844.
54. Terwilliger, T. C. (2003) SOLVE and RESOLVE: automated structure solution and density modification. Methods Enzymol 374, 22–37.
55. Brunger, A. T., Adams, P. D., Clore, G. M., Gros, P., Grosse-Kunstleve, R. W., Jiang, J.-S., Kuszewski, J., Nilges, M., Pannu, N. S., and Read, R. J. (1998) Crystallography and NMR system (CNS): a new software system for macromolecular structure determination. Acta Crystallogr D 54, 905–921.
56. Yao, J.-X. (1983) On the application of phase relationships to complex structures. XX. RANTAN for large structures and fragment development. Acta Crystallogr A 39, 35–37.
57. Foadi, J., Woolfson, M. M., Dodson, E. J., Wilson, K. S., Yao, J.-X., and Zheng, C.-D. (2000) A flexible and efficient procedure for the solution and phase refinement of protein structures. Acta Crystallogr D Biol Crystallogr 56, 1137.
58. Otwinowski, Z. (1991) Maximum likelihood refinement of heavy-atom parameters. In: W. Wolf, P. R. Evans, and A. G. W. Leslie, Editors, *Isomorphous replacement and anomalous scattering, proceedings of the CCP4 study weekend*, Warrington, UK, pp. 80–86.
59. Fortelle, E. d. l. and Bricogne, G. (1997) Maximum-likelihood heavy-atom parameter refinement for multiple isomorphous replacement and multi-wavelength anomalous diffraction methods. Methods Enzymol 276, 472–494.

60. Jain, D., Kim, Y., Maxwell, K., Beasley, S., Zhang, R., Gussin, G. N., Edwards, A. M., and Darst, S. A. (2005) Crystal structure of bacteriophage lambda CII and its DNA complex. Mol Cell 19, 259–269.
61. Cowtan, K. D. and Zhang, K. Y. J. (1999) Density modification for macromolecular phase improvement. Prog Biophys Mol Biol 72, 245–270.
62. Abrahams, J. P. and Leslie, A. G. W. (1996) Methods used in the structure determination of bovine mitochondrial F_1 ATPase. Acta Crystallogr D 52, 30–42.
63. Cowtan, K. (1994) Joint CCP4 and ESF-EACBM Newsletter on Protein Crystallography. 31, 34–38.
64. Zhang, K. Y. J. (1993) SQUASH – combining constraints for macromolecular phase refinement and extension. Acta Crystallogr D 49, 213–222.
65. Terwilliger, T. C. (2000) Maximum-likelihood density modification. Acta Crystallogr D 56, 965–972.
66. Perrakis, A., Morris, R., and Lamzin, V. S. (1999) Automated protein model building combined with iterative structure refinement. Nat Struct Biol 6, 458–463.
67. Langer, G., Cohen, S. X., Lamzin, V. S., and Perrakis, A. (2008) Automated macromolecular model building for X-ray crystallography using ARP/wARP version 7. Nat Protoc 3, 1171–1179.
68. Evrard, G. X., Langer, G. G., Perrakis, A., and Lamzin, V. S. (2007) Assessment of automatic ligand building in ARP/wARP. Acta Crystallogr D 63, 108–117.
69. Terwilliger, T. C. (2003) Automated main-chain model building by template matching and iterative fragment extension. Acta Crystallogr D 59, 38–44.
70. Terwilliger, T. C., Grosse-Kunstleve, R. W., Afonine, P. V., Moriarty, N. W., Zwart, P. H., Hung, L. W., Read, R. J., and Adams, P. D. (2008) Iterative model building, structure refinement and density modification with the PHENIX AutoBuild wizard. Acta Crystallogr D 64, 61–69.
71. Jones, T. A., Zou, J.-Y., Cowan, S., and Kjeldgaard, M. (1991) Improved methods for building protein models in electron density maps and the location of errors in these models. Acta Crystallogr A 47, 110–119.
72. Emsley, P. and Cowtan, K. (2004) Coot: model-building tools for molecular graphics. Acta Crystallogr D 60, 2126–2132.
73. McRee, D. E. (1999). Xtalview/Xfit – a versatile program for manipulating atomic coordinates and electron density. J Struct Biol 125, 156–165.
74. Mills, N. (2006) ChemDraw Ultra 10.0. J Am Chem Soc 128, 13649–13650.
75. Li, Z., Wan, H., Shi, Y., and Ouyang, P. (2004) Personal experience with four kinds of chemical structure drawing software: review on ChemDraw, ChemWindow, ISIS/Draw, and ChemSketch. J Chem Inf Comput Sci 44, 1886–1890.
76. Kleywegt, G. J. (2007) Crystallographic refinement of ligand complexes (CCP4 proceedings). Acta Crystallogr D 63, 94–100.
77. Schuettelkopf, A. W. and van Aalten, D. M. F. (2004). PRODRG – a tool for high-throughput crystallography of protein-ligand complexes. Acta Crystallogr D 60, 1355–1363.
78. Konnert, J. H. (1976) A restrained-parameter structure-factor least-squares refinement procedure for large asymmetric units. Acta Crystallogr A 32, 614–617.
79. Murshudov, G. N., Vagin, A. A., and Dodson, E. J. (1997) Refinement of macromolecular structures by the maximum-likelihood method. Acta Crystallogr D Biol Crystallogr 53, 240–255.
80. Pannu, N. S. and Read, R. J. (1996) Improved structure refinement through maximum likelihood. Acta Crystallogr A 52, 659–668.
81. Bricogne, G. (1997) Bayesian statistical viewpoint on structure determination: basic concepts and examples. Methods Enzymol 276, 361–423.
82. Krikpatrick, Jr., S., Gelatt, C. D., and Vecchi, M. P. (1983) Optimization by simulated annealing. Science 220, 671–680.
83. Rice, L. M., Shamoo, Y., and Brünger, A. T. (1998) Phase improvement by multi-start simulated annealing refinement and structure-factor averaging. J Appl Cryst 31, 798–805.
84. Tronrud, D. E. (2007) Introduction to macromolecular refinement. Methods in molecular biology series, vol. 364: macromolecular crystallography protocols: vol. 2: structure determination. pp. 231–253.
85. Adams, P. D., Grosse-Kunstleve, R. W., Hung, L.-W., Ioerger, T. R., McCoy, A. J. , Moriarty, N. W., Read, R. J., Sacchettini, J. C.,. Sauter N. K., and Terwilliger, T. C. (2002) PHENIX: building new software for automated crystallographic structure determination. Acta Crystallogr D 58, 1948–1954.
86. Brünger, A. T. (1997) Free R value: cross-validation in crystallography. Methods Enzymol 277, 366–396.
87. Winn, M. D., Isupov, M. N., and Murshudov, G. N. (2001) Use of TLS parameters to model anisotropic displacements in macromolecular refinement. Acta Crystallogr D Biol Crystallogr 57, 122–133.

88. Carugo, O. and Bordo, D. (1999) How many water molecules can be detected by protein crystallography? Acta Crystallogr D 55, 479–483.
89. Ramachandran, G. N., Ramakrishnan, C., and Sasisekharan, V. (1963) Stereochemistry of polypeptide chain configurations. Int J Mol Biol 7, 95–99.
90. Laskowski, R. A., MacArthur, M. W., Moss, D. S., and Thornton, J. M. (1993) PROCHECK: a program to check the stereochemical quality of protein structures. J Appl Crystallogr 26, 83–291.
91. Vaguine, A. A., Richelle, J., and Wodak, S. J. (1999) SFCHECK: a unified set of procedures for evaluating the quality of macromolecular structure-factor data and their agreement with the atomic model. Acta Crystallogr D 55, 191–205.
92. Hooft, R. W. W., Vriend, G., Sander, C., and Abola, E. E. (1996) Errors in protein structures. Nature 381, 272.
93. Davis, I. W., Leaver-Fay, A., Chen, V. B., Block, J. N., Kapral, G. J., Wang, X., Murray, L. W., Arendall, III, W. B., Snoeyink, J., Richardson, J. S., and Richardson, D. C. (2007) MolProbity: all-atom contacts and structure validation for proteins and nucleic acids. Nucleic Acids Res 35, W375–W383.
94. Wallace, A. C., Laskowski, R. A., and Thornton, J. M. (1995) LIGPLOT: a program to generate schematic diagrams of protein–ligand interactions. Protein Eng 8, 127–134.
95. Nicholls, A., Sharp, K. A., and Honig, B. (1991) Protein folding and association: insights from the interfacial and thermodynamic properties of hydrocarbons. Proteins 11, 281–296.
96. Petrey, D. and Honig, B. (2003) GRASP2: visualization, surface properties, and electrostatics of macromolecular structures and sequences. Methods Enzymol 374, 492–509.
97. Holm, L. and Sander, C. (1998) Touring protein fold space with Dali/FSSP. Nucleic Acids Res 26, 316–319.
98. Lo Conte, L., Ailey, B., Hubbard, T. J. P., Brenner, S. E., Murzin, A. G., and Chothia, C. (2000) SCOP: a structural classification of proteins database. Nucleic Acids Res 28, 257–259.
99. Orengo, C. A., Michie, A. D., Jones, S., Jones, D. T., Swindells, M. B., and Thornton, J. M. (1997) CATH – a hierarchic classification of protein domain structures. Structure 5, 1093–1108.
100. Krissinel, E. and Henrick, K. (2004) Secondary-structure matching (SSM), a new tool for fast protein structure alignment in three dimensions. Acta Crystallogr D 60, 2256–2268.
101. Maiti, R., Van Domselaar, G. H., Zhang, H., and Wishart, D. S. (2004) SuperPose: a simple server for sophisticated structural superposition. Nucleic Acids Res 32(Web server issue), W590–W594.
102. Sumathi, K., Ananthalakshmi, P., Roshan, M. N., and Sekar, K. (2006) 3dSS: 3D structural superposition. Nucleic Acids Res 34(Web server issue), W128–W132.
103. Mosca, R. and Schneider, T. R. (2008) RAPIDO: a web server for the alignment of protein structures in the presence of conformational changes. Nucleic Acids Res 36(Web server issue), W42–W46.
104. Jmol: an open-source Java viewer for chemical structures in 3D. http://www.jmol.org/ and http://wiki.jmol.org:81/index.php/Main_Page.
105. Sayle, R. A. and Milner-White, E. J. (1995) RASMOL: biomolecular graphics for all. Trends Biochem Sci 20, 374–376.
106. Carson, M. (1997) Ribbons. Methods Enzymol 277, 493–505 (Macromolecular Crystallography, R. M. Sweet and C. W. Carter, Editors, Academic Press).
107. Kraulis, P. J. (1991) MOLSCRIPT: a program to produce both detailed and schematic plots of protein structures. J Appl Crystallogr 24, 946–950.
108. Sanner, M. F., Olson, A. J., and Spehner, J. C. (1996) Reduced surface: an efficient way to compute molecular surfaces. Biopolymers 38, 305–320.
109. Everse, S. J. and Doublié, S. (2007) Crystallographic software: a sustainable resource for the community. Methods Mol Biol 364, 273–278.
110. Panjikar, S., Parthasarathy, V., Lamzin, V. S., Weiss, M. S., and Tucker, P. A. (2005) Auto-Rickshaw – an automated crystal structure determination platform as an efficient tool for the validation of an X-ray diffraction experiment. Acta Crystallogr D 61, 449–457.
111. Minor, W., Cymborowski, M., Otwinowski, Z., and Chruszcz, M. (2006) HKL-3000: the integration of data reduction and structure solution – from diffraction images to an initial model in minutes. Acta Crystallogr D 62, 859–866.

Chapter 9

3-D Structures of Macromolecules Using Single-Particle Analysis in EMAN

Steven J. Ludtke

Abstract

Single-particle reconstruction is a methodology whereby transmission electron microscopy (TEM) is used to record images of individual monodisperse molecules or macromolecular assemblies, then sets of images of individual particles are computationally combined to produce a 3-D volumetric reconstruction. Ideally the TEM specimen will be prepared in vitreous ice (electron cryomicroscopy), but negative stain preparations may be used for lower resolution work. This technique has been demonstrated to produce structures at resolutions as high as ~4 Å, though this is not yet typical. The reconstruction process is quite computationally intensive, and several software packages are available for this task. EMAN is one of the easier to master software suites for single-particle analysis. This protocol explains how to perform an initial low-resolution reconstruction using EMAN.

Key words: Cryo-EM, Transmission electron microscopy, Single-particle analysis, Image processing, Structural biology

1. Introduction

Single-particle reconstruction is a structural biology technique for producing 3-D reconstructions of identical nano-scale objects without requiring crystallization (1). Typical targets are large proteins or macromolecular assemblies. Unlike X-ray crystallography, in general, the larger the object, the easier it is to solve, so long as the individual particles are identical at the targeted resolution. Typically ~200 kDa is viewed as a lower size limit for this technique, but exceptions to this rule are possible if sufficient contrast can be obtained. Typical nonviral targets are in the 500 kDa to 3 MDa range. Several examples of structures being reconstructed to ~4 Å resolution using this technique have been published (2–5), and subnanometer resolution can be achieved in

David Fenyö (ed.), *Computational Biology*, Methods in Molecular Biology, vol. 673,
DOI 10.1007/978-1-60761-842-3_9,

many cases, but in negative stain, or on low-end microscopes, resolutions may be limited to 15–30 Å.

A discussion of the image collection protocol is beyond the scope of this article (1, 6), but a few constraints are important to mention. The various available reconstruction packages have slightly different requirements. In general, for reconstructions performed using the EMAN software package, images should be collected over a range of defocuses typically ~1–3 μm. Unlike some software packages which encourage collecting particles into "defocus groups" of the same value, in EMAN it is advantageous to spread defocus values over a range. In general, the close-to-focus limit should be as close to focus as possible while still producing particles which can be visibly located in the resulting images, and defocus should be biased somewhat toward the close-to-focus end of the spectrum. Images containing significant astigmatism or drift (taking the resolution target into account) should be discarded.

While techniques do exist for studying particles with structural heterogeneity such as flexibility or varying ligation states in EMAN (7, 8) and elsewhere (9), that topic also is beyond the scope of this discussion. We assume that the particles being imaged exist in a homogeneous conformation in solution.

EMAN1 (10) is largely designed for UNIX-like operating systems such as Linux or Mac OS-X. While a few of the GUI programs can be used under Windows, full reconstructions are possible only under Linux/OS-X. In EMAN2 (11), full Windows support is available, but at the time of this writing, it is not yet a complete replacement for EMAN1. EMAN1 and EMAN2 each contain many programs, and may be used in a wide range of different protocols. The protocol described here is designed to complete an initial low-resolution reconstruction using single-particle reconstruction, and serves as an introduction to EMAN1 (see Note 1). After completing this protocol, the more detailed documentation provided with EMAN can be used to obtain higher resolutions. For a full description of how single-particle reconstruction works in EMAN and in other packages, the reader is referred to (10, 12–15).

2. Materials

2.1. Images of the Target Molecule/Assembly

1. Images may be collected on CCD or on film and scanned. The final Å/pixel value must be known.
2. Particles should be sufficiently visible that they can be visually located in the images (see Notes 2–4).

3. Particles should be sufficiently monodisperse that a majority are not overlapping with other particles.
4. Magnification should be selected such that the final image data is ~3× oversampled. That is, if a resolution of 9 Å is being targeted, the images should be ~3 Å/pix (see Note 16).

2.2. Adequate Computational Resources

1. For initial pre-processing, it is important to have a computer with a relatively large, high-resolution display (1,600 × 1,200 or better), with sufficient RAM (minimum 1 Gb).
2. Computational requirements are determined by the size, resolution, and symmetry of the particle. A low-resolution study of a small, moderately symmetric molecule may be completed on a desktop PC overnight, whereas a high-resolution structure of a large virus particle could take a million or more CPU-hours, requiring months on a large Linux cluster.

2.3. EMAN Installation

1. EMAN must be installed on your computer(s). The software can be downloaded and installed from http://ncmi.bcm.edu. EMAN2 may optionally be installed as well.
2. Different versions of EMAN are provided for individual workstations and Linux clusters as well as different platforms. Regardless of whether you have access to a cluster for running the large scale refinements, you will also need an appropriate desktop PC for the initial stages of the process.

3. Methods

Single-particle reconstruction can be broken down into a sequence of major steps: image assessment, particle picking, 2-D analysis, initial model generation, and final refinement. To move beyond ~20 Å resolution, the contrast transfer function (CTF) must also be corrected, but that is beyond the scope of this protocol.

To begin this protocol, an empty directory should be created, and all of the raw micrographs or CCD frames should be copied into it. All images should be at the same magnification from the same instrument under similar conditions.

3.1. Overview of EMAN

1. Rather than a single integrated GUI (graphical user interface), EMAN consists of a range of command-line programs, and several independent GUIs for specific purposes. The major GUI programs include : *eman, boxer, ctfit, v2,* and *v4* (see Note 5).
2. All of the EMAN GUI programs share some common features. When an image display window or a plot is open, clicking

the middle mouse button on the window will cause a control panel window to open, offering adjustments such as brightness and contrast, permitting snapshots to be saved, etc. The right mouse button is generally used to move the image within the display. Mac users are encouraged to obtain a three button mouse, though use of modifier keys with the single mouse button can often substitute for the other buttons.

3. There are many command-line programs in the package. As a general convention, typing the name of a command followed by "help" will produce documentation for that program.
4. The generic programs *iminfo, proc2d,* and *proc3d* are useful utilities for generic image processing and format conversion (see Note 15). EMAN can read and write virtually all TEM file formats.
5. While we attempt to fully describe the single-particle reconstruction process here, if a more comprehensive discussion is desired, running *eman*, and clicking on the step1–4 buttons will produce a fairly detailed, though somewhat out of date, tutorial extending to full, high-resolution reconstructions. (see Note 6 workflow).

3.2. Image Assessment

1. Before beginning to select particles, it is worthwhile to first make a preliminary assessment of the images, and eliminate those which are clearly of low quality. Typically a project would start with a minimum of 30–50 potentially usable images. While low-resolution reconstructions are much more tolerant of astigmatism and drift than high resolution work, it is still best to keep only the best images to avoid possible artifacts.
2. While a trained microscopist can often detect high levels of drift or astigmatism in the images by eye, a better assessment method is to examine the Fourier transform of each image. This can be done by running *eman,* and selecting "Browse Files/History." The resulting file browser can be used to display each of your raw images.
3. When displaying a large image like a CCD frame "Big View Required" will appear in the image display in the browser. To see the image, press the "Detach" button, which will open the image in a large window.
4. The first step in assessment is to observe the contrast level of the particles in the image. A middle-click on the image will permit adjustment of the brightness and contrast of the image to optimize visibility of the particles. Particles should be fairly clear and largely be well separated from each other, though some local aggregation is inevitable in most specimens (Fig. 1). In addition, the images should not be so far from

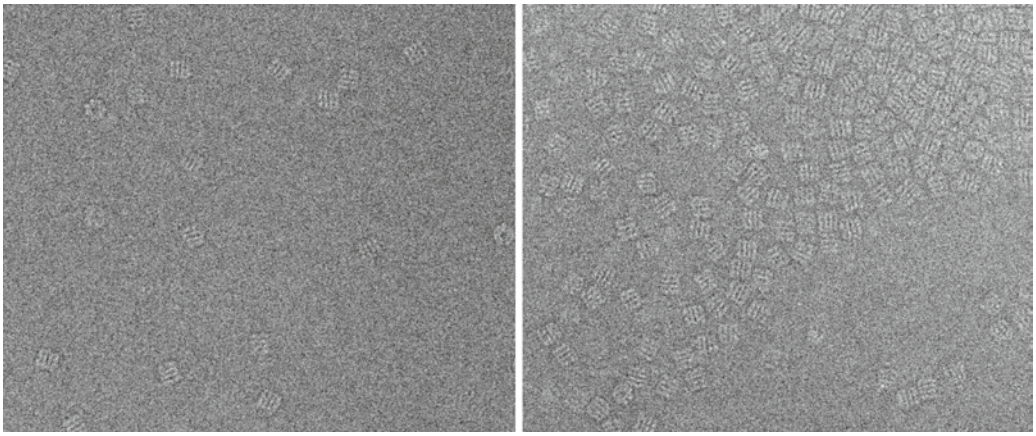


Fig. 1. GroEL in vitreous ice. Both of these images exhibit good contrast, but the *left* image is properly monodisperse, whereas the image on the *right* exhibits much too high a concentration and would not be processed.

focus that internal detail in the particles, or the shape of the particles is obscured. Images with either poor contrast or those too far from focus should be discarded.

5. Next, the "FFT" box in the control panel can be checked. This will display the power spectrum of the image. A detailed discussion of the interpretation of such power spectra is beyond this manuscript, but certain asymmetries in the power spectrum can indicate astigmatism or drift and indicate images, which should be discarded. Though not used here the program *ctfit* can also be useful in this process.

3.3. Particle Picking

1. The image assessment process will typically eliminate anywhere from one to three quarters of the raw micrographs. The next step is to locate particles within each image.
2. In the initial steps only a few of the best micrographs are used, with the goal of initially picking 1,000–2,000 high contrast particles. At this stage it may not be entirely clear what is and is not a particle in the images, and anything which may be a particle should be selected. This will be resolved in Subheading 3.4, after which Subheading 3.3 is revisited for improved particle picking.
3. Prior to selecting particles, the raw images should be normalized. That is, the mean and standard deviation of the images should be adjusted, and optionally the contrast may be inverted. For each image : "*proc2d* <*imagefile*> <*imagefile*> *edgenorm inplace [invert]*". If the images are in .DM3 format (see Note 15), the second filename should be replaced with the ".mrc" extension, and the "image.mrc" file should be used in the next step. The invert option should be specified if

your particles appear dark on a white background. If the particles appear white on a dark background, they do not require inversion. For images that are too oversampled, the shrink=<n> option may be used to reduce the sampling by an integral factor of <n>, and increase Å/pix by the same factor.

4. Executing "*boxer <imagefile>*" will cause three windows to open (see Note 7). One window is the control panel with various input fields and a menu, the second contains the micrograph, and the third will initially be empty. This program must be run once for each micrograph to be boxed.
5. In general the box size should be about 1.5× larger than your particles. "Measure" in the control panel, can be used to estimate the size of your particle in pixels by left-dragging in the image window. The longest axis of a representative particle should be measured, then multiplied by 1.5, and finally rounded down to a "good" size. "Good" box sizes are those with prime factors less than 11 and divisible by 8. "Good" sizes include the following: 40, 48, 64, 80, 96, 112, 128, 144, 160, 192, 256. Once determined, the size should be entered in the boxer control panel, and the same size used thereafter for each micrograph.
6. Particles must now be selected from the image. The image can be panned by right-dragging on the image, or using the panning widget in the control panel. The zoom factor can be adjusted in the boxer control panel. Particles may be selected either manually or semiautomatically at this point. For manual picking, "Select" should be toggled in the control panel, then individual particles must be manually clicked on in the micrograph view. Each particle will appear in the third window as it is selected. Left-dragging can be used to properly center each particle. Bad particles can be deleted using the "Delete" mode in the boxer control panel.
7. For semi-automatic selection, which should be followed by manual pruning, 3–5 particles should be selected manually, then "Autobox" from the "Boxes" menu in the control panel can be used. This will cause a window with four sliders to appear, and some additional particles to be automatically selected. At this point the magnification of the image should be reduced so the entire micrograph is visible in the image display. The automatically picked particles are confined to a region adjacent to the first selected reference. As the sliders are adjusted, it will update the automatically picked particles in this region. The first slider is used to decide how closely the potential particles must match the references. The second and third sliders are used to exclude potential particles whose

contrast is too high or too low. Once the sliders have been adjusted for the preview region, "OK" will trigger autoselection of the entire micrograph.

8. This process should continue until a total of ~1,000 particles have been selected from the set of best micrographs. After each micrograph is complete "Save Boxed Particles" and "Save Box DB" must be selected from the "Boxes" menu. The default filenames are appropriate for both files. *Boxer* can then be exited and restarted for the next micrograph.

3.4. 2-D Analysis

1. To proceed with 2-D analysis the boxed-out particles from the individual images must be combined into a single-particle stack file. This is best done in a subdirectory to avoid having too many output files in one place. Type "*mkdir r2d*; *cd r2d*". The most efficient method for producing a usable stack file is to type "*lstcat.py all.lst ../*.hed*" where "../*.hed" will reference all of the boxed-out particle files you saved in the previous step. This should be followed by "*lstfast.py all.lst*". These operations produce a text file "all.lst" which EMAN will treat as an image file containing all of the particles you have selected.
2. Filtering and/or further downsampling the particle data is recommended for this preliminary analysis. This may produce improved results in addition to speeding the process. The following command will perform a low-pass filter and downsample by a factor of 2: "*proc2d all.lst start.hed apix=<A/pix> lp=20 shrink=2 edgenorm*".
3. Next, the program refine2d.py must be applied to the particle stack. Execute "*refine2d.py all.lst –iter=8 --ninitcls=50*". For those with multi-core workstations, "*--proc=<ncores>*" may be appended to the command for more speed.
4. A number of files will be produced by this command. The file containing the final results is called "iter.final.img" (see Note 15).
5. Display iter.final.img either with the *eman* browser or by running "*v2* iter.final.img" (Fig. 2). Class-averages should represent a variety of different views of the particle under study (see Notes 8 and 9). Class-averages representing contamination, or other undesirable images may also appear.
6. Now that a better idea of what is present in the images has been obtained, the next step is to return to 3.3 and complete the particle selection process for all of the micrographs (see Note 10). If there are a significant number of "bad" class-averages, it is best to also reselect the particles from the micrographs already completed, being more conservative in the selection process. If reboxing is performed, be sure to remove

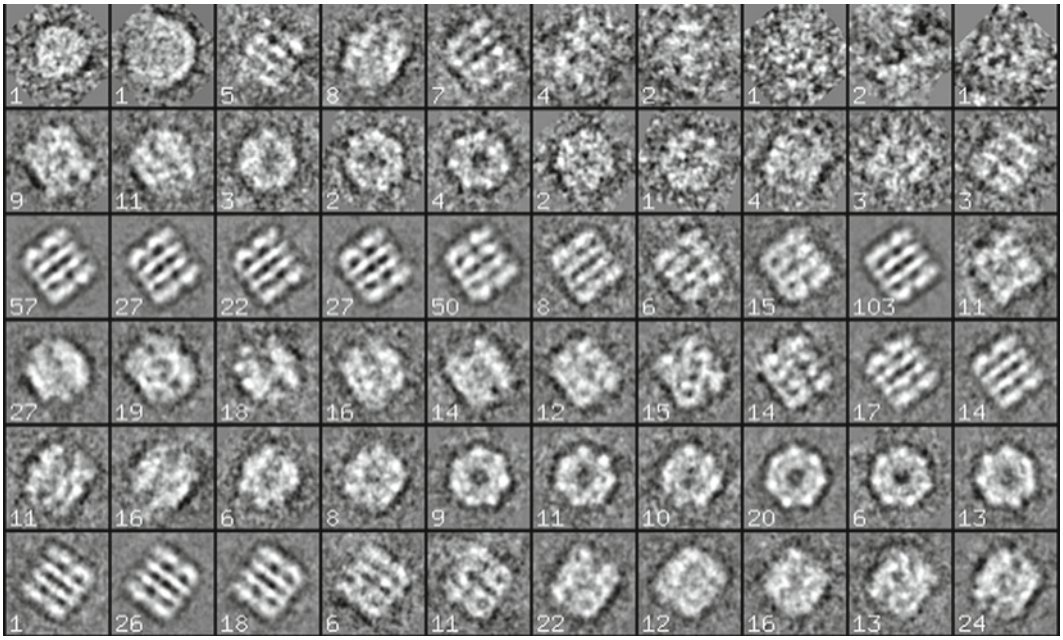


Fig. 2. Class-averages produced from a set of 838 GroEL particles. The number in the corner of each class-average indicates how many particles the average was constructed from. Note that most of the very poor images or images with contamination came from a small number of particles. Also note that GroEL exhibits a fairly strongly preferred orientation, showing many of the characteristic rectangular side views and many of the sevenfold symmetric top views, but very few orientations in between. While this is not optimal, the wide range of different side views is sufficient to obtain an unambiguous 3-D reconstruction.

existing .hed and .img files before beginning. For the low-resolution reconstruction outlined here, 2,000–3,000 particles in total should be more than sufficient (perhaps as many as 5,000 for asymmetric objects).

3.5. Initial Model Determination

1. There are several methods for producing an initial model which will refine to an accurate structure. There is also considerable controversy in the community over how good an initial model needs to be. In EMAN, we believe even abstract shapes or random patterns will have a strong propensity to refine to the correct structure. However for each particle, there will be a small number of "local minima," incorrect structures which the refinement process may "stick" at if obtained. Rather than resort to difficult experimental methods such as random conical tilt, we take the approach of simply refining several random starting models, and assessing the final results of each refinement. Generally one or more of the refinements will lead to the correct solution (see Notes 11 and 17).
2. *makeinitialmodel.py* can be used to produce a manually specified or randomly generated initial model. Simply executing the

program will prompt for the necessary information. The starting model must use the same box size as the particle data, and in general should be approximately the same size as the particle. From the class-averages in Subheading 3.4, it may be fairly obvious what shape the particle has, and one reasonable starting model would be something vaguely similar to this shape. A random model is also quite acceptable (see Note 12).

3. If a structure has or may have symmetry, "*proc3d model.mrc model.mrc sym=<sym spec>*" will impose it. "<sym spec>" may be one of c<n>, d <n>, icos, tet or oct, for example, "sym=d7" could be used for GroEL at low resolution. (see Note 13).
4. The resulting starting model will be written to model.mrc. The model can be visualized in projection using "*v4 model.mrc*" or in isosurface display using UCSF Chimera (http://www.cgl.ucsf.edu/chimera/) or in EMAN2 using "*e2display.py model.mrc*".
5. It is best to run refinements in a subdirectory as with 2-D refinement: "*mkdir initial1; cd initial1*". The starting model must be called threed.0a.mrc, but again, we would like to reduce the size for speed at this point so: "*proc3d ../model.mrc threed.0a.mrc meanshrink=2*".
6. The particles must also be copied into this directory and reduced: "*lstcat.py all.lst ../*.hed*" followed by "*proc2d all.lst start.hed shrink=2 apix=<A/pix> lp=20 edgenorm*".
7. Now a refinement can be run: "*refine 8 mask=<boxsize/4> hard=35 ang=7.5 pad=<see below> classkeep=1 classiter=5 xfiles=<A/pix x2>,<mass in kDa>,99 phasecls [sym=<sym spec>] [proc=<maxproc>]*". "pad=" should be set to the original box size*3/4 rounded to the nearest "good" size. "mask=" should be the original box size/4. If there is no symmetry, "ang=" may be increased to 9 for speed. For very high symmetries such as icosahedral ang= may be reduced to 5. This process may take some time to run depending on the symmetry, box size, and speed of your processor.
8. When the refinement is complete there will be a large number of different files in the directory. The primary files of interest are "threed.?a.mrc" and "classes.?.img".
9. As a first step in assessing the refinement results, "*v4 threed.?a.mrc*" will open one window for each iteration of the reconstruction process, which can be rotated together. The last window will represent threed.8a.mrc, and contains the final results of the refinement run. Ideally, there will be little difference between threed.7a.mrc and threed.8a.mrc. If there are still significant changes from one iteration to the next you may consider running additional iterations of refinement. Running the same "*refine*" command, but replacing "8" with

"12" will continue the refinement process through 12 iterations (for example).

10. The next step is to assess the self-consistency of the reconstruction using the *eman* browser or *v2* to look at classes.8.img (Fig. 3). This file contains pairs of projections of the 3-D model and class-averages generated from the particles. Ideally each adjacent pair of images should be identical, although the second image will inevitably be noisier, as less averaging has been performed. Some orientations may have few particles, meaning these averages will be quite noisy and may not look much like the corresponding projection. This is harmless, as such averages are excluded from the reconstruction. However, if there are several projections with strong averages which match the projection poorly, this is an indication of an incorrect model.
11. Regardless of whether an apparently good starting model was produced, this process, starting at step 2 should be repeated multiple times (using a new directory, initial# for each try). After several tries, the results should be assessed. Ideally, more than one of the refinements will have produced basically the same, clearly accurate, structure. If no clearly correct result has been obtained, the process may be continued additional times. If a self-consistent result still cannot be obtained, there is a possibility that the data contains structural heterogeneity, and cannot form a single self-consistent structure. In this situation other methods may be considered (7–9). Alternatively, contacting the EMAN developers may be worthwhile.

3.6. Refinement

1. Once a reliable initial model has been obtained, a full reconstruction can be completed. This is virtually identical to the refinement process in Subheading 3.5, except the fully sampled data is used, and thus the refinements will be more time consuming.
2. A suitable empty subdirectory should, once again, be created "*mkdir refine1; cd refine1*", and the particle data prepared: "*lstcat.py start.lst ../*.hed; lstfast.py start.lst*". Once again, the data should be low-pass filtered to roughly the first zero of the CTF, which we will assume is at ~20 Å: "*proc2d start.lst start.hed apix=<A/pix> lp=20*".
3. Next, copy threed.8a.mrc from whichever directory contained the best model to be the starting model for this refinement, also returning it to the original box size: "*proc3d ../initial3/threed.8a.mrc threed.0a.mrc scale=2.0 clip=<boxsize>,<boxsize>,<boxsize>*", replacing initial3 with the correct directory. <boxsize> is the original, unreduced box size in pixels.

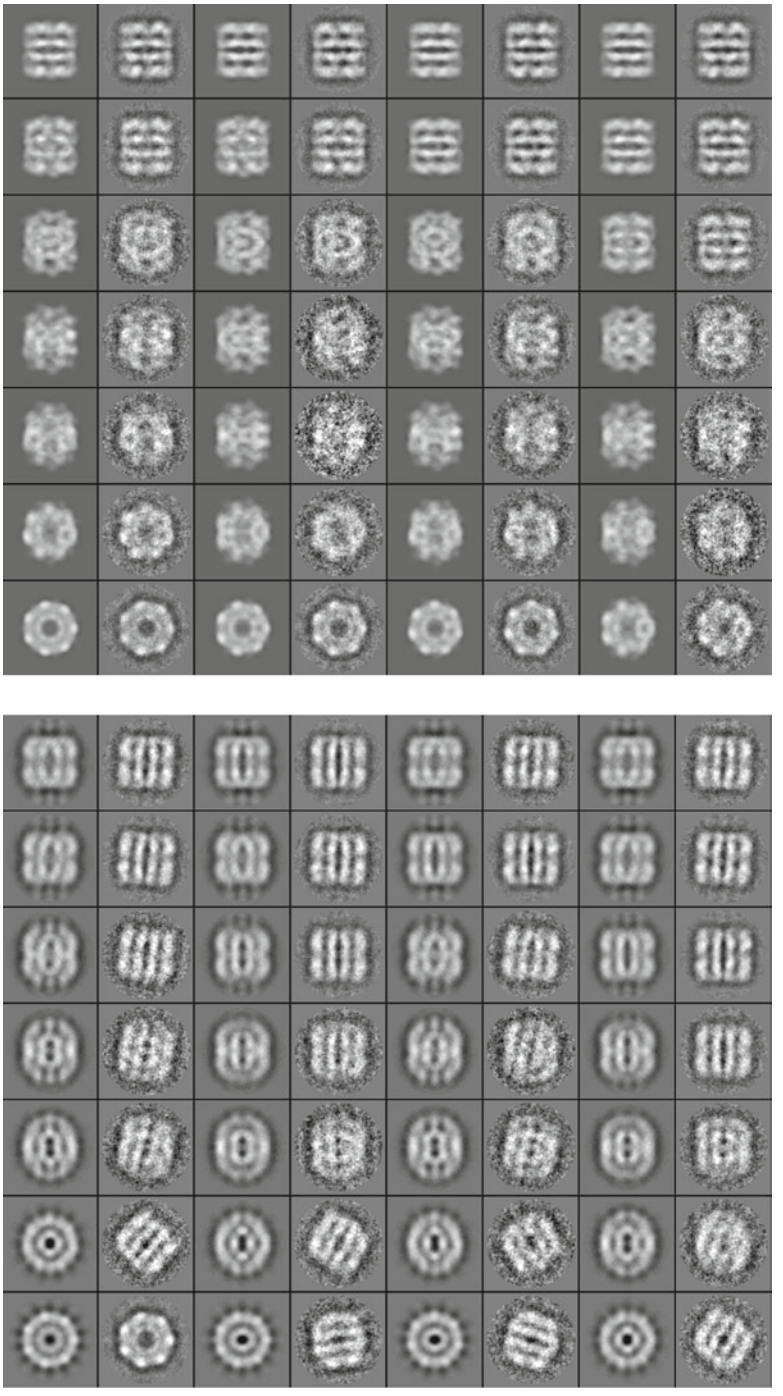


Fig. 3. Comparison of projections and class-averages for a correct *top* and an incorrect *bottom* reconstruction. Note that in the *top* set, there is excellent agreement between each horizontal pair of images. In the *bottom* set, while many of the pairs match well, many also do not. Due to the iterative refinement strategy, some of the projections and averages will always agree, but for a structure to be correct, all of the high contrast averages should agree with their projections.

4. Finally, we are ready to run the refinement: "*refine 8 mask=<boxsize/2> hard=25 ang=<ang> pad=<as above x2> classkeep=1 classiter=3 xfiles=<A/pix>,<mass in kDa>,99 phasecls [sym=<sym spec>] [proc=<maxproc>]*". This is very much like the refinement above, except our box size is now twice as large. "ang=" may also be reduced somewhat to produce finer angular sampling and thus more projections. Since CTF correction still is not being performed, "ang=5" is probably sufficient. "classiter=" has also been reduced from 5 to 3, which provides less protection from model bias (16), but will produce higher resolution reconstructions. There are many other documented options which may be added for potentially improved results, such as "amask=", "usefilt", and "fscls".
5. Once the refinement is complete (this will take as much as ~10–20× longer than the earlier refinement), in addition to examining the output files as above, the resolution of the model should be evaluated. This process is only marginally useful without CTF correction, but should still be completed. The standard resolution assessment method in single-particle analysis is to split the particle data into even and odd halves, and do a 3-D reconstruction for each half, then compare them with a Fourier shell correlation (FSC) function. To produce the two reconstructions: "eotest *mask=<boxsize/2> hard=25 pad=<as above x2> classkeep=1 classiter=3 xfiles=<A/pix>,<mass in kDa>,99 phasecls [sym=<sym spec>] [proc=<maxproc>]*". The options are a subset of the options used for *refine*, though this command will take only a short time to complete.
6. To perform the FSC comparison, execute the *eman* browser and select "Convergence" from the "Analysis" menu. This will run some computations, then prompt for an Å/pixel value. After providing this, a plot will appear. This plot will contain one dark line and a number of thinner, lighter lines. The dark line represents the FSC resolution test.
7. Ideally, this FSC curve will begin (low resolution) at 1.0, at some resolution it will begin falling toward zero, and it will oscillate randomly around zero until the end of the curve (high resolution). In some cases, the curve will fall, but will not reach zero, and may even move higher again. This can be caused by either insufficient sampling (ang=too large), aggressive masking (primarily if the amask=option is used aggressively in refinement or if the box size is too small) or other artifacts. If the curve falls to zero, then the resolution can be estimated as the point at which the FSC falls below 0.5 (see Note 14).
8. The other thinner curves in this plot are not an indication of resolution, but rather of convergence. These curves compare each iteration with the previous iteration in the refinement

process. As the refinement progresses, these lines should gradually move to higher resolution (right) and higher FSC scores (up). When convergence has been reached, from one iteration to the next, the curves will remain basically the same. If convergence has not been reached, additional refinement iterations (step 4) should be executed by increasing the parameter 8 immediately after the "refine" command.

9. The final reconstruction is the highest numbered threed.?a.mrc file.

Note that we have sidestepped the process of CTF correction which is required to achieve a high resolution reconstruction. In addition, this structure will lack CTF amplitude correction, meaning there will be some subtle localized distortions even at low resolution. However, this protocol should have at least produced a low-resolution structure with the correct overall shape, and make a suitable starting point for future CTF corrected reconstructions. The more complicated protocols for CTF corrected reconstruction, running on clusters and handling large numbers of particles are discussed in the built-in tutorial in *eman*, the workflow interface in EMAN2 and in earlier publications (10, 17, 18).

4. Notes

1. As of this writing, EMAN is undergoing a major transition from EMAN1 to EMAN2 (11). While EMAN2 will eventually obsolete EMAN1, and it contains a workflow interface which dramatically simplifies the reconstruction process, it is currently still in development. The overall reconstruction strategy described here for EMAN1 will largely still apply in EMAN2. Notes have been added where significant differences exist, or where EMAN2 may be more suitable. It is completely safe to install EMAN2 within the same user account as EMAN1. All EMAN2 programs begin with the prefix "e2" to avoid naming conflicts between EMAN1 and EMAN2.
2. It can be difficult to optimize experimental conditions to produce the necessary monodisperse particles with good contrast. Many routinely used buffer components have a substantial negative impact on image contrast in cryo-EM experiments. The most important of these is glycerol. The presence of even a small percentage of glycerol can dramatically reduce imaging contrast, and should be eliminated, if at all possible. Detergent, added to stabilize membrane proteins, can also

cause substantial difficulties, and while clearly it cannot be eliminated, reducing its concentration as much as possible without making your particles unstable is an important step. Detergent at concentrations above CMC will produce micelles, which may be visible in the images, and can be confused with the target particles in some situations. The key to remember is that anything present in the specimen will appear in the electron micrographs regardless of whether it is biochemically inert.

3. One common approach for difficult specimens is to use a continuous carbon substrate, but it is important to be aware that this carbon will add to the noise level of the images, and frequently leads to a preferred particle orientation, which in some cases can make a reliable 3-D reconstruction impossible.
4. If you are experiencing problems with preferred particle orientations in the absence of a continuous carbon substrate, one possible solution is to add a very low concentration (well below CMC) of detergent, which may help prevent hydrophobic patches on the particles from sticking to the surface of the buffer.
5. In EMAN2, the main GUI programs are *e2desktop.py, e2workflow.py, e2boxer.py, e2ctf.py*, and *e2display.py*.
6. In the future, the program *e2workflow.py* in EMAN2 will take you step by step through the reconstruction process including CTF correction, but as of this writing it is not yet complete, and cannot yet be run on linux clusters.
7. The EMAN2 program *e2boxer.py* can perform interactive semiautomatic particle picking on several micrographs at once, and is a good an alternative to *boxer*. A number of excellent non-EMAN particle pickers also exist (19). The *proc2d* command can be used for file format conversion even if a non-EMAN particle picker is used.
8. Reference free class-averages should be closely examined for signs of significant dynamics in the particle. If several class-averages seem to be in the same orientation, but one domain is undergoing substantial motion, this may be a sign of a structurally heterogeneous particle. If the motion is relatively small and localized, 3-D reconstruction may still be possible using the method presented here. For larger motions or other forms of heterogeneity, see (7–9).
9. Preferred orientation can be a significant problem. If the class-averages seem to largely represent a single view of the specimen, it may be impossible to achieve an accurate 3-D reconstruction. While it is not necessary to have all possible orientations, the minimum requirement for a complete 3-D reconstruction is to have particles covering at least one

tomographic series of orientations, i.e., a series of particle orientations covering a 180° rotation of the object from any one arbitrary orientation.

10. The "boxes" menu in *boxer* also has an "autobox from references" option. Better particle picking may be achieved by preparing a set of references from the *refine2d.py* results, and using these to rebox the micrographs. After an initial 3-D refinement, projections of the model can also be used in this process with *makeboxref.py*. Note 7 should also be considered, however.
11. It is theoretically impossible to determine absolute handedness from untilted single-particle data, since the recorded images represent near ideal projections of the object. To determine absolute handedness experimentally, some other protocol, such as tomography or random conical tilt, must be followed. At sufficiently high resolutions, however, it may be possible to determine handedness by comparison of fragments to X-ray crystal structures or homology models. At even higher resolutions (~4 Å), the pitch of the alpha-helices may become visible, also solving the handedness problem.
12. For particles with greater than threefold rotational symmetry, the program *startcsym* can be used to generate initial models from raw particle data. Prior to use, the particles should first be centered with *cenalignint*. In some cases the "fixrot=90" option will need to be specified to get an accurate model. Also note that for Dn symmetries, the symmetry must still be specified as Cn.
13. Generally the class-averages produced by *refine2d.py* will give some idea of what symmetry is present, but in general, when dealing with objects of unknown symmetry, the best approach is to try refining with the suspected symmetry, assess the results, then relax the symmetry for a few refinement iterations to see how the model changes. The process can then be repeated for another symmetry choice and compared.
14. The threshold value to use for resolution assessment has been hotly debated over the years. Other values that have been used include 0.33 (20), 0.143 (21), and use of a sigma curve (22). It is generally agreed among reviewers, however, that FSC curves should be included in the supplementary data when publishing. This serves the dual purpose of allowing the reviewer to assess the shape of the FSC curve in addition to applying a threshold of their choice.
15. Note that in many cases EMAN1 will use the IMAGIC file format by default. IMAGIC files separate images into two parts, a ".hed" file containing image header information, and a ".img" file containing the actual image data. These two files

exist as a pair, and one file must never be renamed, moved, or deleted without also removing its companion. When issuing EMAN commands, either of these files may be specified. EMAN also supports most other TEM formats such as MRC, SPIDER, TIFF, DM3, PIF, etc. The Gatan .DM3 format is supported, but is read-only.

16. Image sampling can be quite important. Generally speaking, interpretable images can only be obtained up to resolutions ~3× the pixel size. That is, a 3 Å/pixel image can at best produce a 9 Å resolution reconstruction. However, sampling too finely can also lead to a number of problems. In addition to the obvious issue of computational inefficiency, certain algorithms make assumptions about the sampling level. While some additional oversampling, perhaps as much as 5×, should be fine, beyond this point reconstructions may actually become worse, not better.
17. EMAN2 has a program called e2initialmodel.py, which can automate the entire task of initial model determination from class-averages, but using it will require some familiarity with EMAN2.

References

1. Frank J. Three-dimensional electron microscopy of macromolecular assemblies: visualization of biological molecules in their native state. New York: Oxford University Press, 2006.
2. Ludtke SJ, Baker ML, Chen DH, Song JL, Chuang DT & Chiu W. De novo backbone trace of GroEL from single particle electron cryomicroscopy. Structure 2008;16:441–8.
3. Jiang W, Baker ML, Jakana J, Weigele PR, King J & Chiu W. Backbone structure of the infectious epsilon15 virus capsid revealed by electron cryomicroscopy. Nature 2008;451:1130–4.
4. Zhang X, Settembre E, Xu C, et al. Near-atomic resolution using electron cryomicroscopy and single-particle reconstruction. Proc Natl Acad Sci USA 2008;105:1867–72.
5. Yu X, Jin L & Zhou ZH. 3.88 Å structure of cytoplasmic polyhedrosis virus by cryo-electron microscopy. Nature 2008;453:415–9.
6. Glaeser RM. Electron crystallography of biological macromolecules. New York: Oxford University Press, 2007.
7. Chen DH, Song JL, Chuang DT, Chiu W & Ludtke SJ. An expanded conformation of single-ring GroEL–GroES complex encapsulates an 86 kDa substrate. Structure 2006;14:1711–22.
8. Brink J, Ludtke SJ, Kong Y, Wakil SJ, Ma J & Chiu W. Experimental verification of conformational variation of human fatty acid synthase as predicted by normal mode analysis. Structure 2004;12:185–91.
9. Leschziner AE & Nogales E. Visualizing flexibility at molecular resolution: analysis of heterogeneity in single-particle electron microscopy reconstructions. Annu Rev Biophys Biomol Struct 2007;36:43–62.
10. Ludtke SJ, Baldwin PR & Chiu W. EMAN: semiautomated software for high-resolution single-particle reconstructions. J Struct Biol 1999;128:82–97.
11. Tang G, Peng L, Baldwin PR, et al. EMAN2: an extensible image processing suite for electron microscopy. J Struct Biol 2007;157:38–46.
12. Ludtke SJ, Jakana J, Song JL, Chuang DT & Chiu W. A 11.5 A single particle reconstruction of GroEL using EMAN. J Mol Biol 2001;314:253–62.
13. Frank J, Radermacher M, Penczek P, et al. SPIDER and WEB: processing and visualization of images in 3D electron microscopy and related fields. J Struct Biol 1996;116:190–9.
14. van Heel M, Harauz G, Orlova EV, Schmidt R & Schatz M. A new generation of the IMAGIC image processing system. J Struct Biol 1996;116:17–24.
15. Grigorieff N. FREALIGN: high-resolution refinement of single particle structures. J Struct Biol 2007;157:117–25.

16. Stewart A & Grigorieff N. Noise bias in the refinement of structures derived from single particles. Ultramicroscopy 2004;102:67–84.
17. Ludtke SJ, Chen DH, Song JL, Chuang DT & Chiu W. Seeing GroEL at 6 A resolution by single particle electron cryomicroscopy. Structure 2004;12:1129–36.
18. Saad A, Ludtke SJ, Jakana J, Rixon FJ, Tsuruta H & Chiu W. Fourier amplitude decay of electron cryomicroscopic images of single particles and effects on structure determination. J Struct Biol 2001;133:32–42.
19. Zhu Y, Carragher B, Glaeser RM, et al. Automatic particle selection: results of a comparative study. J Struct Biol 2004;145: 3–14.
20. Penczek PA. Three-dimensional spectral signal-to-noise ratio for a class of reconstruction algorithms. J Struct Biol 2002;138: 34–46.
21. Rosenthal PB & Henderson R. Optimal determination of particle orientation, absolute hand, and contrast loss in single-particle electron cryomicroscopy. J Mol Biol 2003;333:721–45.
22. van Heel M & Schatz M. Fourier shell correlation threshold criteria. J Struct Biol 2005; 151:250–62.

Chapter 10

Computational Design of Chimeric Protein Libraries for Directed Evolution

Jonathan J. Silberg, Peter Q. Nguyen, and Taylor Stevenson

Abstract

The best approach for creating libraries of functional proteins with large numbers of nondisruptive amino acid substitutions is protein recombination, in which structurally related polypeptides are swapped among homologous proteins. Unfortunately, as more distantly related proteins are recombined, the fraction of variants having a disrupted structure increases. One way to enrich the fraction of folded and potentially interesting chimeras in these libraries is to use computational algorithms to anticipate which structural elements can be swapped without disturbing the integrity of a protein's structure. Herein, we describe how the algorithm Schema uses the sequences and structures of the parent proteins recombined to predict the structural disruption of chimeras, and we outline how dynamic programming can be used to find libraries with a range of amino acid substitution levels that are enriched in variants with low Schema disruption.

Key words: Chimera, Directed evolution, Dynamic programming, Optimization, Protein design, Recombination

1. Introduction

Proteins are widely used for synthetic biology applications, but they often do not exhibit the functional properties desired for engineered biological systems. However, protein variants are thought to exist in protein sequence space that meet the specifications of almost any artificially engineered biological system imaginable. Evidence for this comes from studies using knowledge-based protein design, which have identified proteins with structures and functions distinct from those observed in nature (1, 2). Unfortunately, our understanding of protein sequence-structure–function relationships is not yet sophisticated enough to consistently alter protein functions rationally, especially when the design goal is to optimize a preexisting property. Directed

David Fenyö (ed.), *Computational Biology*, Methods in Molecular Biology, vol. 673,
DOI 10.1007/978-1-60761-842-3_10, © Springer Science+Business Media, LLC 2010

evolution, in contrast, has repeatedly proven effective at protein optimization when applied alone and when used with knowledge-based mutagenesis (3). In this approach, a selection (or screen) is used to sieve through libraries of artificial protein variants to find those rare mutations that lead to desired changes in function.

Directed evolution has several limitations that must be considered when engineering a protein. Typically, functional proteins cannot be fished out of libraries encoding random protein sequences. The frequency with which functional proteins occur in protein sequence space is thought to be miniscule compared with the maximum number of protein variants that can be evaluated in a given experiment (4). One way to improve your chances of discovering proteins with a desired function is to increase the fraction of folded variants in your combinatorial library. This can be achieved by infusing into your library design some knowledge about the protein(s) used as parents for directed evolution. This information can draw from our understanding of protein stability (5), family sequences (6), structure–function relationships (7), and laboratory evolution experiments (8). In this chapter, we describe how one can use sequence, structural, and thermodynamic information to enrich the fraction of functional protein variants in a library created using protein recombination. A protocol is outlined for using the Schema algorithm to identify libraries with a user-defined level of amino acid substitutions that minimize structure disruption (9–11).

2. Materials

2.1. Protein Structural Coordinates

With Schema, the three-dimensional structural coordinates from one of the parent proteins being recombined are required to estimate chimera disruption (see Note 1). Only those sequence positions with defined structural coordinates are considered in the calculation of structural disruption (see Note 2). In cases where there are no structural reports for the proteins being recombined, structural coordinates can be generated using algorithms that generate homology models of proteins (see Note 3), such as Swiss-Model (12).

2.2. Protein Sequence Alignment

The primary amino acid sequences of the proteins being recombined must be aligned before performing calculations of structural disruption. If PDB coordinates are available for all of the proteins being recombined, the sequence alignment should be generated using the SwissProt or Combinatorial-Extension algorithms, which use structural information to guide the creation of a sequence alignment (13, 14). In all other cases, multiple

sequence alignments should be created using algorithms that only consider sequence information, such as the BLAST (15) and ClustalW2 (16) algorithms.

3. Methods

Schema posits that the best way to conserve a protein's structure upon recombination of homologous proteins is to minimize the number of residue–residue interactions in the parental structures that are altered by recombination (see Fig. 1). The physiochemical characteristics of residues incorporated in chimeras at each position are ignored because they have been preselected to be compatible with the parental structure (17). Interactions are simply defined as any pair of residues whose side chains are within a defined cutoff distance d_c (see Note 4). The major advantages of Schema are its simplicity (11), proven effectiveness (10), and ability to be used with the library design algorithm *R*ecombination *A*s a *S*hortest-*P*ath *P*roblem (RASPP) (9). For any set of parent proteins being recombined, RASPP uses dynamic programming to identify the optimal tradeoff surface for mutation and structural disruption in library space. For a library of a user-defined size (e.g., n crossovers between two parents), RASPP identifies libraries with a range of average amino acid substitution levels $\langle m \rangle$ that have lower than average Schema disruption $\langle E \rangle$ (9).

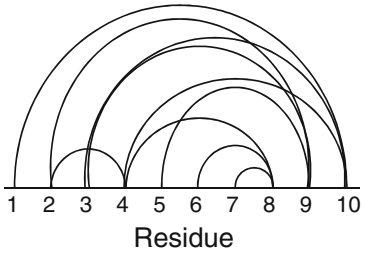
3.1. Library Diversity

Chimeric libraries optimized using Schema and RASPP are created using *S*equence-*I*ndependent *S*ite-*D*irected *C*himeragenesis (SISDC) (8, 18, 19). With SISDC, the number of parents and crossover sites controls library size (see Fig. 2a). Upon creating a n-crossover library using p parents, the number of possible chimeric variants is p^{n+1} (see Note 5). The amino acid substitution level accessible in your chimeras can also be adjusted through your parental choice (see Note 6). The number of amino acid substitutions that can be incorporated into a chimera increases as the sequence identity among the parents used for recombination decreases (17). The accessible substitution level also increases as the number of parents recombined increases.

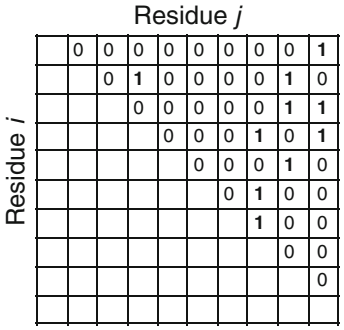
3.2. Choosing Parental Proteins

When recombining homologous proteins, it is thought to be best to recombine the most closely related proteins that will yield your desired level of amino acid substitution (17). Libraries created in this way are predicted to contain a higher fraction of folded (and functional) variants than libraries created using more distantly related parents. In addition, the thermostability of the proteins being recombined should be considered when using SISDC.

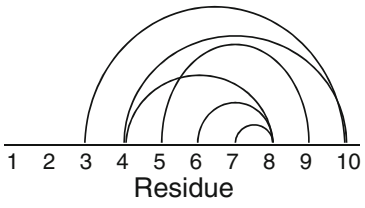
1. Identify all residue-residue interactions (solid lines & *ij* pairs = 1) using structural coordinates from one of the parent proteins recombined.



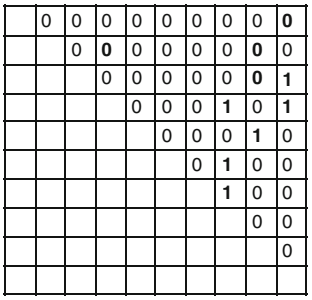
=



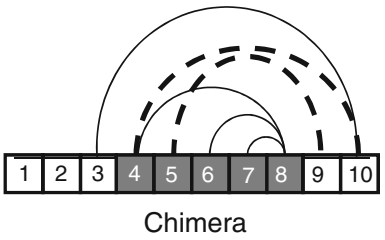
2. Remove interactions that cannot be broken by recombination using the sequence alignment of the parent proteins.



=



3. Count the interactions broken (left = dashed lines; right = black boxes) when a chimera inherits peptides from different parents (left = gray & white boxes).



=

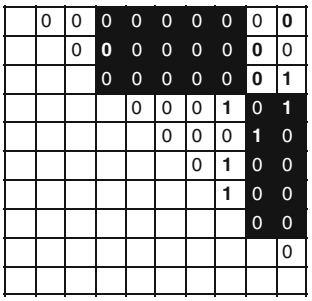


Fig. 1. Protocol for calculating the structural disruption of a chimera. When recombining two structurally related proteins, you first generate a contact matrix that accounts for all pairwise residue–residue interactions in the parent structures. Interactions involving residues that are identical in the parents are removed from the matrix, since they cannot be broken by recombination. The structural disruption *E* is simply the number of residue–residue contacts broken by recombination (11). The chimera shown, which inherits residues 4–8 from parent X and all other residues from parent Y (1–3 and 9–10), has an $E = 2$.

Among protein homologs, those with higher stability have been shown to yield libraries that are enriched in the number of unique, functional, and potentially interesting proteins in both random mutation (20) and recombination (21) experiments. Thus, when you have a choice of multiple proteins for SISDC, you should

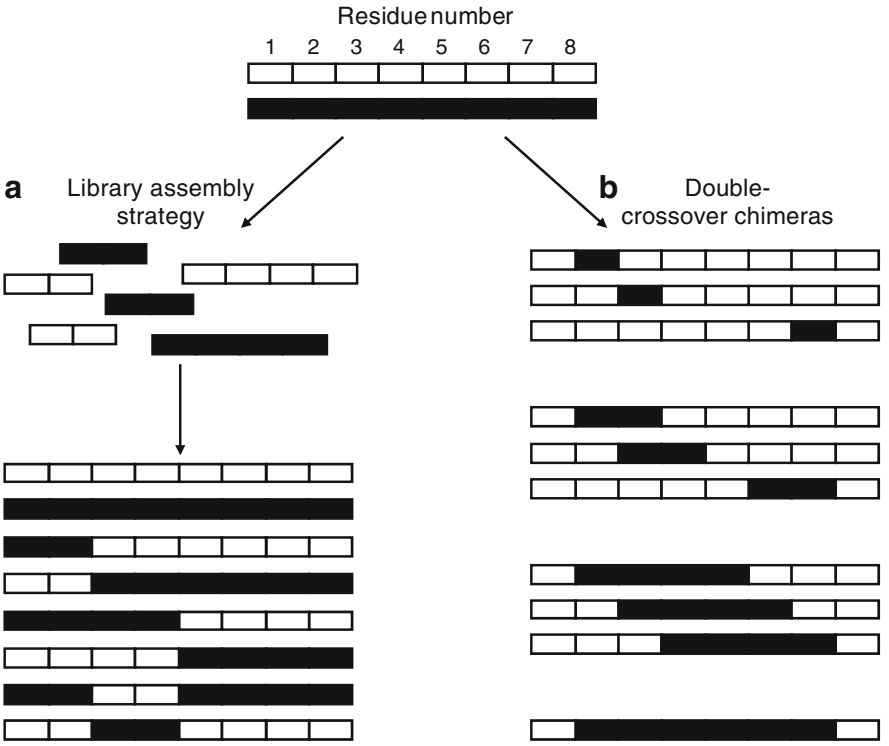


Fig. 2. Sequence-independent site-directed chimeragenesis. (**a**) When constructing a *n*-crossover library, *n* recombination sites are chosen that define the possible polypeptide inheritance in the chimeras. In the example shown, two crossovers between two natural proteins yield 2^3 sequences, two of which are the original parent proteins. (**b**) A simple way to find chimeras for calibration experiments is to calculate *E* for all chimeras in which a single contiguous polypeptide is exchanged among the parent proteins (22). By plotting the *E* vs. *m* for all such chimeras, one can rapidly identify chimeras with a range of amino acid substitution and disruption levels that are easy to build for calibration studies (see Fig. 3).

recombine proteins that exhibit the greatest thermostability available, provided that there are no other functional differences in the enzymes.

3.3. Calculating the Structural Disruption of a Single Chimera

When creating a chimera by recombining homologous proteins, a matrix s_i is created to indicate which parent is incorporated at each position i in the chimeric sequence. For example, if the first residue in a chimera ($i = 1$) is inherited from the first parent, then $s_1 = 1$, but if that residue is inherited from the second parent, then $s_1 = 2$. In addition, the structural coordinates of one parent protein and a sequence alignment of all parents proteins recombined are read into the matrices *pdb* and *align*, respectively (see Note 7). These three matrices are used to calculate the disruption of each chimera, which is defined as

$$E = \sum_{i=1}^{N} \sum_{j=i+1}^{N} C_{ij} \Delta_{ij}$$

where N specifies the number of residues in the structure used for calculations, C_{ij} indicates whether residues i and j are sufficiently close in the parental structure to represent an important interaction that should not be broken, and Δ_{ij} designates whether the interaction between residues i and j in the chimera is present in either of the parents recombined. C_{ij} is given a value of 1 when residues i and j are inherited from different parent proteins ($s_i \neq s_j$) and when these residues are interacting, i.e., within a user-defined cutoff distance d_c in the parental structure. C_{ij} is given a value of 0 in all other cases (see Notes 8 and 9). Because some exchanged polypeptides do not effectively disrupt residue–residue interactions observed in the parents, the delta function Δ_{ij} uses the sequence alignment of the proteins recombined to determine which of the residue–residue interactions in a chimera are distinct from those present in either of the parents. In cases where an interaction between residues i and j in a chimera involves amino acids distinct from those at structurally related positions in all of the parent proteins, Δ_{ij} is given a value of 1 to indicate that this new pairwise interaction is capable of disrupting the chimeras' structure (otherwise $\Delta_{ij} = 0$).

Because of its simplicity, Schema cannot differentiate the structural disruption of chimeras that have opposite polypeptide inheritance (see Note 10). In addition, this algorithm does not account for intersubunit residue–residue contacts that are broken by recombination, although these could be easily considered (see Note 11).

3.4. Calibrating the Disruptive Nature of Substitutions

Functional analysis of chimeric libraries is time consuming and expensive, so you should calibrate the disruption nature of E for the proteins that you are recombining before selecting a library to construct for directed evolution (22). The easiest way to do this is to evaluate the folding (and function) of a small number of chimeras created by swapping single contiguous polypeptides (see Fig. 2b). To do this, you first calculate E for all possible double crossover chimeras and their amino acid substitution level m (see Fig. 3). From this library, you select a handful of chimeras with a broad range of E and m values (e.g., 20), you build the genes encoding these chimeras using splicing by overlap extension in the laboratory (22), and you characterize which of these proteins exhibit parent-like structure (see Note 12). The results from these measurements are then used to identify a threshold level of disruption below which a majority of chimeras retain parent-like structure. This threshold is used to estimate the fraction of variants that are folded in chimeric libraries identified by RASPP and to guide the selection of a chimeric library to construct (8).

3.5. Using RASPP for Library Design

The structural disruption of individual chimeras can be rapidly calculated using Schema as described above (11). However,

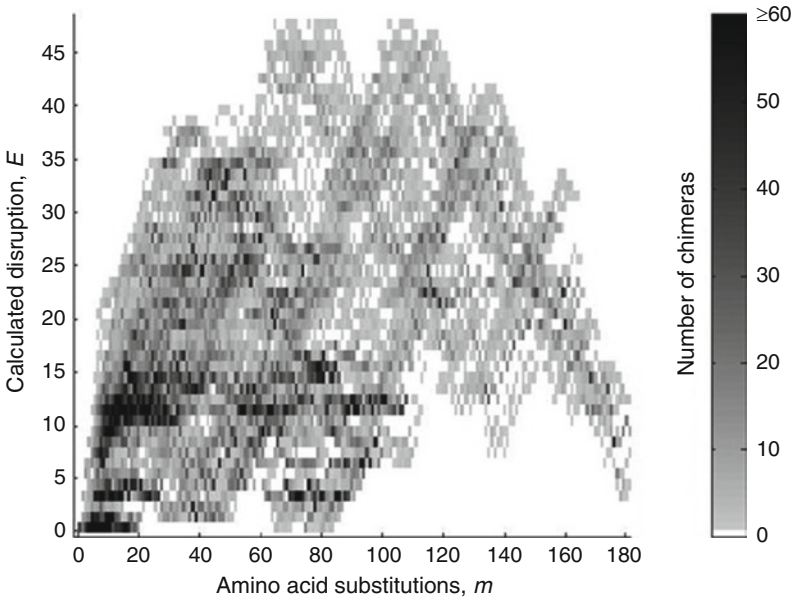


Fig. 3. Structural disruption for chimeras created by swapping a single contiguous polypeptide element. The ATPase domains of *Escherichia coli* HscA and human mitochondrial hsp70 (mtHsp70) were recombined to create all double-crossover chimeras that have a minimum of ten amino acids in each of the contiguous polypeptides recombined. Chimera disruption E is plotted relative to m, the number of chimera residues that differ from HscA. Two residues in the chaperone ATPase domain structure were defined as contacting if any atoms in their side chains were within 5 Å of one another.

this approach is not practical for finding libraries that minimize the average disruption $\langle E \rangle$ of chimeras subject to constraints on the average amino acid substitution level $\langle m \rangle$, i.e., libraries with the best energy-diversity tradeoff (see Note 13). While no optimization protocol has been described for minimizing the $\langle E \rangle$ of a library subject to constraints on the $\langle m \rangle$, one approach that has been applied to this problem identifies libraries that minimize the $\langle E \rangle$ of a library subject to constraints on the length of the polypeptides recombined (8, 23). By posing this surrogate optimization goal, RASPP uses dynamic programming to identify libraries over a range of $\langle m \rangle$ that minimize $\langle E \rangle$ (9).

RASPP uses graph theory to establish the global optimization problem as an all-pairs shortest path problem, where libraries having n crossovers are represented using a directed graph (see Fig. 4 and Note 14), and optimal libraries are found by searching for the shortest paths representing libraries having n crossovers (see Fig. 5). Each path taken is represented by a set of arcs whose individual weights are determined and stored in two matrices. Arcs representing the sets of possible first crossovers $(0,X_1)$ are stored in a matrix designated *arc_singles* and given a weight that represents the $\langle E \rangle$ of the chimeras arising from exchanging the peptide defined by that arc. For the example shown in Fig. 4,

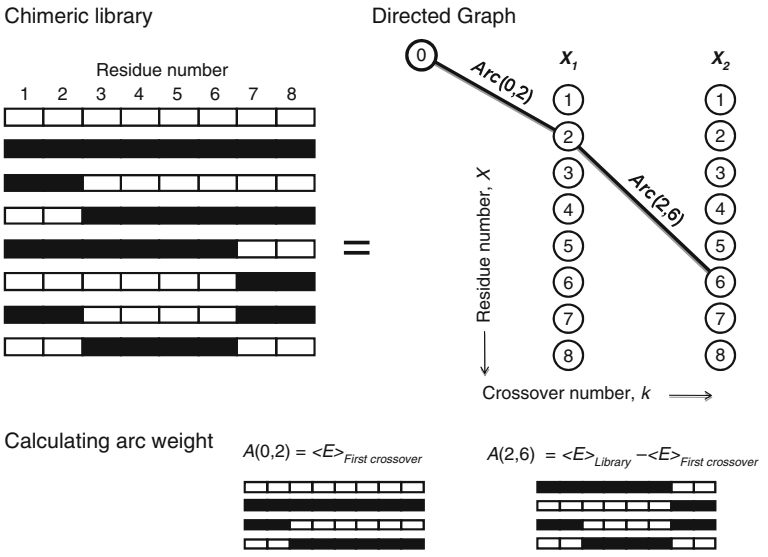


Fig. 4. A directed graph representation of chimeric libraries. To represent a single library (*left*) for RASPP calculations, a directed graph (*right*) is created where each column *k* represents a recombination site, each node X_k within the columns represents the residue after which recombination occurs, and the arcs *A* connecting these columns represent the average structural disruption $\langle E \rangle$ that arises from adding that crossover (*bottom*). The arc weight for the first crossover *A*(0,2) is simply the $\langle E \rangle$ of all chimeras that are created by introducing a single recombination site after the second residue. The subsequent arc *A*(2,6) is given a weight by subtracting $\langle E \rangle$ of the previous arc from the $\langle E \rangle$ for all eight members of two-crossover library.

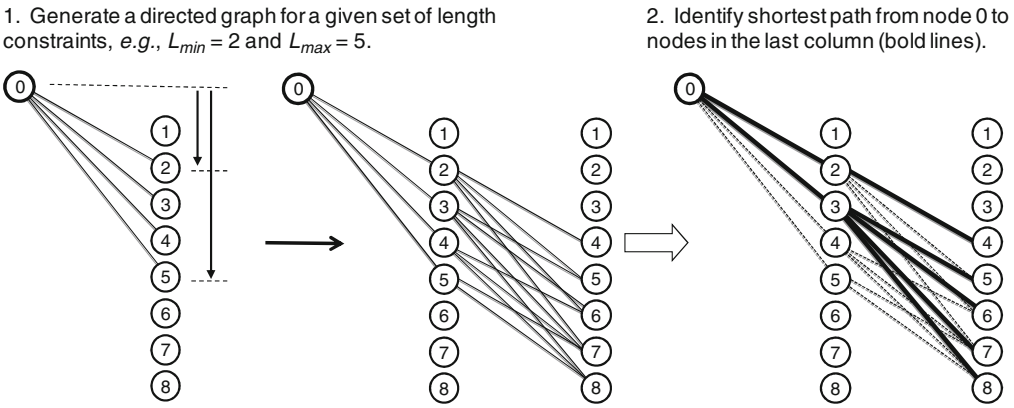


Fig. 5. Length constraints on the exchanged fragments are used to guide the connection of nodes in directed graphs. With RASPP, directed graphs are built for all possible sets of length constraints (L_{min} and L_{max}). In each of these graphs, the shortest paths to each node in the k^{th} column (*bold lines*) are calculated to identify libraries with *n* crossovers that minimize structural disruption (9). The ensemble of libraries identified from the shortest paths in all directed graphs is used to generate the optimal-tradeoff surface shown in Fig. 6.

where the first crossover occurs after residue two, the arc $A(0,2_1)$ is given a length that represents the $\langle E \rangle$ of the four chimeras that arise from allowing that exchange of that peptide. The weight of all subsequent arcs $A(X_k,X_{k+1})$ are stored in a matrix designated

arc_doubles. Their weight is defined as the $\langle E \rangle$ associated with swapping the peptides designated by those arcs. The arc weights in these matrices are calculated once, and all subsequent calculations by RASPP refer back to the *arc_doubles* and *arc_singles* matrices.

In RASPP, the user first dictates the number of crossovers n and the number of residues in the parent proteins recombined N. These values determine the possible range of fragment constraints $L_{\min}$ and $L_{\max}$ that are used to probe all possible n-crossover libraries (see Note 15). Second, directed graphs are generated for all possible combinations of $L_{\min}$ and $L_{\max}$. For every $L_{\min}$ and $L_{\max}$ pair, three arc matrices are generated from *arc_doubles* and *arc_singles* by eliminating those arcs that do not conform to the $L_{\min}$ and $L_{\max}$ parameters. These additional arc matrices define length-constrained arc paths for the first fragment (designated *arc_first*), intermediate fragments (*arc_intermediate*), and the last two fragments (*arc_last*). These constrained arc paths are from:

1. Node 0 to $L_{\min} \le X_1 \le L_{\max}$ (*arc_first*)
2. Node X_k to X_{k+1} if $L_{\min} \le X_{k+1} - X_k \le L_{\max}$ (*arc_intermediate*), and
3. Node X_{n-1} to X_n and X_n to N if $L_{\min} \le X_n - X_{n-1} \le L_{\max}$ and $L_{\min} \le X_N - X_n \le L_{\max}$ (*arc_last*).

Third, these three matrices are then used to find the shortest path through the directed graph by populating a $[(N-1)$ by $n]$ matrix, designated *path*. In *path*, the first column is populated by *arc_first*, the last column by *arc_last*, and the middle columns by *arc_intermediate*. Fourth, the cell array *output* is used to store all optimal chimeric libraries calculated by RASPP. Since there is only one path to each node in column 1 of the directed graph, there is no need to find optimal paths for single crossover libraries, and all crossover loci are placed in *output*. For two crossover libraries, the shortest paths going from node 0 to each node in column 2 are considered optimal. The two crossovers representing each of these optimal libraries are again stored in *output*. More formally, the shortest n path (*see* Fig. 5) is identified by finding the length of the shortest path U from node 0 to node j in column k using the shortest paths from node 0 to all nodes in column $k-1$:

$$U_j^k = \min(U_j^{k-1} + A(i, j))$$

Once the lowest energy libraries are identified by RASPP, the next step is to determine how well these libraries approximate the optimal energy-diversity tradeoff region. To do this, you calculate the average number of mutations in each library identified by RASPP, i.e., those libraries stored in *output* for which the $\langle E \rangle$ has been calculated. A plot of $\langle E \rangle$ as a function of $\langle m \rangle$ is then used to generate a RASPP curve (see Fig. 6), which is defined as those

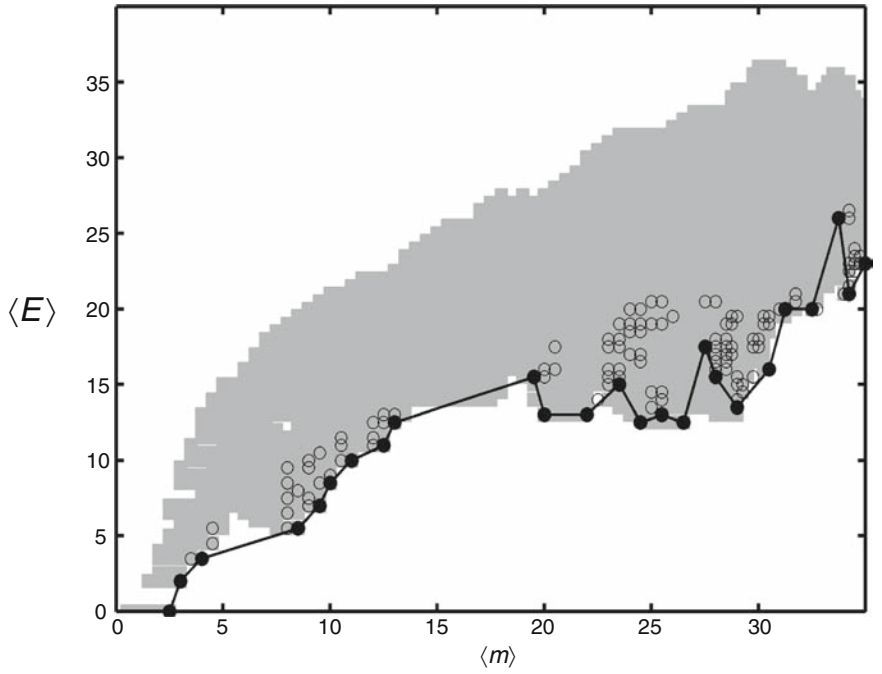


Fig. 6. A RASPP curve approximates the optimal tradeoff surface for libraries. The average disruption ⟨*E*⟩ and amino acid substitution level ⟨*m*⟩ was calculated for all possible three-crossover libraries (~10^6) created by recombining *Bacillus subtilis* and *Thermotoga neapolitana* adenylate kinase (*gray boxes*). Libraries enriched in low disruption chimeras (*black circles*) were identified using RASPP (9), and the libraries with the lowest ⟨*E*⟩ at each ⟨*m*⟩ were used to generate the optimal disruption-diversity tradeoff surface (*black line*). The crystal structure of *Bacillus subtilis* adenylate kinase (PDB = 1P3J) was used for all calculations (30). Pairs of residues were defined as contacting if any atoms in their side chains were within 4.5 Å.

libraries at each level of ⟨*m*⟩ with the lowest ⟨*E*⟩. Libraries populating this curve are the best to use for laboratory evolution experiments, since they are enriched in chimeras with low structural disruption (8, 10, 23) One limitation of RASPP is that it only identifies optimal libraries for a subset of possible ⟨*m*⟩ values.

4. Notes

1. If atomic resolution models exist for each of the proteins recombined, you should use each set of structural coordinates to perform calculations of disruption. This allows one to assess how subtle structural differences affect chimera disruption (22).
2. With many proteins, atomic coordinates are only available for a fraction of the residues that are used for recombination, so only a subset of the residues can be considered in the calculation of structural disruption. The quality of Schema predictions is

expected to decrease as the number of residues lacking atomic coordinates increases, since this algorithm only considers those residues for which structural information is available.

3. Protein coordinates obtained from homology models are only expected to be useful for Schema calculations when they are generated using proteins whose sequences exhibit ≥25% amino acid identity (24).
4. Most studies use a distance cutoff of 4.5 Å and exclude hydrogen, backbone nitrogen, and backbone oxygen atoms when calculating E (10, 11, 19). However, it remains unclear whether the Schema disruption values obtained using this distance cutoff lead to the most accurate predictions of structural disruption.
5. Upon construction of chimeric gene libraries in the lab using SISDC, biases can occur in the frequency with which some chimeric sequences are observed (10). This bias should be quantified prior to screening (or selection) experiments and used to calculate the number of chimeras that must be sampled to thoroughly search your library (25).
6. When recombining two parents, the maximal amino acid substitution level possible is half the number of sequence differences between the parents.
7. The matrix *pdb* contains the x, y, and z coordinates for each atom in one of the parental proteins (see Note 4), as well as the residue number where that atom is found. The matrix *align* indicates whether the residues listed in *pdb* are identical in the parent proteins recombined. Frequently homologous proteins differ in the numbering of their residues owing to differences in the length of their N terminus. For this reason, it is essential that you make sure that the numbering of structurally related residues in *pdb* and *align* is identical prior to performing disruption calculations.
8. The atoms in cofactors are not considered when calculating E, since they typically interact with highly conserved amino acids within the proteins being recombined (8, 22).
9. Protein sequence alignments frequently require the insertion of gaps within the primary amino acid sequence of one or more of the proteins being aligned. These gaps are only considered when calculating E if they are present at a position for which structural information is available. Thus, gaps introduced into the parent used to generate C_{ij} are simply ignored (26), and gaps that occur in any parent other than that used for calculating C_{ij} are treated as nonidentical residues (26).
10. A chimera created using the first five residues of a protein X and the last five residues of a protein Y (XXXXXYYYYY) will

have the same E as YYYYYXXXXX when calculated using Schema. Algorithms are available that assign different calculated disruption values to chimeras having opposite polypeptide inheritance (27, 28). However, these approaches cannot be used with the optimization algorithm RASPP.

11. The quality of Schema predictions has not yet been validated with proteins that require oligomerization for stability. When recombining such proteins, the number of broken interactions within a monomer is not expected to be sufficient to account for all broken residue–residue contacts. For this reason, it is recommended that you account for intersubunit residue–residue interactions $E_{\text{interface}}$ and consider the total calculated disruption for each chimera $E_{\text{total}} = E_{\text{intrasubunit}} + E_{\text{interface}}$. The equation used to calculate interfacial disruption $E_{\text{interface}}$ for an oligomeric chimera is the same as for calculating disruption in a monomer, except that C_{ij} designates whether residue i from chain A is contacting residue j from chain B.
12. Simple assays are available that can rapidly evaluate the conservation of parent-like structure in chimeras, including screens for cofactor binding (8, 22), parent-like activity (10, 23), and parent-like solubility (29).
13. Full enumeration of all chimeras in all possible libraries arising from p^{n+1} crossover combinations is only tractable on a desktop computer when a handful of crossovers are allowed ($n \leq 3$). However, it is intractable when one approaches polypeptide lengths representative of proteins and larger number of crossovers.
14. When an n crossover library is represented using n columns in a directed graph, each column k is given a number of nodes N equal to the number of residues in the proteins recombined. Arcs are used to designate each recombination site, and nodes X visited by an arc $A(X_k, X_{k+1})$ designate crossover sites and define swapped polypeptide.
15. For n crossovers, the minimum fragment length L_{min} has a range from 1 to $N/(n+1)$ and the maximum fragment length L_{max} ranges from $N/(n+1)$ to $N-n(L_{\text{min}})$. It is important to note that RASPP does not consider residues conserved between the two parent proteins as possible crossover sites. Thus, any fragment length L is defined only by nonconserved residues, reducing the effective protein length to ($N-$ *conserved residues*).

Acknowledgments

This work was supported by Robert A. Welch Foundation C-1614 (to J.J.S.), Hamill Innovation Award (to J.J.S.), and National Institutes of Health training grant 2T32-GM008362 (to P.Q.N.).

References

1. Kuhlman, B., Dantas, G., Ireton, G.C., Varani, G., Stoddard, B.L., and Baker, D. (2003) Design of a novel globular protein fold with atomic-level accuracy *Science* **302**, 1364–1368.
2. Park, H.S., Nam, S.H., Lee, J.K., Yoon, C.N., Mannervik, B., Benkovic, S.J., and Kim, H.S. (2006) Design and evolution of new catalytic activity with an existing protein scaffold *Science* **311**, 535–538.
3. Arnold, F.H. (2001) Advances in Protein Chemistry, Vol. 55, 55th Edition, San Diego: Academic Press.
4. Axe, D.D. (2004) Estimating the prevalence of protein sequences adopting functional enzyme folds *J Mol Biol* **341**, 1295–1315.
5. Bloom, J.D., Silberg, J.J., Wilke, C.O., Drummond, D.A., Adami, C., and Arnold, F.H. (2005) Thermodynamic prediction of protein neutrality *Proc Natl Acad Sci U S A* **102**, 606–611.
6. Amin, N., Liu, A.D., Ramer, S., Aehle, W., Meijer, D., Metin, M., Wong, S., Gualfetti, P., and Schellenberger, V. (2004) Construction of stabilized proteins by combinatorial consensus mutagenesis *Protein Eng Des Sel* **17**, 787–793.
7. Wang, L., Brock, A., Herberich, B., and Schultz, P.G. (2001) Expanding the genetic code of Escherichia coli *Science* **292**, 498–500.
8. Otey, C.R., Landwehr, M., Endelman, J.B., Hiraga, K., Bloom, J.D., and Arnold, F.H. (2006) Structure-guided recombination creates an artificial family of cytochromes P450 *PLoS Biol* **4**, e112.
9. Endelman, J.B., Silberg, J.J., Wang, Z.G., and Arnold, F.H. (2004) Site-directed protein recombination as a shortest-path problem *Protein Eng Des Sel* **17**, 589–594.
10. Meyer, M.M., Silberg, J.J., Voigt, C.A., Endelman, J.B., Mayo, S.L., Wang, Z.G., and Arnold, F.H. (2003) Library analysis of SCHEMA-guided protein recombination *Protein Sci* **12**, 1686–1693.
11. Voigt, C.A., Martinez, C., Wang, Z.G., Mayo, S.L., and Arnold, F.H. (2002) Protein building blocks preserved by recombination *Nat Struct Biol* **9**, 553–558.
12. Schwede, T., Kopp, J., Guex, N., and Peitsch, M.C. (2003) SWISS-MODEL: an automated protein homology-modeling server *Nucleic Acids Res* **31**, 3381–3385.
13. Guda, C., Lu, S., Scheeff, E.D., Bourne, P.E., and Shindyalov, I.N. (2004) CE-MC: a multiple protein structure alignment server *Nucleic Acids Res* **32**, W100–W103.
14. Guex, N., and Peitsch, M.C. (1997) SWISS-MODEL and the Swiss-PdbViewer: an environment for comparative protein modeling *Electrophoresis* **18**, 2714–2723.
15. Tatusova, T.A., and Madden, T.L. (1999) BLAST 2 Sequences, a new tool for comparing protein and nucleotide sequences *FEMS Microbiol Lett* **174**, 247–250.
16. Larkin, M.A., Blackshields, G., Brown, N.P., Chenna, R., McGettigan, P.A., McWilliam, H., Valentin, F., Wallace, I.M., Wilm, A., Lopez, R., Thompson, J.D., Gibson, T.J., and Higgins, D.G. (2007) Clustal W and Clustal X version 2.0 *Bioinformatics* **23**, 2947–2948.
17. Drummond, D.A., Silberg, J.J., Meyer, M.M., Wilke, C.O., and Arnold, F.H. (2005) On the conservative nature of intragenic recombination *Proc Natl Acad Sci U S A* **102**, 5380–5385.
18. Hiraga, K., and Arnold, F.H. (2003) General method for sequence-independent site-directed chimeragenesis *J Mol Biol* **330**, 287–296.
19. Meyer, M.M., Hiraga, K., and Arnold, F.H. (2006) Combinatorial recombination of gene fragments to construct a library of chimeras *Curr Protoc Protein Sci* **Chapter 26**, Unit 26.2.
20. Bloom, J.D., Labthavikul, S.T., Otey, C.R., and Arnold, F.H. (2006) Protein stability promotes evolvability *Proc Natl Acad Sci U S A* **103**, 5869–5874.
21. Li, Y., Drummond, D.A., Sawayama, A.M., Snow, C.D., Bloom, J.D., and Arnold, F.H. (2007) A diverse family of thermostable cytochrome P450s created by recombination of stabilizing fragments *Nat Biotechnol* **25**, 1051–1056.
22. Otey, C.R., Silberg, J.J., Voigt, C.A., Endelman, J.B., Bandara, G., and Arnold, F.H. (2004) Functional evolution and structural conservation in chimeric cytochromes p450: calibrating a structure-guided approach *Chem Biol* **11**, 309–318.
23. Meyer, M.M., Hochrein, L., and Arnold, F.H. (2006) Structure-guided SCHEMA recombination of distantly related beta-lactamases *Protein Eng Des Sel* **19**, 563–570.
24. Chung, S.Y., and Subbiah, S. (1996) A structural explanation for the twilight zone of protein sequence homology *Structure* **4**, 1123–1127.
25. Bosley, A.D., and Ostermeier, M. (2005) Mathematical expressions useful in the construction, description and evaluation of protein libraries *Biomol Eng* **22**, 57–61.
26. Silberg, J.J., Endelman, J.B., and Arnold, F.H. (2004) SCHEMA-guided protein recombination *Methods Enzymol* **388**, 35–42.

27. Saraf, M.C., Horswill, A.R., Benkovic, S.J., and Maranas, C.D. (2004) FamClash: a method for ranking the activity of engineered enzymes *Proc Natl Acad Sci USA* **101**, 4142–4147.
28. Saraf, M.C., and Maranas, C.D. (2003) Using a residue clash map to functionally characterize protein recombination hybrids *Protein Eng* **16**, 1025–1034.
29. Sieber, V., Martinez, C.A., and Arnold, F.H. (2001) Libraries of hybrid proteins from distantly related sequences *Nat Biotechnol* **19**, 456–460.
30. Bae, E., and Phillips, G.N., Jr. (2004) Structures and analysis of highly homologous psychrophilic, mesophilic, and thermophilic adenylate kinases *J Biol Chem* **279**, 28202–28208.

Chapter 11

Mass Spectrometric Protein Identification Using the Global Proteome Machine

David Fenyö, Jan Eriksson, and Ronald Beavis

Abstract

Protein identification by mass spectrometry is widely used in biological research. Here, we describe how the global proteome machine (GPM) can be used for protein identification and for validation of the results. We cover identification by searching protein sequence collections and spectral libraries as well as validation of the results using expectation values, rho-diagrams, and spectrum databases.

Key words: Proteomics, Mass spectrometry, Protein identification, Spectrum libraries, Validation

1. Introduction

Mass spectrometry-based protein identification has become an invaluable tool for elucidating protein function, and several methods have been developed for protein identification, including sequence collection searching with masses of peptides or their fragments, spectral library searching, and de novo sequencing (Fig. 1).

The first step in protein identification is to find peaks in the mass spectra that correspond to peptides and their fragments. It is important to find all the relevant peaks and at the same time minimizing the number of background peaks. This can be achieved by scanning the spectra for peaks of the expected width and selecting peaks above a signal to noise threshold (see Note 1), and then picking the monoisotopic peak for each isotope cluster (see Note 2). After picking the peaks, spectra with low information content that could not produce any meaningful results can be removed to increase the speed of subsequent analysis (1).

David Fenyö (ed.), *Computational Biology*, Methods in Molecular Biology, vol. 673,
DOI 10.1007/978-1-60761-842-3_11, © Springer Science+Business Media, LLC 2010

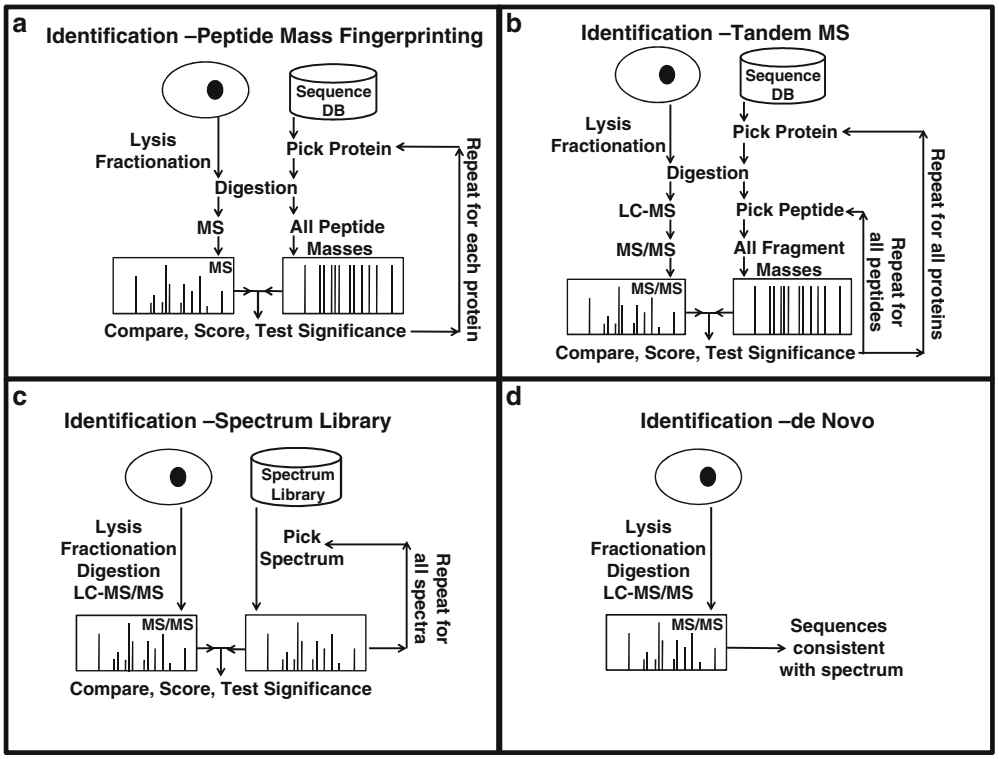


Fig. 1. *Mass spectrometry based workflows for protein identification*: (**a**) searching a protein sequence collection with peptide mass information; (**b**) searching a protein sequence collection with peptide fragment mass information; (**c**) searching a spectrum library with peptide fragment mass information; (**d**) de novo sequencing.

The first method for protein identification developed was peptide mass fingerprinting, PMF (2), i.e., matching measured proteolytic peptide masses to the theoretical proteolytic peptide masses of proteins in a sequence collection and calculating a score based on the matching peptides (see Note 3 and Fig. 1a). A basis of peptide mass fingerprinting is that the mass measurement of a single proteolytic peptide matches the masses of only a few different proteolytic peptide sequences (3). For example, a mass around 2,000 Da measured with an accuracy of 1 ppm matches on the average 4 and 1.5 unmodified tryptic peptides in the entire proteome of human and yeast, respectively (Fig. 2). A single peptide mass measurement is typically not matched uniquely with a single protein species and is therefore not sufficient to identify a protein (the probability for more than one protein identified = 1). But, a set of measured peptide masses from a single digested protein is useful for identification, since the probability is <<1 of randomly matching these mass values to a protein sequence in the collection searched. In theory, not only single proteins but also a large portion of the proteins in a complex protein mixture can be identified by the PMF approach (4).

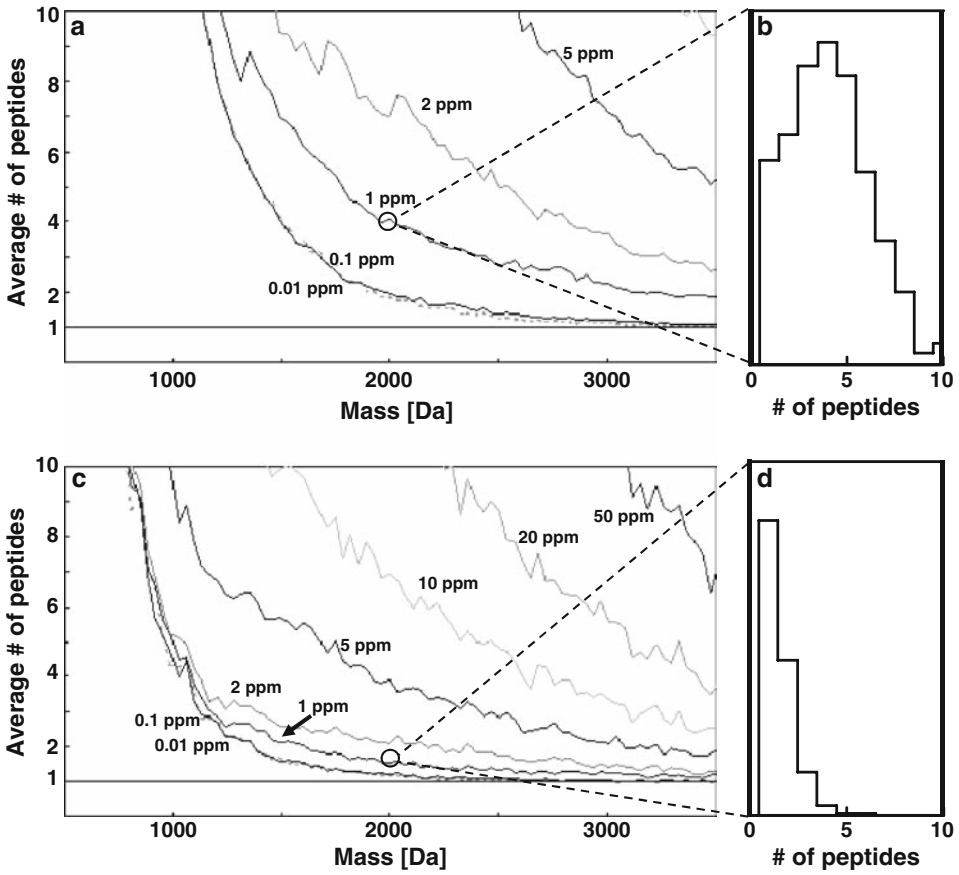


Fig. 2. *The information value of a mass measurement.* The number of unmodified tryptic peptides as a function of peptide mass for different mass accuracies for (**a**) human and (**c**) yeast. The distribution of number of matching unmodified tryptic peptides at mass 2,000 Da and mass accuracy of 1 ppm for (**b**) human and (**d**) yeast.

However, in practice, mass spectrometers fail to detect simultaneously peptides originating from different sample proteins that differ significantly in abundance (5). Hence, a prerequisite for PMF-based protein identification is that the samples analyzed are reasonably pure and only contain a few different proteins (6).

A more robust method for complex protein mixtures is to search sequence collections using the observed mass of an intact individual peptide ion species together with the masses of the fragment ions observed upon inducing fragmentation of the peptide in the mass spectrometer (Fig. 1b). This method requires only one or a few identified peptides to identify a gene. Peptides are fragmented by increasing their internal energy, usually through collisions. When their internal energy is increased, peptides fragment along their backbone, and ions characteristic of the amino acid sequence and the activation method are produced. The masses of these ions are compared with the theoretical fragment masses of the peptides in the sequence collection that match the mass of the

intact peptide, and a score is calculated based on the matching fragments (7, 8). This method is based on the method developed for identifying organic molecules from their fragment mass spectra (9–11). The advantage of using a sequence collection is that it is not necessary to observe fragmentation next to every amino acid in the peptide; a few fragment ions are usually sufficient because the sequence collection can be used to fill in the missing information (see Note 4). The drawback is, however, that if the sequence is not in the sequence collection, it cannot be found using this method, but as more and more complete genome sequences are becoming available, this becomes less of an issue. The probability of fragmentation between a pair of adjacent amino acids is dependent on their chemical properties and to a lesser degree on the amino acids further away from the fragmentation site; therefore, the intensity of fragment ions is highly sequence dependent. The information in the peak intensities cannot fully be utilized when searching protein sequence collections because most implementations use the same intensity for all theoretical fragments owing to the difficulty in accurately predicting their relative intensities from the amino-acid sequence.

One way of utilizing the sequence-specific fragment ion intensities and thereby improving the sensitivity is to instead search spectrum libraries (Fig. 1c), i.e., large collections of experimentally acquired fragment mass spectra that have been annotated. This is currently the predominant method for identification of small organic molecules (12) and has during the last few years been applied to peptide identification (13, 14). In this method, the intensity information is fully utilized (see Note 5) because the matching is between two experimentally acquired fragment mass spectra, and therefore, this is the most sensitive of the identification methods. The challenge is, however, to collect large high-quality sets of spectra that have sufficient coverage of the proteome.

In cases, when the genome has not been sequenced and there are no spectrum libraries available, the only possibility is to use de novo sequencing (Fig. 1d), i.e., use only the information in the fragment mass spectra and the mass of the intact peptide to obtain the peptide sequences (15–18). This requires much higher quality data because the entire space of all possible sequences is the search space (see Note 6). To search the entire space of potential sequences is impractical even for short peptides, but several algorithms have been developed that attempt at searching the relevant part of the search space in a reasonable time frame (15–18).

In all mass spectrometry-based identification methods, a score is calculated to quantify the match between the observed mass spectrum and the collection of possible sequences. These scores are highly dependent on the details of the algorithm used, and they are not always easy to interpret because the interpretation of

the score depends on properties of the data and the search results. Therefore, it is desirable to convert the score to a measure that is easy to interpret, such as the probability that the result is random and false. For this conversion, the distribution of random and false scores is needed (Fig. 3). Estimates of this distribution can be generated using either simulations (19, 20), collecting statistics during the search (21–23), or direct calculations (24).

Here, we describe how the different components of the global proteome machine (GPM) can be used for protein and peptide identification and validation.

2. Methods

2.1. Searching Protein Sequence Collections

X! Tandem (25–27) is a search engine for identifying proteins by searching sequence collections. X! Tandem scores the match between an observed tandem mass spectrum and a peptide sequence, by calculating a score that is based on the intensities of the fragment ions and the number of matching b- and y-ions (see Note 7). This score is converted to an expectation value using the distribution of the scores of randomly matching peptides (Fig. 3). Before the search, the user needs to specify a set of parameters including which sequence collection to search, the mass accuracy of peptides and their fragments, and modifications of the peptide sequence (see Note 8). The search is done iteratively; only proteins that have at least one peptide identified in an iteration are

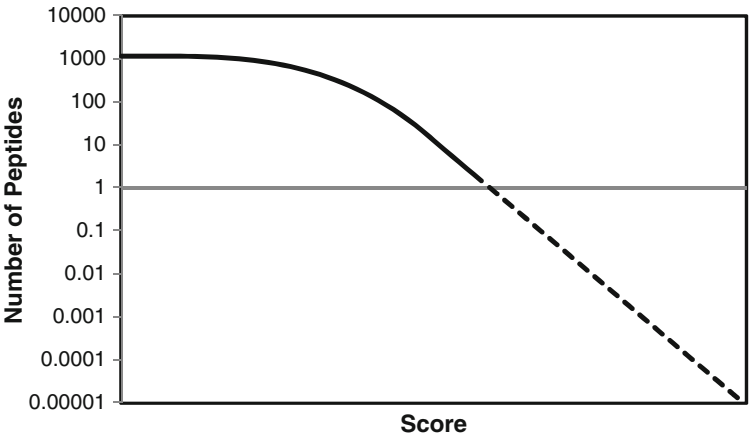


Fig. 3. *Expectation values.* The score can be transformed to an expectation value, i.e., the number of peptides that through random matching generate the score, if the distribution of random scores is known. This random distribution can be obtained for expectation values >1 by collecting statistics during the search because most peptides in a sequence collection match a given mass spectrum purely through random matching. Estimating expectation values <1 can be done by fitting the tail of the distribution to a Gumbel distribution and extrapolating.

searched in subsequent iterations (25). This iterative search can be used to speed up and increase the sensitivity of the identification of modifications, nonspecific enzymatic cleavage, and point mutations by restricting the search to unmodified tryptic peptides in the first iteration, and then widening the search in subsequent iterations. Another way to speed up the searches and make them more sensitive is to restrict the search to proteotypic peptides using X! P3 (27), which searches only peptides that have been previously identified and deposited in the GPM DataBase (GPMDB) (28).

2.2. Searching Spectrum Libraries

X! Hunter (13) is a search engine for searching annotated spectrum libraries. X! Hunter uses the same scoring as X! Tandem, except for that it compares the observed mass spectrum to libraries of spectra derived from experiments. Therefore, the peptide sequence-dependent intensity information can be fully utilized, and the sensitivity of the search is increased. It is, however, critical that the spectrum libraries are constructed carefully. The libraries for X! Hunter are constructed by taking the fragment mass spectra from GPMDB and grouping them so that one library spectrum is constructed for each peptide modification and charge state. The selection criteria are that (1) the spectrum matches to a peptide with an expectation value less than 0.001 and (2) at least 40% of the ion intensity in a spectrum is assignable as y- or b-ions or their corresponding neutral loss products. For the selected spectra, the m/z values of the matching peaks are substituted with the exact theoretical values. The ten spectra with lowest expectation value are selected for each peptide modification and charge state, and a composite spectrum is created and added to the library. These annotated spectrum libraries can also be extended to modification that do not affect the fragmentation pattern (e.g., some types of stable isotope labeling), by using the ion intensities of the fragmented unmodified peptide and reassigning the m/z values to correspond to the modified peptide.

2.3. Validation of Results

The search results for all GPM search engines are displayed in a unified interface that allows the user to get an overview of the results as well as inspect the details of the results when needed. In the basic display, proteins for which there is evidence for their presence in the sample are listed. The strength of the evidence is quantified with an expectation value (see Note 9) (23), and the proteins are listed in the order of increasing expectation value, i.e., in the order of decreasing strength of the evidence. Other information that can be used to assess the identified proteins are also shown, including the sum of the intensity of the matching fragment ions for all peptides, the number of matching peptides, and the fraction of the protein sequence covered by the observed peptides. Details of the evidence for a protein can be displayed,

listing all matching peptides sequences, modifications and charge state together with the peptide expectation values, error in the mass measurement, and the sum of the intensity of the fragment ions matching to the peptide sequences. For an individual peptide, the annotated fragment mass spectrum can be displayed showing the peak assignments. There are also alternative ways to display the list of identified protein, including their distribution among gene ontology categories, pathways, and protein interaction networks. In these displays, a *p*-value is calculated to asses which gene ontology categories, pathways, or interactions are enriched or depleted in the dataset.

Comparison of identification results to the large set of search results collected in GPMDB is an effective way to validate the results. One way to use GPMDB is to visually compare the peptides observed for a protein with observations in other experiments in GPMDB (Fig. 4). Commonly, the same peptides are observed for a given protein in most proteomics experiments, and therefore, an observation of a peptide that has not been observed in other experiments should be investigated manually.

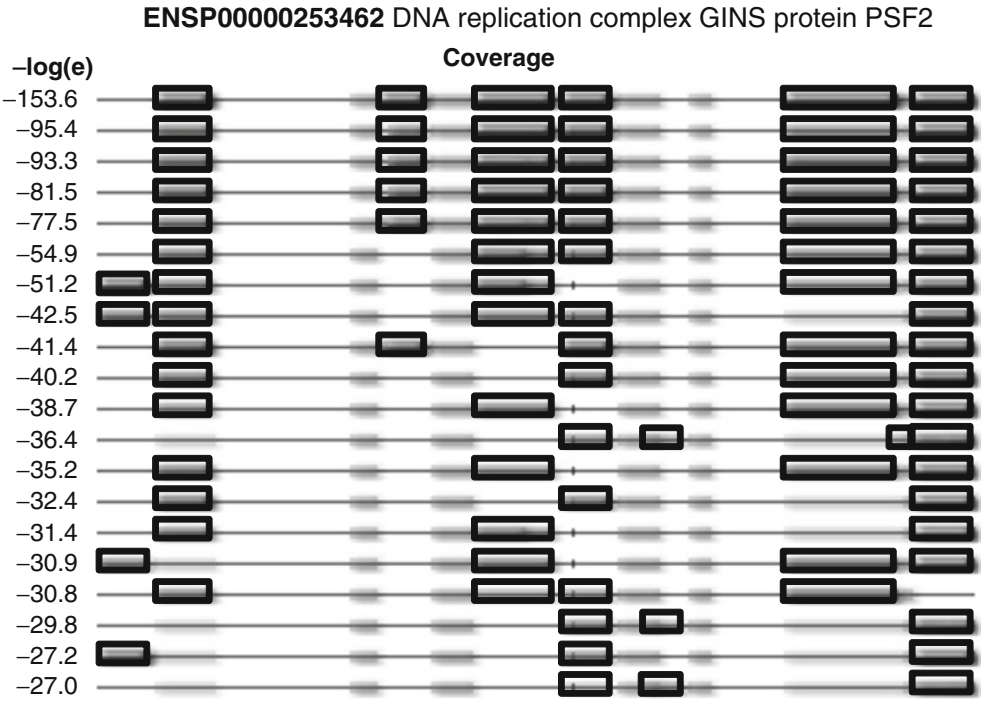


Fig. 4. *Using proteotypic peptides for validation of identification results.* The peptides identified for a protein can be compared with observations in other experiments in GPMDB. Commonly, the same peptides are observed for a given protein in proteomics experiments, and therefore, an observation of a peptide that has not been observed in other experiments should be investigated manually. The peptides observed for PSF2, a protein associated with the replication fork, are shown with black borders and regions of the protein that are difficult to observe in proteomics experiments are shown without borders. In a majority of the 20 experiments shown, the same 5 peptides are observed.

Another way of validating search results is to compare the sequence dependent ion intensity distribution of tandem mass spectra with spectra in GPMDB to evaluate if the fragmentation pattern is similar (Fig. 5). Several frequency measures from GPMDB for proteins and peptides are also reported together with the search results. For peptides, the number of times it has been observed in GPMDB and the fraction of the peptide identifications that are in a specific charge state (ω) are used. For proteins, Ω, a measure of peptide coverage with respect to charge state is used. Ω is a list of ratios denoting what fraction of the peptides in a particular charge state for a given protein was seen in a single protein identification. Proteins expectation values are also compared with other identifications of the protein in GPMDB, and the rank is reported, allowing the user to judge how their result compares with other results. All these measures are shown to make the validation of the results easier by allowing detailed comparison with the large set of experimental results that are available in GPMDB.

The information in GPMDB can also be used to design experiments. It is advisable to start planning an experiment by inspecting the information associated with proteins of interest to find out what has been observed in other proteomics experiments. For example, GPMDB supports the design of experiments targeted to investigate a group of proteins (multiple reaction monitoring (MRM)). Through the MRM module, the information in GPMDB is used to aid in the selection of peptides and their fragment ions that produce a strong signal and are specific to the protein.

The quality of the overall match between the whole dataset and the sequence collection can be evaluated using ρ-diagrams and ρ-scores (29). A ρ-diagram is a comparison between the distribution of peptide expectation values for a dataset and the predicted distribution for random matching (see Note 10). For a dataset that only has random matches to a sequence collection, the data points in the ρ-diagram will fall on the diagonal, $\rho = \log(e)$, i.e., the expectation values for the peptides are distributed as expected from random matching (Fig. 6a). In contrast, for datasets that are of high quality, typically many peptides match well with the sequence collection, and the data points in the ρ-diagram deviate from the diagonal and are closer to $\log(e) = 0$ (Fig. 6b). The ρ-score corresponding to a ρ-diagram is defined as the area between the data points and the diagonal [$\rho = \log(e)$] normalized to a value between 0 and 100, where ρ-score of 0 corresponds to purely random matching and ρ-score of 100 corresponds to no random matching. The ρ-score, being a measure of the quality of a match between an entire dataset and a sequence collection, can be used for optimizing search parameters, for evaluating algorithms, and for controlling the quality of datasets.

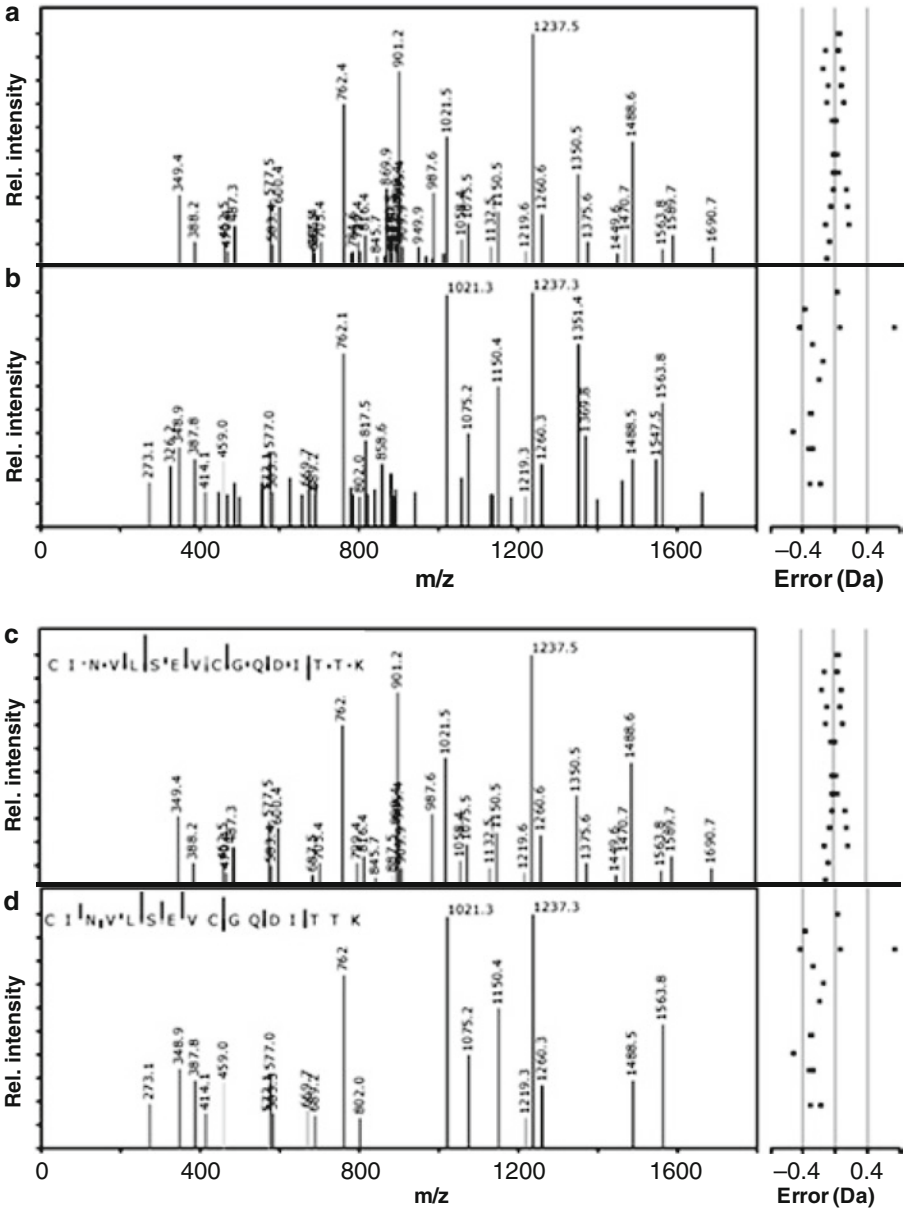


Fig. 5. *Using tandem mass spectra for validation of identification results.* The intensity distribution of tandem mass spectra is mainly dependent on the peptide sequence. Therefore, comparing a fragment mass spectrum with spectra in GPMDB can be used for validation of the results. (**a**, **c**) A stronger [log(*e*) = –12.8] and (**b**, **d**) a weaker [log(*e*) = –3.6] spectrum matching to the sequence CINVLSEVCGQDITTK are shown [(**a**, **b**) – all peaks (**c**, **d**) – peaks matching the sequence]. The stronger spectrum has many peaks matching the peptide sequence and little background, while the weaker spectrum has fewer matching peaks and more background peaks, but the intensity profile of the matching peaks is similar.

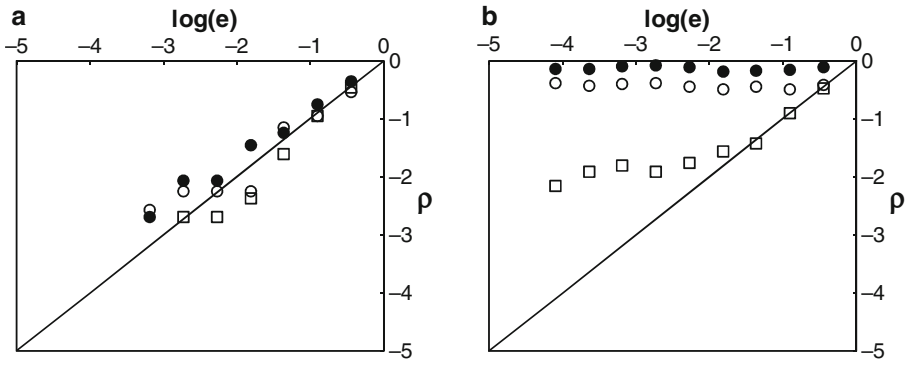


Fig. 6. *ρ-diagram.* A *ρ*-diagram shows the quality of the match between a dataset and a proteome. (**a**) The data points are close to the line $\rho = \log(e)$ when the results are dominated by random matching between the data and the proteome. The three datasets shown were obtained by searching against a collection of reversed sequences. (**b**) Three datasets of different quality are shown (*ρ*-scores are 95, 87, 57, respectively). The highest quality dataset (*filled circles*) is closest to the line $\log(e) = 0$ and the lowest quality dataset (*open squares*) is closest to the line $\rho = \log(e)$.

3. Notes

1. Peaks in mass spectra are detected by finding local maxima in $S(l) = \sum_{|k-l|<w_l/2} I(k)$ over the expected peak width w_l for each point, l, in the spectrum, where $I(k)$ is the measured intensity at a point $k, 0 \le k \le N, 0 \le l \le N$ and N is the total number of points in the mass spectrum. The signal to noise ratio (the ratio of the root mean square deviation of the peak and of the background) is usually used to decide if the peak should be used for identification. The mass of an analyte can be determined using the centroid;

$$C(l) = \frac{\sum_{|k-l|<w'/2} I(k) \cdot \frac{m}{z}(k)}{\sum_{|k-l|<w'/2} I(k)}$$

(where $\frac{m}{z}(k)$ is the mass to charge ratio at a point k) of the corresponding peak in the mass spectrum, where w' is the width of the centroid calculation.

2. Because peptides naturally contain heavy isotopes of atoms (e.g., 1.11% ^{13}C and 0.366% ^{15}N), they are observed as clusters of peaks. The relative intensities of these isotope clusters are dependent on the mass of the peptide because the number of atoms increases with mass, and therefore, the probability of the peptide containing one or more heavy isotopes increases. The largest effect comes from ^{13}C and a first order estimate of the peak intensities is given by, $T_m = \binom{n}{m} p^m (1-p)^{n-m}$ where T_m

is the intensity of peak m in the distribution, m is the number of ^{13}C, n the total number of carbon atoms in the peptide, and p is the probability for ^{13}C (i.e., 1.11%).

3. The simplest method for peptide mass fingerprinting is to count the number of peptides in the mass spectrum that match to each protein in the sequence collection. This count can then be used as a score to rank the proteins. This simple scoring scheme works well when the data are of high-quality, but with low-quality data, typically, a large protein will get the highest score due to random matching. This is because the probability for random matching increases with the size of the protein simply because there are more peptides to match. More sophisticated scoring methods have been developed as an attempt to compensate for this effect (24, 30–32).
4. The sequence collections used for protein identification are based on the genes predicted from the genome sequence, and are therefore a very small subset of all possible sequences. For example, there are ~2.5×10^4 unique tryptic peptides of length 15 in the human proteome compared with $20^{15} = 3.3 \times 10^{19}$ possible unmodified peptides of length 15. Because a vast majority of possible peptides are not used in an organism, the distance between real peptides in sequence space is typically large, and therefore, missing information can be filled in using the sequence collection.
5. Typically, the normalized inner product of the two spectra is used to score how well their intensities match. If the spectra are represented as vectors with the number of dimension equal to the number of matching peaks, n, and the length of the vector in each dimension equal to the intensity of the corresponding ion, the dot product is given by, $\frac{\mathbf{I}\bullet\mathbf{L}}{|\mathbf{I}||\mathbf{L}|} = \sum_{k=1}^{k=n} I_k L_k \Big/ \sqrt{\sum_{k=1}^{n} I_k^{\,2} \sum_{k=1}^{n} L_k^{\,2}}$, where $\mathbf{I} = (I_1, I_2, \ldots, I_n)$ is the observed spectrum, and $\mathbf{L} = (L_1, L_2, \ldots, L_n)$ is the library spectrum. The range of the normalized dot product is from –1 to 1. If the observed and library spectra are identical, the resulting dot product is 1, and any differences between them will result in lower values of the dot product.
6. The search space for de novo sequencing of unmodified peptides is 20^N where N is the length of the peptide. If there are m types of potential modifications, then search space increases to $(20+m)^N$.
7. The score, called hyperscore, is based on the assumption of a hypergeometric distribution and is given by $S_H = S_I \cdot n_b! \cdot n_y!$, where n_y is the number of matching y-ions, n_b the number

of matching b-ions, and S_I is the dot product between the observed spectrum and the spectrum predicted from the peptide sequence. The intensities for the spectrum predicted from the peptide sequence are usually set to 1 for each expected fragment mass and 0 for all other masses. However, X! Tandem also supports using intensities that are dependent on the two amino acids on each side of the fragmented bond.

8. A complete description of the input parameters for X! Tandem, X! P3, and X! Hunter can be found at http://thegpm.org/TANDEM/api/.

9. Protein expectation values can be estimated from the expectation values of its matching peptides. If more than one peptide has been found for a protein, the expectation values for the peptides are combined with a simple Bayesian model for the probability of having two peptides from the same protein having the best score in different spectra: $e_{pro} = \left(\frac{\beta^n(1-\beta^{s-n})}{sN^{n-1}}\right) \times (\Pi_{j=1}^{n} e_j) \times \left(\Pi_{i=0}^{n-1} \frac{s-i}{n-i}\right)$ where n is the number of unique peptide sequences matching the protein, e_j is the expectation value of the jth peptide, N is the total number of peptides scored to find the n unique peptides, s is the number of mass spectra in dataset, and β is $N/$(the total number of peptides in the proteome considered). If only one peptide is matching the protein, then the protein expectation value is set to the peptide expectation value, $e_{pro} = e_1$.

10. ρ is defined as $\rho(i) = \log\left(\frac{E_i}{E_0}\right)$ where i is an integer, $i = \log(e)$, e is the expectation value, and E_i is the number of peptides with expectation values between $\exp(i)$ and $\exp(i-1)$. For purely random matching, $E_i = \int_{\exp(i-1)}^{\exp(i)} N\,de = N\left[\exp(i) - \exp(i-1)\right] = N \bullet \exp(i) \bullet [1 - \exp(-1)]$ where N is the total number of peptides that have been assigned to spectra, and therefore $\rho(i) = \log\left(\frac{E_i}{E_0}\right) = i = \log(e)$ for random matching.

Acknowledgments

This work was supported by funding provided by the National Institutes of Health Grants RR00862 and RR022220, the Carl Trygger foundation, and the Swedish research council.

References

1. K. Flikka, L. Martens, J. Vandekerckhove, K. Gevaert, and I. Eidhammer (2006) Improving the reliability and throughput of mass spectrometry-based proteomics by spectrum quality filtering, *Proteomics*, **6**, 2086–94.
2. W.J. Henzel, T.M. Billeci, J.T. Stults, S.C. Wong, C. Grimley, and C. Watanabe (1993) Identifying proteins from two-dimensional gels by molecular mass searching of peptide fragments in protein sequence databases, *Proc Natl Acad Sci USA*, **90**, 5011–5.
3. D. Fenyo, J. Qin, and B.T. Chait (1998) Protein identification using mass spectrometric information, *Electrophoresis*, **19**, 998–1005.
4. J. Eriksson and D. Fenyo (2005) Protein identification in complex mixtures, *J Proteome Res*, **4**, 387–93.
5. J. Eriksson and D. Fenyo (2007) Improving the success rate of proteome analysis by modeling protein-abundance distributions and experimental designs, *Nat Biotechnol*, **25**, 651–5.
6. O.N. Jensen, A.V. Podtelejnikov, and M. Mann (1997) Identification of the components of simple protein mixtures by high-accuracy peptide mass mapping and database searching, *Anal Chem*, **69**, 4741–50.
7. J.K. Eng, A.L. McCormack, and J.R. Yates (1994) An approach to correlate mass spectral data with amino acid sequences in a protein database, *J Am Soc Mass Spectrom*, **5**, 976.
8. M. Mann and M. Wilm (1994) Error-tolerant identification of peptides in sequence databases by peptide sequence tags, *Anal Chem*, **66**, 4390–9.
9. A.M. Duffield, A.V. Robertson, C. Djerassi, B.G. Buchanan, G.L. Sutherland, E.A. Feigenbaum, and J. Lederberg (1969) Applications of artificial intelligence for chemical inference. II. Interpretation of low-resolution mass spectra of ketones, *J Am Chem Soc*, **91**, 2977–81.
10. J. Lederberg, G.L. Sutherland, B.G. Buchanan, E.A. Feigenbaum, A.V. Robertson, A.M. Duffield, and C. Djerassi (1969) Applications of artificial intelligence for chemical inference. I. The number of possible organic compounds. Acyclic structures containing C, H, O, and N, *J Am Chem Soc*, **91**, 2973–6.
11. G. Schroll (1969) Applications of artificial intelligence for chemical inference. III. Aliphatic ethers diagnosed by their low-resolution mass spectra and nuclear magnetic resonance data, *J Am Chem Soc*, **91**, 2977–81.
12. S. Heller (1999) The history of the NIST/EPA/NIH mass spectral database, *Today's Chemist at Work*, **8**, 45–50.
13. R. Craig, J.C. Cortens, D. Fenyo, and R.C. Beavis (2006) Using annotated peptide mass spectrum libraries for protein identification, *J Proteome Res*, **5**, 1843–9.
14. H. Lam, E.W. Deutsch, J.S. Eddes, J.K. Eng, N. King, S.E. Stein, and R. Aebersold (2007) Development and validation of a spectral library searching method for peptide identification from MS/MS, *Proteomics*, **7**, 655–67.
15. J.A. Taylor and R.S. Johnson (1997) Sequence database searches via de novo peptide sequencing by tandem mass spectrometry, *Rapid Commun Mass Spectrom*, **11**, 1067–75.
16. V. Dancik, T.A. Addona, K.R. Clauser, J.E. Vath, and P.A. Pevzner (1999) De novo peptide sequencing via tandem mass spectrometry, *J Comput Biol*, **6**, 327–42.
17. B. Ma, K. Zhang, C. Hendrie, C. Liang, M. Li, A. Doherty-Kirby, and G. Lajoie (2003) PEAKS: powerful software for peptide de novo sequencing by tandem mass spectrometry, *Rapid Commun Mass Spectrom*, **17**, 2337–42.
18. B. Spengler (2004) De novo sequencing, peptide composition analysis, and composition-based sequencing: a new strategy employing accurate mass determination by fourier transform ion cyclotron resonance mass spectrometry, *J Am Soc Mass Spectrom*, **15**, 703–14.
19. J. Eriksson, B.T. Chait, and D. Fenyo (2000) A statistical basis for testing the significance of mass spectrometric protein identification results, *Anal Chem*, **72**, 999–1005.
20. J.E. Elias and S.P. Gygi (2007) Target-decoy search strategy for increased confidence in large-scale protein identifications by mass spectrometry, *Nat Methods*, **4**, 207–14.
21. H.I. Field, D. Fenyo, and R.C. Beavis (2002) RADARS, a bioinformatics solution that automates proteome mass spectral analysis, optimises protein identification, and archives data in a relational database, *Proteomics*, **2**, 36–47.
22. A. Keller, A.I. Nesvizhskii, E. Kolker, and R. Aebersold (2002) Empirical statistical model to estimate the accuracy of peptide identifications made by MS/MS and database search, *Anal Chem*, **74**, 5383–92.
23. D. Fenyo and R.C. Beavis (2003) A method for assessing the statistical significance of mass spectrometry-based protein identifications using general scoring schemes, *Anal Chem*, **75**, 768–74.

24. J. Eriksson and D. Fenyo (2004) Probity, a protein identification algorithm with accurate assignment of the statistical significance of the results, *J Proteome Res*, **3**, 32–6.
25. R. Craig and R.C. Beavis (2003) A method for reducing the time required to match protein sequences with tandem mass spectra, *Rapid Commun Mass Spectrom*, **17**, 2310–6.
26. R. Craig and R.C. Beavis (2004) TANDEM: matching proteins with tandem mass spectra, *Bioinformatics*, **20**, 1466–7.
27. R. Craig, J.P. Cortens, and R.C. Beavis (2005) The use of proteotypic peptide libraries for protein identification, *Rapid Commun Mass Spectrom*, **19**, 1844–50.
28. R. Craig, J.P. Cortens, and R.C. Beavis (2004) Open source system for analyzing, validating, and storing protein identification data, *J Proteome Res*, **3**, 1234–42.
29. D. Fenyo, B.S. Phinney, and R.C. Beavis (2007) Determining the overall merit of protein identification data sets: rho-diagrams and rho-scores, *J Proteome Res*, **6**, 1997–2004.
30. D.N. Perkins, D.J. Pappin, D.M. Creasy, and J.S. Cottrell (1999) Probability-based protein identification by searching sequence databases using mass spectrometry data, *Electrophoresis*, **20**, 3551–67.
31. W. Zhang and B.T. Chait (2000) ProFound: an expert system for protein identification using mass spectrometric peptide mapping information, *Anal Chem*, **72**, 2482–9.
32. J. Magnin, A. Masselot, C. Menzel, and J. Colinge (2004) OLAV-PMF: a novel scoring scheme for high-throughput peptide mass fingerprinting, *J Proteome Res*, **3**, 55–60.

Chapter 12

Unbiased Detection of Posttranslational Modifications Using Mass Spectrometry

Maria Fälth Savitski and Mikhail M. Savitski

Abstract

A major challenge in proteomics is to fully identify and characterize posttranslational modification (PTM) patterns present at any given time in cells, tissues, and organisms. Currently, the most frequently used method for identifying PTMs is tandem mass spectrometry combined with searching a protein sequence database. Although, database searching has been highly successful for the identification of proteins, it has a number of significant drawbacks for identification of modifications. The user needs to specify all expected modifications, and the search engine needs to consider all possible combinations of these modifications for all peptide sequences. If several potential modifications are considered, the search can take much longer than the data acquisition, creating a bottleneck in high-throughput analysis. In addition, the many possible assignments that need to be tested increase the noise and require better quality data for confident identification of modifications. Here, we describe a method for identifying both known and unknown PTM using mass spectrometry that does not suffer from these problems. The method is based on the observation that, in many samples, peptides are usually present both with and without modifications. By identifying the unmodified peptide with conventional database searches, the modified species of the peptide can be identified by searching for peptides with common and similar fragments as the unmodified peptide. After identifying both the modified and unmodified peptide, the elemental composition of the modification can be deduced if the mass accuracy of the precursor ion is sufficiently high.

Key words: Bioinformatics, Mass spectrometry, Posttranslational modifications, MS/MS, ModifiComb

1. Introduction

1.1. Posttranslational Modifications

Posttranslational modifications (PTMs) are covalent-processing events that often are, as the name indicates, among the last steps in the protein biosynthesis. PTMs can be produced by proteolytic cleavage or by addition of a modifying group to one or more amino acid residues. PTMs change the intrinsic properties of the protein, and can determine its activity state, localization, turnover, and interactions with other proteins. In signaling, for example,

David Fenyö (ed.), *Computational Biology*, Methods in Molecular Biology, vol. 673,
DOI 10.1007/978-1-60761-842-3_12, © Springer Science+Business Media, LLC 2010

kinase cascades are turned on and off by the reversible addition and removal of phosphate groups (1), and in the cell cycle, ubiquitination marks cyclins for destruction at defined time points (2). PTMs are key regulators of protein function, localization, and interactions taking place inside the cell (3, 4).

The reversible phosphorylation of serine, threonine, and tyrosine residues in proteins is considered to be one of the most important PTMs (3, 5–7). It is estimated that in higher organisms more than 25% of all proteins are phosphorylated and more than 2% of the human genes code for protein kinases and the respective phosphatases. Protein phosphorylation is believed to play a major role in many important cellular processes such as the cell cycle, cell differentiation, metabolism, cell motility, and signaling (8). Other PTMs such as glycosylation (9), methylation (10, 11), and ubiquitination (12) play important roles in cellular processes. Other types of known and unknown PTMs are also involved. Many types of PTMs have been discovered serendipitously and many more remain yet to be discovered.

1.2. Mass Spectrometry

Mass spectrometry (MS) is an analytical technique that measures the mass-to-charge (m/z) ratio of ions. A mass spectrometer consists of an ion source where the analytes are ionized, a mass analyzer where the ionized analytes are separated according to their m/z ratio, and a detector that measures the relative abundance of the ions at each m/z value. Ionizing proteins and peptides for MS analysis is most commonly done by electrospray ionization (ESI) (13) or by matrix-assisted laser desorption/ionization (MALDI) (14, 15). ESI and MALDI are called soft ionizations techniques because they can ionize a large molecule (e.g., protein or peptide) without fragmenting it, which makes it possible to determine the mass of the intact peptide or protein.

To be able to decipher the amino acid sequence of a peptide, it has to be broken into pieces. Ions of the peptide of interest are gathered at its specific m/z and the peptide ions are fragmented by, for example, collision with an inert gas such as helium. This is called tandem mass spectrometry (MS/MS) (16). To identify the amino acid sequence, the MS/MS spectra are searched against databases using search engines such as Mascot (17), Sequest (18), and X! Tandem (19).

There are two major techniques to fragment peptides, either through vibrational excitation or through electron capture/transfer. Collision-activated dissociation (CAD) (20, 21) is the most common vibrational dissociation method, whereas electron-capture dissociation (ECD) (22) and electron-transfer dissociation (ETD) (23) are examples of dissociation through electron capture/transfers. The different types of fragmentation methods yield different types of fragment ions. The two most common ion types in CAD fragmentation are b and y ions, which are formed

from backbone cleavage of the peptide. *b* ions are formed when the N-terminal fragment retains the charge and *y* ions are formed when the C-terminal fragment retains the charge. If the ionized peptide contains more than one charge, complementary ion pairs can be produced, for example, fragmentation of a doubly charged peptide can produce a b_n/y_m ion pair, where the sum of *n* and *m* equals the total number of residues in the peptide. Although CAD fragmentation most often yield *b* and *y* ions, ECD/ETD mainly produce *c* and *z*·ions. ECD and ETD provide more extensive peptide fragmentation than CAD and are better for characterizing PTMs, because they remain intact during the backbone fragmentation (*vide infra*) (24).

1.3. Identification of PTMs Using MS

From a mass spectrometric point of view, PTMs can be roughly classified into two distinct classes: labile and stable modifications (Fig. 1). The loss of a labile modification is a preferred fragmentation

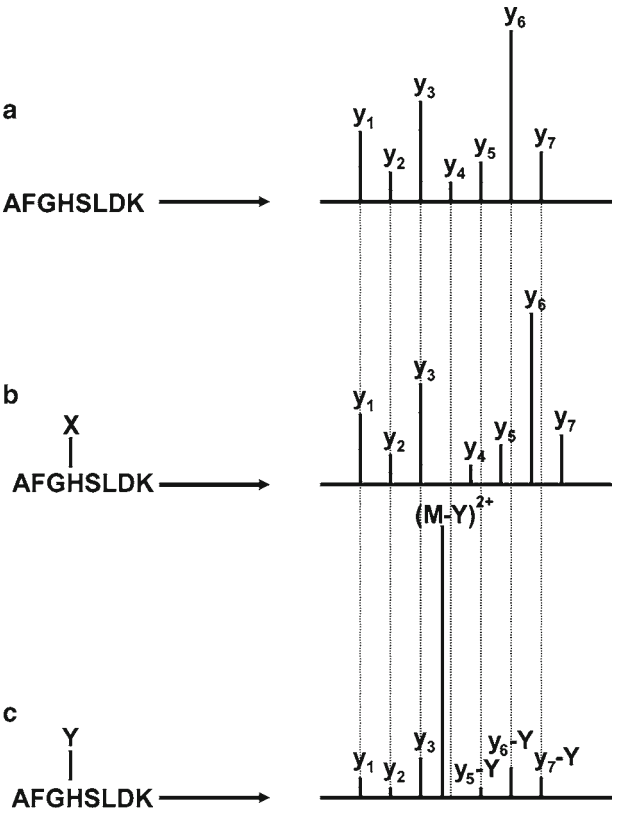


Fig. 1. (**a**) Fragmentation of an unmodified doubly charged peptide (only y^+ ions are shown for simplicity). (**b**) Fragmentation of the same peptide with a stable modification on serine. y_4, y_5, y_6, and y_7 are shifted by the mass of the modification *X* compared with the unmodified peptide. (**c**) Fragmentation of the same peptide with a labile modification on serine. The most dominant peak corresponds to a loss of the neutral modification mass *Y* from the modified peptide. All *y* ions are low abundant and do not contain information on the position of *Y*, because they arise through secondary fragmentation of $(M-Y)^{2+}$.

channel in CAD. A tandem mass spectrum of a peptide with a labile modification frequently, only exhibits one very abundant fragment ion corresponding to the loss of the modification and few to none backbone fragment ions, making it very difficult to derive the identity of the peptide and the location of the modification (Fig. 1c). A stable modification, however, has no such effect. Fragmentation of a peptide with a stable modification gives rise to a spectrum dominated by backbone fragmentation (Fig. 1b). Many modifications fall in the gap between the two classes (4). Phosphorylation is such an example: serine/threonine phosphorylation is a labile type of modification, whereas the less frequent tyrosine phosphorylation is a stable modification.

One of the most striking features of the ECD/ETD approach is that it always produces backbone fragmentation regardless of what type of modification is attached to the peptide, even such labile ones as phosphorylation (25) and glycosylation (26). Basically, ECD/ETD MS/MS spectra of modified peptides always give rise to the type of fragmentation pattern shown in Fig. 1b (with c and z ions instead of y and b). This makes the combined use of ECD/ETD and CAD very effective for the study of phospho-peptides. The strong neutral loss of phosphoric acid in CAD serves as a flag for identifying the spectra of phospho-peptides, and the accompanying ECD spectrum can reveal the identity of the peptide by providing sequence information and possibly also the location of the phosphorylation site (27).

In order to detect stable modifications using database searching, the following strategy is employed. Assume that one is looking for a modification of mass ΔM located on the amino acid residue X (X is any of the 20 common amino acids). Then, the search engine will expand the search space by adding for every peptide containing X an additional modified peptide with ($X+\Delta M$) instead of X (e.g., a peptide containing three X will give rise to seven additional modified peptides, three with one modification, three with two modifications, and one with three modifications). The experimental spectra will be compared in addition to the "usual" theoretical spectra, also to the spectra stemming from these added modified peptides.

The addition of a large number of new peptides to the database leads to a longer overall search time and to a decreased sensitivity. It is very computationally demanding to perform searches assuming the presence of more than ten types of modifications. Also no new types of modifications can be discovered by using this standard approach.

Craig and Beavis (28) have suggested a strategy for reducing the time required to search MS/MS data with variable modifications. The strategy works under the assumption that an unmodified, tryptic (without missed cleavages) peptide will be selected for MS/MS for each identifiable protein in the sample.

In the first stage of identification, only search criteria for unmodified tryptic peptides (if desired one or more potential modifications can be added) will be used. In subsequent stages, searches allowing for modifications, non-specific hydrolysis and missed cleavages will be conducted against proteins that were identified in the first stage by tryptic peptides. This strategy allows for a more sensitive and specific identification of known modifications.

Here, we are going to describe a method for how to identify both known and unknown PTMs utilizing MS. These types of approaches have already uncovered various new and hitherto unreported types of modifications.

2. Materials

The type of data required for the analysis described in this chapter would preferentially come from a high-resolution mass spectrometer such as the Fourier Transform mass spectrometer [29] or the Orbitrap [30]. The data analysis would require access to a regular PC and to one of the established search engines, e.g., X! Tandem or Mascot.

3. Methods

This strategy for identifying PTMs is based on the assumption that a modified peptide (dependent peptide) will be present along with its unmodified counterpart (base peptide) in the sample (most PTMs are substoichiometric) (Fig. 2). The computational approach can identify PTMs without making any assumptions about which PTMs are present in a given sample. It requires that the MS data are in high resolution, and preferably the MS/MS data should be in high resolution as well. The mass resolution is the dimensionless ratio of the mass of the peak divided by its width. Usually, the peak width is taken as the full width at half maximum intensity. The resolution required for a reliable unbiased PTM identification should be above 50,000. If the MS data are of low resolution, it will be very difficult to reliably assign the elemental composition to a potential modification mass. If the MS/MS data are of low resolution, the procedure can still work, but only for high-quality spectra and will require more matching fragments.

Step 1. Perform a regular database search to identify base peptides in the dataset.

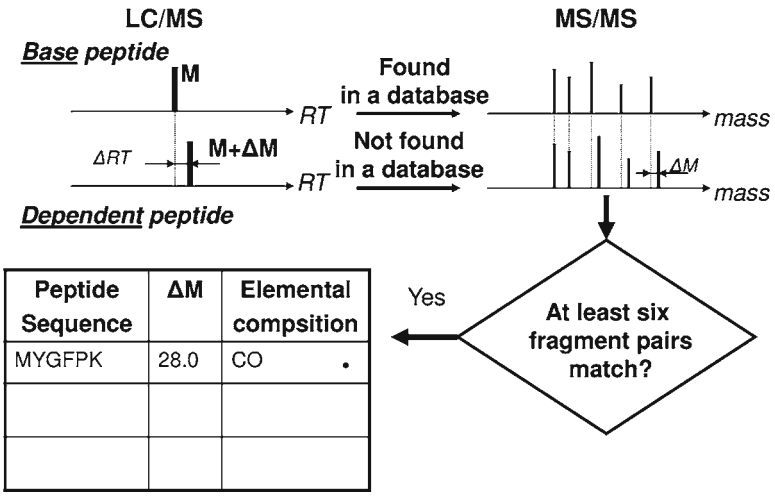


Fig. 2. Flowchart for the unbiased PTM detection. The text explaining the individual steps can be found in the methods part.

Step 2. Make a new dataset of the unidentified MS/MS spectra, these are potential-dependent peptides.

Step 3. For every identified base peptide, search the dataset of unidentified MS/MS spectra for spectra with similar fragment ions (same mass or shifted by the mass of the potential modification, $\Delta M = M_{\text{modified}} - M_{\text{unmodified}}$) (see Note 1). For an MS/MS spectrum to be considered as a dependent peptide to the base peptide, there should be at least six common fragments (see Note 2). The validity of the six fragment approach was tested on high-resolution data using a special decoy database approach described in (31) and was shown to yield less than 2% false-positive hits.

Step 4. The last step is to identify the elemental composition of the modification. This can be done, unambiguously in most cases, when the mass of the peptide is measured within 5 ppm (see Notes 3 and 4).

4. Notes

1. Retention time difference can also be used to identify the pairs of base and dependent peptides, because the dependent peptide may be expected to elute within a limited time window before or after the base peptide. For instance, in a typical 2 h liquid chromatography gradient, the phosphorylated version of a peptide will rarely elute 10 min later or earlier than its unmodified version. By using the retention time, the number of MS/MS spectra of possible dependent peptides can be limited. Although the retention time resolution is relatively low compared with the *m/z* resolution (a standard deviation of around 5 min can

be expected in replicate experiments using a 90 min gradient), it is still a highly useful parameter. For instance, retention time alone can predict whether formylation (+CO) occurs on a serine/threonine side-chain or the N-terminus.

2. If there are MS/MS spectra from both CAD and ECD fragmentation available, these can be added together to filter out real peptide peaks by using golden complementary pairs (32, 33). Golden complementary pairs are fragment ions pairs that can be reliably identified from both CAD and ECD spectra by utilizing the fact that different bonds are cleaved (in CAD the peptide bond and in ECD the N–Cα bond). The fragment ions identified in such a way have a very high, >95%, probability of being true fragment ions. The golden pair matching also gives information about whether the ions are N-terminal or C-terminal fragments. When using golden pair, a criterion of four common fragments between the base and the dependant peptides could be used.
3. The mass accuracy of the precursor ion is important when it comes to identifying the elemental composition of the modification. The unique elemental composition of modifications can be deduced in the majority of the cases when working with MS data of <5 ppm accuracy.
4. An important part of the characterization of PTMs is to evaluate their biological significance –do the modifications have a function in vivo, or are they in vitro modifications due to the sample preparation.

References

1. Cohen P. (2000) The regulation of protein function by multisite phosphorylation – a 25 year update. *Trends in Biochemical Sciences* **25**, 596–601.
2. Tyers M, Jorgensen P. (2000) Proteolysis and the cell cycle: with this RING I do thee destroy. *Current Opinion in Genetics & Development* **10**, 54–64.
3. Jensen ON. (2004) Modification-specific proteomics: characterization of post-translational modifications by mass spectrometry. *Current Opinion in Chemical Biology* **8**, 33–41.
4. Mann M, Jensen ON. (2003) Proteomic analysis of post-translational modifications. *Nature Biotechnology* **21**, 255–61.
5. Blagoev B, Ong SE, Kratchmarova I, Mann M. (2004) Temporal analysis of phosphotyrosine-dependent signaling networks by quantitative proteomics. *Nature Biotechnology* **22**, 1139–45.
6. Blume-Jensen P, Hunter T. (2001) Oncogenic kinase signalling. *Nature* **411**, 355–65.
7. Steen H, Mann M. (2002) A new derivatization strategy for the analysis of phosphopeptides by precursor ion scanning in positive ion mode. *Journal of the American Society for Mass Spectrometry* **13**, 996–1003.
8. Johnson SA, Hunter T. (2005) Kinomics: methods for deciphering the kinome. *Nature Methods* **2**, 17–25.
9. Wells L, Vosseller K, Hart GW. (2001) Glycosylation of nucleocytoplasmic proteins: signal transduction and O-GlcNAc. *Science* **291**, 2376–78.
10. Huang Y, Fang J, Bedford MT, Zhang Y, Xu RM. (2006) Recognition of histone H3 lysin e-4 methylation by the double tudor domain of JMJD2A. *Science* **312**, 748–51.
11. Philips MR, Pillinger MH, Staud R, et al. (1993) Carboxyl methylation of ras-related proteins during signal transduction in neutrophils. *Science* **259**, 977–80.
12. Cenciarelli C, Hou D, Hsu KC, et al. (1992) Activation-induced ubiquitination of the T-cell antigen receptor. *Science* **257**, 795–7.

13. Yamashita M, Fenn JB. (1984) Electrospray ion-source – another variation on the free-jet theme. *Journal of Physical Chemistry* **88**, 4451–9.
14. Karas M, Bachmann D, Hillenkamp F. (1985) Influence of the wavelength in high-irradiance ultraviolet laser desorption mass spectrometry of organic molecules. *Analytical Chemistry* **57**, 2935–9.
15. Tanaka K, Waki H, Ido Y, et al. (1988) Protein and polymer analyses up to m/z 100 000 by laser ionization time-of-flight mass spectrometry. *Rapid Communications in Mass Spectrometry* **2**, 151–3.
16. McLafferty FW. (1981) Tandem mass spectrometry. *Science (New York, NY)* **214**, 280–7.
17. Perkins DN, Pappin DJ, Creasy DM, Cottrell JS. (1999) Probability-based protein identification by searching sequence databases using mass spectrometry data. *Electrophoresis* **20**, 3551–67.
18. Eng JK, McCormack AL, Yates JR. (1994) An approach to correlate tandem mass-spectral data of peptides with amino-acid-sequences in a protein database. *Journal of the American Society for Mass Spectrometry* **5**, 976–89.
19. Craig R, Beavis RC. (2004) TANDEM: matching proteins with tandem mass spectra. *Bioinformatics (Oxford, England)* **20**, 1466–67.
20. Hunt DF, Yates JR, 3rd, Shabanowitz J, Winston S, Hauer CR. (1986) Protein sequencing by tandem mass spectrometry. *Proceedings of the National Academy of Sciences of the United States of America* **83**, 6233–7.
21. Paizs B, Suhai S. (2005) Fragmentation pathways of protonated peptides. *Mass Spectrometry Reviews* **24**, 508–48.
22. Zubarev RA, Kelleher NL, McLafferty FW. (1998) Electron capture dissociation of multiply charged protein cations. A nonergodic process. *Journal of the American Chemical Society* **120**, 3265–6.
23. Syka JE, Coon JJ, Schroeder MJ, Shabanowitz J, Hunt DF. (2004) Peptide and protein sequence analysis by electron transfer dissociation mass spectrometry. *Proceedings of the National Academy of Sciences of the United States of America* **101**, 9528–33.
24. Zubarev RA. (2004) Electron-capture dissociation tandem mass spectrometry. *Current Opinion in Biotechnology* **15**, 12–6.
25. Stensballe A, Jensen ON, Olsen JV, Haselmann KF, Zubarev RA. (2000) Electron capture dissociation of singly and multiply phosphorylated peptides. *Rapid Communications in Mass Spectrometry* **14**, 1793–800.
26. Mirgorodskaya E, Roepstorff P, Zubarev RA. (1999) Localization of O-glycosylation sites in peptides by electron capture dissociation in a Fourier transform mass spectrometer. *Analytical Chemistry* **71**, 4431–6.
27. Kocher T, Savitski MM, Nielsen ML, Zubarev RA. (2006) PhosTShunter: a fast and reliable tool to detect phosphorylated peptides in liquid chromatography Fourier transform tandem mass spectrometry data sets. *Journal of Proteome Research* **5**, 659–68.
28. Craig R, Beavis RC. (2003) A method for reducing the time required to match protein sequences with tandem mass spectra. *Rapid Communications in Mass Spectrometry* **17**, 2310–6.
29. Marshall AG, Hendrickson CL. (2002) Fourier transform ion cyclotron resonance detection: principles and experimental configurations. *International Journal of Mass Spectrometry* **215**, 59–75.
30. Olsen JV, de Godoy LMF, Li GQ, et al. (2005) Parts per million mass accuracy on an orbitrap mass spectrometer via lock mass injection into a C-trap. *Molecular & Cellular Proteomics* **4**, 2010–21.
31. Savitski MM, Nielsen ML, Zubarev RA. (2006) ModifiComb, a new proteomic tool for mapping substoichiometric post-translational modifications, finding novel types of modifications, and fingerprinting complex protein mixtures. *Molecular & Cellular Proteomics* **5**, 935–48.
32. Horn DM, Zubarev RA, McLafferty FW. (2000) Automated de novo sequencing of proteins by tandem high-resolution mass spectrometry. *Proceedings of the National Academy of Sciences of the United States of America* **97**, 10313–7.
33. Nielsen ML, Savitski MM, Zubarev RA. (2005) Improving protein identification using complementary fragmentation techniques in fourier transform mass spectrometry. *Molecular & Cellular Proteomics* **4**, 835–45.

Chapter 13

Protein Quantitation Using Mass Spectrometry

Guoan Zhang, Beatrix M. Ueberheide, Sofia Waldemarson, Sunnie Myung, Kelly Molloy, Jan Eriksson, Brian T. Chait, Thomas A. Neubert, and David Fenyö

Abstract

Mass spectrometry is a method of choice for quantifying low-abundance proteins and peptides in many biological studies. Here, we describe a range of computational aspects of protein and peptide quantitation, including methods for finding and integrating mass spectrometric peptide peaks, and detecting interference to obtain a robust measure of the amount of proteins present in samples.

Key words: Proteomics, Quantitation, Proteins, Peptides, Mass spectrometry

1. Introduction

Mass spectrometry (MS)-based quantitative proteomics has been applied to solve a wide variety of biological problems, and several MS-based workflows have been developed for protein and peptide quantitation (Fig. 1). In mass spectrometric quantitation methods it is usually assumed that the measured signal has a linear dependence on the amount of material in the sample for the entire range of amounts being studied. A prerequisite for accurate quantitation is that unwanted experimental variations in sample extraction, preparation, and analysis be minimized, and it is therefore critical that each step in the workflow is optimized for reproducibility.

One way of optimizing the reproducibility is to label the samples with stable isotopes, mix them together and perform the subsequent sample-handling steps on the mixed sample. The earlier in the workflow that the stable isotope label is introduced and the samples mixed, the smaller is the effect of variations in sample handling.

David Fenyö (ed.), *Computational Biology*, Methods in Molecular Biology, vol. 673,
DOI 10.1007/978-1-60761-842-3_13,

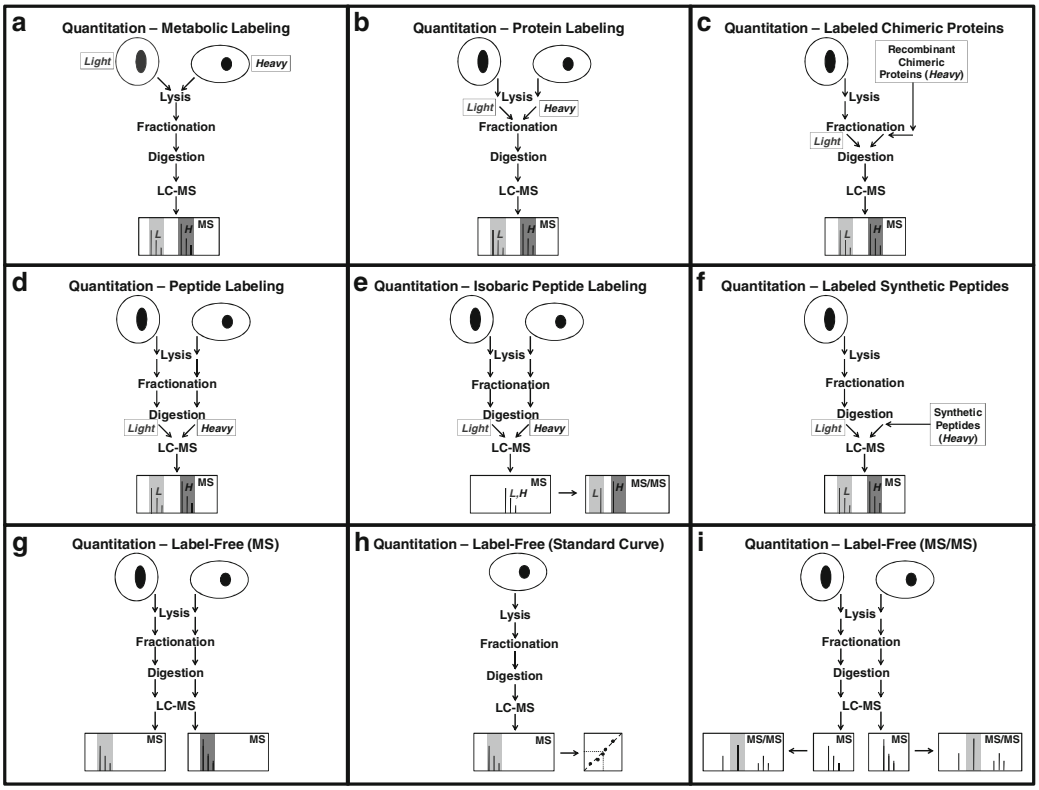


Fig. 1. Workflows for mass spectrometry-based protein and peptide quantitation. (**a**) Metabolic labeling (1, 2). (**b**) Protein labeling (4). (**c**) Chimeric recombinant protein labeling (8, 9). (**d**) Peptide labeling (4, 5). (**e**) Isobaric peptide labeling (7). (**f**) Synthetic peptide labeling (6). (**g**) Label-free quantitation using the intensity of precursor ions (11–13). (**h**) Label-free quantitation using the intensity of precursor ions and a standard curve. (**i**) Label-free quantitation using the intensity of fragment ions.

Metabolic labeling (1, 2) provides the earliest possible introduction of stable isotope labels into the sample (Fig. 1a). Here, labels are introduced as isotopically distinct metabolic precursors, and the samples can be mixed before all subsequent steps in the workflow. It is important to monitor the level of incorporation of the label, but this can, for example, be done by using two heavy labels that are incorporated into the samples with equal efficiency (3). In cases when metabolic labeling is not feasible, the stable isotope labels also can be introduced later in the workflow (4–9) by heavy isotope labeling of proteins (Fig. 1b, c) or peptides (Fig. 1d–f). In general, stable isotope labels need to be designed carefully in order to prevent introducing systematic errors caused by dissimilar behavior of the compounds with different labels. For example, it has been observed that using hydrogen/deuterium substitution in the heavy label can affect the retention time of the labeled peptides, while $^{12}C/^{13}C$ substitution does not have any observable effect on the retention time (10).

Label-free methods (11–13) for quantitation are often used when the introduction of stable isotopes is impractical (e.g., in many animal studies) or the cost is prohibitive (e.g., in biomarker studies where a relatively large number of samples need to be analyzed). Three label-free quantitation workflows are shown in Fig. 1g–i. In these workflows the different samples are analyzed separately and it is therefore critical that each step of the workflow is carefully optimized for reproducibility. In label-free quantitation workflows, usually the peptide ion peaks are integrated and used as a measure of quantity. This allows the quantity of protein and peptides to be compared in different samples (Fig. 1g) or the absolute quantity can be calculated using a standard curve (Fig. 1h). The peptide fragment ions can also be used for quantitation by integrating one or more of their peaks (Fig. 1i) as, for example, in Multiple Reaction Monitoring (MRM) (14). Using fragment ions for quantitation provides increased specificity because in addition to requiring the mass of the precursor ion be close to its predicted mass, the masses of the fragment ions are also required to be correct. Because peptides fragment in a sequence-specific manner, additional specificity can be gained by requiring that the relative intensities of the fragment ions do not deviate from the expected intensities. Alternative methods for quantitation using fragment mass spectra do not integrate peaks but are based on the results of searching protein sequence collections (see Note 1).

Currently, there are several software packages available for analysis of data from these different workflows where the quantitation is done by integrating peaks of ions that correspond to peptides or their fragments (see Note 2 for a few examples). Here, we describe how the mass spectra are processed to allow for finding the peptide peaks, detecting interference, and integrating the peaks to obtain a measure of the amount of material present in the samples.

2. Methods

Step 1: Detecting peptide peaks. Peptide peaks of interest for quantitation may range between smooth peaks with a large signal-to-noise ratio and noisy peaks that are barely above the background. The width of these peaks is, however, characteristic of the resolution of the mass spectrometer, the data acquisition parameters used, as well as the mass-to-charge ratio (m/z) of the peptide. Therefore, peaks can readily be detected by scanning the mass spectra for local maxima of the expected width (see Note 3). In addition, peptides are not observed as a single peak in mass spectrometry, but as a cluster of peaks, because of the presence of

small amounts of stable heavy isotopes in nature (e.g., 1.11% ^{13}C) and each peptide contains many carbon atoms. The relative intensities of the peaks in these isotope clusters are characteristic of the atomic composition of the peptides and they are strongly dependent on the peptide mass (Fig. 2a–c, see Note 4).

A majority of quantitation experiments are performed by coupling liquid chromatography with mass spectrometry, which introduces a retention time dimension. During these experiments, usually the same peptide is observed during several adjacent time points (Fig. 2d–g) with highly abundant peptides typically being observed over larger time windows than low-abundance peptides. But even with separation in both m/z and retention time, it is not uncommon to have unwanted interference between peaks from different peptides (Fig. 2e, g).

Step 2: Detecting interference. The following characteristics of peptide peaks can be used as filters to differentiate them from interfering and non-peptide peaks: (1) the width of individual peaks in m/z and retention time, (2) the intensity distribution of the isotope clusters, and (3) the measured peptide m/z. These characteristics are shown in Fig. 3 for two peptides. The width of individual peaks as a function of m/z is highly characteristic of the instrument parameters with very little variation and therefore a narrow peak width filter can be used. The width of individual peaks as a function of retention time (Fig. 3a–c, j–l) shows larger variation. This variation is mainly dependent on the peak intensity and the elution time, although strong peptide sequence dependent variation can also be observed, and therefore a wider filter must be applied. High-accuracy measurement of peptide mass is a sensitive and selective filter that is highly reproducible even at the tails of the peak where the intensity is low (Fig. 3g–i, p–r). The shape of the isotope distribution is also a sensitive and selective filter that can be used to detect interference from other peaks (Fig. 3d–f, m–o). A convenient measure of the similarity of isotope distributions is the dot product (see Note 5) between them (Fig. 3f, o). The dot product can be applied to compare sets containing any number of peaks, for example, to detect interferences when a set of fragment ions is monitored in a MRM experiment. In the example shown in Fig. 3, dot product analysis of the chromatograms shown in the panels on the right shows that only the first isotope cluster corresponds to the peptide of sequence YVLTQPPSVSVAPGQTAR, while the second and third peaks are interfering peaks from peptides whose first three isotope peaks have a similar m/z, but their relative intensity is different.

Step 3: Measuring peptide quantity. The quantity of peptides is measured by calculating the height or the area of the corresponding peaks in the ion chromatograms. Careful background subtraction is essential for accurate determination of both the height and the

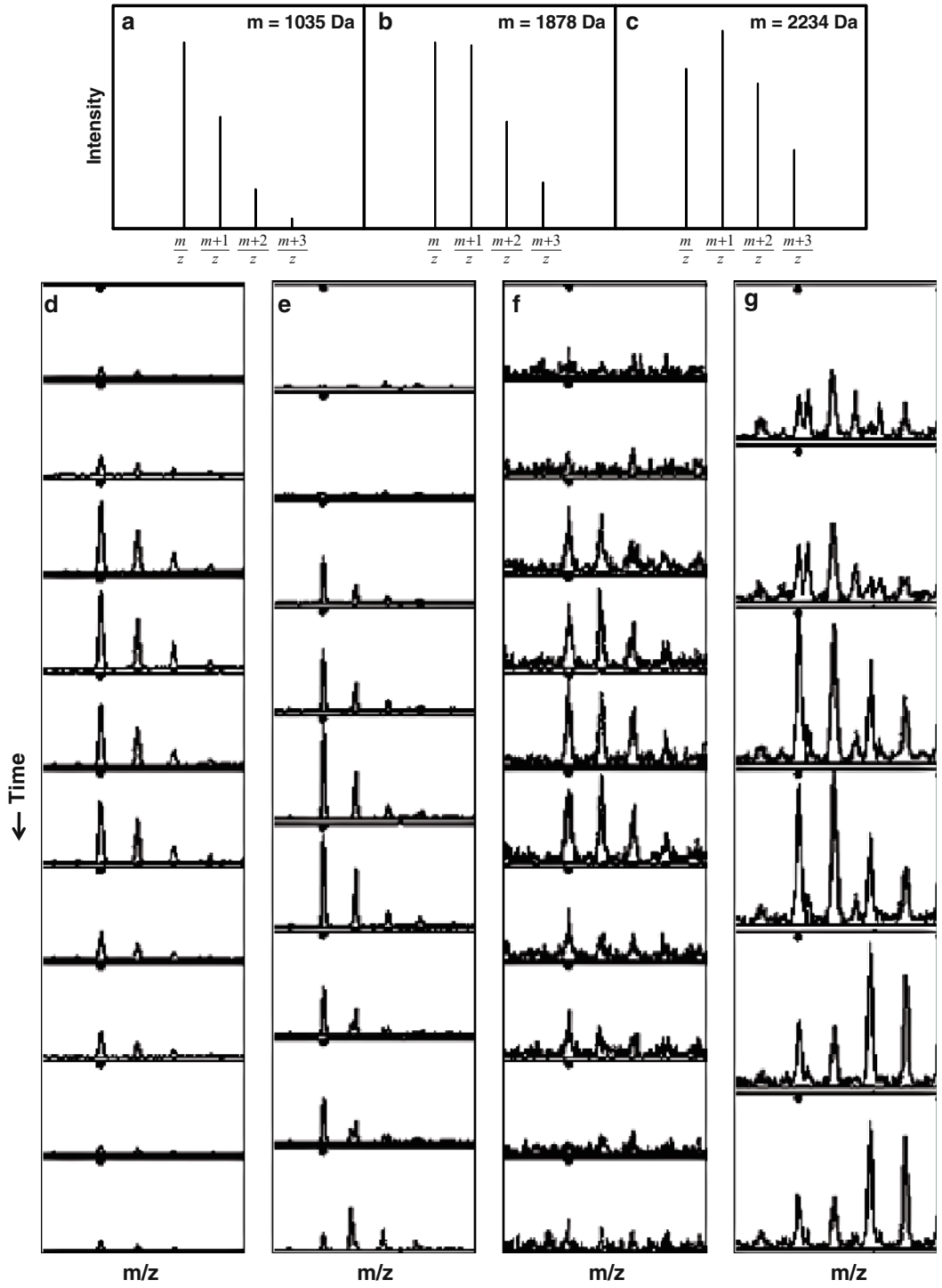


Fig. 2. Isotope distributions of peptides. (**a**–**c**) The isotope distribution of peptides is strongly dependent on the peptide mass (see Note 4). (**d**–**g**) Examples of peptide isotope distributions observed by LC-MS with different levels of interference from other peaks acquired using quadrupole time-of-flight MS.

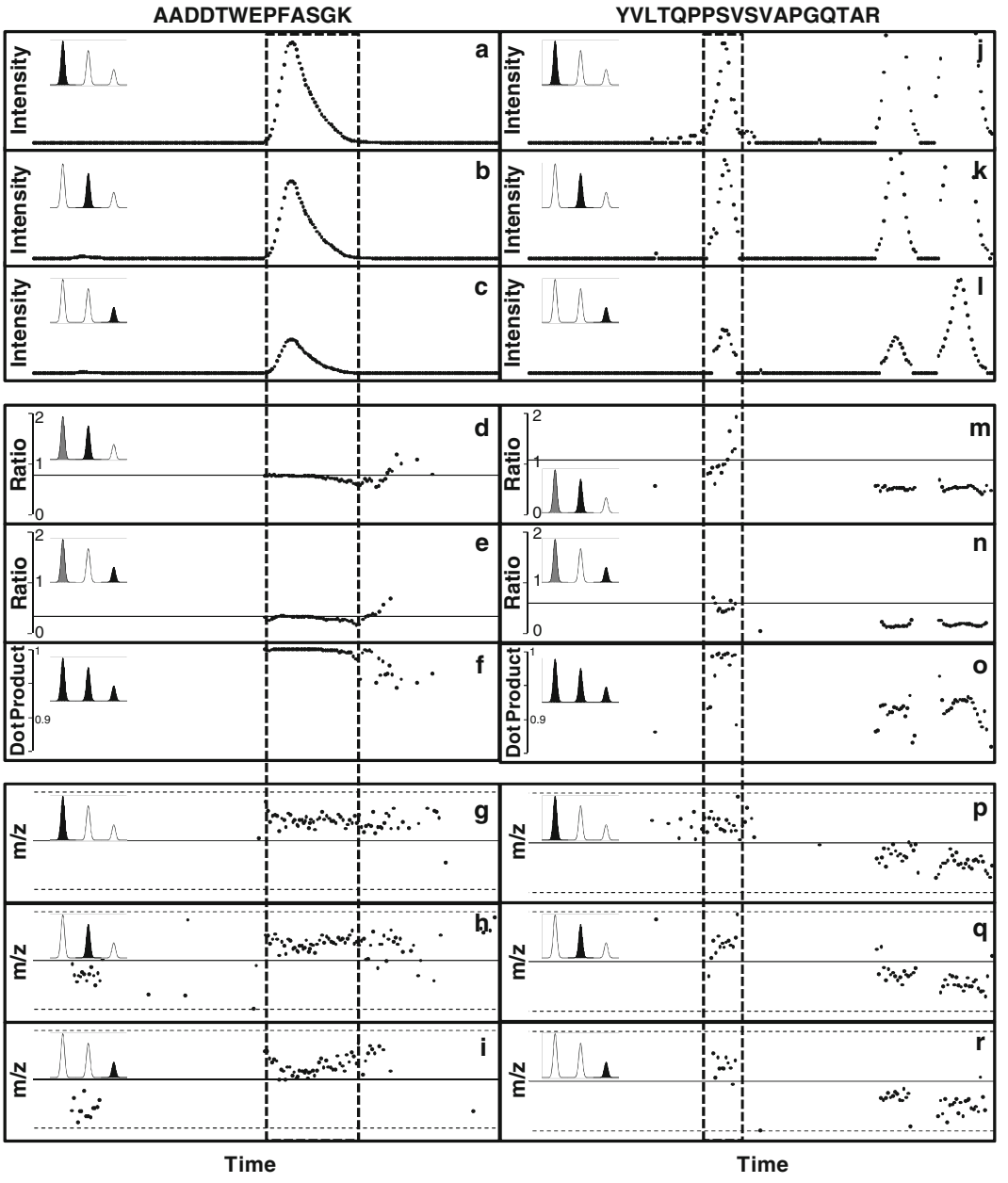


Fig. 3. Examples of the variation in mass measurements and the shape of isotope distributions. (**a**–**i**) Peptide with amino acid sequence: AADDTWEPFASGK; (**j**–**r**) Peptide with amino acid sequence: YVLTQPPSVSVAPGQTAR. Panels from *top* to *bottom*: The intensity distribution of the first (**a**, **j**), second (**b**, **k**), and third (**c**, **l**) isotope peaks as a function of time; the distribution of normalized intensities of the second (**d**, **m**) and the third (**e**, **n**) isotope peaks normalized to the first isotope peak (the *line* represents the expected ratio based on the amino acid sequence); The normalized *dot* product of the three first peaks of the measured and the theoretical isotope distributions (**f**, **o**); the *m*/*z* distribution of the first (**g**, **p**), second (**h**, **q**), and third (**i**, **r**) isotope peaks as a function of time (the *solid line* represents the mass predicted from the amino acid sequence and the *dotted lines* correspond to ±5 ppm).

area of peaks (*see* **Note 6**). The advantage of using the height of the peak as the measure of quantity is the simplicity and robustness of its calculation (e.g., the average or median height for a few points around the centroid can be used). The peak height is a good measure of quantity if the width of the peak does not vary between samples and the signal is strong with little noise. In contrast, the peak area is a better measurement of quantity when there is substantial noise because many more data points are used, but it is much more sensitive to interference from other peaks because of the larger area in the m/z and retention time space that is used. The difficulty in calculating the peak area is in deciding where the peak ends and the background starts in both m/z and retention time dimensions. This determination can be very challenging for peaks with long tails. It is also important to use the same peak limits for a specific peptide in all samples. One way of circumventing the problem of finding the peak limits is to select a function and fit its parameters (e.g., centroid, width, skewness, etc.) to the peak and integrate the function. However, often it is not straightforward to find a function that fits well to all peaks in the spectrum.

Step 4: Matching peptides from different experiments. In many quantitation studies more than one experiment (i.e., replicates and/or multiple samples) is performed. This requires the matching of the peptides quantified in the different experiments. For successful matching of peptides, the retention time scales of all experiments have to be aligned, because there are always uncontrolled variations in the experimental conditions that affect the peptide retention times in a nonlinear manner. This alignment can be done by identifying peaks present in all experiments that can be used as landmarks. These peaks are matched across experiments using either their mass and retention time, or their identity as determined by tandem MS. A smooth function is fitted to the retention times of these landmarks and used for aligning the retention times of all quantified peptides. The residual difference in retention time for the landmarks can be used to estimate the uncertainty in the alignment.

For some mass spectrometers, the m/z scale needs to be calibrated between experiments. This mass calibration can be done using the same landmarks as used for retention time alignment. When experiments are aligned in retention time and are mass calibrated, the quantified peptides can be matched within windows determined by the uncertainty in the retention time and the m/z.

The measured intensities of peptide peaks commonly vary from experiment to experiment in a global manner. It is therefore advisable to design experiments so that only a few of the quantified peptides have changes related to the hypothesis, and the majority of peptides change because of random variations in the experimental conditions. The randomly changing peptides can be used to

normalize the overall intensity using either their median change in the intensity ratios or by fitting an intensity dependent smooth function to the measured intensity ratios.

Step 5: Calculating protein quantity. Protein quantity can be estimated by measuring of peptide quantities. There are, however, several factors that can make the estimates of protein quantity uncertain even when highly accurate peptide quantities have been obtained. Because only a few peptides are typically measured for a given protein, these peptides might not be sufficient to define all isoforms of the protein that are present in the sample – i.e., some of the peptide sequences might be shared with other proteins, making them only suitable for quantitating the group of proteins. A few peptides might also be modified, and the change in the amount of the modified and unmodified forms of the protein is often not the same. Despite these issues, a reasonable estimate of the protein quantity can often be obtained even when only a few of its peptides are quantified. When many peptides are observed for a given protein it can be possible to even calculate the variation in quantity of several isoforms.

Step 6: Determination of the significance of the change in quantity. The significance of a measured change in quantity can be calculated if the distribution of random quantity changes (due to uncontrolled variation of experimental conditions) is known (Fig. 4a). This distribution can be obtained by analysis of technical and biological replicates. When the distribution of random quantity changes is known, the significance of a measured change in quantity can be calculated by integrating under the curve from the measured change in quantity to infinity and dividing this area by the area under the entire distribution of random changes. This value represents the probability that the measured quantity change was obtained from purely random variations, that is, the probability of rejecting the null hypothesis that there is no change in the experimental conditions. The distribution of random quantity changes is strongly dependent on the experimental conditions and the workflow that is chosen. For example, for label-free quantitation the distribution of random quantity changes depends on the number of replicates obtained (Fig. 4b–g). It is important to design quantitation experiments to minimize the width of the distribution of random quantity changes to allow for detection of small nonrandom changes.

3. Notes

1. Alternative methods for quantitation search fragment mass spectra against a protein sequence collection and use the search results for quantitation. One method uses the number

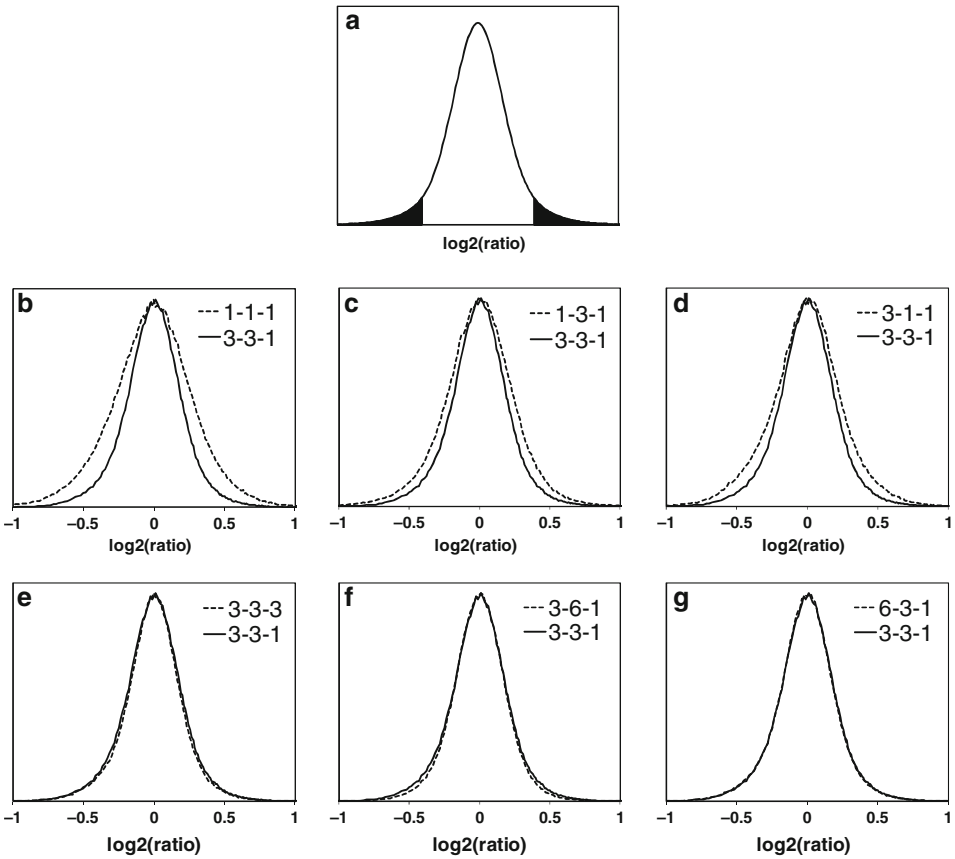


Fig. 4. (**a**) The distribution that represents the null hypothesis, that is, that a given ratio is random. This distribution can be obtained by analysis of samples where only random variation is expected (technical and biological replicates). Then the significance of a ratio measurement can be calculated by integrating this distribution from the measured ratio to infinity. (**b**–**g**) Combining data from repeat analysis makes the distribution that represents the null hypothesis narrower, and smaller changes can be detected. Examples of the effect of replicate analysis on the protein ratio distribution for a workflow comprising immunoprecipitation, protein fractionation, and digestion (simulated data based on measurements of the variation in the individual steps) (26). Only limited improvements are observed beyond 3, 3, 1 repeat analyses for immunoprecipitation, protein fractionation and digestion, respectively (*solid curves*). *Dotted curves*: (**b**) 1, 1, 1; (**c**) 1, 3, 1; (**d**) 3, 1, 1; (**e**) 3, 3, 3; (**f**) 3, 6, 1; (**g**) 6, 3, 1 repeat analyses for immunoprecipitation, protein fractionation and digestion, respectively.

of different fragment mass spectra that identifies a peptide as a measure of its quantity (15). Another method calculates a measure that is based on the fraction of the protein sequence that the identified peptides cover (16). However, these alternative methods that are not based on peak integration are generally less accurate when only a few fragment spectra or peptides are observed for a given protein because of the limited statistics. On the other hand, they are less sensitive to interference and can often be more robust.

2. There are many software packages available for quantitation. A few examples of freely available software are listed below:

Name	Type	Location
ASAPratio (17)	ICAT SILAC	http://tools.proteomecenter.org/wiki/index.php?title=Software:ASAPRatio
MaxQuant (18, 19)	SILAC	http://www.maxquant.org/
MSQuant (20)	SILAC	http://msquant.sourceforge.net/
Pview (21)	SILAC Label-free	http://compbio.cs.princeton.edu/pview/
Quant (22)	iTRAQ	http://sourceforge.net/projects/protms/
RAAMS (23)	$^{16}O/^{18}O$	http://informatics.mayo.edu/svn/trunk/mprc/raams/index.html
Skyline (24)	MRM	http://proteome.gs.washington.edu/software.html

3. For a mass spectrum where $I(k)$ is the measured intensity at a point k with $0 \le k \le N$, and N is the total number of points in the mass spectrum. The peaks are detected by calculating the sum, $S(l) = \sum_{|k-l|<w_l/2} I(k)$ over the expected peak width w_l for each point, l, in the spectrum, and detecting local maxima in $S(l)$. In cases where there is sufficient noise in the spectrum the signal-to-noise ratio is calculated by taking the ratio of the root mean square (RMS) of the intensities over the peak ($\text{RMS} = \sqrt{\sum_{|k-l|<w_l/2} (I(k) - \hat{I})^2 / w_l}$, where $\hat{I}$ is the mean intensity over the peak) and the RMS of the intensities in a nearby region where there are no peaks (see Note 6).
4. Peptides are observed as clusters of peaks in mass spectrometry, because of the presence of small amounts of stable heavy isotopes in nature (e.g., 0.015% ^{2}H, 1.11% ^{13}C and 0.366% ^{15}N, 0.038% ^{17}O, 0.200% ^{18}O, 0.75% ^{33}S, 4.21% ^{34}S, 0.02% ^{36}S). The intensities of the isotope distribution are calculated accurately by including all possible isotopes. The largest effect comes from ^{13}C and a first order estimate of the relative peak intensities is given by $T_m = \binom{n}{m} p^m (1-p)^{n-m}$, where T_m is the intensity of peak m in the distribution, m is the number of ^{13}C, n the total number of carbon atoms in the peptide, and p is the probability for ^{13}C (i.e., 1.11%). The isotope distribution of peptides is strongly dependent on the peptide mass because the number of atoms increases with mass, and therefore the probability increases for having one or more of the naturally occurring heavy isotopes.
5. The normalized dot product between the measured intensities, $\mathbf{I} = (I_1, I_2, \ldots, I_n)$ and theoretical intensities $\mathbf{T} = (T_1, T_2, \ldots, T_n)$ of the isotope distribution is given by

$\frac{\mathbf{I}\cdot\mathbf{T}}{|\mathbf{I}||\mathbf{T}|} = \sum_{k=1}^{n} I_k T_k \Big/ \sqrt{\sum_{k=1}^{n} I_k^2 \sum_{k=1}^{n} T_k^2}$. The range of the normalized dot product is from –1 to 1. If the measured and theoretical intensities are identical the resulting dot product is 1 and any differences between them will result in lower values of the dot product.

6. Low-frequency background can be removed by fitting a smooth curve to the regions of the mass spectrum where there are no peaks. This smoothing can, for example, be achieved by applying a very wide and strong smoothing function to the entire spectrum, which will result in a smooth function slightly higher than the background. Subsequently, points in the original spectrum that are far above this smooth curve are removed (i.e., the peaks). The smoothing procedure is repeated, this time without including the peaks, to produce a smooth function that will closely follow the background of the spectrum (25).

Acknowledgments

This work was supported by funding provided by the National Institutes of Health Grants RR00862, RR022220, NS050276, and CA126485, the Carl Trygger foundation, and the Swedish research council.

References

1. Y. Oda, K. Huang, F.R. Cross, D. Cowburn, and B.T. Chait (1999) Accurate quantitation of protein expression and site-specific phosphorylation, *Proc Natl Acad Sci USA*, **96**, 6591–6.
2. S.E. Ong, B. Blagoev, I. Kratchmarova, D.B. Kristensen, H. Steen, A. Pandey, and M. Mann (2002) Stable isotope labeling by amino acids in cell culture, SILAC, as a simple and accurate approach to expression proteomics, *Mol Cell Proteomics*, **1**, 376–86.
3. B. Schwanhausser, M. Gossen, G. Dittmar, and M. Selbach (2009) Global analysis of cellular protein translation by pulsed SILAC, *Proteomics*, **9**, 205–9.
4. S.P. Gygi, B. Rist, S.A. Gerber, F. Turecek, M.H. Gelb, and R. Aebersold (1999) Quantitative analysis of complex protein mixtures using isotope-coded affinity tags, *Nat Biotechnol*, **17**, 994–9.
5. O.A. Mirgorodskaya, Y.P. Kozmin, M.I. Titov, R. Korner, C.P. Sonksen, and P. Roepstorff (2000) Quantitation of peptides and proteins by matrix-assisted laser desorption/ionization mass spectrometry using (18)O-labeled internal standards, *Rapid Commun Mass Spectrom*, **14**, 1226–32.
6. S.A. Gerber, J. Rush, O. Stemman, M.W. Kirschner, and S.P. Gygi (2003) Absolute quantification of proteins and phosphoproteins from cell lysates by tandem MS, *Proc Natl Acad Sci USA*, **100**, 6940–5.
7. P.L. Ross, Y.N. Huang, J.N. Marchese, B. Williamson, K. Parker, S. Hattan, N. Khainovski, S. Pillai, S. Dey, S. Daniels, S. Purkayastha, P. Juhasz, S. Martin, M. Bartlet-Jones, F. He, A. Jacobson, and D.J. Pappin (2004) Multiplexed protein quantitation in *Saccharomyces cerevisiae* using amine-reactive isobaric tagging reagents, *Mol Cell Proteomics*, **3**, 1154–69.
8. R.J. Beynon, M.K. Doherty, J.M. Pratt, and S.J. Gaskell (2005) Multiplexed absolute quantification in proteomics using artificial QCAT proteins of concatenated signature peptides, *Nat Methods*, **2**, 587–9.

9. L. Anderson and C.L. Hunter (2006) Quantitative mass spectrometric multiple reaction monitoring assays for major plasma proteins, *Mol Cell Proteomics*, **5**, 573–88.
10. E.C. Yi, X.J. Li, K. Cooke, H. Lee, B. Raught, A. Page, V. Aneliunas, P. Hieter, D.R. Goodlett, and R. Aebersold (2005) Increased quantitative proteome coverage with (13)C/(12)C-based, acid-cleavable isotope-coded affinity tag reagent and modified data acquisition scheme, *Proteomics*, **5**, 380–7.
11. P. Schulz-Knappe, H.D. Zucht, G. Heine, M. Jurgens, R. Hess, and M. Schrader (2001) Peptidomics: the comprehensive analysis of peptides in complex biological mixtures, *Comb Chem High Throughput Screen*, **4**, 207–17.
12. W. Wang, H. Zhou, H. Lin, S. Roy, T.A. Shaler, L.R. Hill, S. Norton, P. Kumar, M. Anderle, and C.H. Becker (2003) Quantification of proteins and metabolites by mass spectrometry without isotopic labeling or spiked standards, *Anal Chem*, **75**, 4818–26.
13. M.C. Wiener, J.R. Sachs, E.G. Deyanova, and N.A. Yates (2004) Differential mass spectrometry: a label-free LC-MS method for finding significant differences in complex peptide and protein mixtures, *Anal Chem*, **76**, 6085–96.
14. T.A. Addona, S.E. Abbatiello, B. Schilling, S.J. Skates, D.R. Mani, D.M. Bunk, C.H. Spiegelman, L.J. Zimmerman, A.J. Ham, H. Keshishian, S.C. Hall, S. Allen, R.K. Blackman, C.H. Borchers, C. Buck, H.L. Cardasis, M.P. Cusack, N.G. Dodder, B.W. Gibson, J.M. Held, T. Hiltke, A. Jackson, E.B. Johansen, C.R. Kinsinger, J. Li, M. Mesri, T.A. Neubert, R.K. Niles, T.C. Pulsipher, D. Ransohoff, H. Rodriguez, P.A. Rudnick, D. Smith, D.L. Tabb, T.J. Tegeler, A.M. Variyath, L.J. Vega-Montoto, A. Wahlander, S. Waldemarson, M. Wang, J.R. Whiteaker, L. Zhao, N.L. Anderson, S.J. Fisher, D.C. Liebler, A.G. Paulovich, F.E. Regnier, P. Tempst, and S.A. Carr (2009) Multi-site assessment of the precision and reproducibility of multiple reaction monitoring-based measurements of proteins in plasma, *Nat Biotechnol*, **27**, 633–41.
15. H. Liu, R.G. Sadygov, and J.R. Yates, 3rd (2004) A model for random sampling and estimation of relative protein abundance in shotgun proteomics, *Anal Chem*, **76**, 4193–201.
16. Y. Ishihama, Y. Oda, T. Tabata, T. Sato, T. Nagasu, J. Rappsilber, and M. Mann (2005) Exponentially modified protein abundance index (emPAI) for estimation of absolute protein amount in proteomics by the number of sequenced peptides per protein, *Mol Cell Proteomics*, **4**, 1265–72.
17. X.J. Li, H. Zhang, J.A. Ranish, and R. Aebersold (2003) Automated statistical analysis of protein abundance ratios from data generated by stable-isotope dilution and tandem mass spectrometry, *Anal Chem*, **75**, 6648–57.
18. J. Cox and M. Mann (2008) MaxQuant enables high peptide identification rates, individualized p.p.b.-range mass accuracies and proteome-wide protein quantification, *Nat Biotechnol*, **26**, 1367–72.
19. J. Cox, I. Matic, M. Hilger, N. Nagaraj, M. Selbach, J.V. Olsen, and M. Mann (2009) A practical guide to the MaxQuant computational platform for SILAC-based quantitative proteomics, *Nat Protoc*, **4**, 698–705.
20. P. Mortensen, J.W. Gouw, J.V. Olsen, S.E. Ong, K.T. Rigbolt, J. Bunkenborg, J. Cox, L.J. Foster, A.J. Heck, B. Blagoev, J.S. Andersen, and M. Mann (2010) MSQuant, an open source platform for mass spectrometry-based quantitative proteomics, *J Proteome Res*, 9(1):393–403.
21. Z. Khan, J.S. Bloom, B.A. Garcia, M. Singh, and L. Kruglyak (2009) Protein quantification across hundreds of experimental conditions, *Proc Natl Acad Sci USA*, **106**, 15544–8.
22. A.M. Boehm, S. Putz, D. Altenhofer, A. Sickmann, and M. Falk (2007) Precise protein quantification based on peptide quantification using iTRAQ, *BMC Bioinformatics*, **8**, 214.
23. C.J. Mason, T.M. Therneau, J.E. Eckel-Passow, K.L. Johnson, A.L. Oberg, J.E. Olson, K.S. Nair, D.C. Muddiman, and H.R. Bergen, 3rd (2007) A method for automatically interpreting mass spectra of 18O-labeled isotopic clusters, *Mol Cell Proteomics*, **6**, 305–18.
24. B. MacLean, D.M. Tomazela, N. Shulman, M. Chambers, G. Finney, B. Frewen, R. Kern, D.L. Tabb, D.C. Liebler and M.J. Maccoss (2010) Skyline: an open source document editor for creating and analyzing targeted proteomics experiments, Bioinformatics, 26, 966–8.
25. E.M. Woo, D. Fenyo, B.H. Kwok, H. Funabiki, and B.T. Chait (2008) Efficient identification of phosphorylation by mass spectrometric phosphopeptide fingerprinting, *Anal Chem*, **80**, 2419–25.
26. G. Zhang, D. Fenyo, and T.A. Neubert (2009) Evaluation of the variation in sample preparation for comparative proteomics using stable isotope labeling by amino acids in cell culture, *J Proteome Res*, **8**, 1285–92.

Chapter 14

Modeling Experimental Design for Proteomics

Jan Eriksson and David Fenyö

Abstract

The complexity of proteomes makes good experimental design essential for their successful investigation. Here, we describe how proteomics experiments can be modeled and how computer simulations of these models can be used to improve experimental designs.

Key words: Proteomics, Mass spectrometry, Experimental design, Simulations, Modeling

1. Introduction

The proteomics researcher that aims at *comprehensive* proteome analysis using mass spectrometry (MS)-based methods will face experimental challenges. These challenges are due to the many different proteins encoded by a genome, the rich variation of protein posttranslational modifications, and the large concentration differences between different proteins. The range of protein concentrations have been measured to be six orders of magnitude in *Saccharomyces cerevisiae* (1) and estimated to be larger than ten orders of magnitude in body fluids (2). In contrast to these wide abundance ranges, the MS detection methods typically employed in proteomics span only a few orders of magnitude in range, hampering the identification and quantitation of low-abundance proteins. A good experimental design for proteomics should manage to keep the detection of low-abundance proteins and the cost for instrumentation and analysis at reasonable and desired levels.

Proteomics researchers have realized that the complexity and the range of protein abundance of a proteome make it necessary to apply various separation protocols prior to the MS-analysis.

David Fenyö (ed.), *Computational Biology*, Methods in Molecular Biology, vol. 673,
DOI 10.1007/978-1-60761-842-3_14,

Most current experimental designs in proteomics (3) involve (1) taking samples of proteins relevant to the biological hypothesis or phenomenon explored; (2) protein separation by liquid chromatography (LC) and/or gel electrophoresis (4); (3) protein digestion using an enzyme of high specificity; (4) chromatographic (5) or electrophoretic separation (6) of the proteolytic peptides; (5) mass spectrometric (MS) analysis (7); and (6) searching a protein sequence collection to identify proteins (8–10) based on the MS and MS/MS information. There are many choices available for each step in the workflow, and this makes the parameter space for the workflow design large (Fig. 1).

Optimization of experimental design in the large parameter space by relying on experiments only would be prohibitively expensive, and it is therefore bound to yield an incomplete investigation. Instead, we have proposed a simulation-based optimization approach (11) that employs an experimental model. This approach can be used to evaluate the success of current designs, predict the performance of future, and further optimized proteomics experimental designs. Here, we describe methods for building the experimental model, and show an example how the model can be applied to optimize proteomics experiments.

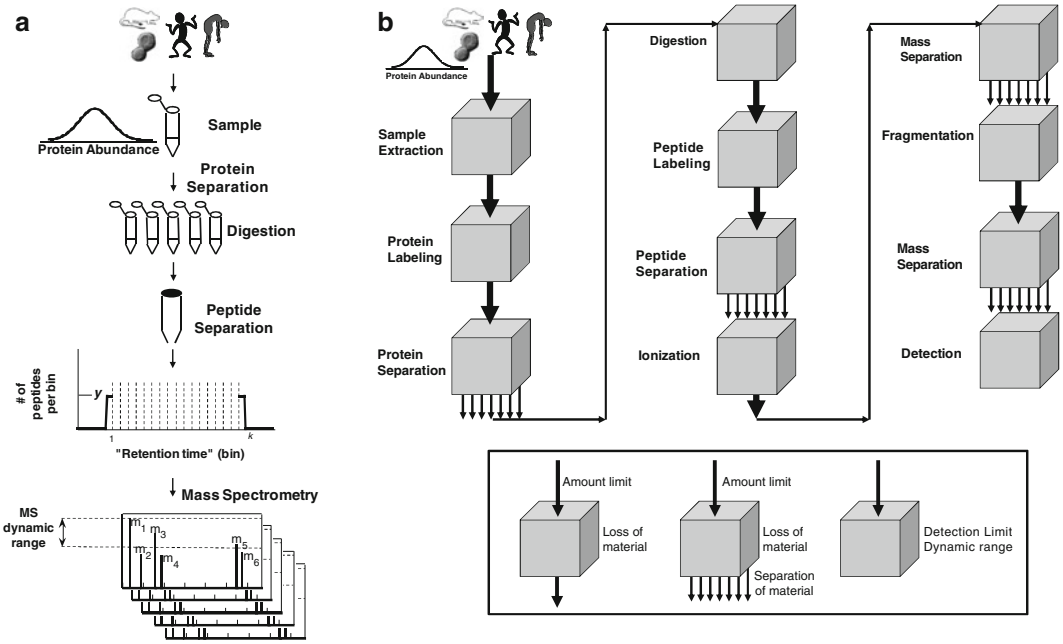


Fig. 1. (**a**) Model of a common proteomics experiment. (**b**) Generalized model of a proteomics experiment.

2. Methods

Any computer simulation (see Note 1) needs input of reasonable assumptions about the model parameters in order to generate meaningful predictions about the experimental reality. The best overall strategy to improve experimental design is to use simulations together with good background information about the experimental components. In a general model of proteomics experiments there are many parameters (Fig. 1b), and many of these can be very difficult to determine (see Note 2), but often a simple model is sufficient to find the bottle-necks in the experimental design. The benefit of simulations is that once there is meaningful information available about parts of a system, this information can be employed in many different combinations in the computer to generate predictions much more rapidly than by experimental investigation. The simulations can also be used to determine which parameters are important to determine experimentally. Therefore, the proteomics researcher that would like to investigate and improve an experimental design should perform some model experiments or by other means determine the important model parameters. Pertinent information about all the parts that are important for the experimental design should be derived. The task can be viewed as containing three parts: (1) the protein sample, (2) the peptide sample, and (3) the mass spectrometry.

2.1. The Protein Sample

The protein abundance distribution in the sample is always uncertain, but models describing two major groups of distributions, tissue (Fig. 2a) and body fluid (Fig. 2b), have been suggested (11). The tissue distribution is based on protein quantitation experiments using an antibody against a tag engineered into the protein sequence of individual *S. cerevisiae* genes, followed by quantitative western blot analysis (1). This experiment revealed a bell-shaped distribution of proteins ranging about six orders of magnitude in abundance. The body fluid distribution was

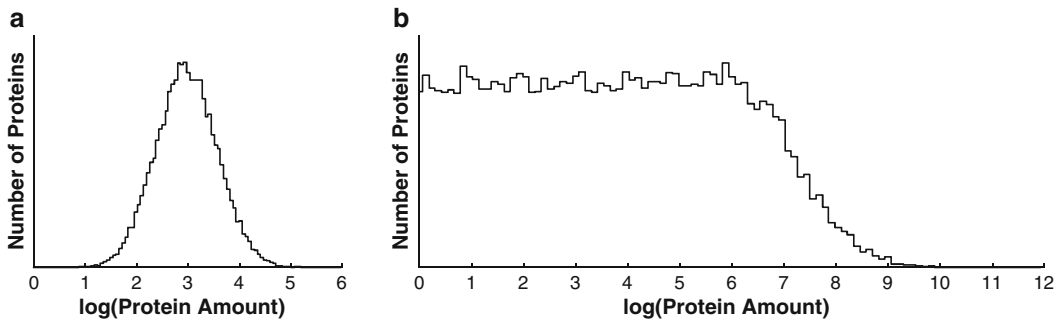


Fig. 2. Protein abundance distributions for (**a**) tissue and (**b**) body fluid.

assumed to cover a larger range of protein abundances (2), and to be bell-shaped at high abundances, and flat at low abundances. The flat shape at low abundances was chosen because many different tissues in the body contribute proteins at a low level to the body fluid proteome. These distributions need to be calibrated based on the specific details of the experiment modeled. For example, these models do not take into account modified proteins. A scaling toward lower abundances is needed if, e.g., phospho-proteins are to be detected.

The next steps in the workflow that need to be modeled are protein separation by electrophoresis or chromatography and the subsequent digestion of the proteins with endoproteases. The losses associated with these steps need to be estimated. Here we refer this mixture of proteolytic peptides originating from the digested proteins as the peptide sample.

2.2. The Peptide Sample

Separation of peptides is typically done using a reverse phase chromatography (RPC) column. The loading capacity and the resolving power of the RPC column should be estimated and incorporated in the model. The elution time of peptides in RPC is dependent on their sequence and can be estimated (12). There are many possible sources of losses for peptides: they can stick to walls, not bind to the column, or bind too hard to the column so that they cannot be eluted. All these losses are sequence dependent and difficult to elucidate in detail, but they can be estimated from model experiments.

2.3. The Mass Spectrometry

In model experiments, samples from peptide libraries can be employed to estimate the detection sensitivity and dynamic range of the mass spectrometer. Note that the dynamic range of the mass spectrometer is the *ratio* of concentrations for *two different peptide species* that can be *detected simultaneously*, and it is much narrower than the range of concentrations over which a single peptide species can be detected when there are no other peptides in the sample. The rate of acquisition of the mass spectrometer can be determined in various modes of operation. In experimental designs with the mass spectrometer coupled online with the RPC column, the limited rate of acquisition will cause losses of peptides that are potentially detectable. Other sources of losses in the mass spectrometry step include low ionization and fragmentation efficiencies.

3. Results

Using a simple model for a typical proteomics experiment, we investigated the effect of changing the dynamic range and detection limit of the mass spectrometer on the *success rate* and the

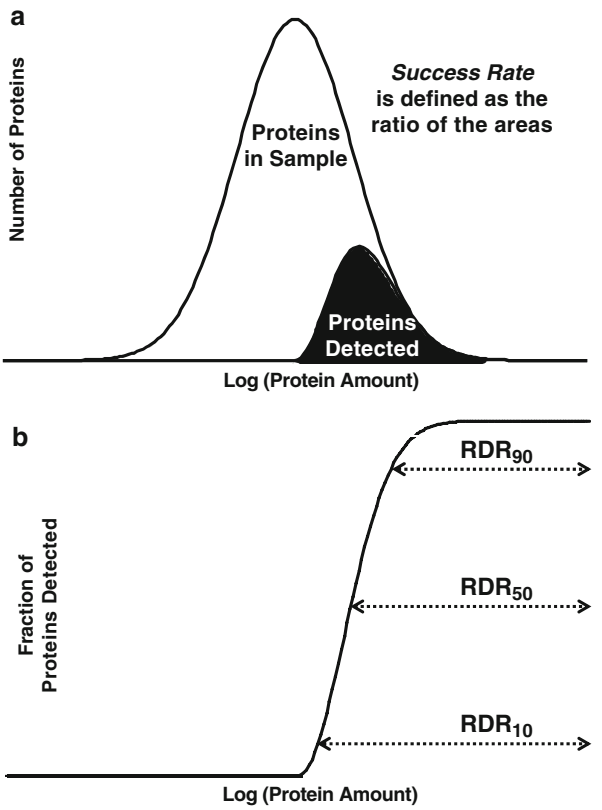


Fig. 3. Definitions: (**a**) The success rate of an experiment (**b**) and the relative dynamic range of an experiment.

relative dynamic range (*RDR*). The success rate indicates what fraction of the proteome is detected (Fig. 3a), and the *RDR* indicates how deep down into the low abundance proteins an experimental design can manage to detect proteins (Fig. 3b). The assumptions of the simple model are that (1) the abundance distribution of proteins in the sample is given by Fig. 2a; (2) proteins are separated into a number of fractions each having the same number of proteins without any losses; (3) the proteins in fraction are digested with trypsin and loaded onto a reverse phase column with the peptides having a probability of being lost; (4) the peptides are separated by RPC and analyzed by MS with a certain probability of not being detected.

Figure 4 displays an example of how simulations (see Note 1) can reveal the impact on the *success rate* and the *RDR* by one feature of the sample preparation and two features of the mass spectrometer: the degree of *protein separation*, the *MS detection limit*, and the *MS dynamic range*. The top left panel of Fig. 4a indicates how the *Success rate* and the *RDR* vary when *first* improving the protein separation, *then* improving the *MS detection limit*, and *finally* improving the *MS dynamic range*. The right

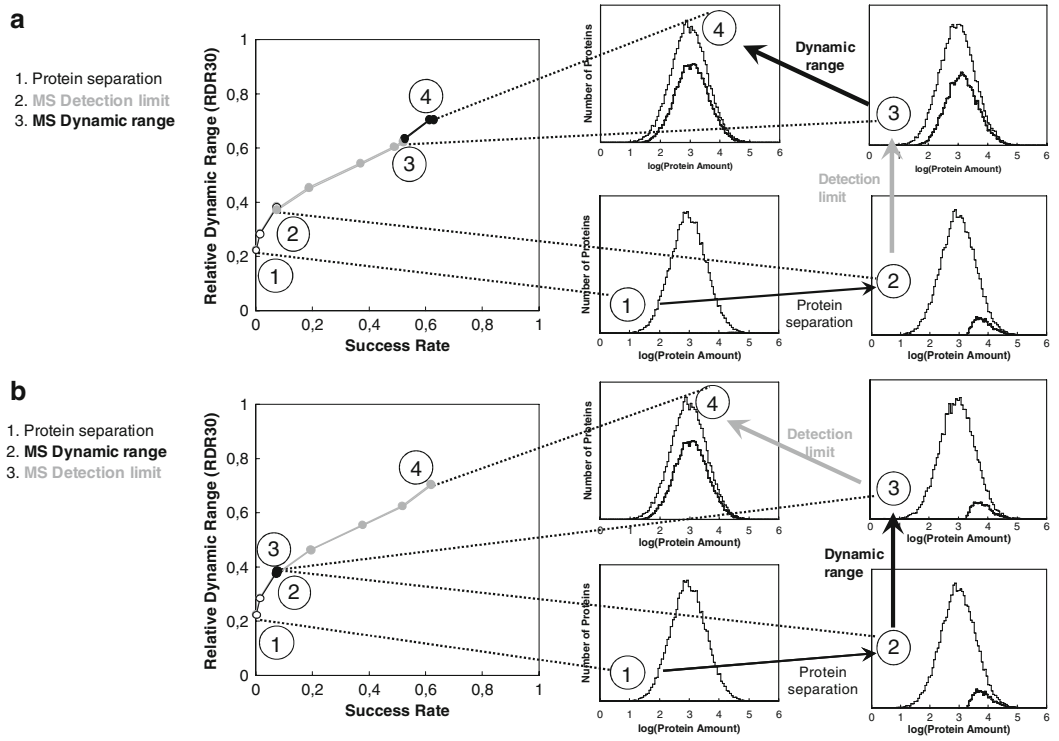


Fig. 4. Results from simulations showing the effect of protein separation and the effect of MS detection limit and MS dynamic range on the success rate, and the relative dynamic range (RDR) for detection of proteins from *Homo sapiens* tissue samples. (**a**) *Left*: RDR as a function of success rate when first improving the protein separation and going from 30,000 (1) to 300 proteins (2) in each fraction, then enhancing the sensitivity of the mass spectrometer from 1 fmol to 1 amol (3), and finally improving the MS dynamic range from 10^2 to 10^4 (4). *Right*: The protein abundance distribution assumed for human tissue together with the distribution of the proteins detected for the experimental designs (1–4). (**b**) Same as in (**a**), but with the MS dynamic range improved prior to improving the MS detection sensitivity. Note that the effect of improving the dynamic range is negligible compared with the effect of improving the detection sensitivity.

panel of Fig. 4a shows the protein abundance distribution model employed in the simulation together with the distribution of the proteins detected for the initial design (Fig. 4a, 1), the design with better protein separation (Fig. 4a, 2), after improving the detection limit (Fig. 4a, 3), and after enhancing the MS dynamic range (Fig. 4a, 4). It is evident that all these three features of the experimental design can influence strongly the outcome of an experiment. The *way* in which design parameters are changed can however be critical. For example, if instead upon improving the protein separation, the MS dynamic range is enhanced, there is no improvement of the success rate and the *RDR* until also the MS detection limit is improved (Fig. 4b, 1–4).

Simulations also reveal that improving the detection sensitivity of the mass spectrometer is analogous to increasing the amount of peptide material loaded in the peptide separation step, and that

improving the MS dynamic range is analogous to enhancing the proteolytic peptide separation (11). The starting point in Fig. 4 assumes no protein separation, a load of 0.1 µg of peptides in the peptide separation step that separates the peptides into 100 fractions, and a mass spectrometer with a detection sensitivity of 1 fmol and a dynamic range of 100. This setup is not uncommon in proteomics, but is obviously the wrong choice for comprehensive analysis. If comprehensive analysis is desired, Fig. 4 and results in ref. 11 show clearly that the practitioner should avoid the design (Fig. 4a, 1) and employ some protein separation and either load more material in the peptide separation step or choose a mass spectrometer with better detection sensitivity prior to either improving separation of peptides or improving the MS dynamic range.

4. Notes

1. In the simulations, a mixture of human proteins is randomly selected. The estimated distribution of protein amounts in the sample (Fig. 2a) is used to assign an amount to each protein in the mixture, and the protein mixture is digested. The resulting proteolytic peptides are randomly selected based on a precolumn survival probability. The surviving peptides are separated into fractions according to a separation model (12). The separated peptides are randomly selected based on a postcolumn survival probability. The surviving peptides are considered detected by MS if their amount is above the detection limit and their peak intensity is within the dynamic range of the mass spectrometer. The entire process is repeated many times to obtain sufficient statistics.
2. A general model for a proteomics experiment has many parameters and it is often not feasible to determine many of them experimentally. An alternative to experimental determination of model parameters is to investigate how sensitive the conclusions are to the model parameters. The experimental effort does not need to be focused on parameters that do not affect the conclusions when varied within a wide range. For example, the loss of material in the different workflow steps are often difficult to estimate in absolute numbers, therefore their impact was investigated by changing the pre- and postcolumn peptide survival rates between 10 and 100%. Within this range of peptide survival rates the conclusions drawn from Fig. 4 did not change.

Acknowledgments

This work was supported by funding provided by the National Institutes of Health Grants RR00862 and RR022220, the Carl Trygger foundation, and the Swedish research council.

References

1. S. Ghaemmaghami, W.K. Huh, K. Bower, R.W. Howson, A. Belle, N. Dephoure, E.K. O'Shea, and J.S. Weissman (2003) Global analysis of protein expression in yeast, *Nature*, **425**, 737–41.
2. N.L. Anderson and N.G. Anderson (2002) The human plasma proteome: history, character, and diagnostic prospects, *Mol Cell Proteomics*, **1**, 845–67.
3. R. Aebersold and M. Mann (2003) Mass spectrometry-based proteomics, *Nature*, **422**, 198–207.
4. H. Wang, S.G. Clouthier, V. Galchev, D.E. Misek, U. Duffner, C.K. Min, R. Zhao, J. Tra, G.S. Omenn, J.L. Ferrara, and S.M. Hanash (2005) Intact-protein-based high-resolution three-dimensional quantitative analysis system for proteome profiling of biological fluids, *Mol Cell Proteomics*, **4**, 618–25.
5. Y. Ishihama (2005) Proteomic LC-MS systems using nanoscale liquid chromatography with tandem mass spectrometry, *J Chromatogr A*, **1067**, 73–83.
6. B.J. Cargile, J.L. Bundy, T.W. Freeman, and J.L. Stephenson, Jr. (2004) Gel based isoelectric focusing of peptides and the utility of isoelectric point in protein identification, *J Proteome Res*, **3**, 112–9.
7. J.J. Coon, J.E. Syka, J. Shabanowitz, and D.F. Hunt (2005) Tandem mass spectrometry for peptide and protein sequence analysis, *Biotechniques*, **38**, 519, 521, 523.
8. R.S. Johnson, M.T. Davis, J.A. Taylor, and S.D. Patterson (2005) Informatics for protein identification by mass spectrometry, *Methods*, **35**, 223–36.
9. L. McHugh and J.W. Arthur (2008) Computational methods for protein identification from mass spectrometry data, *PLoS Comput Biol*, **4**, e12.
10. D. Fenyo (2000) Identifying the proteome: software tools, *Curr Opin Biotechnol*, **11**, 391–5.
11. J. Eriksson and D. Fenyo (2007) Improving the success rate of proteome analysis by modeling protein-abundance distributions and experimental designs, *Nat Biotechnol*, **25**, 651–5.
12. O.V. Krokhin, R. Craig, V. Spicer, W. Ens, K.G. Standing, R.C. Beavis, and J.A. Wilkins (2004) An improved model for prediction of retention times of tryptic peptides in ion pair reversed-phase HPLC: its application to protein peptide mapping by off-line HPLC-MALDI MS, *Mol Cell Proteomics*, **3**, 908–19.

Chapter 15

A Functional Proteomic Study of the *Trypanosoma brucei* Nuclear Pore Complex: An Informatic Strategy

Jeffrey A. DeGrasse and Damien Devos

Abstract

The nuclear pore complex (NPC) is the sole mediator of transport between the nucleus and the cytoplasm. The NPC is composed of about 30 distinct proteins, termed nucleoporins or nups. The yeast (Rout et al., J Cell Biol 148:635–651, 2000) and mammalian (Cronshaw et al., J Cell Biol 158:915–927, 2002) NPC have been extensively studied. However, the two species are relatively closely related. Thus, to reveal details about NPC evolution, we chose to characterize the NPC of a distantly related organism, *Trypanosoma brucei*. We took a subcellular proteomic approach and used several complementary strategies to identify 865 proteins associated with the nuclear envelope. Over 50% of ~8,100 open reading frames of *T. brucei* have little or no known function because *T. brucei* is distantly related to model metazoa and fungi (Berriman et al., Science 309:416–422, 2005). By sequence similarity alone, we could identify only five nucleoporins. This chapter outlines our strategy to identify 17 additional nucleoporins as well as contribute functional annotation data to the *T. brucei* genome database.

Key words: *Trypanosoma brucei*, Functional proteomics, Functional annotation, Informatics, Evolution, Nuclear pore complex, Structure prediction

1. Introduction

The reductionist scientific experiment focuses on one molecule, gene, or protein. Rapidly advancing and accessible computational tools have allowed scientists to probe complex biological networks with an integrative strategy. This interdisciplinary, and often collaborative, field is known as systems biology. Genomics and proteomics are both focused on the complicated interaction networks of biological macromolecules and how these networks respond to stimuli.

Functional genomics uses prediction algorithms to identify and functionally annotate putative open reading frames (ORFs).

David Fenyö (ed.), *Computational Biology*, Methods in Molecular Biology, vol. 673,
DOI 10.1007/978-1-60761-842-3_15, © Springer Science+Business Media, LLC 2010

In a large proportion of the cases, function is predicted by the similarity of a query ORF to a growing library of known genes or domains in an automated fashion. Inevitably, there is a subset of ORFs that do not significantly match to a previously functionally annotated gene due to the loss of sequence similarity through divergent evolution. In the case of *Trypanosoma brucei*, over 50% of the ORFs have insufficient functional annotation (3).

The union of separation science, biological mass spectrometry, and informatics led to the field of proteomics. With appropriate sample preparation and separation (e.g., chromatography), the current generation of mass spectrometers are sufficiently efficient and sensitive to analyze and indentify a few thousand peptides, corresponding to several hundred proteins, in a single experiment (4). The identification of peptides from the raw mass spectra relies heavily on informatics and genomics (5). At a minimum, an early draft genome sequence is required for a large-scale proteomic project.

The flow of information between functional genomics and proteomics is bidirectional. Functional genomics enables the proteomic community to quickly identify the function of an identified protein. Proteomic data can reveal that an ORF is expressed under the conditions of the proteomic experiment and thus confirming that it is not a pseudogene.

We discuss here the role of informatics as a bridge between functional genomics and proteomics to reveal the functional nature of a protein. As a case study, we outline our general informatics strategy to identify the protein components (collectively known as nucleoporins or nups) of the *T. brucei* nuclear pore complex (NPC)(23). *T. brucei* is a member of the order Kinetoplastida, which is distantly related to other model organisms, such as mammals and yeast. This distant relationship challenges automated functional genomics. Thus, despite identifying over 865 proteins associated with the *T. brucei* nuclear envelope (NE), the paucity of functional annotation data challenged our ability to readily ascribe function to a significant number of the experimentally identified proteins. The integrative strategy we outline in this chapter overcomes that challenge and allowed us to successfully identify the vast majority of *T. brucei* nups.

2. Materials

The programs used in this strategy are outlined below (see Note 1).

1. Sequence alignment:
   (a) PSI-BLAST (http://www.ncbi.nlm.nih.gov/BLAST/) (6).

   (b) FASTA (http://fasta.bioch.virginia.edu/) (7).
   (c) HMMER (http://hmmer.janelia.org/) (8).
2. Pattern matching: ProteinInfo (http://prowl.rockefeller.edu/).
3. Motif prediction:
   (a) Phobius (http://phobius.cbr.su.se/) (9).
   (b) DISOPRED (http://bioinf.cs.ucl.ac.uk/disopred/) (10).
   (c) COILS (http://www.ch.embnet.org/software/COILS_form.html) (11).
   (d) Nucleo (http://pprowler.itee.uq.edu.au/Nucleo-Release-1.0/) (12).
4. Secondary structure prediction: PSIPRED (http://bioinf.cs.ucl.ac.uk/psipred/) (13).
5. Fold prediction: HHSearch (http://toolkit.tuebingen.mpg.de/hhpred) (14).
6. Multiple sequence alignments: ClustalW/ClustalX (http://www.clustal.org/) (15).

3. Methods

A subcellular proteomic approach was utilized to identify proteins associated with the *T. brucei* NE and, specifically, the trypanosome nuclear pore complex (TbNPC). To that end, we isolated the NE from the cytoplasm and the nucleoplasm (16, 17). Using several complementary proteomic strategies, such as hydroxyapatite chromatography LC-MS and SDS–PAGE MALDI-MS, we identified 865 proteins associated with the NE.

To identify the putative nucleoporins (nups) present in our dataset, the following strategy was developed (see Fig. 1). This general strategy is necessary when significant primary structure similarity between species has been lost due to divergent evolution. As in the case described here, inferring protein function between distantly related species by homology can be particularly difficult. In trypanosomes, sequence homology alone is sufficient to identify only five constituents (of ~30 predicted nups in yeast (1) and humans (2)) of the TbNPC (TbSec13, TbNup96, TbNup158, TbNup144, and TbNup62). Using the below strategy, we identified 17 additional nups, allowing us to characterize the majority of the trypanosome nups.

3.1. Parsing the Dataset and Homology Searching

1. We begin parsing the 865 member dataset by cross referencing each identified protein to its functional annotation page on GeneDB, the functional annotation database of the

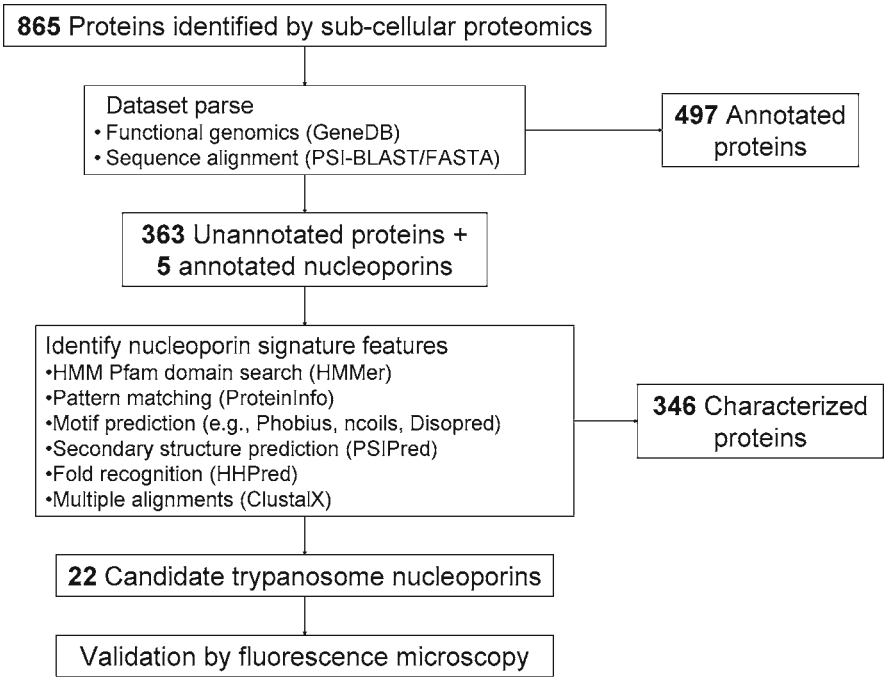


Fig. 1. Flowchart illustrating the logical parsing of the proteomic dataset as described in this strategy.

Wellcome Trust Sanger Institute. If a protein had been functionally annotated, and found not to be related to the NE or the NPC, then the protein was cataloged and set aside. There is a possibility that these proteins may functionally interact with the NE, but further study would be required to demonstrate such an interaction or localization. In this way, 497 proteins were set aside from further interrogation and five proteins were immediately identified as putative nups.

2. Following genome sequencing, ORFs are automatically predicted and functionally annotated. To reduce the number of incorrectly predicted and functionally annotated ORFs, high-confidence thresholds are set for these predictions. This leads to a large number of proteins without functional annotation due to low pairwise alignment scores. Thus, we first manually functionally annotated and characterized the remaining 368 proteins identified by mass spectrometry by pairwise alignments. The sequences were queried using both PSI-BLAST and FASTA against the National Center for Biotechnology Information (NCBI) nonredundant database (see Note 2). The alignments were seeded with a word size of 2, filters were turned off, and alignments were scored against the BLOSUM45 matrix. Setting the word size to 2 increases the sensitivity of the alignment at the expense of computational efficiency, and the BLOSUM45 matrix is

calibrated for low-similarity alignments. The resulting alignments were individually inspected by hand to avoid false positives (see Note 3).

3. Querying the domain architectures between distant orthologs is more sensitive than querying the entire primary structure (18). To identify conserved domains, we conducted a hidden Markov model (HMM) alignment using HHMer, to the Pfam HMM-profile database of domain families (19). This search is particularly sensitive, because the query sequence is aligned to a profile, or consensus sequence, generated by a family of sequences.

3.2. Pattern Matching

Occasionally, a domain will have a conserved sequence pattern. Within the nups, the FG-repeat domain contains the FG dipeptide at regular or irregular intervals. Based on the FG-repeat domains examined prior to this study, we established the following criteria for the prediction of FG-repeat domains: the FG dipeptide is present at least five times within 200 residues; there is a depletion of arginine; and the intervening sequence is compositionally enriched in proline, serine, threonine, asparagine, and glutamine. Aside from the FG dipeptide, the intervening sequence is not conserved enough to be confidently identified by BLAST alone. Thus, we also scanned both the trypanosome protein database (http://www.genedb.org) and our dataset for the presence of FG-repeat domains by using a pattern recognition algorithm within ProteinInfo (20). The pattern is entered using regular expressions to allow for plasticity in the sequence. The search revealed ten putative FG-repeat containing nups within our proteomic dataset that met the overall criteria for an FG-repeat domain. All recognizable FG-repeat domains in the trypanosome genomic database were identified in the proteomics dataset.

3.3. Motif Prediction

At this point, we have identified ten FG-repeat containing nups (two of which were previously annotated) as well as three previously annotated nups. To continue parsing the remaining ~200 proteins, we predicted the presence of several types of motifs with various prediction algorithms. We concentrated on the motifs that are present within proteins known to associate with the NPC and NE, which include transmembrane helices (Phobius), natively disordered regions (Disopred and PONDR), coiled coils (COILS), and putative nuclear localization sequence (NLS, Nucleo) (see Note 4). We kept for further investigation those results that had better than an 80% predictive confidence score, based on the benchmarks of the individual algorithm.

3.4. Secondary Structure Prediction

Exhausting primary structure similarity and motif prediction, we turn next to secondary structure prediction to identify nups in

our proteomic dataset. Previous work showed that nups fall into eight major fold types with characteristic secondary structure patterns (21). To facilitate the detection of these specific fold types, we predicted the secondary structure of all 368 unannotated proteins using PSI-PRED. When interpreting the results, one should bear in mind that these algorithms require primary structure similarity for accurate prediction. For the purposes of this study, we are not concerned by the details of element size and boundaries. We are concerned only whether a domain is primarily β-sheet rich or α-helix rich, as the previously identified fold types have these characteristics. This method allowed us to identify nine additional putative *T. brucei* nups for a total of 22 *T. brucei* nups.

3.5. Fold Prediction

Complementary to secondary structure prediction, we predicted the three-dimensional fold type of the putative trypanosome nups using the HHSearch algorithm. We considered a protein a nup if it was confidently predicted to have one of the fold types previously defined as a nup-specific fold type (21). The confidence increased, if the arrangement of the domains was consistent with previously described architectures (21, 22).

3.6. Multiple Alignments

For several known domains and conserved sequences, multiple sequence alignments were conducted with ClustalX using default settings. Conserved residues are indicative of a functional role within the NPC and nucleocytoplasmic transport and, whenever possible, were cross checked with the literature for any mutational analysis. The multiple alignments also elucidate phylogenetic relationships and probe the evolutionary history of the system of interest.

3.7. Concluding Remarks

This strategy yielded a comprehensive inventory of the *T. brucei* nups. By searching for modules using the algorithms outlined in the previous sections, rather than sequence similarity alone, we identified an additional 17 putative nups, for a total of 22 nups. Based on comparisons with the nup inventories of NPCs from vertebrates and yeast, we estimate that we have identified at least 80%, by mass, of the trypanosome NPC. We anticipate that the balance is most likely species-specific or highly divergent nups, which would be difficult to identify within the proteomic dataset by comparative methods. Of the 22 putative TbNups identified in this work, 21 were confirmed by fluorescence microscopy localization in vivo.

Aside from the 497 annotated, but unrelated, proteins and the 22 nups, we characterized the remaining 346 functionally unannotated proteins within the proteomic dataset. Using the strategies noted above, we found 23% of these proteins contain at least one coiled coil, 30% of the unannotated proteins are predicted to have at least one transmembrane helix (TMH), and 9%

of the proteins do contain an NLS. At least one Pfam domain was identified in 33% of the unannotated proteins. The details of these domains and motifs was uploaded to the *T. brucei* functional genomics database to participate in the functional characterization of this organism.

4. Notes

1. This is not an exhaustive list of available programs. Others may be found at sites such as the European Bioinformatics Institute (EBI) toolbox (http://www.ebi.ac.uk/Tools/) and the Swiss Institute of Bioinformatics (SIB) ExPASy (Expert Protein Analysis System) server (http://us.expasy.org/). Advanced users may consider downloading and executing local versions of informatic programs described in this chapter. Doing so allows the user to search custom self-curated databases and scoring matrices (to reduce false-positive rate), and the outputs can be readily saved. Also, because the program will run faster on a local computer, the user can run several experiments with different parameters to test the robustness of a result.
2. Unless otherwise stated, the default parameters were used for each program.
3. While Expect scores are a good indicator of a confident alignment, a trained eye can inspect the alignment closely to see if the score has not been artificially inflated due to low-complexity regions (regions that contain repetitive sequences) or artificially deflated due to small, but significant, regions of similarity.
4. We hasten to add that NLS prediction is not as sophisticated as other motif prediction algorithms, and the results should be used with caution until further benchmark standards have been established.

Acknowledgments

The authors would like to acknowledge Brian T. Chait, Mark C. Field, Michael P. Rout, and Andrej Sali for the helpful advice and discussions. The work was supported by the Training Program in Chemical Biology (JAD).

References

1. Rout MP, Aitchison JD, Suprapto A, Hjertaas K, Zhao YM, Chait BT. (2000) The yeast nuclear pore complex: Composition, architecture, and transport mechanism. The Journal of Cell Biology; 148:635–51.
2. Cronshaw JA, Krutchinsky AN, Zhang WZ, Chait BT, Matunis MJ. (2002) Proteomic analysis of the mammalian nuclear pore complex. The Journal of Cell Biology; 158:915–27.
3. Berriman M, Ghedin E, Hertz-Fowler C, et al. (2005) The genome of the African trypanosome Trypanosoma brucei. Science; 309:416–22.
4. Atwood JA, Weatherly DB, Minning TA, et al. (2005) The Trypanosoma cruzi proteome. Science; 309:473–6.
5. McHugh L, Arthur JW. (2008) Computational methods for protein identification from mass spectrometry data. PLoS Computational Biology; 4:e12.
6. Altschul SF, Madden TL, Schaffer AA, et al. (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Research; 25:3389–402.
7. Pearson WR, Lipman DJ. (1988) Improved tools for biological sequence comparison. Proceedings of the National Academy of Sciences of the United States of America; 85:2444–8.
8. Eddy SR. (1998) Profile hidden Markov models. Bioinformatics; 14:755–63.
9. Kall L, Krogh A, Sonnhammer ELL. (2004) A combined transmembrane topology and signal peptide prediction method. Journal of Molecular Biology; 338:1027–36.
10. Ward JJ, Sodhi JS, McGuffin LJ, Buxton BF, Jones DT. (2004) Prediction and functional analysis of native disorder in proteins from the three kingdoms of life. Journal of Molecular Biology; 337:635–45.
11. Lupas A, Vandyke M, Stock J. (1991) Predicting coiled coils from protein sequences. Science; 252:1162–4.
12. Hawkins J, Davis L, Boden M. (2007) Predicting nuclear localization. Journal of Proteome Research; 6:1402–9.
13. McGuffin LJ, Bryson K, Jones DT. (2000) The PSIPRED protein structure prediction server. Bioinformatics; 16:404–5.
14. Soding J, Biegert A, Lupas AN. (2005) The HHpred interactive server for protein homology detection and structure prediction. Nucleic Acids Research; 33:W244–8.
15. Larkin MA, Blackshields G, Brown NP, et al. (2007) Clustal W and Clustal X version 2.0. Bioinformatics; 23:2947–8.
16. Rout MP, Field MC. (2001) Isolation and characterization of subnuclear compartments from Trypanosoma brucei – Identification of a major repetitive nuclear lamina component. Journal of Biological Chemistry; 276:38261–71.
17. DeGrasse JA, Chait BT, Field MC, Rout MP. High-yield isolation and subcellular proteomic characterization of nuclear and subnuclear structures from trypanosomes. In: Hancock R, ed. Methods in Molecular Biology: The Nucleus. New York: Humana Press; 2008:77–92.
18. Bateman A, Birney E, Durbin R, Eddy SR, Howe KL, Sonnhammer ELL. (2000) The Pfam protein families database. Nucleic Acids Research; 28:263–6.
19. Finn RD, Tate J, Mistry J, et al. (2008) The Pfam protein families database. Nucleic Acids Research; 36:D281–8.
20. Fenyö D, Zhang W, Beavis RC, Chait BT. (1996) Internet-based analytical chemistry resources – a model project. Analytical Chemistry; 68:A721–6.
21. Devos D, Dokudovskaya S, Williams R, et al. (2006) Simple fold composition and modular architecture of the nuclear pore complex. Proceedings of the National Academy of Sciences of the United States of America; 103:2172–7.
22. Devos D, Dokudovskaya S, Alber F, et al. (2004) Components of coated vesicles and nuclear pore complexes share a common molecular architecture. PLoS Biology; 2:e380.
23. DeGrasse JA, DuBois KN, Devos D, Siegel TN, Sali A, Field MC, Rout MP, Chait BT. (2010) Evidence for a Shared Nuclear Pore Complex Architecture That Is Conserved from the Last Common Eukaryotic Ancestor. Molecular & Cellular Proteomics; 8:2119–30.

Chapter 16

Inference of Signal Transduction Networks from Double Causal Evidence

Réka Albert, Bhaskar DasGupta, and Eduardo Sontag

Abstract

Here, we present a novel computational method, and related software, to synthesize signal transduction networks from single and double causal evidences. This is a significant and topical problem because there are currently no high-throughput experimental methods for constructing signal transduction networks, and because the understanding of many signaling processes is limited to the knowledge of the signal(s) and of key mediators' positive or negative effects on the whole process. Our software NET-SYNTHESIS is freely downloadable from http://www.cs.uic.edu/~dasgupta/network-synthesis/.

Our methodology serves as an important first step in formalizing the logical substrate of a signal transduction network, allowing biologists to simultaneously synthesize their knowledge and formalize their hypotheses regarding a signal transduction network. Therefore, we expect that our work will appeal to a broad audience of biologists. The novelty of our algorithmic methodology based on nontrivial combinatorial optimization techniques makes it appealing to computational biologists as well.

Key words: Computational biology, Network inference, Signal transduction, Systems biology, Double causal evidence

1. Introduction

Most biological characteristics of a cell involve complex interactions between its numerous constituents such as DNA, RNA, proteins, and small molecules (1). Cells use signaling pathways and regulatory mechanisms to coordinate multiple functions, allowing them to respond to and acclimate to an ever-changing environment. In a signal transduction network (pathway), there is typically an input, perceived by a receptor, followed by a series of elements through which the signal percolates to the output node, which represents the final outcome of the signal transduction process. For a cellular signal transduction pathway not involving alterations in

David Fenyö (ed.), *Computational Biology*, Methods in Molecular Biology, vol. 673,
DOI 10.1007/978-1-60761-842-3_16,

gene expression, elements often consist of proteinaceous receptors, intermediary signaling proteins, metabolites, effector proteins, and a final output that represents the ultimate combined effect of the effector proteins. If the signal transduction process includes regulation of the transcript level of a particular gene, the intermediate signaling elements will also include the gene itself and the transcription factors that regulate it, as well any small RNAs that regulate the transcript's abundance, with the final output being presence or absence of transcript. Genome-wide experimental methods now identify interactions among thousands of proteins (2–5). However, the state-of-the-art understanding of many signaling processes is limited to the knowledge of key mediators and of their positive or negative effects on the whole process.

The experimental evidence about the involvement of specific components in a given signal transduction network frequently belongs to one of these three categories:

(a) *Biochemical evidence.* This type of evidence provides information on enzymatic activity or protein–protein interactions. These are "direct," physical interactions. Examples include:
   - Binding of two proteins,
   - A transcription factor activating the transcription of a gene, or
   - A simple chemical reaction with a single reactant and single product.

(b) *Pharmacological evidence.* This type of experimental evidence is generated by processes in which a chemical is used either to mimic the elimination of a particular component or to exogenously provide a certain component, leading to observed relationships that are not direct interactions but indirect causal relationships most probably resulting from a chain of direct interactions and/or reactions.

(c) *Genetic evidence of differential responses to a stimulus.* Such evidence in a wild-type organism versus a mutant organism implicates the product of the mutated gene in the signal transduction process. This category is a three-component inference as it involves the stimulus, the mutated gene product, and the response. We will call this category as a *double causal inference.*

In this chapter, we describe a method for synthesizing single and double causal information into a consistent network. Our method significantly expands the capability for incorporating indirect (pathway-level) information. Previous methods of synthesizing signal transduction networks only include direct biochemical interactions, and are therefore restricted by the incompleteness of the experimental knowledge on pair-wise interactions. Figure 1 shows a sche-

matic diagram of our overall goal. Mathematical and more technical details about our method are available in our publications (6–9).

A starting point in applying our method involves distilling experimental conclusions into qualitative regulatory relations between cellular components. We differentiate between positive and negative regulation by using the verbs "promote" and "inhibit" and representing them graphically as → and ⊣, respectively (see Fig. 2). Biochemical and pharmacological evidence is represented as a component-to-component relationship, such as "A promotes B," and is incorporated as a directed edge (also called link) from vertex (also called node) A to B (see Fig. 2). Edges corresponding to "known" (documented) direct interactions are marked as "critical." Genetic evidence leads to double

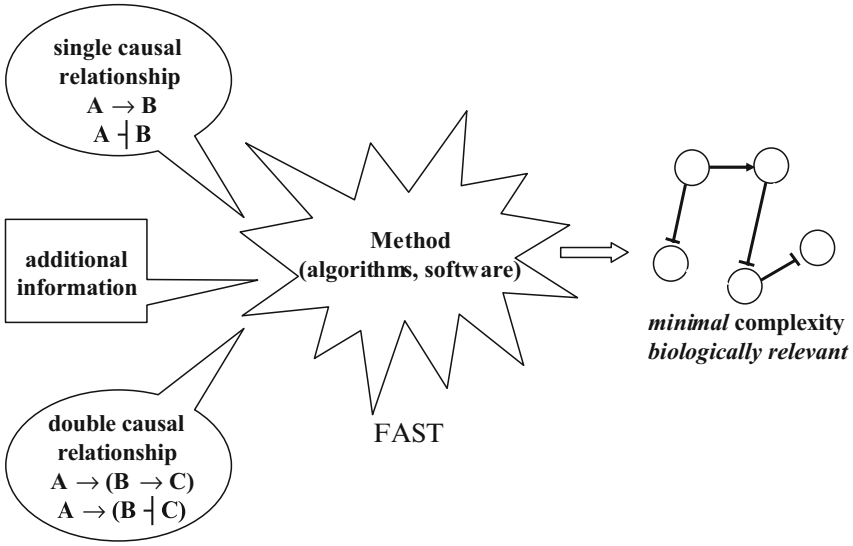


Fig. 1. A schematic diagram of the overall goal of our method.

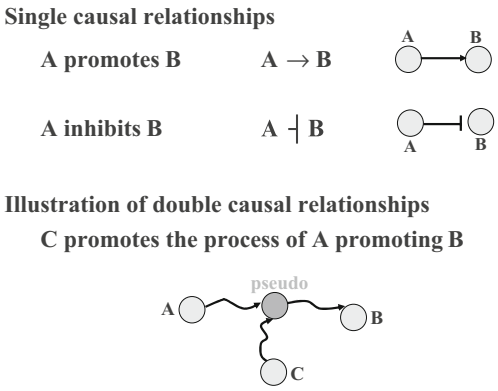


Fig. 2. *Direct and double causal interactions*. Illustration of graph–theoretic interpretations of various types of interactions.

causal inferences of the type "C promotes the process through which A promotes B." We assume that a three-node double causal inference corresponds to an intersection of two paths (one path from A to B and another path from C to B) in the interaction network; in other words, we assume that C activates an unknown intermediary (*pseudo*) vertex of the AB path; see Fig. 2 for a pictorial illustration.

The main idea of our method is to find a minimal graph, both in terms of pseudo-vertex numbers and noncritical edge numbers, that is consistent with all reachability relationships between nonpseudo ("real") vertices. A schematic diagram of an overall high-level view of our method is shown in Fig. 3 and a detailed diagram appears in Fig. 4. Two main computational steps involved are: (1) *binary transitive reduction* (BTR) of a resulting graph subject to the constraints that *no* edges flagged as direct are eliminated and (2) *pseudo-vertex collapse* (PVC) subject to the constraints that real vertices are *not* eliminated. In the next two subsections, we discuss these two computational substeps in more detail.

1.1. Pseudo-vertex Collapse

Intuitively, the PVC problem is useful for reducing the pseudo-vertex set to the minimal set that maintains the graph consistent with all double causal experimental observations. Computationally, an exact solution of this problem can be obtained in polynomial time.

The PVC operation is shown schematically in Fig. 5. Mathematically, the PVC computational problem can be defined as follows. Our input is a signal transduction network $G=(V, E)$ with vertex set V and edge set E in which a subset of vertices are pseudo-vertices. For any vertex v, the vertex sets are defined as follows:

$$\text{in}(v)=\{(u,x) \mid u \text{ has a path to } v \text{ of type } x \text{ with } x \in \{\rightarrow$$

$$\text{out}(v)=\{(u,x) \mid u \text{ has a path to } v \text{ of type } x \text{ with } x \in \{\rightarrow$$

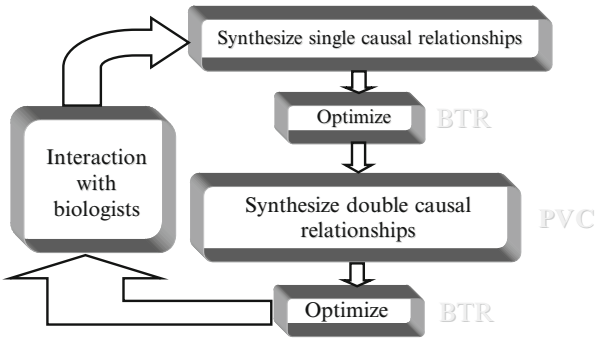
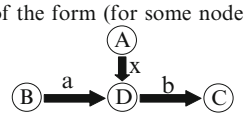
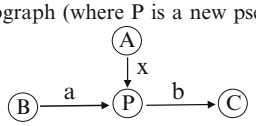


Fig. 3. *High-level description of the network synthesis process.* PVC and BTR refer to the pseudo-vertex collapse and the binary transitive reduction computational steps, respectively.

1. **[encode single causal relationships]**
   1.1 Build networks for connections like A → B and A ⊣ B noting each critical edge.
   1.2 Apply BTR
2. **[encode double causal reltionships]**
   2.1 For each double causal relationship of the form $A \xrightarrow{x} (B \xrightarrow{y} C)$ with x,y ∈ {0,1}, add new nodes and/or edges as follows:
   - if $B \xrightarrow{y} C \in E_{critical}$ then add $A \xrightarrow{x} (B \xrightarrow{y} C)$
   - if no subgraph of the form (for some node D with b = a+b = y (mod 2))



   then add the subgraph (where P is a new pseudo-node and b = a+b = y (mod 2))



   2.2 Apply PVC
3. **[final reduction]** Apply BTR

Fig. 4. *Algorithmic details of the basic network synthesis procedure* (8). In this diagram, a *right arrow* → labeled by 0 denotes a "promotes" relation and a *right arrow* → labeled by 1 denotes an "inhibits" relation. Similarly, a *right double arrow* ⇒ labeled by 0 denotes a "promotes" *path* and a *right double arrow* ⇒ labeled by 1 denotes an "inhibits" *path*. $E_{critical}$ denotes the set of critical edges. The mathematical notation like $a+b=c$ (mod 2) indicates that $a+b$ has the same remainder as c when divided by 2.

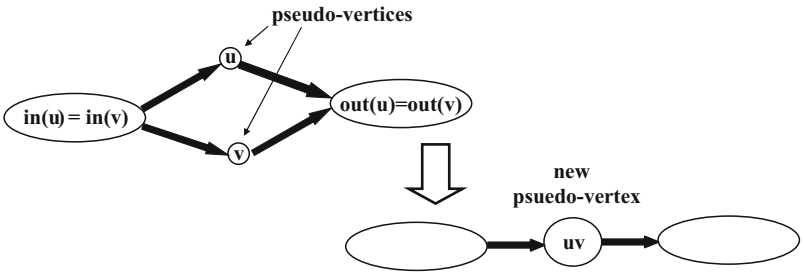


Fig. 5. *Pictorial illustration of a PVC operation*. Repeatedly performing this operation results in a graph with fewer nodes and edges

Collapsing two vertices u and v is *permissible* provided *both* are *not* real vertices, $\text{in}(u)=\text{in}(v)$ and $\text{out}(u)=\text{out}(v)$. A PVC operation is as follows: if permissible, *collapse* two vertices u and v to create a new vertex w, make every incoming (respectively, outgoing) edge to (respectively, from) either u or v an incoming (respectively, outgoing) edge from w, remove any parallel edges that may result from the collapse operation and also remove both vertices u and v. A *valid solution* consists of a network $G'=(V', E')$ obtained from G by a sequence of permissible collapse operations; the goal is to *minimize* the number of edges in E'.

1.2. Binary Transitive Reduction

Intuitively, the BTR problem is useful for determining a sparsest graph consistent with a set of experimental observations. Computationally, in contrast to the PVC problem, an exact solution of this problem is *hard*.

The BTR operation is shown schematically in Fig. 6. Mathematically, the BTR computational problem can be defined as follows. Our input is a signal transduction network $G=(V, E)$ with a subset $E_c \subseteq E$ of edges marked as *critical*. A valid solution is a subset of edges E', with $E_c \subseteq E' \subseteq E$, that maintains the same "reachability": u has a path to v in G of nature x ($x \in \{\rightarrow, \dashv\}$) if and only if u has a path to v in $G'=(V, E')$ of the *same nature*. The goal is to *minimize* the size of E'.

The BTR problem is known to be NP-hard as a consequence of the results in (10). A few results were obtained for certain versions of BTR (11, 12) before our work in (6–9), but they were either special cases or biologically more restrictive versions. A special case of the BTR problem, called the *minimum-equivalent-digraph* problem, has been of special interest to computer scientists for a long time with regard to optimizing computer networks with connectivity requirements (13–17) and has also found applications in the context of visualization of social networks (18). Our theoretical results (6) resulted in efficient 2-approximation algorithms for BTR, which has been recently improved further to a 1.5-approximation (19).

The final product of our method led to a custom software package NET-SYNTHESIS (available at http://www.cs.uic.edu/~dasgupta/network-synthesis/) that can be simply downloaded and run in almost any machine with Microsoft Windows as the operating system (for LINUX users, source C/C++ codes for a nongraphic version of the software can be provided on request). The input to NET-SYNTHESIS is a list of relationships among biological components (single causal and double causal)

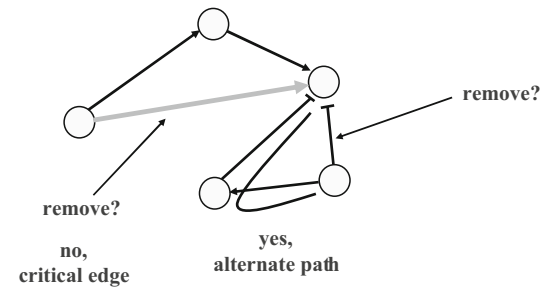


Fig. 6. *Pictorial illustration of a BTR operation.* The *lighter edge* is a critical edge and thus cannot ever be removed. The indicated inhibitory edge can be removed because there is an alternate inhibitory path from the beginning node of the edge to the end node of the edge.

and its output is a network diagram and a text file with the edges of the signal transduction network.

Below is a summary of the standard steps necessary for carrying out the network synthesis and simplification task using NET-SYNTHESIS:

1. Gather the direct interactions, single causal inferences, and double causal inferences regarding your signal transduction network.
2. Read the single inferences into NET-SYNTHESIS to form a graph. Perform BTR on the graph.
3. Integrate the double causal inferences into the graph.
4. Perform PVC.
5. Perform a follow-up round of BTR and vertex collapse until the graph cannot be reduced further.
6. If warranted, simplify the graph further by designating known vertices as pseudo-vertices and performing PVCs.

2. Materials

2.1. Information and Data Sources

Large-scale repositories such as Many Microbe Microarrays (http://m3d.bu.edu/cgi-bin/web/array/index.pl?read=aboutM3D), NASCArrays (http://affymetrix.arabidopsis.info/narrays/experimentbrowse.pl), and Gene Expression Omnibus (http://www.ncbi.nlm.nih.gov/geo/) contain expression information for thousands of genes under tens to hundreds of experimental conditions. In addition, information about differentially expressed genes responding to a combination of two experimental perturbations, e.g., the presence of a signal in normal versus mutant organisms, can be expressed as double causal inferences. Signal transduction pathway repositories such as TRANSPATH (http://www.gene-regulation.com/pub/databases.html#transpath) and protein interaction databases such as the Search Tool for the Retrieval of Interacting Proteins (http://string.embl.de/) contain up to thousands of interactions, a large number of which are not supported by direct physical evidence and thus are best treated as single causal inferences.

2.2. Software

The input to the NET-SYNTHESIS software package is a list of relationships among biological components (single causal and double causal) and its output is a network diagram and a text file with the edges of the signal transduction network. We note that "nodes" and "vertices" are used interchangeably in the software and in this chapter. In the following, we explain a few menu

options for NET-SYNTHESIS; a user manual is available at the software's webpage, http://www.cs.uic.edu/~dasgupta/network-synthesis/help.html.

2.2.1. File Menu

- *Read*. Reads an input file from your local directory. After reading, it builds a network for single causal inferences (i.e., edges) only.
- *Write*. Writes the current result to a text file in your local directory.

2.2.2. Action Menu

- *Redundant edges*. Finds out and removes if there are duplicate edges in your file or in the current graph.
- *Add pseudonodes*. Adds the double causal (i.e., three-vertex) inferences in the input file to the network via introducing pseudo-nodes if necessary.
- *Collapse pseudonodes*. Collapses pseudo-nodes using the PVC algorithm.
- *Reduction* (*slower*). Performs BTR on the current network. Recommended for networks of no more than 150 nodes.
- *Reduction* (*faster*). Performs BTR on the current network. Recommended for networks of more than 150 nodes.
- *Collapse degree-2 pseudonodes*. Collapses pseudo-nodes that have a single incoming edge and a single outgoing edge.
- *Randomize before reduction*. The transitive reduction algorithm has steps where ties are broken arbitrarily. If you turn on this action, then such tie-breaking steps will be randomized, thus potentially giving different solutions at different runs of the transitive reduction. This option may be useful if you wanted to check out more than one solution for the transitive reduction step.

2.2.3. View Menu

- *Info*. Shows basic information about the current graph such as the number of vertices and edges.
- *Edge handle*. Displays the edges more visibly (and, hopefully more nicely).
- *Show critical*. Shows critical edges with a different color.

2.2.4. Other Functions

- You can right click on a vertex on the canvas to change the name of that node. This may be especially useful in changing a real node to a pseudo-node or vice versa because the program assumes that nodes whose names start with an asterisk (*****) are pseudo-nodes.
- You can right click on the edge handle to change the nature of an edge (e.g., from excitatory to inhibitory or vice versa).

3. Methods

3.1. Gather the Direct Interactions, Single Causal Inferences, and Double Causal Inferences Regarding Your Signal Transduction Network

First, thoroughly read the relevant literature concerning the signal transduction pathway of interest. After reading all available literature on the topic, assess whether sufficient information is on hand such that network synthesis is necessary. For example, if all that is known about a system is that component/process X activates component Y which in turn activates component Z, one can draw a simple linear network and deduce that knockout of Y will eliminate signaling, but a formal analysis is hardly required.

In assessing the literature, the modeler should especially focus on experiments that provide information of the type relevant to network construction. Experiments that identify nodes belonging to a signaling pathway and the relationships between them include: (1) in vivo or in vitro experiments which show that the properties (e.g., activity or subcellular localization) of a protein change upon application of the input signal or upon modulation of components already definitively known to be associated with the input signal; (2) experiments that directly assay a small molecule or metabolite (e.g., imaging of cytosolic Ca^{2+} concentrations) and show that the concentration of that metabolite changes upon application of the input signal or modulation of its associated elements; (3) experiments that demonstrate physical interaction between two nodes, such as protein–protein interaction observed from yeast two-hybrid assays or in vitro or in vivo coimmunoprecipitation; (4) pharmacological experiments which demonstrate that the output of the pathway of interest is altered in the presence of an inhibitory agent that blocks signaling from the candidate intermediary node (e.g., a pharmacological inhibitor of an enzyme or strong buffering of an ionic species); (5) experiments which show that artificial addition of the candidate intermediary node (e.g., exogenous provision of a metabolite) alters the output of the signaling pathway; (6) experiments in which genetic knockout or overexpression of a candidate node is shown to affect the output of the signaling pathway. The first three types of experiments correspond to single causal inferences that will become edges of the network; the third also corresponds to direct interactions that will become critical edges of the network. The fourth to sixth types of experiments correspond to double causal inferences.

The experimental conclusions need to be distilled into two kinds of regulation: positive (usually described by the verbs "promotes," "activates," and "enhances") and negative (usually described by the verbs "inhibits," "reduces," and "deactivates"), and represented graphically as $\rightarrow$ and $\dashv$ (see Fig. 2). As the input to NET-SYNTHESIS is simple text files, the graphical symbols are replaced by "$\rightarrow$" and "$\dashv$." Component-to-component relationships are represented such

as "A→B." Double causal inferences are of the type "C promotes the process through which A activates B." The only way this statement can correspond to a direct interaction is if C is an enzyme catalyzing a reaction in which A is transformed into B. We represent *supported* enzyme-catalyzed reactions as both A (the substrate) and C (the enzyme) activating B (the product). If the interaction between A and B is direct and C is *not* a catalyst of the A–B interaction, we assume that C activates A. In all other cases, we represent the double causal inference such as "C→(A→B)."

Note that some choices may have to be made in distilling the relationships, especially in the case where there are two conflicting reports in the literature. For example, imagine that in one report it is stated that proteins X and Y do not physically interact based on yeast two-hybrid analysis, while in a second report, it is described that proteins X and Y do interact, based on coimmunoprecipitation from the native tissue. The modeler will need to decide which information is more reliable, and proceed accordingly. Such aspects dictate that human intervention will inevitably be an important component of the literature curation process, even as automated text search engines such as GENIES (20–22) grow in sophistication.

We will illustrate the five analysis steps following the data-gathering phase on a sample collection of single and double causal inferences. This sample is a small subset of the evidence gathered for the signal transduction network responsible for abscicic acid-induced closure of plant stomata (23). The vertices correspond to the signal, denoted "ABA," the output, denoted "Closure," and seven mediators of ABA-induced closure, the heterotrimeric G protein α subunit (GPA1), the small molecules NO and phosphatidic acid (PA), the enzymes Phospholipase C (PLC) and Phospholipase D (PLD), K^+ efflux through slowly activating outwardly rectifying K^+ channels at the plasma membrane (KOUT). The compilation includes nine single causal inferences, two of which correspond to direct interactions and two double causal inferences.

The input to NET-SYNTHESIS is given as follows:

ABA ⊣ NO
ABA → PLD
ABA → GPA1
ABA → PLC
GPA1 → PLD Y
PLD → PA
NO ⊣ KOUT
KOUT → Closure Y
PA → Closure
PLC → (ABA → KOUT)
PLD → (ABA → Closure)

The single inferences need to precede the double inferences. The direct interactions are marked by the letter "Y" following the component-to-component relationship.

3.2. Read the Single Inferences into NET-SYNTHESIS to Form a Graph. Perform BTR on the Graph

To use NET-SYNTHESIS on this example, it needs to be saved into a text file, e.g., "example.txt." After starting NET-SYNTHESIS, select the command "Read" from the File menu, and open the input file "example.txt." This will display the vertices and edges corresponding to the single inferences. You can move the nodes by clicking and holding your left mouse button on them. Try to arrange the nodes so the edges do not cross each other. Note that the small circles correspond to edge handles (if you have the option of edge handles chosen in the View menu) which can also be moved to make the graph clearer. Clicking on Info in the View menu indicates that currently the network contains eight vertices and nine edges. To perform BTR, select "Reduction (slower)" from the Action menu. This reduction method is the better choice for networks smaller than 150 vertices. A pop-up window will indicate that one edge was removed. Indeed, the edge from ABA to PLD was superfluous as it did not indicate a direct interaction and had no effect on the reachability of any node in the network.

3.3. Integrate the Double Causal Inferences into the Graph

To read in the double causal inferences, select "Add pseudonodes" from the Action menu. The pop-up window will indicate that two pseudo-vertices and six edges were added to account for the two double causal inferences. Rearrange the nodes to see what is new. Indeed, the PLD→(ABA→Closure) inference created a new pseudo-vertex, indicated by a circle with a star in it, and three new edges, one from PLD to the pseudo-vertex, one from ABA to the pseudo-node, and one from the pseudo-node to Closure. The second inference was incorporated in a similar manner. The newly added edges created new redundancies in the network. For example, the newly introduced pseudo-node connecting ABA and PLD to Closure has the same in and out reachability as the node PA, i.e., it can be reached from ABA, GPA1, and PLD and it can reach Closure. Therefore, the pseudo-vertex is a candidate for PVC.

3.4. Perform PVC

To perform PVC, select "Collapse pseudonodes" from the Action menu. The pop-up window will indicate that one pseudo-node was removed. An inspection of the network will tell you that indeed the pseudo-vertex indicated above was collapsed with the real node PA. This decreased the number of vertices by one and the number of edges by two. As an effect of the collapse, ABA is now directly connected to PA in addition to being connected by the chain GPA1–PLD. The ABA→PA edge is redundant with the path, thus it is a candidate for BTR. In addition, an edge

among the three that connect ABA, PLC, and the remaining pseudo-vertex is also redundant. Thus, we should try to simplify the network further.

3.5. Perform a Follow-up Round of BTR and Vertex Collapse Until the Graph Cannot be Reduced Further

Select "Reduction (slower)" again and you will see that indeed the two edges have been removed. The remaining pseudo-vertex is now simply a mediator between PLC and KOUT. But because its existence does not add any further information, it should be removed. You can do that by selecting "Collapse degree-2 pseudonodes" from the action menu. Now the network has eight vertices and nine edges. Select "Reduction (slower)" to make sure no more reduction is possible.

3.6. If Warranted, Simplify the Graph Further by Designating Known Vertices as Pseudo-vertices and Performing PVC

In the example above, we succeeded in integrating single and double causal inferences into a signal transduction network whose nodes are all known (i.e., they are not pseudo-nodes). For a real situation, as opposed to an illustrative example, the resulting network can be quite large and complex. In cases when some of the nodes are clearly more documented, more important, or more interesting than others, it may be beneficial to focus on the reachability among these more important nodes and disregard the others without explicitly removing them. One can do this by designating the less important nodes as pseudo-nodes and then simplifying the network by using PVC and BTR.

Let us designate the node NO as a pseudo-node. We can do this by right-clicking on the node, prepending a * to the node name that appears in a pop-up window, and press Enter. The node will now become a pseudo-node, indicated by the fact that the symbol corresponding to the node becomes a small circle with a star in the middle. Selecting "Collapse degree-2 pseudonodes" will remove the pseudo-node and connect ABA and KOUT with a positive edge. This is because a path with an even number of negative edges is positive. The new edge is redundant with the path going through PLC and "Reduction (slower)" will delete it.

4. Conclusion

We have previously successfully illustrated the usefulness of our software by applying it to synthesize an improved version of a previously published signal transduction network (7, 23) and by using it to simplify a novel network corresponding to activation-induced cell death of T cells in large granular lymphocyte leukemia (7, 24). It is our hope that this method, in assistance with interactive human intervention as discussed before, will be useful in the future in synthesizing and analyzing networks in a broader context.

References

1. B. Alberts. Molecular Biology of the Cell. Garland Publishing: New York, 1994.
2. T. I. Lee, N. J. Rinaldi et al. Transcriptional regulatory networks in *Saccharomyces cerevisiae*, Science, 298, 799–804, 2002.
3. L. Giot, J. S. Bader et al. A protein interaction map of *Drosophila melanogaster*, Science, 302, 1727–1736, 2003.
4. J. D. Han, N. Bertin et al. Evidence for dynamically organized modularity in the yeast protein–protein interaction network, Nature, 430, 88–93, 2004.
5. S. Li, C. M. Armstrong et al. A map of the interactome network of the metazoan *C. elegans*, Science, 303, 540–543, 2004.
6. R. Albert, B. DasGupta et al. Inferring (biological) signal transduction networks via transitive reductions of directed graphs, Algorithmica, 51 (2), 129–159, 2008.
7. S. Kachalo, R. Zhang et al. NET-SYNTHESIS: A software for synthesis, inference and simplification of signal transduction networks, Bioinformatics, 24 (2), 293–295, 2008.
8. R. Albert, B. DasGupta et al. A novel method for signal transduction network inference from indirect experimental evidence, Journal of Computational Biology, 14 (7), 927–949, 2007.
9. R. Albert, B. DasGupta et al. A novel method for signal transduction network inference from indirect experimental evidence, in 7th Workshop on Algorithms in Bioinformatics, R. Giancarlo and S. Hannenhalli (Eds.), LNBI 4645, Springer: Berlin/Heidelberg, 407–419, 2007.
10. A. Aho, M. R. Garey and J. D. Ullman. The transitive reduction of a directed graph, SIAM Journal of Computing, 1 (2), 131–137, 1972.
11. A. Wagner. Estimating coarse gene network structure from large-scale gene perturbation data, Genome Research, 12, 309–315, 2002.
12. T. Chen, V. Filkov and S. Skiena, Identifying gene regulatory networks from experimental data, in 3rd Annual International Conference on Computational Molecular Biology, 94–103, 1999.
13. S. Khuller, B. Raghavachari and N. Young. Approximating the minimum equivalent digraph, SIAM Journal of Computing, 24 (4), 859–872, 1995.
14. S. Khuller, B. Raghavachari and N. Young. On strongly connected digraphs with bounded cycle length, Discrete Applied Mathematics, 69 (3), 281–289, 1996.
15. S. Khuller, B. Raghavachari and A. Zhu. A uniform framework for approximating weighted connectivity problems, in 19th Annual ACM-SIAM Symposium on Discrete Algorithms, 937–938, 1999.
16. G. N. Frederickson and J. JàJà. Approximation algorithms for several graph augmentation problems, SIAM Journal of Computing, 10 (2), 270–283, 1981.
17. A. Vetta. Approximating the minimum strongly connected subgraph via a matching lower bound, in 12th ACM-SIAM Symposium on Discrete Algorithms, 417–426, 2001.
18. V. Dubois and C. Bothorel. Transitive reduction for social network analysis and visualization, in IEEE/WIC/ACM International Conference on Web Intelligence, 128–131, 2008.
19. P. Berman, B. DasGupta and M. Karpinski. Approximating Transitivity in Directed Networks, arXiv:0809.0188v1 (available online at http://arxiv.org/abs/0809.0188v1).
20. C. Friedman, P. Kra, H. Yu, M. Krauthammer and A. Rzhetsky. GENIES: a natural-language processing system for the extraction of molecular pathways from journal articles, Bioinformatics, 17 (Suppl 1), S74–S82, 2001.
21. E. M. Marcotte, I. Xenarios and D. Eisenberg. Mining literature for protein-protein interactions. Bioinformatics, 17 (4), 359–363, 2001.
22. L. J. Jensen, J. Saric and P. Bork. Literature mining for the biologist: from information retrieval to biological discovery, Nature Reviews Genetics, 7 (2), 119–129, 2006.
23. S. Li, S. M. Assmann and R. Albert. Predicting essential components of signal transduction networks: a dynamic model of guard cell abscisic acid signaling, PLoS Biology, 4 (10), e312, 2006.
24. R. Zhang, M. V. Shah, J. Yang et al. Network model of survival signaling in large granular lymphocyte leukemia. Proceedings of the National Academy of Sciences of the United States of America, 105 (42), 16308–16313, 2008.

Chapter 17

Reverse Engineering Gene Regulatory Networks Related to Quorum Sensing in the Plant Pathogen *Pectobacterium atrosepticum*

Kuang Lin, Dirk Husmeier, Frank Dondelinger, Claus D. Mayer, Hui Liu, Leighton Prichard, George P.C. Salmond, Ian K. Toth, and Paul R.J. Birch

Abstract

The objective of the project reported in the present chapter was the reverse engineering of gene regulatory networks related to quorum sensing in the plant pathogen *Pectobacterium atrosepticum* from microarray gene expression profiles, obtained from the wild-type and eight knockout strains. To this end, we have applied various recent methods from multivariate statistics and machine learning: graphical Gaussian models, sparse Bayesian regression, LASSO (least absolute shrinkage and selection operator), Bayesian networks, and nested effects models. We have investigated the degree of similarity between the predictions obtained with the different approaches, and we have assessed the consistency of the reconstructed networks in terms of global topological network properties, based on the node degree distribution. The chapter concludes with a biological evaluation of the predicted network structures.

Key words: *Pectobacterium atrosepticum*, Quorum sensing, Transposon mutagenesis, Microarrays, Graphical Gaussian models, Sparse Bayesian regression, LASSO, Bayesian networks, Nested effects models, Degree distribution, Power law, Gene ontologies

1. Introduction

Pectobacterium atrosepticum (Pba), which is a plant pathogen on potato in temperate regions, synthesizes and secretes large quantities of plant cell wall degrading enzymes that are responsible for the soft rot disease phenotype, earning it the epithet "brute force" pathogen. In particular, the "brute force" attack utilizes a population density-dependent regulatory mechanism called quorum sensing (QS), which controls a wide range of phenotypes in many

David Fenyö (ed.), *Computational Biology*, Methods in Molecular Biology, vol. 673,
DOI 10.1007/978-1-60761-842-3_17, © Springer Science+Business Media, LLC 2010

different bacteria. Utilizing the production and secretion of certain signalling molecules, QS serves as a communication network that allows bacteria to coordinate their activities based on the local density of their population. A recent study by Liu *et al.* (1) provides the first evidence that Pba uses QS to target host defences simultaneously with a physical attack on the plant cell wall. Moreover, Liu *et al.* (1) demonstrate that a wide range of previously known and unknown virulence regulators lie within the QS regulon, revealing it to be the master regulator of virulence. The objective of the present study is to shed further light on the QS regulatory mechanism by applying current methods from multivariate statistics and machine learning to reconstruct putative gene regulatory networks from gene expression profiles obtained from wild-type and various knockout strains.

2. Material

2.1. Gene Knockout via Transposon Mutagenesis

Mutated bacterial strains were generated via transposon mutagenesis. Transposons are relatively short pieces of mobile DNA that can insert into pieces of DNA within a genome. Transposon mutagenesis is a process that allows transposons to be transferred to a host organism's chromosome. This is accomplished by way of a plasmid from which a transposon is extracted and inserted into the host chromosome. The insertion can result in the interruption or modification of the function of an extant gene on the chromosome, effectively creating a mutant knockout strain. In the present study, nine mutant Pba strains were generated, where the following genes were knocked out: *expM, hor, hrpL, expI, expR, aepA, virR*, and *virS*. Additionally, a double mutation event was induced, where both *virR* and *expI* were knocked out. For further details and an exact specification of the experimental protocol, see ref. (1).

2.2. Genome-Wide Transcriptomic Profiling with Miroarrays

Wild-type and mutant Pba strains were grown in a nutrient broth to stationary phase, and then used to inoculate sterilized potato tubers. At 12 h postinoculation, the bacterial cells were isolated from the tuber by scraping infected tissue into sterilised water. RNA was isolated by following the protocol described in Liu *et al.* (1), then reverse transcribed and cDNA labelled. 60-Mer oligonucleotide probes were designed to Pba-coding sequences and used, together with controls, to generate 11K custom arrays with 99.5% genome coverage (Agilent, Inc., Santa Clara, CA, USA). All microarray experiments were carried out in triplicate, for each of the eight single Pba knockout mutants in *expM, hor, hrpL, expI, expR, aepA, virR*, and *virS*, and the double knockout mutant in *virR/expI*, to obtain relative gene expression levels with respect to Pba wild-type.

2.3. Preprocessing of Gene Expression Profiles

All microarray images were visually assessed for quality prior to feature extraction, whereby standard probe quality control standards were applied[1]. Features flagged as poor were removed. Box plots and principal components analysis of whole data sets were used to assess array to array variation. Any outlying microarrays were repeated as necessary. The microarray data were preprocessed using GeneSpring[2] software (version 7.2) and normalized using the Lowess algorithm (Agilent Technologies Inc.). This nonparametric normalization technique first fits a nonlinear curve to the plot of the log-ratios of the two dye intensities *Cy*5 and *Cy*3, $M = \log_2(Cy5/Cy3)$, versus the average log-ratio $A = \log_2(Cy3 * Cy5)/2$. It then uses the residuals of the fit as normalized log-ratio values. This method was first suggested by Yang et al. (2) and has become the standard method of normalizing two-colour microarrays.[3]

2.4. Assessing Differential Gene Expression

To assess differential gene expression in Pba knockout strains with respect to Pba wildtype, we computed *p*-values with the empirical Bayes method proposed in Smyth (3). Recall that all knockout experiments were carried out in triplicate. The three resulting log-ratio values for each mutant versus wild-type comparison were tested against 0 by using the moderated *t*-test available in the Bioconductor library LIMMA (4). This test differs from a standard *t*-test by using a standard error estimate that is obtained from an empirical Bayesian analysis. The individual estimate for a single gene is shrunk towards the average estimate for all genes, which stabilizes the analysis particularly for small sample sizes, as in our example. This method was first introduced by Lönnstedt and Speed (5), later generalized and implemented in Bioconductor[4] by Smyth (3), and it is one of the most widely used tools for detecting differential gene expression. As a result of this analysis, we obtain a *p*-value for each gene, which indicates whether the corresponding average log-ratio between mutant and wild-type is significantly different from 0.

2.5. Clustering of Gene Expression Profiles

It would be difficult to visualize and interpret regulatory networks involving several thousand genes. Moreover, there would be considerable inference uncertainty, as the likelihood in network space would be diffuse, with many different networks having very similar scores. We therefore resorted to a clustering approach as a preliminary

[1] See further information in ArrayExpress-http://www.ebi.ac.uk/microarray-as/aer/.

[2] http://www.chem.agilent.com/en-US/Products/software/lifesciences-informatics/genespringgx/Pages/default.aspx.

[3] Note that in contrast to most home-spotted cDNA microarrays, Agilent arrays are not printed by different print tips and thus are not subdivided into separate subblocks within the array. For this reason, it was not necessary to use the print-tip lowess algorithm, which applies the same curve fitting technique separately to each subblock on the array.

[4] http://www.bioconductor.org/.

complexity reduction step, and then inferred regulatory interactions among about 100 inferred clusters and nine key target regulatory genes. In order to infer biologically plausible clusters, we combined the outputs from several clustering algorithms based on a biological scoring scheme using gene ontologies. The basic idea is depicted in the flow chart of Fig. 1. We applied five clustering algorithms: K-means and hierarchical agglomerative average linkage clustering (6), using both Euclidean and correlation distances, as well as mixtures of factor analyzers inferred with variational Bayesian Expectation Maximization (7). We assessed the biological plausibility of the inferred clusters by testing for significantly enriched GO (Gene Ontology) terms. We collected GO annotations for both Pba genes and close homologues from the EBI[5] and ASAP[6] databases. In total, 18,996 GO terms were assigned to the 3,616 genes

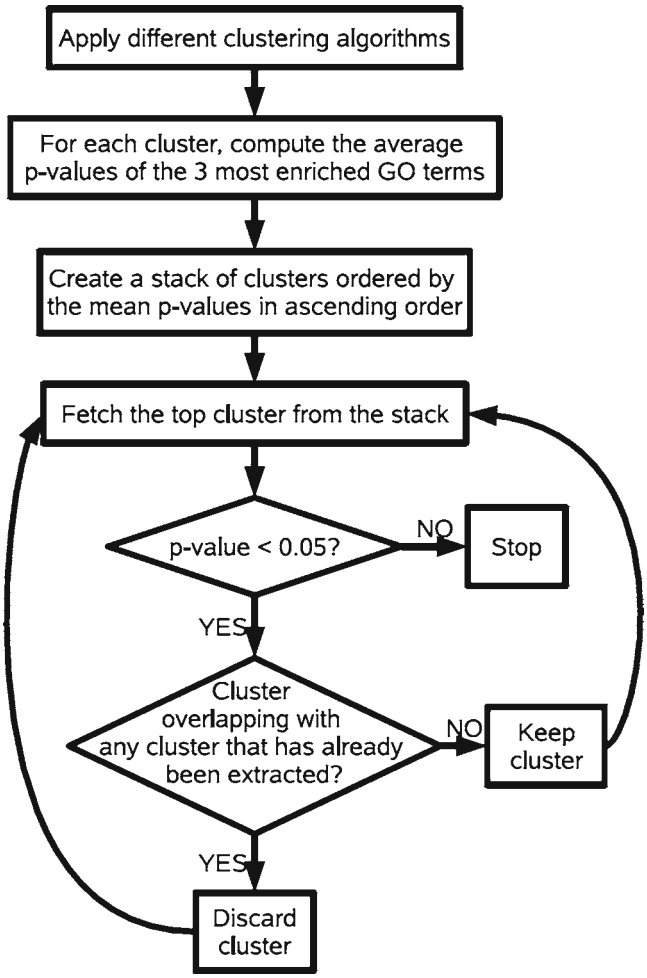


Fig. 1. Flow chart of the algorithm used for clustering gene expression profiles.

[5] http://www.ebi.ac.uk/GOA/proteomes.html.
[6] https://asap.ahabs.wisc.edu/asap/logon.php.

in our set. We computed the significance of GO term enrichment in the clusters using the program Ontologizer[7] with the default options. We applied a standard 5% threshold cutoff on the Bonferroni-corrected *p*-values, and considered a cluster to be biologically plausible when the *p*-value of a GO term enrichment was smaller than 5%. We then combined clusters from different clustering methods by the application of the algorithm depicted in Fig. 1, resulting in 110 clusters. The composition of these clusters, as well as further details of the clustering scheme, can be obtained from the supplementary material[8].

3. Methods

A simple method for reconstructing gene regulatory networks, proposed by Butte and Kohane (8) and termed "relevance networks," is based on the following procedure. First, compute all pairwise similarity scores between gene expression profiles. Standard measures of similarity that are commonly used are the Pearson correlation or the mutual information. Next, apply a randomization test to test for significant deviation from zero. Finally, connect all nodes by edges whose pairwise similarity scores are significantly greater than zero. The method is computationally cheap and easy to apply. However, its main disadvantage is that it is intrinsically impossible to distinguish between direct and indirect interactions. If two genes are regulated by a set of common regulators, their gene expression profiles tend to be similar. The relevance network approach will therefore tend to infer an edge between these genes even if there is no direct interaction between them. For instance, in the scenario depicted in the left panel of Fig. 2, where a set of m genes $x_1, x_2, \ldots, x_m$ are regulated by the

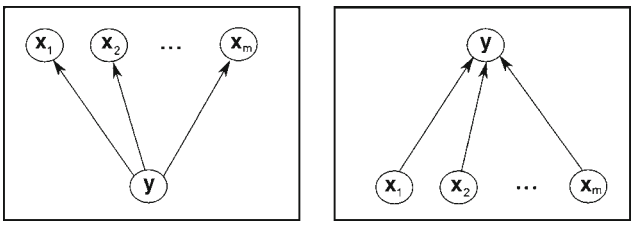


Fig. 2. Schematic of the approach of partial correlation (*left*) and sparse regression (*right*). *Left*: Conditional on $\mathbf{y}$, the gene expression profiles $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m$ are independent, and the partial correlation coefficients will be small. *Right*: The approach of sparse regression aims to find a minimal set of predictors $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m$ to explain gene expression profile $\mathbf{y}$.

[7] http://www.charite.de/ch/medgen/ontologizer/commandline/Ontologizer.jar.
[8] http://www.bioss.ac.uk.testweb.bioss.sari.ac.uk/staff/dirk/Supplements/FF842/.

common regulator y, the approach of relevance networks is prone to inferring spurious edges between the genes x_1, x_2, …, x_m; see Werhli *et al.* (9) for an empirical corroboration.

In the following subsections, we will briefly review various more sophisticated methods that aim to distinguish direct interactions from indirect ones, and which also give some indication about the putative direction of the regulatory interactions. We first assume that we have complete observability, i.e. that the gene expression profiles provide a good indication of the corresponding protein activities. We review four approaches aiming to infer gene regulatory networks from expression profiles: Graphical Gaussian models, LASSO, sparse Bayesian regression, and Bayesian networks. We conclude this section with a review of nested effects models, which aim to infer interactions among regulatory genes that are themselves subject to post-transcriptional modification. This approach allows regulatory gene interactions to be inferred when the data are incomplete, i.e., when the relevant changes at the protein level are not indicated by changes at the gene expression level.

3.1. Graphical Gaussian Models

Graphical Gaussian models (GGMs) are undirected probabilistic graphical models that allow the identification of conditional independence relations among the nodes under the assumption of a multivariate Gaussian distribution of the data. The inference of GGMs is based on a (stable) estimation of the covariance matrix of this distribution. The element C_{ik} of the covariance matrix C is proportional to the correlation coefficient between nodes X_i and X_k. A high correlation coefficient between two nodes may indicate a direct interaction, an indirect interaction, or a joint regulation by a common (possibly unknown) factor. However, only the direct interactions are of interest to the construction of a regulatory network. The strengths of these direct interactions are measured by the partial correlation coefficient ρ_{ik}, which describes the correlation between nodes X_i and X_k conditional on all the other nodes in the network. From the theory of normal distributions, it is known that the matrix of partial correlation coefficients ρ_{ik} is related to the inverse of the covariance matrix $\mathbf{C}$, $\mathbf{C}^{-1}$ (with elements C_{ik}^{-1} (10).

$$\rho_{ik} = -\frac{C_{ik}^{-1}}{\sqrt{C_{ii}^{-1} C_{kk}^{-1}}}. \qquad (1)$$

To infer a GGM, one typically employs the following procedure. From the given data, the empirical covariance matrix is computed, inverted, and the partial correlations ρ_{ik} are computed from (1). The distribution of $|\rho_{ik}|$ is inspected, and edges (i, k) corresponding to significantly small values of $|\rho_{ik}|$ are removed from the

graph. The critical step in the application of this procedure is the stable estimation of the covariance matrix and its inverse. Note that the covariance matrix is only nonsingular if the number of observations exceeds the number of nodes in the network. This condition is not satisfied for many real applications in systems biology. In order to learn a GGM from a data set in such a scenario, Schäfer and Strimmer (11) explored various stabilization methods, based on the Moore–Penrose pseudo inverse and bagging. In the present work, we apply an alternative regularization approach based on shrinkage, which Schäfer and Strimmer (11) found to be superior to their earlier schemes. The idea is to add a weighted nonsingular regularization matrix, e.g., the unity matrix, to the covariance matrix so as to guarantee its nonsingularity. The optimal weight parameter is estimated based on the Ledoit Wolf lemma from statistical decision theory so as to minimize the expected deviation of the regularized covariance matrix from the (unknown) true covariance matrix. The method of GGMs, which are undirected graphs, can be extended to infer putative directions of causal interactions, as proposed in Opgen-Rhein and Strimmer (12). This scheme is based on the computation of the standardized partial variance, which is the proportion of the variance that remains if the influence of all other variables is taken into account. All significant edges in the GGM network are directed in such a fashion that the direction of the arrow points from the node with the larger standardized partial variance (the more *exogeneous node*) to the node with the smaller standardized partial variance (the more *endogeneous* node), provided the ratio of the two partial variances is significantly different from 1. For further details, see ref. (12).

3.2. Sparse Regression and the LASSO

The approach discussed in the previous subsection aims to predict interactions between genes based on the partial correlations between their expression profiles. In the present subsection, we review an alternative paradigm, which pursues a regression approach: given the gene expression profile $\mathbf{y}_g$ of some target gene g, we aim to find a set of regulator genes $\{r\}$ whose gene expression profiles $\{\mathbf{x}_r\}$ are good predictors of gene expression profile $\mathbf{y}_g$:

$$\hat{\mathbf{y}}_g = \sum_r w_{gr} \mathbf{x}_r, \tag{2}$$

where $\hat{\mathbf{y}}_g$ is a predictor of $\mathbf{y}_g$, and the regression parameters w_{gr} represent interaction strengths between the target gene g and the putative regulator genes r. The different concepts are illustrated in Fig. 2. We denote the vector of interaction strengths as $\mathbf{w}_g$, which has w_{gr} as its rth component. The mismatch between the predicted and measured expression profile of target gene g is typically measured by the L2 norm

$$E(\mathbf{w}_g) = \left\| \mathbf{y}_g - \hat{\mathbf{y}}_g(\mathbf{w}_g) \right\|^2. \quad (3)$$

Obtaining the optimal interaction parameters $\hat{\mathbf{w}}_g$ by minimizing $E(\mathbf{w}_g)$ corresponds to a maximum likelihood estimator under the assumption of isotropic Gaussian noise. In practice, this approach is usually susceptible to overfitting, which calls for the application of some regularization scheme. The standard method of ridge regression is given by

$$\hat{\mathbf{w}}_g = \arg\min_{\mathbf{w}_g} \left(E(\mathbf{w}_g) + \lambda \sum_r w_{gr}^2 \right). \quad (4)$$

This can be interpreted in three different ways: (1) maximizing the penalized likelihood with an L2-norm penalty term and regularization parameter λ; (2) constrained maximization of the likelihood under the L2-norm constraint $\sum_r w_{gr}^2 < C$, where λ is a Lagrange parameter; (3) Bayesian *maximum a posteriori* estimate under a zero-mean Gaussian prior on $\mathbf{w}_g$ with diagonal isotropic covariance matrix $\lambda^{-1}\mathbf{I}$: $P(\mathbf{w}_g) = \mathcal{N}(0, \lambda^{-1}\mathbf{I})$. A disadvantage of ridge regression is that the set of interaction parameters $\{w_{gr}\}$ does usually not tend to be sparse. This is a consequence of the fact that the derivative of the regularization term with respect to w_{gr} approaches zero as $w_{gr} \to 0$. Consequently, there is no "force" pulling the parameters to zero when they are small. According to our current knowledge, gene regulatory networks are usually sparse, and a stronger regularization term is therefore desirable. This is can be effected with an L1-norm instead of the L2-norm regularization term:

$$\hat{\mathbf{w}}_g = \arg\min_{\mathbf{w}_g} \left(E(\mathbf{w}_g) + \lambda \sum_r |w_{gr}| \right), \quad (5)$$

which can be interpreted as a Bayesian *maximum a posteriori* estimate under a Laplacian prior on $\mathbf{w}_g$, as first proposed by Williams (13). The derivative of the regularization term with respect to the parameters is now constant, which provides a stronger "force" driving small parameters to zero. The discontinuity of the derivative at $w_{gr} \to 0$ can be exploited to implement an effective pruning scheme for discarding interactions, as discussed in Williams (13). The L1-norm regularization term was introduced to the statistics community by Tibshirani (14), where it was termed the LASSO (least absolute shrinkage and selection operator). One of the first applications to the reconstruction of gene regulatory networks is reported by van Someren et al. (15). Grandvalet and Canu (16) showed that the LASSO estimate of the interaction strengths is equivalent to ridge regression with r-dependent regularization hyperparameters

$$\widehat{\mathbf{w}}_g = \arg\min_{\mathbf{w}_g}\left(E(\mathbf{w}_g) + \sum_r \lambda_r w_{gr}^2 \right) \tag{6}$$

subject to the constraint $\sum_{r=1}^{R} 1/\lambda_r = R/\lambda$, for some predefined constant λ. The regulatory network between the target gene g and the regulatory genes $\{r\}$ is defined by the set of interactions with non-zero interaction strengths w_{gr}. The degree of sparsity is determined by the regularization hyperparameter λ, with larger values of λ resulting in sparser networks. The question, then, is how to set λ. Williams (13) suggested integrating λ out; this approach has been subject to some controversy, though (17). A standard non-Bayesian approach is to estimate λ with k-fold cross-validation. This is the approach that was implemented in the software we applied in the present study, with $k = 10$; see Table 1. An alternative Bayesian approach would be to estimate λ by maximizing the evidence, as discussed in the next subsection.

Note that the generalization of the sparse regression approach to more target genes g is straightforward: $E(\mathbf{w}_g)$ in Eq. (3) just needs to be replaced by

$$E(\mathbf{W}) = \sum_g \left\| \mathbf{y}_g - \hat{\mathbf{y}}_g(\mathbf{w}_g) \right\|^2, \tag{7}$$

where $\mathbf{W}$ is a matrix with column vectors $\mathbf{w}_g$. If there is no clear separation between the set of target and regulatory genes, the effect of gene g needs to be excluded when forming the predictor $\hat{\mathbf{y}}_g(\mathbf{w}_g)$. Again, this requirement is straightforward to implement. To avoid notational opacity, we have not described this approach in its full generality, though.

Table 1
Software packages used for the application of the network reconstruction methods described in Subheading 3.

Method	Software	Web address
GGM	GeneNet	http://strimmerlab.org/software/genenet/
LASSO	Lars	http://www-stat.stanford.edu/~hastie/Papers/LARS/
SBR	SparseBayes	http://www.miketipping.com/index.php?page=rvm
BNet	BNlearn	http://crantastic.org/packages/bnlearn
NEM	Nem	http://www.bioconductor.org/packages/2.3/bioc/html/nem.html

All software packages are freely available from the specified web addresses. All programs are written in R, except for SparseBayes, which is written in Matlab

3.3. Sparse Bayesian Regression

As mentioned in the previous subsection, the minimization of $E(\mathbf{w}_g)$ in Eq. (3) corresponds to maximizing the likelihood $P(\mathbf{D}|\mathbf{w}_g)$ under the assumption of isotropic Gaussian noise, where $\mathbf{D} = \{\mathbf{y}_g, \{\boldsymbol{x}_r\}\}$ is used to denote the data. The estimates $\hat{\mathbf{w}}_g$ in Eqs. (4) and (6) are equivalent to the *maximum a posteriori* estimates

$$\hat{\mathbf{w}}_g = \arg\max_{\mathbf{w}_g} P(\mathbf{w}_g \mid \mathbf{D}, \lambda) = \arg\max_{\mathbf{w}_g} \left[\log P(\mathbf{D} \mid \mathbf{w}_g) + \log P(\mathbf{w}_g \mid \lambda) \right] \quad (8)$$

under the assumption of an isotropic Gaussian or Laplacian prior $P(\mathbf{w}_g|\lambda)$ on the interaction strengths $\mathbf{w}_g$. Within the Bayesian framework, the hyperparameter λ is optimized by maximizing the marginal likelihood or evidence

$$P(\boldsymbol{D}|\lambda) = \int P(\mathbf{D}|\mathbf{w}, \lambda) P(\boldsymbol{w}|\lambda) \mathrm{d}\lambda \quad (9)$$

as discussed by MacKay (18). In the present study, we applied the "sparse Bayesian regression" (SBR) approach of Rogers and Girolami (19), which is based on the work of Tipping and Faul (20). Here, the prior on the interaction parameters is chosen to be a product of zero-mean Gaussian distributions

$$P(\mathbf{w}_g|\lambda) = \prod_r \mathcal{N}(w_{gr}|0, \lambda_r^{-1}) \quad (10)$$

with separate hyperparameters for the regulatory genes r. This scheme is similar to Eq. (6), except that the constraint $\sum_{r=1}^{R} 1/\lambda_r = R/\lambda$ is missing. The hyperparameters λ_r are optimized with the evidence scheme described above[9] Tipping and Faul (20) showed that the marginal likelihood can be decomposed into separate contributions from the individual regulatory genes $\{r\}$. This leads to a fast, iterative maximization algorithm not only for the hyperparameters λ_r, but also the network structure: interactions between the target gene g and the putative regulatory genes $\{r\}$ are progressively added and removed until a local maximum of the marginal likelihood is reached. Specific details of the algorithm can be found in Tipping and Faul (20).

3.4. Bayesian Networks

Bayesian networks (BNets) have received substantial attention from the computational biology community as models of gene regulatory networks, following up on pioneering work by Friedman et al. (21) and Hartemink et al. (22). Several tutorials on Bayesian networks have been published (23–25). We therefore only qualitatively recapitulate some aspects that are of relevance to the present study, and refer the reader to the above tutorials for a thorough and more rigorous introduction.

[9] In statistics, this is called a type-II maximum likelihood estimation.

The structure $\mathcal{H}$ of a Bayesian network is defined by a directed acyclic graph (DAG) indicating how different variables of interest, represented by nodes and connected by directed edges, "interact." The edges of a Bayesian network are associated with conditional probabilities, defined by a functional family and their parameters. The interacting entities are associated with random variables, which represent some measured quantities of interest, like relative gene expression levels or protein concentrations. We are interested in learning a network of causal relations between interacting nodes. While such a causal network forms a valid Bayesian network, the inverse relation does not always hold: when we have learned a Bayesian network from the data, the resulting graph does not necessarily represent the correct causal graph. One reason for this discrepancy is the existence of unobserved nodes. When we find a probabilistic dependence between two nodes, we cannot necessarily conclude that there exists a causal interaction between them, as this dependence could have been brought about by a common yet unobserved regulator. Even under the assumption of complete observation the inference of causal interaction networks is impeded by symmetries within so-called equivalence classes, which consist of networks that define the same conditional independence relations. As such, each Bayesian network represents a whole equivalence class, represented by a complete partially directed acyclic graph (CPDAG). Under the assumption of complete observation, directed edges in a CPDAG can be taken as indications of putative causal interactions.

We denote the set of all measurements of all random variables as the data, represented by the letter **D**. As a consequence of the acyclicity of the network structure, the joint probability of all the random variables can be factorized into a product of lower-complexity conditional probabilities according to conditional independence relations defined by the graph structure $\mathcal{H}$. Under certain regularity conditions, the parameters associated with these conditional probabilities can be integrated out analytically. This allows us to compute the marginal likelihood $P(\boldsymbol{D} \mid \mathcal{H})$, which captures how well the network structure $\mathcal{H}$ explains the data **D**. In the present study, we compute $P(\boldsymbol{D} \mid \mathcal{H})$ under the assumption of a linear Gaussian distribution. The resulting score was derived by Geiger and Heckerman (26) and is referred to as the BGe score.

The objective of inference is to find the DAG (or CPDAG) that is most supported by the data. Mathematically, this is the mode of the posterior distribution

$$P(\mathcal{H} \mid \mathbf{D}) \propto P(\mathbf{D} \mid \mathcal{H}) P(\mathcal{H}), \tag{11}$$

where $P(\mathcal{H})$ is the prior distribution over network structures, which represents the biological knowledge that we might have prior to measuring the data **D**. Since the number of structures $\mathcal{H}$ increases super-exponentially with the number of nodes, an

exhaustive search for the mode of $P(\mathcal{H}|\boldsymbol{D})$ is usually intractable, and some greedy search procedure based on hill climbing is usually pursued: the network structure is locally modified, and the modification is accepted if the score $P(\mathcal{H}|\boldsymbol{D})$ increases. This procedure is iterated until some convergence criterion is satisfied.

Note that in systems biology, where we aim to learn complex interaction patterns involving many components, the amount of information from the data and the prior is usually not sufficient to render the distribution $P(\mathcal{H}|\boldsymbol{D})$ sharply peaked at a single graph. Instead, the distribution is usually diffusely spread over a larger set of networks. Summarizing this distribution by a single network is therefore usually not appropriate. A more sophisticated procedure is to sample network structures $\mathcal{H}$ from the posterior distribution $P(\mathcal{H}|\boldsymbol{D})$ with MCMC, as pursued, e.g., (27–29). As a heuristic simplification of this approach, a hill climbing optimization scheme can be run repeatedly on bootstrap replicated data, as pursued in Friedman *et al.* (21), and carried out in the present work; *see* Subheading 4 for further details.

3.5. Nested Effects Models

3.5.1. Model Overview

Many approaches towards reverse engineering gene regulatory networks are based on analyzing expression levels of the regulators and comparing them to those of the genes they regulate. This is a reasonable method if there is reason to believe that highly expressed regulators have more influence on the genes they regulate. However, this may not always be the case. For example, some regulatory genes may be posttranscriptionally modified, with the consequence that the amount of mRNA present in the cell is not a good indicator of the corresponding protein activity.

Nested effects models (NEMs) (*30*) are one approach to dealing with this problem. Rather than looking at the expression levels of regulating genes (called *S-genes* for *signalling genes*), NEMs look at the effect that knocking out each of these genes

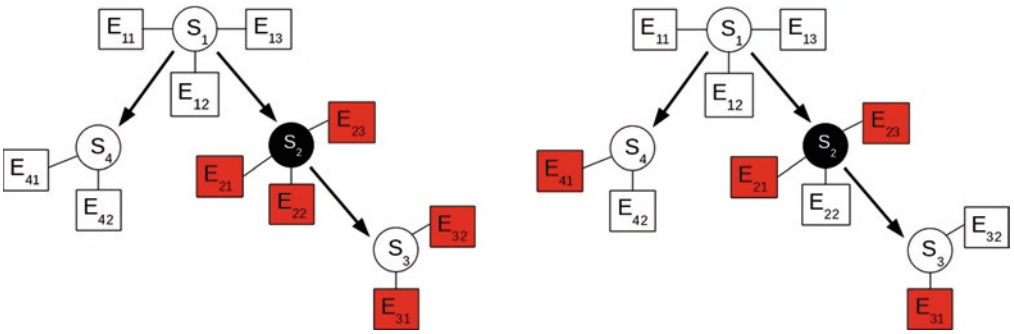


Fig. 3. Illustration of nested effects models (NEMs). *Circles* represent *S*-genes, *boxes* represent E-genes. *Left panel:* The black circle represents an S-gene that has been knocked out. The *shaded boxes* represent E-genes where significant effects are expected. *Right panel:* Due to noise in the data, some true effects are missed (here: E_{22} and E_{32}), and some spurious effects are observed (here: E_{41}).

has on the expression levels of the genes that they regulate (called *E-genes* for *effect reporting genes*). Based on these effects, it is possible to determine the structure of the signalling pathway that the S-genes are a part of. For example, if gene S_1 regulates E_1 and S_2 regulates E_2, and additionally S_2 is downstream from S_1, then we would expect to see an effect on E_1 and E_2 if we knock out S_1, but only an effect on E_2 if we knock out S_2. Note that for real world applications, the situation is more difficult than described in this simple example. This is because gene regulation is a stochastic process and measurements are susceptible to noise. An illustration is given in Fig. 3.

To complete the specification of an NEM, we need two sets of parameters: A network hypothesis $\mathcal{H}$, which describes the relations between the S-genes, and a model Θ for the regulation of the E-genes, where $\theta_i = j$ if E-gene i is regulated by S-gene j. We assume that an E-gene can only be regulated by one S-gene and use model averaging to account for all possibilities. Using Bayes' theorem, the score for a network hypothesis given data $\mathbf{D}$ is:

$$P(\mathcal{H} \mid \mathbf{D}) = \frac{P(\mathbf{D} \mid \mathcal{H})P(\mathcal{H})}{P(\mathbf{D})}. \tag{12}$$

If we assume that the observations of each E-gene, the parameters θ_i and the knockout experiments are independent, then the likelihood $P(\boldsymbol{D} \mid \mathcal{H})$ for a data set consisting of m E-genes and n S-genes decomposes as:

$$P(\mathbf{D}|\mathcal{H}) = \prod_{i=1}^{m} \sum_{j=1}^{n} \prod_{k=1}^{n} P(\boldsymbol{D}_{ik} \mid \mathcal{H}, \theta_i = j) P(\theta_i = j \mid \mathcal{H}), \tag{13}$$

where $P(\boldsymbol{D}_{ik} \mid \mathcal{H}, \theta_i = j)$ is the likelihood of the effect observed at E-gene i when knocking out S-gene k and $P(\theta_i = j \mid \mathcal{H})$ is the prior probability of E-gene i being regulated by S-gene j. Note that we usually do not know which E-genes are controlled by which S-genes. For this reason, Eq. (13) includes a marginalization over all possible assignments of E-genes to S-genes. More details can be found in refs. (30, 31).

3.5.2. Modelling the Effects

In order to find the likelihood of observing an effect at E-gene i when knocking out S-gene k, Markowetz *et al.* (30, 32) first used a discretization scheme based on thresholding to transform the continuous expression values of the E-genes into binary indicators. Then they calculated the likelihood based on the expected false-positive and false-negative rates. This approach incurs an inevitable loss of information, and also requires both positive and negative controls to estimate the error rates, which may not always be available. Fröhlich *et al.* (31) developed an alternative method which uses p-values that correspond to the likelihood of an E-gene

i being differentially expressed when S-gene k is knocked out. They obtain the raw p-value using LIMMA (3), as described in Subheading 2.4, and fit a three-component Beta-uniform mixture (BUM) model to those values. The BUM model consists of a uniform distribution (reflecting the null hypothesis) and two Beta distributions such that:

$$P(D_{ik}) = \pi_{1k} + \pi_{2k} Beta(D_{ik}, \alpha_k, 1) + \pi_{3k} Beta(D_{ik}, 1, \beta_k), \quad (14)$$

where D_{ik} is the p-value of E_i at knockout S_k, the π_{*k} are the mixing coefficients and we have the constraints that $\alpha_k < 1$ and $\beta_k > 2$. If $\hat{\pi} = P(D_{ik} = 1)$ is the maximum uniform part of the model, then we have:

$$P(D_{ik} \mid \mathcal{H}, \theta_i) = \begin{cases} \dfrac{P(D_{ik}) - \hat{\pi}}{1 - \hat{\pi}} & \text{if } \mathcal{H} \text{ predicts an effect} \\ 1 & \text{otherwise.} \end{cases} \quad (15)$$

3.5.3. A Priori Filtering of Effects

A typical microarray experiment can measure the expression levels of thousands of genes, not all of which will be affected by the knockout of an S-gene. For that reason, it makes sense to apply an a priori filtering step to remove E-genes that only show random effects. Fröhlich *et al.* (31) use a scheme that finds patterns of differentially expressed genes that are statistically significant. Given the multiple-testing corrected p-value p_k of an E-gene expression level in experiment k, and a false-positive rate α, we can set:

$$b_k = \begin{cases} 1 & \text{if } p_k < \alpha \\ 0 & \text{otherwise.} \end{cases} \quad (16)$$

If s_k is the number of significant genes in experiment k, then the probability of observing a pattern $\mathbf{b} = (b_1, \ldots, b_n)$ under the null hypothesis H_0 is:

$$P(\mathbf{b} \mid H_0) = \prod_{k=1}^{n} \left(b_k \alpha \frac{s_k}{M} + (1 - b_k)(1 - \alpha) \frac{M - s_k}{M} \right), \quad (17)$$

where M is the total number of E-genes. This allows us to calculate the number of times that we should expect to see **b** by chance. Using a binomial test, we can calculate the statistical significance of seeing **b** more often than expected, and keep only those effects which show a significant pattern.

3.5.4. Network Inference Methods

We now know how to calculate the likelihood for a given network hypothesis. Unfortunately, unless the number of S-genes is very small, it is impractical to score all possible network structures. To circumvent this problem, Markowetz et al. (32) developed a method based on scoring networks consisting of triples and combining

them. Two alternative methods are greedy hill climbing (33) and a module approach based on hierarchical clustering (31).

In the triples approach, we consider all possible triples of S-genes and score the networks that can be formed using only three nodes. Then we select the highest-scoring network for each triple and use model averaging to combine them into a complete network. We calculate the frequency of each edge and include all the edges whose frequency exceeds a certain threshold.

Greedy hill climbing is a more basic approach where we start from a network (usually with no edges) and at each step add the edge that gives the biggest improvement to the score. If no more improvements are possible, the algorithm terminates. This only gives us a local optimum, so it is usually advisable to use bootstrapping (repeat the greedy hill climbing algorithm several times, each time sampling with replacement from the E-genes) to get a measure of the confidence we have in each edge.

The module networks method starts out by creating a hierarchical clustering of the gene expression profiles using a standard clustering method (Frohlich *et al.* (33) suggest average linkage). Then, starting from the top, we look for clusters containing at most four S-genes. When the network has been decomposed into non-overlapping clusters (or modules) of at most four S-genes, we find the highest-scoring network for each cluster using an exhaustive search. Finally, the modules are connected using a constrained greedy hill climbing approach, which only adds edges between S-genes in different modules.

A feature that NEMs have in common with Bayesian networks is the existence of equivalence classes. Two pathway hypotheses $\mathcal{H}_1$ and $\mathcal{H}_2$ have the same likelihood, $P(\mathbf{D}|\mathcal{H}_1) = P(\mathbf{D}|\mathcal{H}_2)$ if they only differ in transitive edges. Consider three S-genes *A*, *B*, and *C*. If *A* is upstream of *B*, and *B* is upstream of *C* in the regulatory hierarchy, then silencing *A* will affect *C*. The structure of the model and the scoring scheme do not allow distinguishing between a direct interaction $A \rightarrow C$, an indirect interaction $A \rightarrow B \rightarrow C$, or the existence of both regulatory paths. Assuming parsimony, we can select among all score-equivalent graphs the one with the minimum number of edges. This technique is called transitive reduction and was adopted in our study.

4. Notes

This section contains notes on how we applied the methods described in the previous section in practice. Table 1 includes an overview of the software packages used, with their web addresses from which they can be downloaded. Networks are obtained with these programs as follows.

BNlearn (for BNets) learns a network using a greedy learning algorithm (the growth-shrink algorithm) proposed by Margaritis (34). To estimate the confidence in the edges, we follow Friedman *et al.* (21) and apply a bootstrap procedure. To this end, the optimization is repeated on 100 bootstrap replicas, from which bootstrap support values are computed as the relative frequency of occurrence of the edges. These bootstrap support values provide an indication of the confidence we have in the edges. They also allow us to obtain sparser network structures by only keeping those edges whose bootstrap support values exceed a specified threshold.

SparseBayes (for SBR) predicts a network that results from a greedy optimization procedure where, starting from an empty graph, parent nodes are added and removed for each node until the Bayesian score can no longer be improved. Again, confidence scores for the edges are obtained via bootstrapping, in the same way as obtained for BNets.

GeneNet (for GGMs) computes p-values for both non-zero edges and edge directions. From these p-values, a network can be predicted when a target false-discovery rate (FDR) is specified. In our application, we used the default threshold provided by the software. In principle, networks at different connectivity densities can be obtained by varying the FDR threshold. Alternatively, one can keep the threshold fixed and obtain support values for the edges via a bootstrap analysis, as described above. We found that both procedures lead to similar results. The bootstrap approach is computationally more expensive, but avoids the assumption of an asymptotic functional form for the distribution under the null hypothesis, on which the computation of the p-values in GeneNet is based.

Given a fixed value for the regularization parameter, Lars (for LASSO) predicts a network using a greedy optimization scheme. The program automatically optimizes the regularization hyperparameter via tenfold cross-validation. Networks with different connectivity densities can be obtained by varying the threshold. As an alternative, we use a 100-fold bootstrapping procedure, where for each bootstrap replica the hyperparameter is inferred on the basis of the same tenfold cross-validation procedure.

5. Simulations

Owing to the absence of a gold-standard network for the real data, we evaluated the performance of the methods on simulated data.

We first compare the performance of the four methods for complete data: GGM, BNet, LASSO, and SBR. We would also like to compare their individual performance with that of the consensus network obtained from model averaging. In machine learning, it is well known that model averaging leads to an

improvement of the generalization performance of a predictor or classifier. The performance of the combined model is better than the average performance of the individual model: see e.g. Chapter 9 in Bishop (35). We would like to test whether model averaging also results in an improved network reconstruction. We take as the gold-standard network the graph shown in the top left panel of Fig. 11, and generate data in the following way. For the root node (*expM*), we sample a new value Y from a normal distribution:

$$Y \sim \mathcal{N}(0, \sigma_a^2), \tag{18}$$

with $\sigma_a = 1.0$. For a node with parent set $\{\pi\}$, new values are sampled from a Gaussian distribution whose mean is given by the average over the parent set:

$$Y \sim \mathcal{N}(\xi \langle X_\pi \rangle, \sigma_a^2); \langle X_\pi \rangle = \frac{1}{K} \sum_{\pi=1}^{K} X_\pi; \xi = \sqrt{\frac{K}{\sigma_a^2 + \sigma_b^2}}, \tag{19}$$

where X_π is the value of the πth parent node, and K is the cardinality of the parent set. The factor ξ is chosen to yield a constant average signal-to-noise ratio of σ_a^2 / σ_b^2 across all nodes in the network. We chose three signal-to-noise ratios: $\sigma_b^2 = 0.1, 0.2$, and 0.5. We generated data sets with $N = 30$ instances, which is about the same number as available for the real data. For each value of σ_b^2, we inferred networks from the data by applying the programs listed in Table 1. We repeated this process on 100 independent data instantiations and computed, for each method and each edge, the marginal probability of the edge occurring. We also averaged the probabilities over the individual methods; this gives the marginal probability of an edge obtained from model averaging. Imposing a threshold on these probabilities, we can determine the number of true-positive (TP), false-positive (FP), true-negative (TN), and false-negative (FN) edges by comparison with the gold-standard network. We then compute the sensitivity = TP/(TP + FN), the specificity = TN/(TN + FP), and the complementary specificity = 1 – specificity = FP/(TN + FP). Rather than selecting an arbitrary value for the threshold, we repeat this scoring procedure for all possible threshold in the interval $(0,1)$, and plot the ensuing sensitivity scores against the corresponding complementary specificity scores. This gives the receiver operating characteristics (ROC) curves of Fig. 4, where a larger area under the curve (AUC) indicates, overall, a better performance of the method. We found that the AUC score obtained with model averaging was slightly but consistently larger than the average AUC score, in corroboration of our conjecture.

In order to assess the reliability of the NEM results, we decided to perform a simulation study similar to the one described in Fröhlich *et al.* (31). However, rather than sample network structures, we restricted ourselves to one of the networks inferred

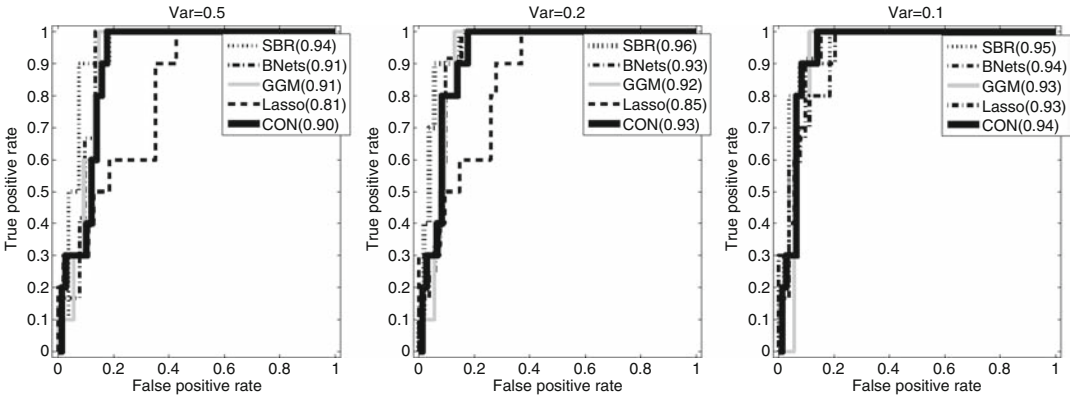


Fig. 4. ROC curves obtained on the synthetic data for GGMs, BNets, LASSO, SBR, and the consensus network. For each graph, the proportion of true-positive edges (*vertical axis*) is plotted against the proportion of false-positive edges (*horizontal axis*). The three panels refer to different noise levels. The areas under the ROC curves (AUC) obtained for the different methods are shown in the legends. Note that the AUC scores obtained with model averaging were found to be slightly but consistently larger than the average AUC scores. *Left panel* ($\sigma_b^2 = 0.5$): 0.90 versus 0.89; *centre panel* ($\sigma_b^2 = 0.2$): 0.93 versus 0.915; *right panel* ($\sigma_b^2 = 0.1$): 0.94 versus 0.9375.

from the real knockout data. We chose, rather arbitrarily, the graph in the top left panel of Fig. 11. This has the advantage that we evaluate the NEM performance on a network that is semirealistic, rather than the transitively closed ideal networks of Fröhlich *et al.* (31). Like Fröhlich *et al.* (31), we sample *p*-values for each knockout from the mixture distribution in Eq. (14). Each S-gene is linked to 100 E-genes. The *p*-values for E-genes where we do not expect an effect due to the network structure are sampled from the uniform distribution. There is a slight subtlety in when to expect an effect if there are different paths between two S-genes (e.g. between *expM* and *aepA* in the network we use here). If one path is disabled by a network, do we expect to see an effect downstream (AND model) or will the signal travel via the alternative path (OR model). We chose to adopt the AND model, which corresponds, e.g. to heterodimerization. For each S-gene where one would expect an effect, we calculate the probability of observing that effect, based on the distance of the current S-gene to the knockout gene. The observed effects are sampled from the beta distributions according to the mixing coefficients π_{1k} and π_{2k}, while for unobserved effects we sample from the uniform distribution. For each knockout, all parameters are drawn from the same ranges as in Fröhlich *et al.* (31), and for each E-gene a small amount of Gaussian noise is added to these parameters.

The results are shown in Fig. 5. There is no significant difference between the two optimization schemes: triples versus greedy search, whereas the filtering versus unfiltering scheme shows a significant difference (at the 0.05 significance level). Surprisingly, the effect of filtering is not consistent, though, leading to an improvement

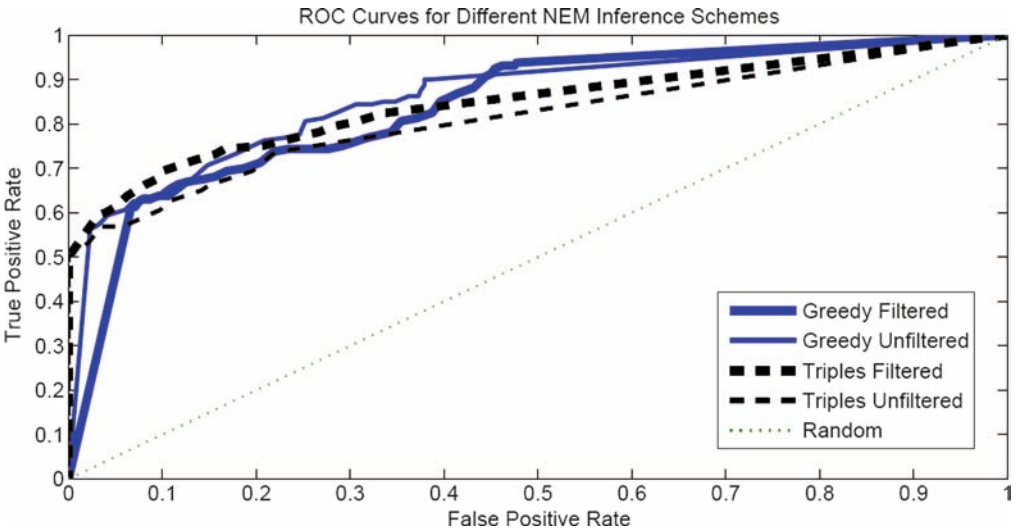


Fig. 5. ROC curves obtained for different training schemes of the NEMs. Two training methods (triples versus greedy search) were combined with two data filtering schemes; see Subheadings 3.5.3 and 3.5.4 for details. The different line types refer to the four different combinations. The areas under the ROC curves are as follows: Greedy filtered (*solid thick line*): 0.84 ± 0.04. Greedy unfiltered (*solid thin line*): 0.86 ± 0.04. Triples filtered (*dashed thick line*): 0.84 ± 0.05. Triples unfiltered (*dashed thin line*): 0.81 ± 0.04. The graphs and estimates were obtained from ten independent data instantiations.

in the performance of the triple method, but a deterioration in the performance of the greedy search. Given these inconsistencies, we decided to apply all four methods to the real data.

6. Results

In order to assess the networks inferred with the different methods, we use a three-prong approach. First, we compute global network properties based on the degree distribution and check if they are consistent with typical patterns found in gene regulatory networks. Second, we assess how consistent the different predictions are, based on a bootstrap analysis and ROC (receiver operating characteristics) curves. Finally, we investigate the biological plausibility of the inferred network structures. Owing to our limited knowledge of regulatory networks and signalling pathways in Pba, the last approach is only partially feasible for the key regulatory genes targeted in the knockout experiments, as listed in Subheading 2.1. We have therefore only applied it to the graphs inferred with NEMs.

6.1. Global Network Properties and Degree Distribution

A comparison of topological features in networks has revealed certain common characteristics among gene regulatory networks. In particular, these networks tend to be scale free, as discussed for instance in Guelzim *et al.* (36). This means that gene regulatory

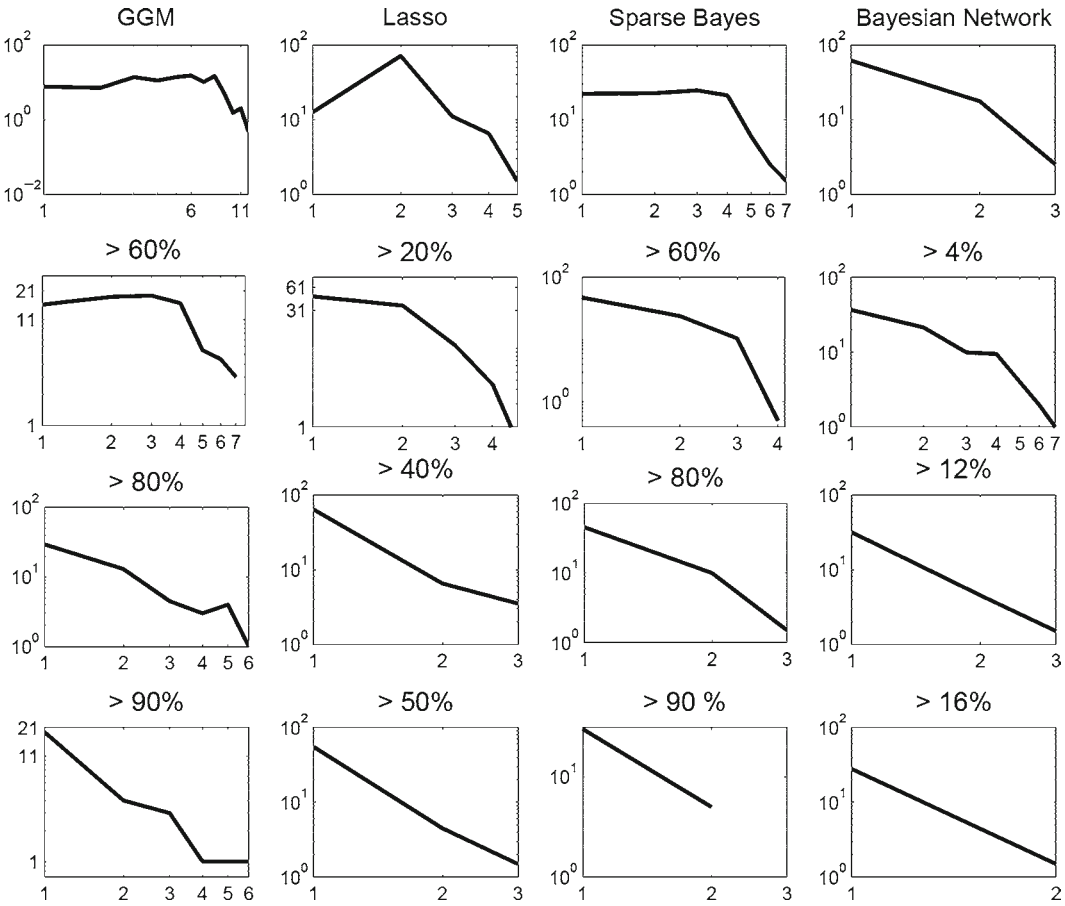


Fig. 6. Double logarithmic plot of the number of nodes $N(k)$ with a given degree k for networks inferred with different methods. *Left column*: Graphical Gaussian models (GGMs). *Centre left column*: LASSO. *Centre right column*: Sparse Bayesian regression (SBR). *Right column*: Bayesian networks (BNets). The top row shows the degree distributions for networks obtained as outputs of the programs shown in Table 1. The remaining rows show the degree distributions for sparser networks obtained from a bootstrap analysis, as described in Subheading 4. Percentage scores over the graphs indicate the thresholds on the bootstrap support values. For each panel, the *x*-axis represents the degree of a node, $N(k)$, and the *y*-axis represents the number of nodes with a specified degree, k. Both axes are on a logarithmic scale. Hence, a non-trivial straight line is indicative of a power law distribution.

networks tend to be characterized by a power law, where the number of nodes with a certain degree k is a power of that degree: $N(k) \propto k^{\alpha}$. The degree of a node k is the number of connections it has. The polynomial dependence of $N(k)$ on k implies a linear dependence in a double-logarithmic representation. Figure 6 suggests that none of the networks inferred from the original gene expression profiles with the programs listed in Table 1 satisfies this dependence. This means that none of the inferred networks exhibits a global structure that is consistent with what is expected to be found in gene regulatory networks. We suspect that this deviation is a consequence of the prediction of several spurious edges. To proceed, we carried out the bootstrap analysis described

in Subheading 4. We then discarded all edges with a bootstrap support value below a specified threshold. The results are shown in the bottom rows of Fig. 6. For GGMs, a threshold on the bootstrap support value of over 80 or 90% results in a network that approximately follows a power law. This finding is encouraging, as the resulting network has a topological feature that is consistent with other gene regulatory networks. For the other methods, the results are less encouraging though. The resulting networks either do not exhibit a power law, or otherwise become too sparse, without a degree greater than 3 or even 2, and hence only a trivial linearity in the log–log plot[10].

In machine learning it has been known for a long time that combining the predictions from several models, ideally of different nature and differently trained, leads on average to more accurate predictions than what can be obtained with a single model; see for instance Battiti and Colla (34), and our discussion in Subheading 5. While this approach of model averaging has mainly been applied to regression and classification tasks, we here pursue the same idea for predicting network structures. Our aim is to combine the networks predicted with BNets, GGMs, LASSO, and SBR. One approach is to combine the original predictions via a filter that ignores all interactions unless they are supported by at least m different methods. The top two rows of Fig. 7 show plots of the number of nodes against their connectivity degree for different values of m. It is seen that by increasing m, the network first shows a better compliance with the power law, until eventually (for $m = 4$) we obtain a trivial linearity defined by only two non-vanishing node degrees. An alternative approach is to combine bootstrap-thresholded rather than the originally predicted networks. Guided by Fig. 6, we chose the following thresholds on the bootstrap support values. GGMs: 80%, LASSO: 40%, SBR: 80%, and BNets: 8%. These values were selected on the basis that we want to find the lowest threshold (and hence the densest connectivity) subject to the requirement that the double-logarithmic plot of the number of nodes $N(k)$ against the degree k shows approximately a linear relationship. The bottom row of Fig. 7 shows the double logarithmic $N(k)$ versus k plot for different consensus networks. This figure suggests that requiring edges to be independently predicted by two different methods gives the best compromise between sparsity versus power law, and the resulting consensus network is shown in Fig. 9.

6.2. Consistency Among the Predictions

It would be interesting to study how consistent or different the four methods for reconstructing networks from complete observation (GGMs, BNets, SBR, and LASSO) are. To this end, we infer a network with each of the four methods, using the software packages of Table 1 with default options. We then carry out the bootstrap

[10] Note that one can always draw a straight line through two points.

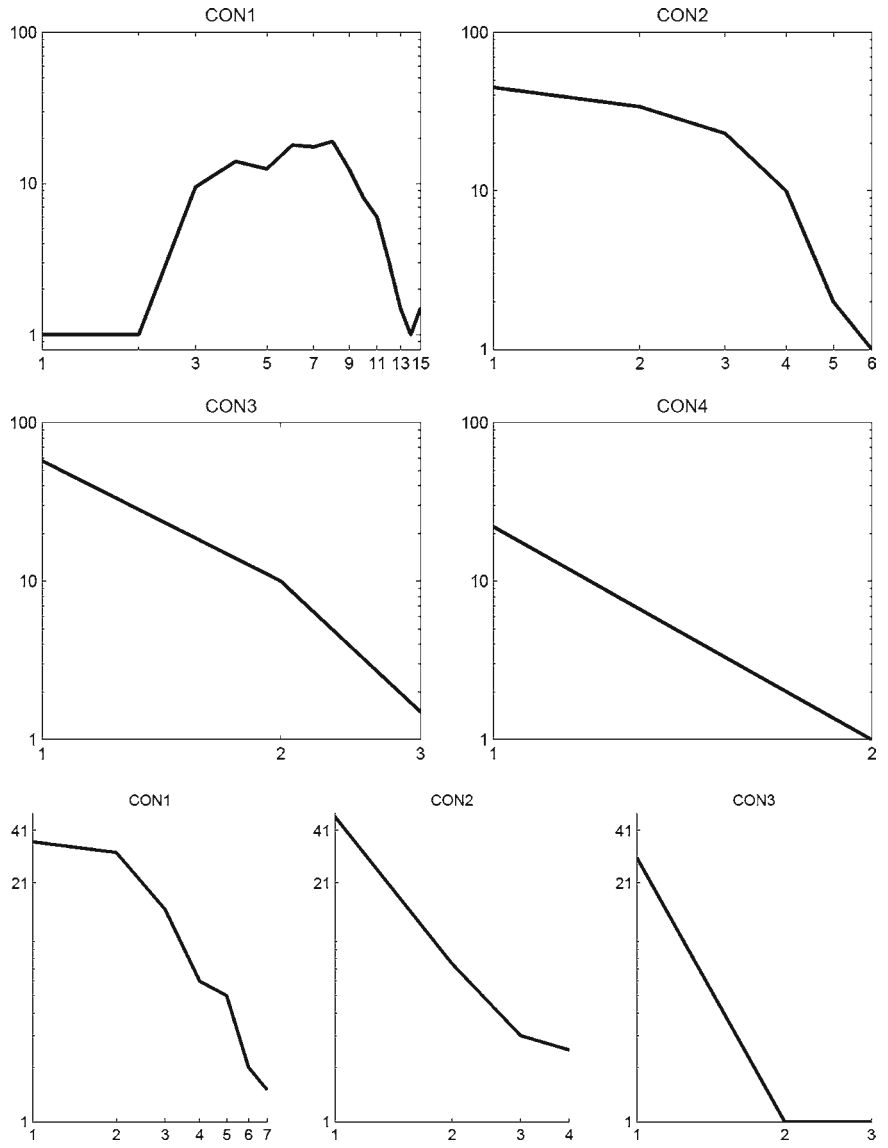


Fig. 7. Double logarithmic plot of the number of nodes $N(k)$ (*vertical axis*) with a given degree k (*horizontal axis*) for different consensus networks. *Top two rows*: The consensus networks were obtained from the networks constructed with GGMs, BNets, LASSO, and SBR, using the programs listed in Table 1 with default values. *Bottom row*: Individual networks were obtained from the bootstrap analysis described in Subheading 4, which were then combined to form a consensus network; see the main text for further details. CON1 is the union of all networks, i.e. an edge is contained in the consensus network if it is present in any of the individual networks. CON2 is a network that contains all those edges that are predicted by at least two reconstruction methods. Likewise, CON3 and CON4 are stricter consensus networks, which contain only edges predicted independently by at least three or all four methods, respectively.

analysis described in Subheading 4 to obtain confidence scores for the edges, on the basis of which the latter are ranked. Given a network structure and a ranking of the edges, we can obtain the receiver operating characteristic (ROC) curve. For instance, taking the network learned with GGM as a gold standard, and taking the bootstrap support values for SBR, we obtain the ROC curve

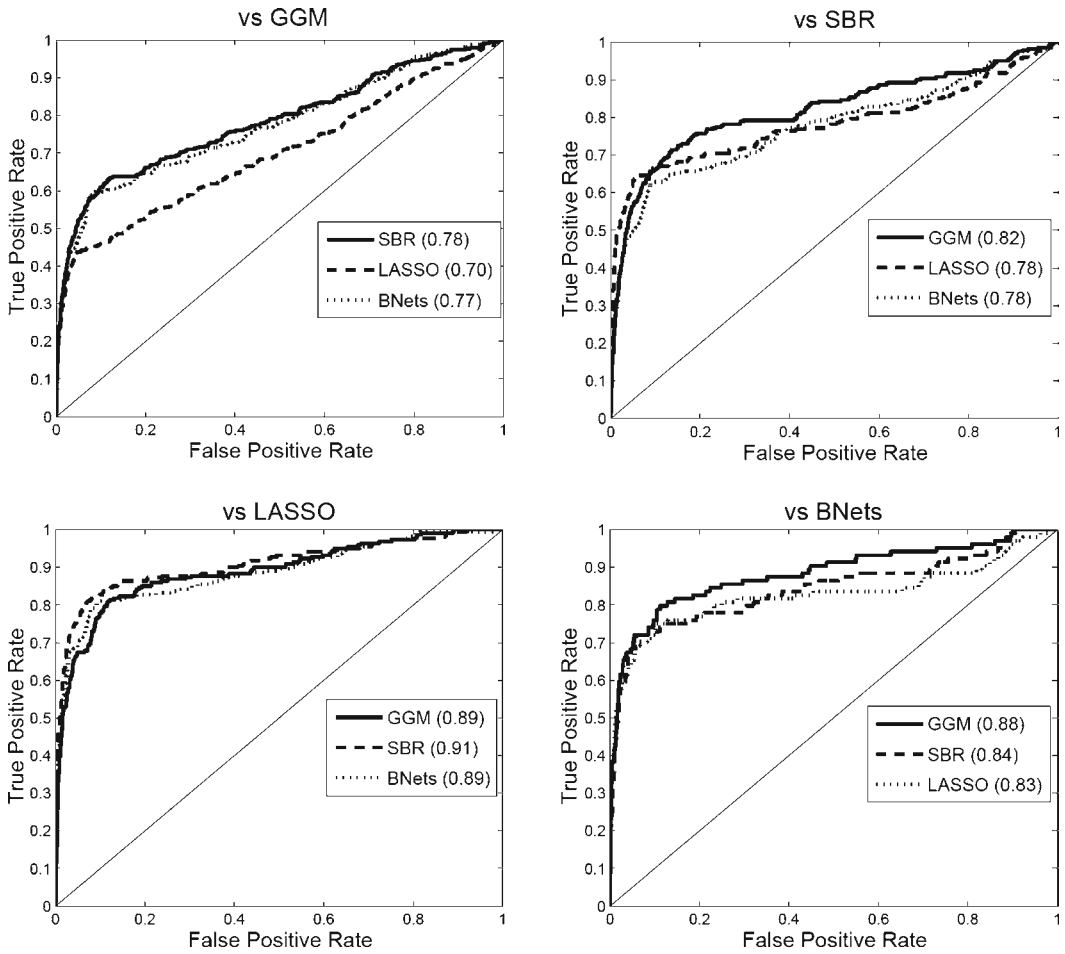


Fig. 8. ROC (receiver operating characteristic) curves for a comparison between different network reconstruction methods. For each of the four investigated methods – GGMs, BNets, LASSO, and SBR – a network was inferred, using the software listed in Table 1. Following the bootstrap procedure described in Subheading 4, bootstrap support values for the edges were obtained with each of the remaining three methods. Taking the network from the first stage as a gold standard, the ROC curves were obtained, with larger areas under the curve indicating a better agreement between the respective methods. The names of the reference methods are shown at the top of each panel.

shown as a solid line in the top left panel of Fig. 8. Repeating the same with the bootstrap support values obtained for BNets, we obtain the ROC curve shown as a dotted line in the top left panel of Fig. 8, and so on. The resulting ROC curves, shown in Fig. 8, are consistently better than what would be expected for a random ordering of the edge scores, with areas under the ROC curve ranging between 0.7 and 0.9. This suggests that the agreement between the different methods is significantly better than random.

6.3. Predicted Networks

The consensus network obtained from the reconstruction methods for complete observation – BNets, GGMs, LASSO, and SBR – is shown in Fig. 9. Each node represents a gene cluster, obtained

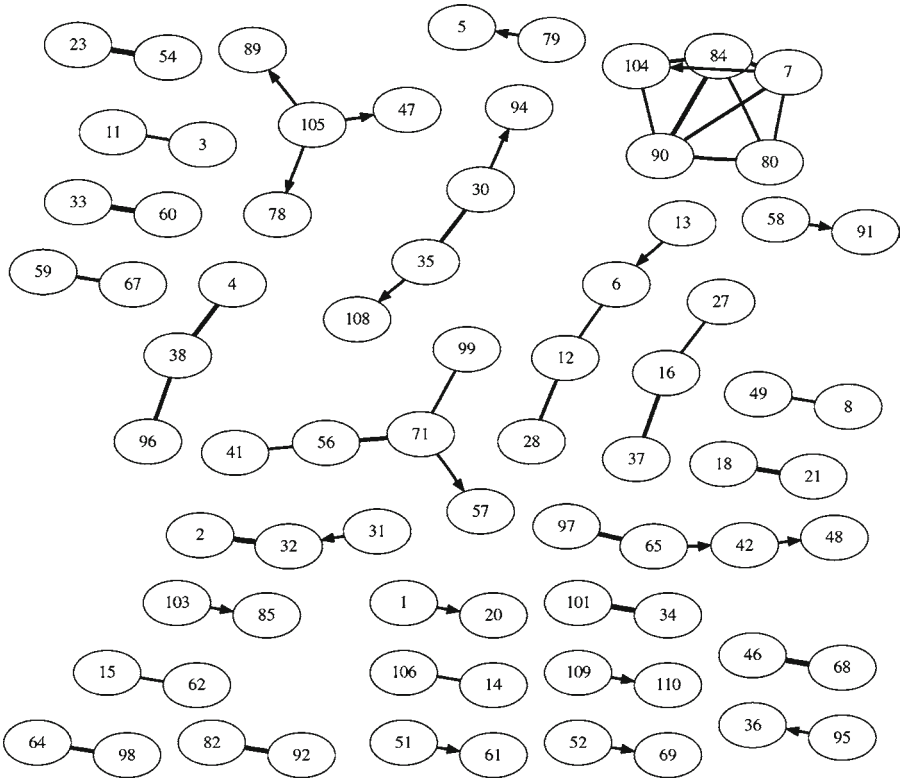


Fig. 9. Consensus network obtained from the networks inferred with GGMs, BNets, LASSO, and SBR. The edges were inferred by at least two different methods. Edges that were inferred by three or four methods are shown as *thick lines*. Each node refers to a gene cluster, described in detail in Table 2 of the supplementary material at http://www.bioss.ac.uk/staff/dirk/Supplements/FF842/.

with the pre-processing method described in Subheading 2.5. Information about the composition of the clusters, with a complete list of genes they contain and GO terms that are significantly enriched, is available from Table 2 of the supplementary material.[11] Note that the threshold on the edges was chosen conservatively to reduce the number of false positives and to ensure that the degree distribution is approximately consistent with a power law. This implies that we incur a certain proportion of false-negative edges, which is indicated by the existence of many disconnected modules. Hence, Fig. 9 only shows the most salient gene regulatory interactions in which we place a comparatively high confidence. To our knowledge, the reconstruction of gene regulatory networks in Pba from expression profiles of several knockout strains has not been attempted before, and no gold-standard network is available for assessing our prediction. The biological interpretation of the

[11] http://www.bioss.ac.uk/staff/dirk/Supplements/FF842/.

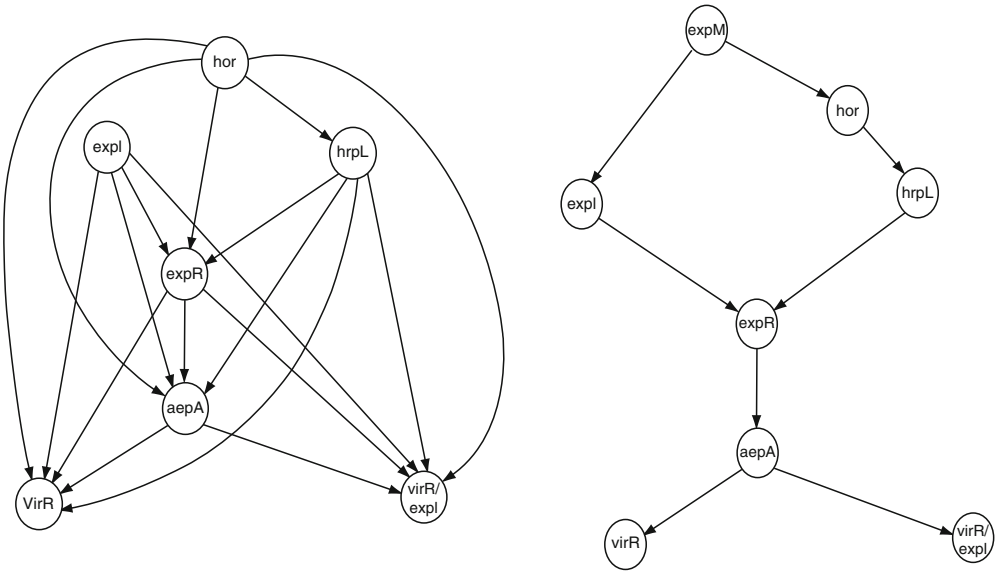


Fig. 10. NEMs trained with greedy search on filtered data. This figure shows the network obtained using a greedy search with bootstrapping, where only the edges that were present in all the bootstrap samples have been retained. The set of effector genes (E-genes) used for the search has been pre-filtered, as described in subheading 3.5.3, to retain only those genes that show a non-random expression pattern over all knockouts. *Left*: Owing to an intrinsic symmetry feature of NEMs, the network has to be transitively closed, meaning that if there is an edge from A to B and an edge from B to C, then there has to be an edge from A to C. *Right*: Transitively reduced graph, where all shortcut edges have been removed. This is the sparsest graph in its equivalence class. The nodes represent the genes targeted in the knock-out experiments: expM, hor, hrpL, expI, expR, aepA, virR, virS, and the double knock-out expI/virS. The genes virS and expM are not included in the *left panel* for visibility purposes (expM was found to be a universal regulator, and virS was found to be regulated by every node).

inferred gene regulatory interactions is still the subject of current research and can be expected to shed new light on the mechanism of quorum sensing in Pba.

The results obtained with NEMs are shown in Fig. 10 and 11. Recall that the objective here is to infer the interactions among the genes targeted in the transposon mutagenesis experiment; these genes are listed in Subheading 2.1. The method allows for the fact that these regulator genes might be subject to post-transcriptional regulation: interactions are inferred from the effects gene knockouts have on the down-stream regulated genes rather than from their gene expression profiles. Recall from Subheading 3.5.4 that due to intrinsic symmetries of the scoring scheme, NEMs cannot distinguish between two network structures that are related via a transitive closure operation. This implies that the two network structures in Fig. 10 are score equivalent, that is, they cannot be distinguished on the basis of the data and the inference scheme. As such, the method only infers regulatory hierarchies rather than actual interaction networks.

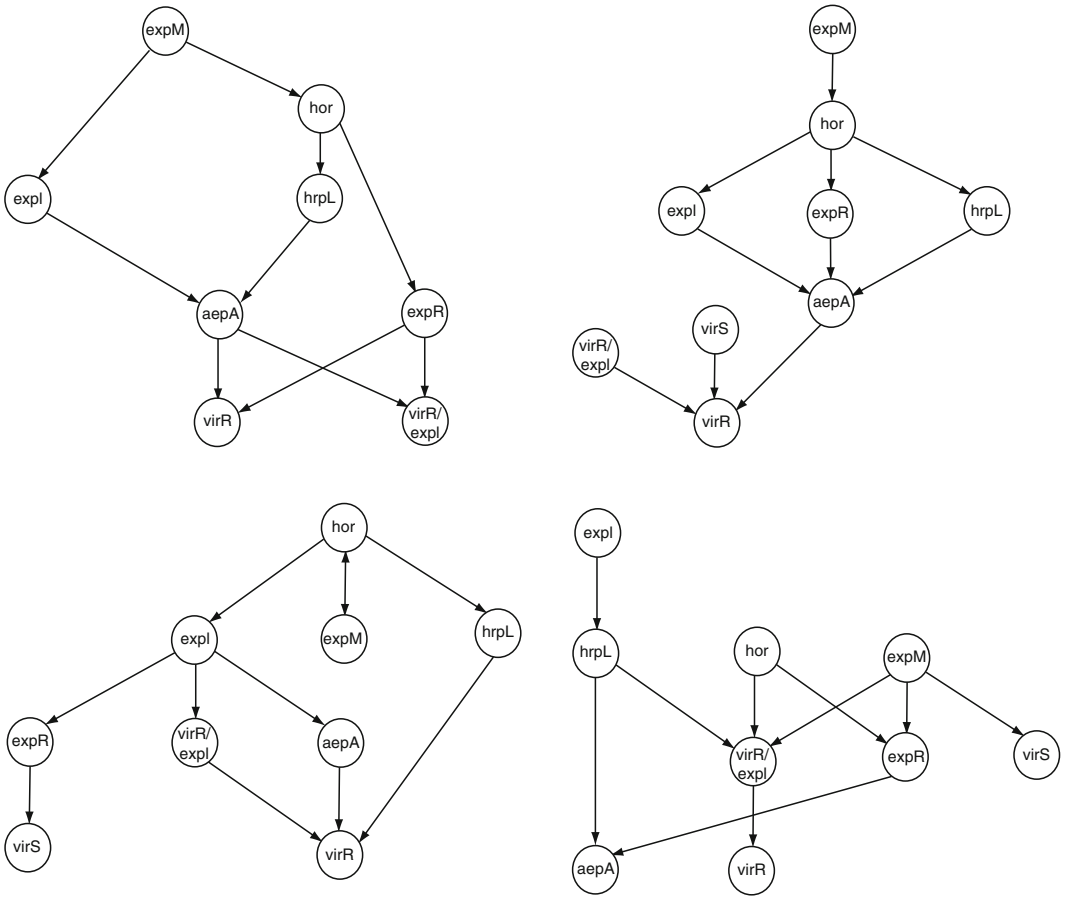


Fig. 11. NEMs learned with different optimization schemes and filtering methods. *Top left*: Transitive reduction of the network found using the same method as shown in this figure, but excluding the *virS* knockout data. There was reason to believe that the *virS* data could affect filtering (and possibly network construction) because it was generated under different conditions than the other knockouts, which might lead to spurious effects. *Top right*: Transitive reduction of the network found using the same method as for Fig. 10, but without applying the filtering method of Subheading 3.5.3. *Bottom left*: Transitive reduction of the network found using the triples scoring method, described in Subheading 3.5.4. Only edges with 100% support have been retained. The same gene filtering method as for Fig. 10 has been applied. *Bottom right:* This graph was obtained with the same method as for the panel on the *bottom left*, but without filtering the genes. All edges with support values over 50% support were retained.

To resolve ambiguities, we have adopted the principle of transitive reduction, which means that we always present the most parsimonious graph of an equivalence class. For instance, the right panel of Fig. 10 shows the transitively reduced graph of the one shown in the left panel. This has to be considered when interpreting the networks in Fig. 11: whenever two nodes are connected by a path, an interaction via a shortcut path is supported by the data also. The networks in Fig. 10 and 11 were obtained with different inference schemes and pre-filtering methods, as described in Subheadings 3.5.3 and 3.5.4. We have also repeated the analysis with and

without *virS* included; this is motivated by the fact that the *virS* knockout experiment was carried out under different conditions, which might add an unwanted source of noise.

Owing to the lack of a gold standard, the direct evaluation of the predicted networks is not feasible. However, some patterns of the regulatory interactions among the selected genes have been reported in the literature. According to our current understanding, summarized in Fig. 5 in ref. (1). *expI* is upstream of both *virR* and *aepA* in the regulatory hierarchy. Figures 10 and 11 suggest that this order is, in fact, consistently predicted by all the graphs learned in our study. Moreover, in none of the predicted networks does the double knockout *expI/virR* appear above both the individual knockouts *expI* and *virR*. This can only be explained by some antagonism between *expI* and *virR*, which is again in agreement with the regulatory structure reported by Liu et al. (1). There are also some interesting deviations, though. Liu et al. (1) predict *expM* to be quite low in the regulatory hierarchy. However, all the graphs learned in our study concur in predicting *expM* to be at the top of the regulatory hierarchy. This finding might point to some flaws in the current hypotheses about the regulatory mechanisms in Pba, with the prospect to obtain a revised and improved model of gene regulatory interactions from the novel data analysis tools explored in the present work. A comparison of the predicted graphs points to some disagreement between them. This is an inevitable consequence of the noise in the data, the complexity of the inference problem, and the different nature of the approaches adopted for dealing with both. Our work is one of the first studies to investigate the robustness of learning NEMs from real data and provides insight into the degree of variation in graph structure that results from a variation of the learning algorithm and pre-filtering scheme.

7. Conclusion

To our knowledge, the present work is the first study that aims to reverse engineer regulatory networks related to quorum sensing in the plant pathogen *Pectobacterium atrosepticum* (Pba) from gene expression profiles obtained after transposon mutagenesis. We have applied four different methods for reconstructing networks from complete data: graphical Gaussian models (GGMs), LASSO, sparse Bayesian regression (SBR), and Bayesian networks (BNets). We have complemented these methods with nested effects models (NEMs), which allow for post-transcriptional modification and which infer regulatory interactions between regulatory genes from their downstream regulation effects. We first tested the selected methods on synthetic data. The insight

gained from these studies has guided our application to the gene expression profiles from mutated Pba strains. We observed a significant degree of agreement among the different methods and found that some known features of regulatory interactions between key regulator genes in Pba could be consistently recovered. This suggests that the network structures reconstructed in our study contain relevant information and have the potential to contribute to the elucidation of the nature of signalling pathways and regulatory processes related to quorum sensing in plant pathogens.

Acknowledgement

This work was supported by the Scottish Government Rural and Environment Research and Analysis Directorate (RERAD) under the flexible funds scheme, grant number 842.

References

1 Liu, H., Coulthurst, S. J., Pritchard, L., Hedley, P. E., Ravensdale, M., Humphris, S., Burr, T., Takle, G., Brurberg, M.-B., Birch, P. R. J., Salmond, G. P. C. and Toth, I. K. (2008) Quorum sensing coordinates brute force and stealth modes of infection in the plant pathogen *Pectobacterium atrosepticum*. *PLoS Pathogens*, **4**, 29.

2 Yang, Y., Dudoit, S., Luu, P. and Speed, T. (2001) Normalization for cDNA microarray data. In Bittner, M., Chen, Y., Dorsel, A. and Dougherty, E. (eds.), *Microarrays: Optical Technologies and Informatics*, volume 4266 of Proceedings of SPIE.

3 Smyth, G. K. (2004) Linear models and empirical Bayes methods for assessing differential expression in microarray experiments. *Statistical Applications in Genetics and Molecular Biology*, **3**, Article 3.

4 Smyth, G. K. (2005) Limma: linear models for microarray data. In Gentleman, R., Carey, V., Huber, W., Irizarry, R. and Dudoit, S. (eds.), *Bioinformatics and Computational Biology Solutions Using R and Bioconductor, Statistics for Biology and Health*, pp. 397–420. Springer, New York.

5 Lönnstedt, I. and Speed, T. (2002) Replicated microarray data. *Statistica Sinica*, **12**, 31–46.

6 Hastie, T., Tibshirani, R. and Friedman, J. (2001) *The Elements of Statistical Learning*. Springer-Verlag, New York.

7 Beal, M. J. (2003) Variational algorithms for approximate Bayesian inference. Ph.D. thesis, Gatsby Computational Neuroscience Unit, University College London.

8 Butte, A. S. and Kohane, I. S. (2000) Mutual information relevance networks: functional genomic clustering using pairwise entropy measurements. *Pacific Symposium on Biocomputing*, **2000**, 418–429.

9 Werhli, A. V., Grzegorczyk, M. and Husmeier, D. (2006) Comparative evaluation of reverse engineering gene regulatory networks with relevance networks, graphical Gaussian models and Bayesian networks. *Bioinformatics*, **22**, 2523–2531.

10 Edwards, D. M. (2000) *Introduction to Graphical Modelling*. Springer Verlag, New York.

11 Schäfer, J. and Strimmer, K. (2005) A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. *Statistical Applications in Genetics and Molecular Biology*, **4**, Article 32.

12 Opgen-Rhein, R. and Strimmer, K. (2007) From correlation to causation networks: a simple approximate learning algorithm and its application to high-dimensional plant gene expression data. *BMC Systems Biology*, **1**, 37.

13 Williams, P. M. (1995) Bayesian regularization and pruning using a Laplace prior. *Neural Computation*, **7**, 117–143.

14 Tibshirani, R. (1996) Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, **58**, 267–288.

15 van Someren, E. P., Vaes, B. L. T., Steegenga, W. T., Sijbers, A. M., Dechering, K. J. and

Reinders, M. J. T. (2006) Least absolute regression network analysis of the murine osterblast differentiation network. *Bioinformatics*, **22**, 477–484.

16 Grandvalet, Y. and Canu, S. (1998) Outcomes of the equivalence of adaptive ridge with least absolute shrinkage. In Kearns, M., Solla, S.A. and Cohn, D.A. (eds.), *Advances in Neural Information Processing Systems* 11, pp. 445–451. MIT Press, Cambridge

17 MacKay, D. J. C. (1996) Hyperparameters: optimize, or integrate out. In Heidbreder, G. (ed.), *Maximum Entropy and Bayesian Methods*, pp. 43–59. Kluwer Academic Publisher, Santa Barbara.

18 MacKay, D. J. C. (1992) Bayesian interpolation. *Neural Computation*, **4**, 415–447.

19 Rogers, S. and Girolami, M. (2005) A Bayesian regression approach to the inference of regulatory networks from gene expression data. *Bioinformatics*, **21**, 3131–3137.

20 Tipping, M. and Faul, A. (2003) Fast marginal likelihood maximisation for sparse Bayesian models. In M., B. C. and J., F. B. (eds.), *Proceedings of the International Workshop on Artificial Intelligence and Statistics*, volume 9.

21 Friedman, N., Linial, M., Nachman, I. and Pe'er, D. (2000) Using Bayesian networks to analyze expression data. *Journal of Computational Biology*, **7**, 601–620.

22 Hartemink, A. J., Gifford, D. K., Jaakkola, T. S. and Young, R. A. (2001) Using graphical models and genomic expression data to statistically validate models of genetic regulatory networks. *Pacific Symposium on Biocomputing*, **6**, 422–433.

23 Husmeier, D., Dybowski, R. and Roberts, S. (2005) Probabilistic Modeling in Bioinformatics and Medical Informatics. Advanced Information and Knowledge Processing. Springer, New York.

24 Heckerman, D. (1999) A tutorial on learning with Bayesian networks. In Jordan, M. I. (ed.), *Learning in Graphical Models, Adaptive Computation and Machine Learning*, pp. 301–354. MIT Press, Cambridge, Massachusetts.

25 Grzegorczyk, M., Husmeier, D. and Werhli, A. (2008) Reverse engineering gene regulatory networks with various machine learning methods. In Emmert-Streib, F. and Dehmer, M. (eds.), *Analysis of Microarray Data: A Network-Based Approach*, pp. 101–142. Wiley-VCH, Weinheim.

26 Geiger, D. and Heckerman, D. (1994) Learning Gaussian networks. In *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, pp. 235–243. Morgan Kaufmann, San Francisco, CA.

27 Madigan, D. and York, J. (1995) Bayesian graphical models for discrete data. *International Statistical Review*, **63**, 215–232.

28 Friedman, N. and Koller, D. (2003) Being Bayesian about network structure. *Machine Learning*, **50**, 95–126.

29 Grzegorczyk, M. and Husmeier, D. (2008) Improving the structure MCMC sampler for Bayesian networks by introducing a new edge reversal move. *Machine Learning*, **71**, 265–305.

30 Markowetz, F., Bloch, J. and Spang, R. (2005) Non-transcriptional pathway features reconstructed from secondary effects of RNA interference. *Bioinformatics*, **21**, 4026–4032.

31 Fröhlich, H., Fellmann, M., Sultmann, H., Poustka, A. and Beissbarth, T. (2008) Estimating large scale signaling networks through nested effect models with intervention effects from microarray data. *Bioinformatics*, **24**, 2650–2656.

32 Markowetz, F., Kostka, D., Troyanskaya, O. and Spang, R. (2007) Nested effects models for highdimensional phenotyping screens. *Bioinformatics*, **23**, i305–i312.

33 Fröhlich, H., Tresch, A. and Beissbarth, T. (2009) Nested effects models for learning signaling networks from perturbation data. *Biometrical Journal*, **51**, 304–323.

34 Margaritis, D. (2003) Learning Bayesian network model structure from data. Ph.D. thesis, School of Computer Science, Carnegie-Mellon University.

35 Bishop, C. M. (1995) *Neural Networks for Pattern Recognition*. Oxford University Press, New York, ISBN 0-19-853864-2.

36 Guelzim, N., Bottani, S., Bourgine, P. and Kepes, F. (2002) Topological and causal structure of the yeast transcriptional regulatory network. *Nature Genetics*, **31**, 60–63.

37 Battiti, R. and Colla, A. M. (1994) Democracy in neural nets: voting schemes for classification. *Neural Networks*, **7**, 691–707.

Chapter 18

Parameter Inference and Model Selection in Signaling Pathway Models

Tina Toni and Michael P. H. Stumpf

Abstract

To support and guide an extensive experimental research into systems biology of signaling pathways, increasingly more mechanistic models are being developed with hopes of gaining further insight into biological processes. In order to analyze these models, computational and statistical techniques are needed to estimate the unknown kinetic parameters. This chapter reviews methods from frequentist and Bayesian statistics for estimation of parameters and for choosing which model is best for modeling the underlying system. Approximate Bayesian computation techniques are introduced and employed to explore different hypothesis about the JAK-STAT signaling pathway.

Key words: Statistical inference, Approximate Bayesian computation, Bayesian model selection, Ordinary differential equation models, Signal transduction, Systems biology

1. Introduction

It is crucial for cells to be able to sense the environment and react to changes in it. This is done through signaling pathways, which involve complex networks of often nonlinear interactions between molecules. These interactions transduce a signal from outside the cell to trigger a functional change within a cell. Signaling pathways are important for differentiation, survival, and adaptation to varying external conditions. The dynamics of these pathways are currently a subject of extensive experimental and computational research (1–5); and some of the most important biological signaling pathways have received considerable attention from mathematical modelers and theoretical systems biologists. These signaling networks include MAPK and Ras-Raf-ERK (6–14), JAK-STAT (15–17), GPCR (18), NF-κB (4, 19). It has become abundantly clear, however, that these signaling pathways cannot be separated

David Fenyö (ed.), *Computational Biology*, Methods in Molecular Biology, vol. 673,
DOI 10.1007/978-1-60761-842-3_18,

from one another, but that they interact; this phenomenon is known as crosstalk (20).

A series of modeling approaches have been applied to the study of signaling pathways. Most widely used are ordinary differential equation (ODE) models that follow mass action kinetics. Also Boolean and Bayesian networks and Petri nets have been employed for modeling and simulation. The rich formalism underlying these different approaches has provided us with efficient tools for the analysis of signaling models.

Computational approaches have been mainly used for simulation and for studying qualitative properties of signaling pathways such as motifs and feedback loops (21, 22), and quantitative properties such as signal duration, signal amplitude, and amplification (23–25).

To develop and utilize detailed quantitative signaling models we require the values of all the parameters, such as kinetic rates of protein turnover and posttranslational modifications (e.g., phosphorylation or dimerization). Due to technological restrictions and cost it is impossible to measure all the parameters experimentally. In this chapter, we review computational tools that can be used for parameter inference for ODE models. While many studies have dealt with the subject of parameter estimation, relatively little attention has been given to model selection; that is, which model(s) to use for inference. Despite this, "What is the best model to use?" is probably the most critical question for making valid inference from the data (26), and this is the second topic that we touch on in this chapter.

There are two broad schools of thought in statistical inference: frequentist and Bayesian. In frequentist statistics one talks about point estimates and confidence intervals around them. The likelihood function is a central concept in statistical inference, and is used in both frequentist and Bayesian settings. It equals the probability of the data given the parameters, and it is a function of parameters.

$$L(\theta) = P(D \mid \theta)$$

The canonical way of obtaining the point estimate is by taking a maximum likelihood estimate; i.e., the set of parameters for which the probability of observing the data is highest. On the other hand, Bayesian statistics is based on probability distributions. Here one aims to obtain the posterior probability distribution over the model parameters, which is proportional to the product of a suitable prior distribution (which summarizes the user's prior knowledge or expectations) and the likelihood (the information that is obtained from the data).

In the following sections we will review how frequentist and Bayesian statistics can be used to estimate parameters of ODE models of signaling pathways, and how to choose which model

has the highest support from the data. We then outline an approximate Bayesian computation (ABC) algorithm based on Sequential Monte Carlo and apply it to the JAK-STAT signaling pathway where we will illustrate aspects related to parameter estimation and model selection.

2. Parameter Inference

Signaling pathway models include numerous parameters, and it is generally impossible to obtain all of these values by experimental measurements alone. Therefore parameter inference (also referred to as model calibration, model fitting, or parameter estimation by different authors) algorithms can be used to estimate these parameter values computationally. A variety of different approaches has been developed and is being used; they all share the two main ingredients: a cost function, which reflects and penalizes the distance between the model and experimental data, and an optimization algorithm, which searches for parameters that optimize the cost function. The most commonly used cost functions in a frequentist approach are the likelihood (one wants to maximize it) and the least squares error (one wants to minimize it). The Bayesian equivalent to a cost function is the Bayesian posterior distribution.

There are many different kinds of optimization algorithms. Their goal is to explore the landscape defined by cost function and find the optimum (i.e., minimum or maximum, depending on the type of cost function used). The simplest are the local gradient descent methods (e.g., Newton's method, Levenberg–Marquardt). These methods are computationally fast, but are only able to find local optima. When the cost function landscape is complex, which is often the case for signaling models with high dimensional parameter space, these methods are unlikely to find the global optimum, and in this case more sophisticated methods need to be used. Multiple shooting (27) performs better in terms of avoiding getting stuck in local optima, but, as argued by Brewer et al. (28) may perform poorly when measurements are sparse and data are noisy. A large class of optimization methods is the global optimization methods that try to explore complex surfaces as widely as possible; among these, genetic algorithms are particularly well known and have been applied to ODE models (25). Moles et al. (29) tested several global optimization algorithms on a 36-parameter biochemical pathway model and showed that the best performing algorithm was a stochastic ranking evolutionary strategy (30) (software is available (31, 32)). Further improvements in computational efficiency of this algorithm were obtained by hybrid algorithms incorporating local gradient search and multiple shooting methods (17, 33).

To obtain an ensemble of good parameter values, an approach based on simulated annealing (34) and Monte Carlo search though parameter space can be used (35, 36). In a Bayesian setting, MCMC methods (37) (software is available (38)) and unscented Kalman filtering (39) have been applied to estimate the posterior distribution of parameters. Bayesian methods do not only estimate confidence intervals, but provide even more information by estimating of the whole posterior parameter distribution. To obtain confidence intervals for a point estimate in a frequentist setting, a range of techniques can be applied that include variance–covariance matrix based techniques (40), profile likelihood (41), and bootstrap methods (42).

Parameter estimation should be accompanied by identifiability and sensitivity analyses. If a parameter is nonidentifiable, this means it is difficult or impossible to estimate due to either model structure (structural nonidentifiability) or insufficient amount or quality or data measurements (statistical nonidentifiability) (19, 43, 44). Structurally nonidentifiable parameters should ideally be removed from the model. Sensitivity analysis studies how model output behaves when varying parameters (45). If model output changes a lot when parameters are varied slightly, we say that the model is sensitive to changes in certain parameter combinations. Recently, the related concept of sloppiness has been introduced by Sethna and coworkers (35, 46). They call a model sloppy when the parameter sensitivity eigenvalues are roughly evenly distributed over many decades; those parameter combinations with large eigenvalues are called sloppy and those with low eigenvalues stiff. Sloppy parameters are hard to infer and carry very little discriminatory information about the model. The concepts of identifiability, sloppiness, and parameter sensitivity are, of course, related: nonidentifiable parameters and sloppy parameters are hard to estimate precisely because they can be varied a lot without having a large effect on model outputs; the corresponding parameter estimates will thus have large variances. A parameter with large variance can, in a sensitivity context, be interpreted as one to which the model is not sensitive if the parameter changes.

3. Model Selection

Model selection methods strive to rank the candidate models, which represent hypothesis about the underlying system, relative to each other according to how well they explain the experimental data. Crucially, the chosen model is not the "true" model, but the best model from the set of candidate models. It is the one which we should probably use for making inferences from the data. Generally, the more parameters are included in the model;

the better a fit to the data can be achieved. If the number of parameters equals the number of data points, there is always a way of setting the parameters so that the fit will be perfect. This is called overfitting. Wel (47) famously addressed a question of "how many parameters it takes to fit an elephant," which practically suggests that if one takes a sufficiently large number of parameters, a good fit can always be achieved. The other extreme is underfitting, which results from using too few parameters or too inflexible a model. A good model selection algorithm should follow the principle of parsimony, also referred to as Occam's razor, which aims for to determine the model with the smallest possible number of parameters that adequately represents the data and what is known about the system under consideration.

The probably best known method for model selection is (frequentist) hypothesis testing. If ODE models are nested (i.e., one model can be obtained from the other by setting some parameter values to zero), then model selection is generally performed using the likelihood ratio test (16, 48). If both models have the same number of parameters and if there is no underlying biological reason to choose one model over the other, then we choose the one which has a higher maximum likelihood. However, if the parameter numbers differ, then the likelihood ratio test penalizes overparameterization.

If the models are not nested, then model selection becomes more difficult but a variety of approaches have been developed that can be applied in such (increasingly more common) situations. Bootstrap methods (42, 48) are based on drawing many so-called bootstrap samples from the original data by sampling with replacement, and calculating the statistic of interest (e.g., an achieved significance level of a hypothesis test) for all of these samples. This distribution is thin compared to the real data.

Other model selection methods applicable to non-nested models are based on information-theoretic criteria (26) such as the Akaike Information Criteria (AIC) (16, 48–50). These methods involve a goodness-of-fit term and a term measuring the parametric complexity of the model. The purpose of this complexity term is to penalize models with high number of parameters; the criteria by which this term should be chosen can differ considerably among the various proposed measures.

In a Bayesian setting, model selection is done through so-called Bayes factors (for comprehensive review see (51)). We consider two models, m_1 and m_2 and would like to determine which model explains the data x better. The Bayes factor measuring the support of model m_1 compared to model m_2, is given by:

$$B_{12} = \frac{p(x \mid m_1)}{p(x \mid m_2)} = \frac{\int p(x \mid m_1, \theta_1) p(\theta_1 \mid m_1) \mathrm{d}\theta_1}{\int p(x \mid m_2, \theta_2) p(\theta_2 \mid m_1) \mathrm{d}\theta_2}$$

To compute it, marginal likelihoods have to be computed, and this is done by integrating nonlinear functions over all possible parameter combinations. This can be a challenging problem when the dimension of the parameter space is high, and Vyshemirsky and Girolami (37) asses various methods how this can be done efficiently. A Bayesian version of information-theoretic model selection techniques introduced above is the Bayesian Information Criterion (BIC) (35, 52), which is an approximation of the logarithm of the Bayes factor. Unlike the AIC, which tends toward overly complex models as the data saturates, the BIC chooses correct models in the limit of infinite data availability.

There are several advantages of Bayesian model selection compared to traditional hypothesis testing. Firstly, the models being compared do not need to be nested. Secondly, Bayes factors do not only weigh the evidence against a hypothesis (in our case m_2), but can equally well provide evidence in favor of it. This is not the case for traditional hypothesis testing where a small p value only indicates that the null model has insufficient explanatory power. However, one cannot conclude from a large p value that the two models are equivalent or that the null model is superior, but only that there is not enough evidence to distinguish between them. In other words, "failing to reject" the null hypothesis cannot be translated to "accepting" the null hypothesis (51, 53). Thirdly, unlike the posterior probability of the model, the p value does not provide any direct interpretation of the weight of evidence (the p value is not the probability that the null hypothesis is true). We expect that Bayesian methods will also deal better with so-called sloppy parameters because they are based on explicit marginalization over model parameters.

4. Approximate Bayesian Computation Techniques

When formulating the likelihood for an ODE model, one normally assumes the Gaussian error distribution on the data points: by definition this is the only way of defining a likelihood for a deterministic model. Moreover, it might be hard to analytically work with the likelihood (e.g., finding maximum likelihood estimate and integrating the marginal probabilities). ABC methods have been conceived with the aim of inferring posterior distributions by circumventing the use of the likelihood, in favor of exploiting the computational efficiency of modern simulation techniques by replacing calculation of the likelihood with a comparison between the observed data and simulated data. These approaches are also straightforwardly applied to ODE model of signaling networks.

Let θ be a parameter vector to be estimated. Given the prior distribution $p(\theta)$, the goal is to approximate the posterior distribution, $p(\theta|x) \propto f(x|\theta)p(\theta)$, where $f(x|\theta)$ is the likelihood of θ given the data x. ABC methods have the following generic form:

1. Sample a candidate parameter vector θ^* from some proposal distribution $p(\theta)$.
2. Simulate a data set x^* from the model described by a conditional probability distribution $f(x|\theta)$.
3. Compare the simulated data set, x^*, to the experimental data, x_0, using a distance function, d, and tolerance ε; if $d(x_0,x^*) \leq \varepsilon$, accept θ^*. The tolerance $\varepsilon \geq 0$ is the desired level of agreement between x_0 and x^*.

The output of an ABC algorithm is a sample of parameters from a distribution $p(\theta|d(x_0,x^*) \leq \varepsilon)$. If ε is sufficiently small then the distribution $p(\theta|d(x_0,x^*) \leq \varepsilon)$ will be a good approximation for the "true" posterior distribution, $p(\theta|x_0)$.

The most basic ABC algorithm outlined above is known as the ABC rejection algorithm; however, recently more sophisticated and computationally efficient ABC methods have been developed. They are based on Markov Chain Monte Carlo (ABC MCMC) and Sequential Monte Carlo (ABC SMC) techniques (54, 55), respectively. They have recently been applied to dynamical systems modeled by ODEs and stochastic master equations; ABC SMC has been developed for approximating the posterior distribution of the model parameters and for model selection using Bayes factors (56). In the next section we illustrate the use of ABC SMC for parameter estimation and model selection in the context of the JAK-STAT signaling pathway.

5. Application to JAK-STAT Signaling Pathway

The JAK-STAT signaling pathway is involved in signaling through several surface receptors and STAT proteins, which act as signal transducers and activators of transcription (57, 58). Here we look at models of signaling though the erythropoietin receptor (EpoR), transduced by STAT5 (Fig. 1). Signaling through this receptor is crucial for proliferation, differentiation, and survival of erythroid progenitor cells (59).

When the Epo hormone binds to the EpoR receptor, the receptor's cytoplasmic domain becomes phosphorylated, which creates a docking site for signaling molecules, in particular the transcription factor STAT5. Upon binding to the activated receptor, STAT5 first becomes phosphorylated, then dimerizes and translocates to the nucleus, where it acts as a transcription factor.

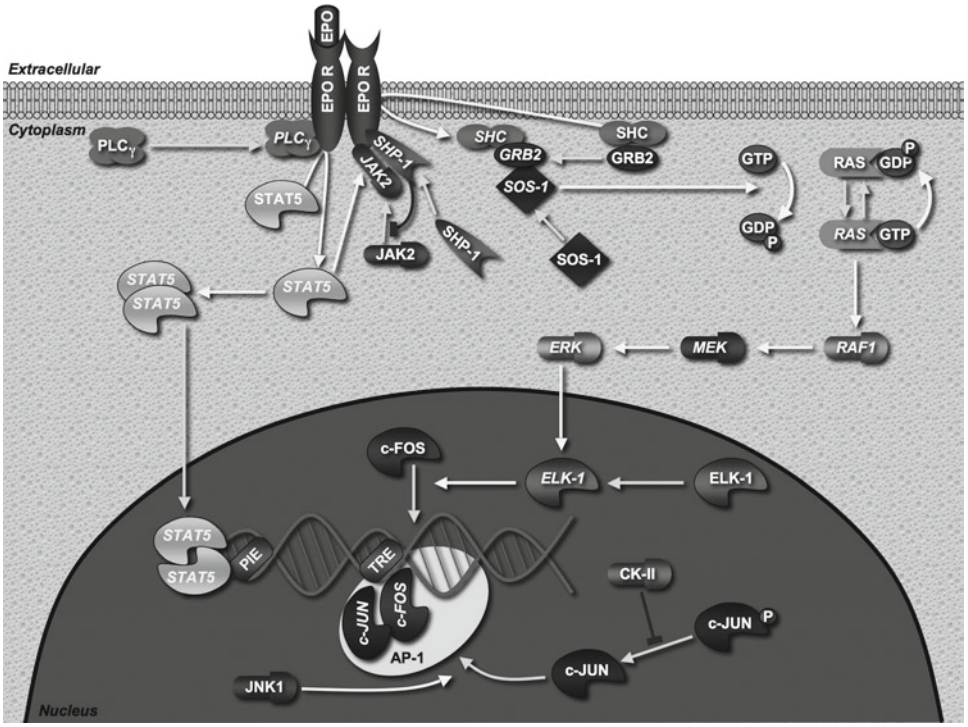


Fig. 1. STAT5 signaling pathway. Adapted from Biocarta.

There have been competing hypotheses about what happens with STAT proteins at the end of the signaling pathway. Originally it had been suggested that STAT proteins get degraded in the nucleus in an ubiquitin-asssociated way (60), while other evidence suggests that they are dephosphorylated in the nucleus and then transported back to the cytoplasm (61).

Here we want to understand how STAT5 protein transduces the signal from the receptor in the membrane through the cytoplasm into the nucleus. We have approached this problem by applying ABC SMC for model selection and parameter estimation to data collected for the JAK-STAT signaling pathway. The most suitable model from model of a STAT5 part of the JAK-STAT signaling pathway among the three proposed models was chosen and parameters have been estimated.

The ambiguity about the shutoff mechanism of STAT5 in the nucleus triggered the development of several mathematical models (16, 48, 62), each describing a different hypothesis. These models were then fitted to experimental data and systematically compared to each other using statistical methods of model selection. The model selection procedure ruled in favor of a cycling model, where STAT5 reenters the cytoplasm.

Timmer et al. (16, 48, 62) developed a continuous mathematical model for STAT5 signaling pathway, comprising of four

differential equations. They assume mass action kinetics and denote the amount of activated Epo-receptors by $\mathrm{EpoR_A}$, monomeric unphosphorylated STAT5 molecules by x_1, monomeric phosphorylated STAT5 molecules by x_2, dimeric phosphorylated STAT5 in the cytoplasm by x_3, and dimeric phosphorylated STAT5 in the nucleus by x_4. The most basic model Timmer et al. developed, under the assumption that phosphorylated STAT5 does not leave the nucleus, consists of the following kinetic equations:

$$\begin{aligned}\dot{x}_1 &= -k_1 x_1 \mathrm{EpoR_A} \\ \dot{x}_2 &= -k_2 x_2^2 + k_1 x_1 \mathrm{EpoR_A} \\ \dot{x}_3 &= -k_3 x_3 + \frac{1}{2} k_2 x_2^2\end{aligned} \tag{1}$$

$$\dot{x}_4 = k_3 x_3 \tag{2}$$

One can then assume that phosphorylated STAT5 de-dimerizes and leaves the nucleus and this can be modeled by adding appropriate kinetic terms to Eqs. 1 and 2 of the basic model:

$$\dot{x}_1 = -k_1 x_1 \mathrm{EpoR_A} + 2k_4 x_4$$

$$\dot{x}_4 = k_3 x_3 - k_4 x_4$$

Timmer et al. develop their cycling model further by assuming a delay in moving of STAT5 out of the nucleus. They write ODE equations for x_1 and x_4 for this model as

$$\dot{x}_1 = -k_1 x_1 \mathrm{EpoR_A} + 2k_4 x_3(t-\tau) \tag{3}$$

$$\dot{x}_4 = k_3 x_3 - k_4 x_3(t-\tau) \tag{4}$$

while equations for x_2 and x_3 remain as above. The outcome of their statistical analysis is that this model fits the data best, which leads them to the conclusion that this is the most appropriate model.

Instead of Timmer's chosen model, we propose a similar model with clear physical interpretation. Instead of $x_3(t-\tau)$, we propose to model the delay of phosphorylated STAT5 x_4 in the nucleus with $x_4(t-\tau)$:

$$\dot{x}_1 = -k_1 x_1 \mathrm{EpoR_A} + 2k_4 x_4(t-\tau)$$

$$\dot{x}_4 = k_3 x_3 - k_4 x_4(t-\tau)$$

We have performed the ABC SMC model selection algorithm (56) on the following models: (1) Cycling delay model with

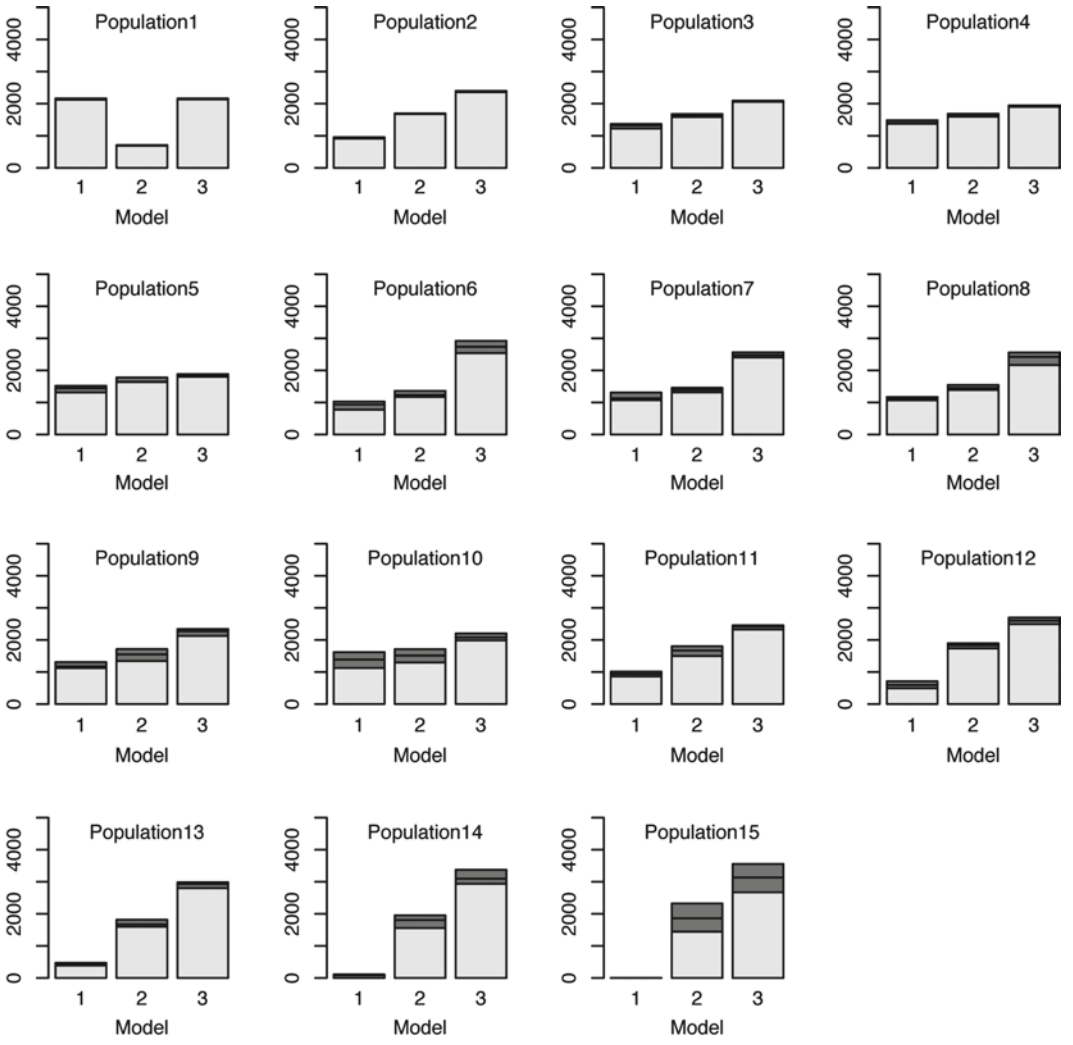


Fig. 2. Histograms show populations of the model parameter *m*. Population 13 represents the approximation of the marginal posterior distribution of *m*. The *dark shaded sections* present 25% and 75% quantiles around the median.

$x_3(t-\tau)$, (2) Cycling delay model with $x_4(t-\tau)$, and (3) Cycling model without a delay. The model parameter m can therefore take values 1, 2, and 3.

Figure 2 shows intermediate populations leading to the approximation of the marginal posterior distribution of m (population 13). Bayes factors can be calculated from the last population and according to the conventional interpretation of Bayes factors (51), it can be concluded that there is very strong evidence in favor of models 2 and 3 compared to model 1. However, there is only weak evidence for model 3 being more suitable than model 2.

6. Discussion

Modeling biological signaling or regulatory systems requires reliable parameter estimates. But the experimental dissection of signaling pathways is costly and laborious; it furthermore seems unreasonable to believe that the same set of parameters describes a system across all possible environmental, physiological, and developmental conditions. We are therefore reliant on efficient and reliable statistical and computational methods in order to estimate parameters and, more generally, reverse engineer mechanistic models.

As we have argued above, any such estimate must include a meaningful measure of uncertainty. A rational approach to modeling such systems should furthermore allow for the comparison of competing models in light of available data. The relative new ABC approaches are able to meet both objectives. Furthermore, as we have shown elsewhere they are not limited to deterministic modeling approaches but are also readily applied to explicitly stochastic dynamics; in fact it is possible to compare the explanatory power of deterministic and stochastic dynamics in the same mechanistic model.

One of the principal reasons for applying sound inferential procedures in the context of dynamical systems is to get a realistic appraisal of the robustness of these systems. If, as has been claimed, only a small set of parameters determines the system outputs then we have to ascertain these with certainty. It is here, in the reverse engineering of potentially sloppy dynamical systems, where the Bayesian perspective may be most beneficial.

References

1. Klipp, E. and Liebermeister, W. (2006) Mathematical modeling of intracellular signaling pathways. *BMC Neurosci.* **7** (Suppl 1), S10.
2. Neves, S. and Iyengar, R. (2002) Modeling of signaling networks. *BioEssays.* **24**, 1110–1117.
3. Levchenko, A. (2003) Dynamical and integrative cell signaling: challenges for the new biology. *Biotechnol Bioeng.* **84**, 773–782.
4. Cho, K. and Wolkenhauer, O. (2003) Analysis and modeling of signal transduction pathways in systems biology. *Biochem Soc Trans.* **31**, 1503–1509.
5. Papin, J., Hunter, T., Palsson, B., and Subramaniam, S. (2005) Reconstruction of cellular signaling networks and analysis of their properties. *Nat Rev Mol Cell Biol.* **6**, 99–111.
6. Fujioka, A., Terai, K., Itoh, R.E., Aoki, K., Nakamura, T., Kuroda, S., Nishida, E., and Matsuda, M. (2006) Dynamics of the Ras/ERK MAPK cascade as monitored by fluorescent probes. *J Biol Chem.* **281**, 8917–8926.
7. Apgar, J.F., Toettcher, J.E., Endy, D., White, F.M., and Tidor, B. (2008) Stimulus design for model selection and validation in cell signaling. *PLoS Comput Biol.* **4**, e30.
8. Markevich, N.I., Hoek, J.B., and Kholodenko, B.N. (2004) Signaling switches and bistability arising from multisite phosphorylation in protein kinase cascades. *J Cell Biol.* **164**, 353–359.
9. Babu, C., Yoon, S., Nam, H., and Yoo, Y. (2004) Simulation and sensitivity analysis of phosphorylation of EGFR signal transduction pathway in PC12 cell model. *Syst Biol.* **1**, 213–221.

10. Babu, C.S., Song, E.J., and Yoon, Y. (2006) Modeling and simulation in signal transduction pathways: a systems biology approach. *Biochimie*. **88**, 277–283.
11. Conzelmann, H., Saez-Rodriguez, J., and Sauter, T. (2004) Reduction of mathematical models of signal transduction networks: simulation-based approach applied to EGF receptor signalling. *Syst Biol.* **1**, 159–169.
12. Kolch, W., Calder, M., and Gilbert, D. (2005) When kinases meet mathematics: the systems biology of MAPK signalling. *FEBS Lett.* **579**, 1891–1895.
13. Andrec, M., Kholodenko, B., Levy, R., and Sontag, E. (2005) Inference of signaling and gene regulatory networks by steady-state perturbation experiments: structure and accuracy. *J Theor Biol.* **232**, 427–441.
14. Schoeberl, B., Eichler-Jonsson, C., Gilles, E., and Müller, G. (2002) Computational modeling of the dynamics of the MAP kinase cascade activated by surface and internalized EGF receptors. *Nat Biotechnol.* **20**, 370–375.
15. Aaronson, D. and Horvath, C. (2002) A road map for those who don't know JAK-STAT. *Science*. **296**, 1653.
16. Swameye, I., Muller, T.G., Timmer, J., Sandra, O., and Klingmuller, U. (2003) Identification of nucleocytoplasmic cycling as a remote sensor in cellular signaling by data-based modeling. *Proc Natl Acad Sci USA*. **100**, 1028–1033.
17. Balsa-Canto, E., Peifer, M., Banga, J.R., Timmer, J., and Fleck, C. (2008) Hybrid optimization method with general switching strategy for parameter estimation. *BMC Syst Biol.* **2**, 26.
18. Modchang, C., Triampo, W., and Lenbury, Y. (2008) Mathematical modeling and application of genetic algorithm to parameter estimation in signal transduction: trafficking and promiscuous coupling of G-protein coupled receptors. *Comput Biol Med.* **38**, 574–582.
19. Yue, H., Brown, M., Knowles, J., Wang, H., Broomhead, D.S., and Kell, D.B. (2006) Insights into the behaviour of systems biology models from dynamic sensitivity and identifiability analysis: a case study of an NF-kappaB signalling pathway. *Mol Biosyst.* **2**, 640–649.
20. Schwartz, M.A. and Baron, V. (1999) Interactions between mitogenic stimuli, or, a thousand and one connections. *Curr Opin Cell Biol.* **11**, 197–202.
21. Tyson, J., Chen, K., and Novak, B. (2003) Sniffers, buzzers, toggles and blinkers: dynamics of regulatory and signaling pathways in the cell. *Curr Opin Cell Biol.* **15**, 221–231.
22. Bhalla, U.S. and Iyengar, R. (1999) Emergent properties of networks of biological signaling pathways. *Science*. **283**, 381–387.
23. Heinrich, R., Neel, B., and Rapoport, T. (2002) Mathematical models of protein kinase signal transduction. *Mol Cell.* **9**, 957–970.
24. Saez-Rodriguez, J., Kremling, A., and Conzelmann, H. (2004) Modular analysis of signal transduction networks. *Control Syst Mag.* **24**, 35–52.
25. Vera, J., Bachmann, J., Pfeifer, A., Becker, V., Hormiga, J., Darias, N., Timmer, J., Klingmüller, U., and Wolkenhauer, O. (2008) A systems biology approach to analyse amplification in the JAK2-STAT5 signalling pathway. *BMC Syst Biol.* **2**, 38.
26. Burnham, K. and Anderson, D. (2002) Model selection and multimodel inference: a practical information-theoretic approach. Springer, New York.
27. Peifer, M. and Timmer, J. (2007) Parameter estimation in ordinary differential equations for biochemical processes using the method of multiple shooting. *IET Syst Biol.* **1**, 78–88.
28. Brewer, D., Barenco, M., Callard, R., Hubank, M., and Stark, J. (2007) Fitting ordinary differential equations to short time course data. *Philos Transact A Math Phys Eng Sci*. **366**, 519–544.
29. Moles, C., Mendes, P., and Banga, J. (2003) Parameter estimation in biochemical pathways: a comparison of global optimization methods. *Genome Res.* **13**, 2467–2674.
30. Runarsson, T. and Yao, X. (2000) Stochastic ranking for constrained evolutionary optimization. *IEEE Trans Evol Comput.* **4** ,284–294.
31. Ji, X. and Xu, Y. (2006) libSRES: a C library for stochastic ranking evolution strategy for parameter estimation. *Bioinformatics.* **22**, 124–126.
32. Zi, Z. and Klipp, E. (2006) SBML-PET: a Systems Biology Markup Language-based parameter estimation tool. *Bioinformatics.* **22**, 2704–2705.
33. Rodriguez-Fernandez, M., Mendes, P., and Banga, J. (2006) A hybrid approach for efficient and robust parameter estimation in biochemical pathways. *Biosystems.* **83**, 248–265.
34. Kirkpatrick, S., Gelatt, C., and Vecchi, M. (1983) Optimization by simulated annealing. *Science*. **220**, 671–680.
35. Brown, K.S. and Sethna, J.P. (2003) Statistical mechanical approaches to models with many poorly known parameters. *Phys Rev E.* **68**, 021904.
36. Brown, K.S., Hill, C.C., Calero, G.A., Myers, C.R., Lee, K.H., Sethna, J.P., and Cerione,

R.A. (2004) The statistical mechanics of complex signaling networks: nerve growth factor signaling. *Phys Biol.* **1**, 184–195.

37. Vyshemirsky, V. and Girolami, M.A. (2008) Bayesian ranking of biochemical system models. *Bioinformatics.* **24**, 833–839.
38. Vyshemirsky, V. and Girolami, M. (2008) BioBayes: a software package for Bayesian inference in systems biology. *Bioinformatics.* **24**, 1933–1934.
39. Quach, M., Brunel, N., and d'Alche Buc, F. (2007) Estimating parameters and hidden variables in non-linear state-space models based on odes for biological networks inference. *Bioinformatics.* **23**, 3209–3216.
40. Bard, Y. (1974) Nonlinear parameter estimation. Academic Press, New York.
41. Venzon, D. and Moolgavkar, S. (1988) A method for computing profile-likelihood-based confidence intervals. *Appl Stat.* **37**, 87–94.
42. Efron, B. and Tibshirani, R. (1993) An introduction to the bootstrap. CRC Press, Boca Raton, FL.
43. Hengl, S., Kreutz, C., Timmer, J., and Maiwald, T. (2007) Data-based identifiability analysis of non-linear dynamical models. *Bioinformatics.* **23**, 2612–2618.
44. Schmidt, H., Madsen, M.F., Danø, S., and Cedersund, G. (2008) Complexity reduction of biochemical rate expressions. *Bioinformatics.* **24**, 848–854.
45. Saltelli, A., Ratto, M., Andres, T., Campolongo, F., Cariboni, J., Gatelli, D., Saisana, M., and Tarantola, S. (2008) Global sensitivity analysis: the primer. John Wiley and Sons, England.
46. Gutenkunst, R., Waterfall, J., Casey, F., Brown, K., Myers, C., and Sethna, J. (2007) Universally sloppy parameter sensitivities in systems biology models. *PLoS Comput Biol.* **3**, e189.
47. Wei, J. (1975) Least squares fitting of an elephant. *Chem Tech.* **5**, 128–129.
48. Timmer, J. and Muller, T. (2004) Modeling the nonlinear dynamics of cellular signal transduction. *Int J Bifurcat Chaos.* **14**, 2069–2079.
49. Akaike, H. (1973) Information theory as an extension of the maximum likelihood principle. Second International Symposium on Information Theory, Akademiai Kiado, Budapest, 267–228.
50. Akaike, H. (1974) A new look at the statistical model identification. *Automat Contr.* **19**, 716–723.
51. Kass, R. and Raftery, A. (1995) Bayes factors. *J Am Stat Assoc.* **90**, 773–795.
52. Schwarz, G. (1978) Estimating the dimension of a model. *Ann Stat.* **6**, 461–464.
53. Cox, R.D. and Hinkley, D.V. (1974) Theoretical statistics. Chapman & Hall/CRC, New York.
54. Marjoram, P., Molitor, J., Plagnol, V., and Tavare, S. (2003) Markov chain Monte Carlo without likelihoods. *Proc Natl Acad Sci USA.* **100**, 15324–15328.
55. Sisson, S.A., Fan, Y., and Tanaka, M.M. (2007) Sequential Monte Carlo without likelihoods. *Proc Natl Acad Sci USA.* **104**, 1760–1765.
56. Toni, T., Welch, D., Strelkowa, N., Ipsen, A., and Stumpf, M. (2009) Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems. *J Roy Soc Interface.* **6**, 187–202.
57. Darnell, J.E. (1997) STATs and gene regulation. *Science.* **277**, 1630–1635.
58. Horvath, C.M. (2000) STAT proteins and transcriptional responses to extracellular signals. *Trends Biochem Sci.* **25**, 496–502.
59. Klingmuller, U., Bergelson, S., Hsiao, J.G., and Lodish, H.F. (1996) Multiple tyrosine residues in the cytosolic domain of the erythropoietin receptor promote activation of STAT5. *Proc Natl Acad Sci USA.* **93**, 8324–8328.
60. Kim, T.K. and Maniatis, T. (1996) Regulation of interferon-gamma-activated STAT1 by the ubiquitin-proteasome pathway. *Science.* **273**, 1717–1719.
61. Köster, M. and Hauser, H. (1999) Dynamic redistribution of STAT1 protein in IFN signaling visualized by GFP fusion proteins. *Eur J Biochem.* **260**, 137–144.
62. Muller, T.G., Faller, D., Timmer, J., Swameye, I., Sandra, O., and Klingmüller, U. (2004) Tests for cycling in a signalling pathway. *J R Stat Soc Ser C.* **53**, 557.

Chapter 19

Genetic Algorithms and Their Application to *In Silico* Evolution of Genetic Regulatory Networks

Johannes F. Knabe, Katja Wegner, Chrystopher L. Nehaniv, and Maria J. Schilstra

Abstract

A genetic algorithm (GA) is a procedure that mimics processes occurring in Darwinian evolution to solve computational problems. A GA introduces variation through "mutation" and "recombination" in a "population" of possible solutions to a problem, encoded as strings of characters in "genomes," and allows this population to evolve, using selection procedures that favor the gradual enrichment of the gene pool with the genomes of the "fitter" individuals. GAs are particularly suitable for optimization problems in which an effective system design or set of parameter values is sought.

In nature, genetic regulatory networks (GRNs) form the basic control layer in the regulation of gene expression levels. GRNs are composed of regulatory interactions between genes and their gene products, and are, inter alia, at the basis of the development of single fertilized cells into fully grown organisms. This paper describes how GAs may be applied to find functional regulatory schemes and parameter values for models that capture the fundamental GRN characteristics. The central ideas behind evolutionary computation and GRN modeling, and the considerations in GA design and use are discussed, and illustrated with an extended example. In this example, a GRN-like controller is sought for a developmental system based on Lewis Wolpert's French flag model for positional specification, in which cells in a growing embryo secrete and detect morphogens to attain a specific spatial pattern of cellular differentiation.

Key words: Evolutionary computation, Genetic algorithm, Genetic regulatory network, Modeling, Simulation, Gene regulation logic, Developmental program

Abbreviations

CPM	Cellular Potts model
CRM	*Cis*-regulatory module
EC	Evolutionary computing
GA	Genetic algorithm
PR	Gene product (protein or RNA)
GRN	Genetic regulatory network
TF	*Trans*-regulatory factor

David Fenyö (ed.), *Computational Biology*, Methods in Molecular Biology, vol. 673,
DOI 10.1007/978-1-60761-842-3_19, © Springer Science+Business Media, LLC 2010

1. Introduction

With the sequencing of the human genome, and the inventory of its total protein-coding gene content, a full understanding of the programs controlling metabolism, development, and adaptation may seem to be within reach. However, over the years it has become increasingly clear that regulation of gene expression is achieved through highly complex dynamic interaction networks, often called genetic regulatory networks (GRNs). There is now a large amount of information available about gene expression levels in different cells, tissues, and organisms, at different stages of development, response, and adaptation. Nonetheless, unraveling the intricate structure of the GRNs that underlie these processes is difficult, and requires the integration of a great number of experimental and computational resources. Techniques are being developed to pinpoint and quantitatively describe the interactions – of so-called *cis*-regulatory sites with *trans*-regulatory factors (TF) – that lead to enhanced or reduced gene expression. In addition, network structures can, at least partially, be inferred using sophisticated statistical techniques that detect correlations and dependencies in gene expression levels. On the basis of the progress already made in these fields, it is widely expected that one day it will be possible to use the knowledge about the basic interactions, constrained by information obtained with network inference techniques, to automatically create mathematical GRN models that not only show similar dynamics and responses, but also have the same network structure as the intracellular GRNs that they aim to describe. Procedures for automatic or computer-aided dynamic GRN generation will almost certainly make use of computational evolutionary techniques to search for networks that exhibit the intended behavior. It is likely that computational tools will be using a limited number of predefined dynamic components that describe processes such as transcription, translation, and transport, and subnetworks such as signaling cascades. The greatest challenges in the development of such tools lie, in the first instance, in deciding what the nature and characteristics of these predefined components should be, and in collecting sufficient information to be able to describe their dynamics in adequate detail.

The purpose of this chapter is to give readers an impression of what evolutionary computation (EC) is, and what its advantages and limitations are, and how it may be used. We illustrate how EC techniques, specifically a class of techniques called genetic algorithms, can be used to generate GRN models that show a particular, relatively complex behavior on the basis of an example taken from our own investigations into the potential of GRN-like structures as control systems. An in-depth discussion of the requirements for quantitative modeling and simulation of biological GRNs is beyond the scope of this chapter, and we will focus mostly on "conceptual"

or artificial GRNs, mathematical models that qualitatively capture the most distinctive features of biological GRNs.

1.1. Evolutionary Computation and Genetic Algorithms

Relatively early in the history of modern computation, engineers had the idea to harness the power of evolution for optimization purposes. After all, evolution, driven by the related mechanisms of *genetic drift* and *natural selection*, has led to the huge diversity of well-adapted species we see on earth nowadays. Evolution, it is argued, is a massively parallel search that tests the ability of millions of populations to adapt to their habitats.

In its most general definition, biological evolution is "change in the gene pool of a population from one generation to the next." A *gene* is a hereditary, functional unit that can be passed on unaltered for many generations, and a *gene pool* is the collection of all genes in a *population* – here, a group of individuals that are able to interbreed and produce fertile offspring (see Note 1). Genetic drift is the accumulation of random change in the gene pool of a population over time. Each individual member of a population has its own, unique *genotype*, the full set of genes in its *genome*. Genetic variation within a population is brought about by mutation, heritable change in the genotype of a single individual, or recombination, the reshuffling, and exchange of genetic material between two individuals. An individual's ability to adapt to its environment is determined by its *phenotype*, the physical expression of its genotype in that environment. While mutation and recombination increase the diversity within a population, natural selection makes it decrease. "Fitter" individuals – those whose phenotype is best adapted to the environment – are the ones that, on average, produce more offspring. Greater fitness may be due to greater *viability*, the probability to survive to a given age, or to increased *fertility*, which is related to the total number of offspring produced during a lifetime. In a population whose size remains the same, traits that make individuals fitter will accumulate as the population evolves over many generations, while the less favorable ones will disappear.

The *evolutionary cycle*, in which a number of individuals are selected from a population on the basis of their fitness in step 1, to produce offspring that receives, in step 2, a combination of the – possibly mutated – parental genomes, and where, in step 3, the new generation replaces the whole previous generation in the population, is depicted in Fig. 1. This simple cycle of selection, variation, and replacement forms the basis of all evolutionary computation. Evolutionary computation (EC) has been found particularly appropriate for solving multidimensional problems whose parameter space is too large to search exhaustively or too complex to investigate systematically. By probabilistically searching and improving a "population" of "candidate solutions" – rather than trying to find a single analytical solution, or applying deterministic search rules – the algorithms used in EC often find good, albeit not necessarily the best, solutions. As with all such global search heuristics, EC

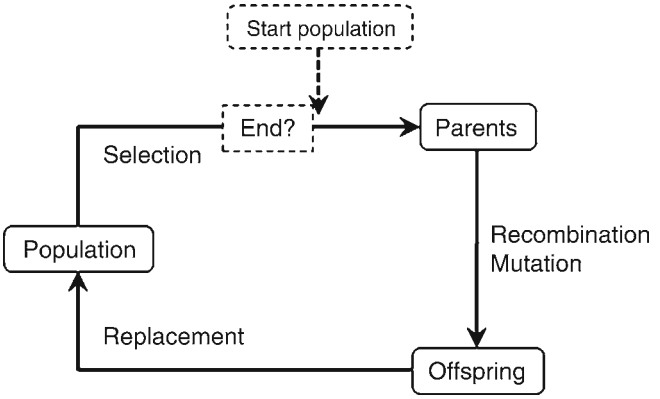


Fig. 1. The evolutionary cycle, consisting of parent "selection" (fitter parents produce, on average, more offspring), introduction of genetic variation in the offspring through cross-over and mutation mechanisms, and replacement of the parent generation with their offspring, who will act as parents in the next round. In Evolutionary Computation, an evolutionary run must be initiated with a starting population, and is terminated when a predefined fitness or time criterion is met. The population size is kept constant throughout the full evolutionary run.

techniques are better suited to certain types of problems than to others, and are certainly not guaranteed to solve any problem optimally. However, because they have fewer restrictions, evolutionary algorithms are, on the whole, applicable to a wider range of problems than many other optimization techniques.

Genetic algorithms (GAs) form a particular class of EC techniques (see Note 2), first developed in the 1960s by John Holland in an attempt to use natural adaptation mechanisms in computer science (1, 2). GAs use selection methods and so-called genetic operators, such as cross-over, mutation, and inversion, to slowly change the information content in a population of "genomes." Using a set of rules that is different for different problems, the information in each genome is translated into a "phenotype." The characteristics of each phenotype are then compared with a set of desired characteristics, and assigned a "fitness value" on the basis of the comparison. The selection process is set to favor genomes that produce fitter phenotypes, allowing the fitter individuals to produce most of the offspring.

Thus, GAs employ procedures that simulate, in a highly simplified fashion, the processes that operate in biological evolution. The assignment of a fitness value to a genotype on the basis of a comparison with a target behavior is clearly unrealistic, as in nature there are no targets, just survival and reproduction probabilities in a given environment. However, the primary purpose of a GA is not to accurately simulate biological evolution, but to solve computational problems. In that context, GAs are often capable of providing satisfactory solutions to complex optimization problems that cannot so easily be solved by other, more formal methods. For further reading we recommend *An Introduction to Genetic Algorithms* by M.

Mitchell (3), and the two volumes in the series *Evolutionary Algorithms*, edited by Back, Fogel, and Michalewicz (4).

1.2. Genetic Regulatory Networks

In its most basic definition, a GRN is a collection of regulatory interactions between genes and gene products. A gene product (PR) (see Note 3) is a macromolecule that has been constructed on the basis of the information present in the coding region of the gene, and may be a polyribonucleotide (RNA) or a polypeptide (protein). A gene is said to be *expressed* in a cell when its gene products can be found in that cell. Some genes are constitutively expressed, and their PRs always present. Most genes, however, are conditionally expressed: they are only activated under certain conditions. In spite of the fact that all cells that form an organism have the same genome, expression levels within a single organism often vary widely from tissue to tissue, from cell to cell, and over time. This differential gene expression may originate in variations in certain extracellular conditions, or in variable detection of, or response to the external conditions, but can also be caused by asymmetries in the distribution of certain key molecules after cell division. Thus, the PRs participating in the cell signaling apparatus that detects these differences in conditions form the "input" to GRNs, whereas the PRs that contribute to the cell structure and metabolism are the "output."

Many PRs are involved in the activation or repression of the expression of one or more genes: they are said to function as TFs (see Note 4) to those genes. Thus, the statement "gene A regulates gene B" in actual fact means that the PR of gene A is involved as a TF in the regulation of the expression of gene B. Regulation is mediated through so-called *cis-regulatory* TF-binding sites. Physically, a *cis*-regulatory binding site is a stretch of DNA that has a high affinity for a specific TF or TF complex. If TF molecules are present in a given cell over a certain period of time (i.e., their genes are being expressed), they will bind to their *cis*-regulatory binding sites for at least part of that time, during which they act to increase or decrease the gene *transcription rate*. A gene may have many *cis*-regulatory sites that bind as many different TFs, each of which may affect the transcription rate differently. TFs often act synergistically, with their combined effect very different from the sum of their individual effects. A simple example of synergy is a situation in which only the complex of two different TFs (a hetero-dimer) is able to repress transcription, whereas either of the constituents of the complex have no effect. In that case, repression will only occur under conditions in which both TFs – which, as the products of different genes, may require different sets of conditions for their own expression – are being expressed. Because the expression of a single gene may be regulated by many different TFs, and a particular PR may take part in the regulation of multiple genes including its own, the connection patterns in GRNs are often highly convoluted, and feedback loops are in abundance.

Mature PRs are created in a series of consecutive transformations of the primary transcript that include chemical modification (carried out by enzymes, a class of proteins that catalyze chemical reactions), transport from the nucleus to other parts in the cell, and, for proteins, translation of the information in the mRNA into a polypeptide chain, followed by posttranslational modification into a mature protein. Activation of mature PRs may require further modification or complex formation. Furthermore, mature RNA and protein molecules are often actively broken down as soon as they are somehow surplus to requirement (see Note 5). Modification, translation, transport, and breakdown are all as tightly controlled as gene transcription. In the following, however, we shall focus on the regulation of transcription, not only because it forms the basis for all higher-level regulation, but also because it can be highly adaptive, and has the ability to integrate the information contained in a large amount of incoming data.

All GRN models include components that fulfill the roles of genes and TFs in biological GRNs. In their most abstracted form, GRNs are simply represented by "directed graphs": sets of nodes (boxes or circles) connected by arrows. The nodes represent genes, and the arrows indicate control: the arrowhead points at the gene whose expression is being regulated by the PR of the gene connected to the other end of the arrow. The arrows may have weights that express the effectiveness of a PR as a TF.

In more elaborate models, PRs, their precursors, and the various processes that contribute to changes in their levels may have individual representations. Although no one way of representing GRNs is "the best," the inclusion or omission of detail will certainly make particular models more fit for purpose than others.

Graphical representations such as the ones described above specify which processes can and cannot occur. However, the response of a network to an incoming signal can only be assessed on the basis of rules that describe how rapidly the value of a node changes when the values of other system components change. In their most simplified form, GRNs are modeled as Boolean networks, in which the genes are either "on" or "off." Boolean network models usually let a full response to changed conditions take place in a single, discrete time step. More complex dynamic models may include delays and fractional or exponential response mechanisms, and PR levels are allowed to assume (positive) values on discrete or continuous scales. Most models define a functional dependency of the rate of PR production on the amount of TFs present, so that increasing the amount of an activator, or decreasing the amount of a repressor increases the rate of PR production. In order to prevent unlimited growth, a PR breakdown rate or an upper level must be specified. Synergistic effects may be modeled in different ways, some of which are outlined in (5).

The parameters associated with the delayed, fractional, or exponential response determine the rate at which these networks can adapt to changed conditions. Time may be modeled discretely or continuously. To simulate the responses of a network over time, discrete time models require solution of a set of difference equations, whereas continuous-time models translate to sets of differential equations that are numerically integrated. Stochasticity (randomness) may be introduced in various ways in both types of models; however, introduction of stochasticity tends to be computationally costly, and, for highly abstracted models, does not usually yield a more accurate reflection of reality.

Again, no particular modeling framework is necessarily better or more realistic than any of the others. The mechanisms that operate within a biological cell or organism are so diverse and complex, and the models so abstract, that it is possible to find some "biological justification" for most. Altogether, there are far more ways of modeling GRNs than can be evaluated here. Many excellent reviews have appeared over the last decade, some discussing specific systems, some examining specific modeling techniques, and others attempting to evaluate the whole (6–28). An example of how the above concepts and considerations can be incorporated in a dynamic model is presented in Subheading 2.2.

1.3. Running Example: Evolving a GRN-Like Control System for Cellular Differentiation into a French Flag Pattern

Throughout this chapter, we shall illustrate the most important concepts in the field of evolutionary computation and GAs on the basis of an example application in which a GRN is evolved that controls cell division and gradient formation in a system based on Lewis Wolpert's famous French flag model of positional specification in the embryo ((29, 30) and references therein; see also (31)). Wolpert proposed this model in 1969 to explain a mechanism for retrieving positional information by arrays of cells in an embryo or tissue. The model is based on the assumption that, during early development, gradients of chemical signals called morphogens build up across the growing embryo. The cells in the embryo detect the morphogen concentration and compare it with certain built-in thresholds, to decide in which domain of the embryo – here, in the red, white, or blue section of the French flag – they are located. Since this model was first proposed, several variations of have been explored, e.g. (32, 33). Although these models differ in detail, all include a 2D spatial arrangement of cells that can detect the concentration of morphogen in an underlying stratum. In our version, all cells can also produce this morphogen, and secrete it into the stratum. Gradients form when the morphogen diffuses away from the cells that secreted it, and eventually decays. Cells only "communicate" through these morphogen gradients; there is no other exchange of information between cells. Each cell has its own independent control unit that regulates, inter alia, its morphogen secretion rate, and determines its color.

These control units are instances of a GRN, and have the same components and connectivity in each cell.

The task for the GA in this example is to find a GRN that controls cellular morphogen secretion in such a way that a gradient forms along the long axis of the flag-shaped environment as the cells in the embryo multiply and eventually fill the whole flag. The gradient has to be steep enough to allow the cells to determine whether its level at their position is above, in between, or below two separate thresholds, in order for them to adopt a blue, white, or red color. More information on the development of this GRN-controlled French flag system, including a detailed description and discussion of the results, is found in (34).

2. Materials

2.1. Cellular Model

To mimic the growth and multiplication of cells in a developing embryo, we used an existing implementation of the so-called cellular Potts model (CPM). The CPM was originally developed by Glazier and Graner to simulate differential adhesion-driven cell arrangement (35), but has been re-used for a variety of cell-level modeling tasks, recently reviewed by Merks and Glazier (36). The CPM has two layers, both consisting of pixel lattices (here of 60×40 pixels each). In the top layer, "cells" are represented as domains of adjacent pixels. Every pixel is associated with an integer number that identifies the cell to which it belongs, with zero indicating that it does not belong to any cell. The system advances in time steps in which each pixel attempts once, on average, to copy its value into a randomly chosen neighboring pixel. Copying success is limited by so-called effective energy constraints: each cell has an ideal size (number of pixels in the cell) at which its energy content is minimal, and deviation from the ideal incurs an energy penalty. A copying attempt will succeed in any case if the copying decreases the energy of the whole system. The copying event may even proceed if the overall energy is predicted to increase, but the probability of success decreases exponentially with increasing energy differences. Because of this, cells remain dynamic, even if their energy is close to minimal. Other cell properties that can be constrained, and therefore externally controlled, include shape (the ratio of its projections on the horizontal and vertical axes), its tendency to "stick" to neighboring cells, and the direction in which to divide. Importantly, all cells in our model have the ability to "secrete" one or more types of morphogen into the "stratum" in the bottom layer, and to read out its current level in each time step. The rate of secretion is open to external control. Each type of morphogen has fixed diffusion and decay

rates, and its distribution over time over the stratum is computed by numerically solving the partial differential equations that describe the diffusion, decay, and production (secretion) of the morphogen. We used a flexible open source CPM implementation called CompuCell3D (see http://CompuCell3D.org and (36) for implementation and formalism details), which allowed us to choose predefined constraints, and also to define new ones where necessary. CompuCell3D also facilitates the use of external control modules to control the various parameters that determine cell behavior.

2.2. GRN Model

The central components of the GRN control unit used to control the French Flag development are genes and gene products (PRs). The genes contain one or more *cis*-regulatory modules (CRMs), and each of these modules consists of one or more binding sites for TFs. Each gene produces one type of PR, and each type of PR can function as a TF in the expression of any gene (including its own) if at least one of the CRMs of that gene contains a binding site for that PR. A PR can function as a TF in the regulation of many genes, and a gene can have many TF-binding sites. A PR may have many multiple binding sites on one gene, even within a single CRM. Genes may be constitutive or facultative. Constitutive genes are "on" in the absence of their TFs, and need overall repression to reduce their expression level; facultative genes are "off" by default, and need to be activated by their TFs to produce their PRs. TFs that bind to the same CRM act together, whereas the CRMs make independent contributions to the gene expression level. These contributions may either be activating or inhibitory (repressive).

These effects are quantitatively implemented as follows. At any one point in time, each PR in each cell is associated with a number, n, that represents the amount of PR molecules in that cell. The TFs that bind to a single CRM do so (conceptually) as a tight complex (a hetero-oligomer), so that the TF with the smallest number molecules present determines the total amount of complex that can be formed (see Note 6). The contribution of one CRM to the overall expression level is taken to be proportional to the amount of complex formed. Activating CRMs make a positive, and inhibitory CRMs make a negative contribution. The sum S of the individual CRM contributions is then translated into a PR production rate ν^{p} between zero and a maximum rate (e.g., 150 PR molecules per time unit). The relationship between ν^{p} and S is sigmoid, with its inversion point below zero for constitutive, and above zero for facultative genes. Figure 2a shows a graphical representation of a single gene, and demonstrates how ν^{p} is calculated from the quantities of its TFs. Each PR also decays at a rate ν^{d} that is proportional to the amount n of PR present: $\nu^{d} = k^{d} \times n$. The proportionality constant k^{d} is different for each PR.

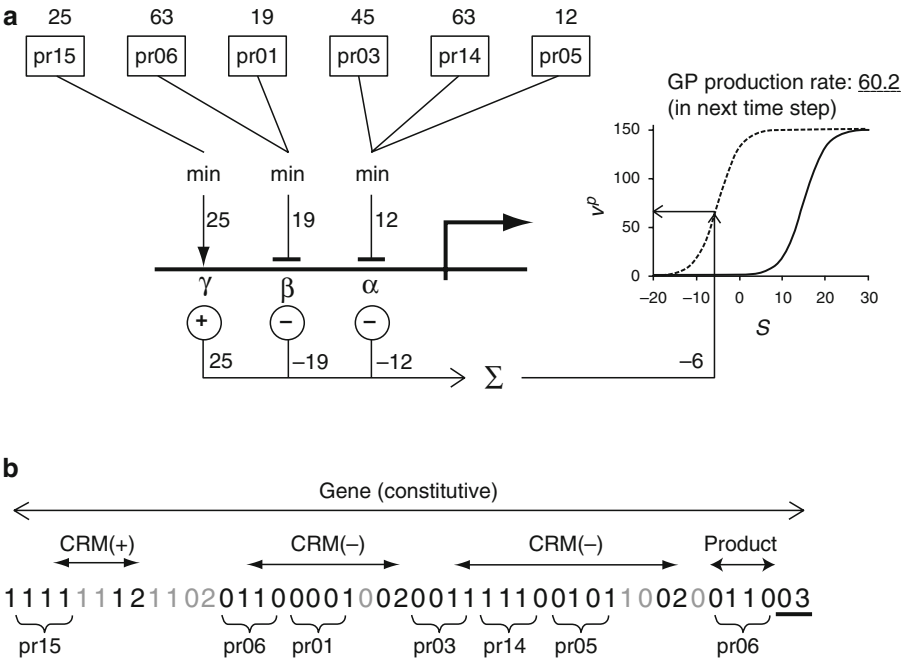


Fig. 2. Gene representation. (**a**) Dynamic model, illustrating how a PR production rate is computed from the TF quantities present in one cell, at one particular point in time. The gene depicted here (as a *horizontal line* crossed by a *bent arrow*) is constitutive, and has three CRMs, α, β, and γ, one of which (γ) is activating, and the other two inhibitory. The boxes indicate the names of the PRs that act as TFs to the gene, with their quantities shown in *bold* above the boxes. For each CRM, the minimum TF quantity is carried through, and negated for inhibitory CRMs. The sum *S* of these values is then used to compute v^p, the PR production rate. The relationship between *S* and v^p is sigmoid, as shown in the graphs to the right of the gene. The *broken line*, here with a midpoint at –5, represents $v^p(S)$ for constitutive genes; the *solid line* (midpoint at 15) that for facultative genes. (**b**) Gene encoding: genes are encoded in the genome using 3 as the gene delimiter, 2 as CRM delimiters, and zeros and ones immediately to the right of the delimiter to indicate a constitutive (13) or facultative (03) gene (*bold*, *underlined*), and an activating (12) or inhibitory (02) CRM (*bold*). The rest of the gene representation consists solely of zeros and ones, and represent the PR of the gene (the four characters immediately to the left of the 31 or 30), and the TFs (as many quadruplets to the left of each 21 and 20 that will fit). The residual characters (of which there may be zero or more in the PR area, and zero to three in the CRM areas), *indicated in gray*, do not have a function.

To simulate the response of the GRN in each cell to the changing conditions, time is divided in discrete steps Δt of unit size ($\Delta t = 1$). The quantity of the *i*th PR in the *j*th cell at the end of time step *t*, $n_{i,j}(t)$, is calculated by simply adding the number of PR molecules formed in time step *t* to the quantity $n_{i,j}(t-1)$ of the same PR in the previous time step, and subtracting the number of molecules that have disappeared, as shown in Eq. 1. This is done for all PRs in all cells.

$$n_{i,j}(t) = n_{i,j}(t-1) + (v^p_{i,j}(t) - v^{\mathrm{d}}_{i,j}(t)) \times \Delta t \qquad (1)$$

The GRN for the French Flag system uses 20 genes to produce a maximum of 16 different PRs (justified in Subheading 3.1). All PRs may be used as TFs (to regulate the expression of other genes). Furthermore, a total of 12 PRs have the following

predefined functions. For each cell, the only input into the GRN comes from the average concentration of two morphogens, m_1 and m_2, in the part of the stratum directly underneath the cell. This is done at the start of each time step by reading out, and setting the number of molecules of two particular PR to values proportional to m_1 and m_2. A further 10 PRs form the GRNs output, and act as controllers for the CPM. One PR controls the cell's target size: when there is a large amount present, the cell will try to grow until it reaches the size threshold for division (and then divides); if its quantity is zero, the cell will shrink, and eventually disappear (undergo "apoptosis"). Five more PRs set the cell's shape, stickiness, and division direction targets, essentially in the same way. Another two govern the morphogen production–secretion rate: again, the more PR, the greater the rate. Finally, two PRs determine the color of the cell: high quantities of both (compared with a set threshold) make the cell blue, one high and one low gives white, and two low quantities yields a red cell.

2.3. Simulation

In a simulation round, the CPM is combined with a "wired" (equipped with a particular connectivity pattern, as encoded in the genome; see Subheading 3.1) and parameterized GRN model. The simulation starts with a single cell, and therefore one copy of the GRM. After setting the PR quantities in that cell to their initial values (usually zero), the system is left to develop. In viable individuals, the cell will begin to grow as soon as the PR that controls the target size accumulates, and will divide when its threshold for division is reached. Upon division, the cell's PR content is distributed between the daughter cells in quantities proportional to their sizes, and each daughter receives an exact copy of the GRN. Because of the way the CPM is designed, cells cannot grow on top of each other, or outside the pixel lattice, so that the lattice fills up with cells over the course of the simulation. The fraction of the lattice that is covered by cells by the end of a simulation round (of 200 time steps, in our case) is dependent on the growth rate that is achieved by the system. Numerical integration of the rate equations for the GRN system, as expressed in Eq. 19.1, is deterministic: if a simulation is started with the same initial PR quantities in cells with equal and constant sizes, the PR quantities will develop equally in all cells. However, the CPM is a nondeterministic system: it uses a random number generator (RNG) to choose a neighboring pixel in each copying attempt (see Subheading 2.1). Provided they are carried out with a different sequence of random numbers, no two simulation rounds will be the same (see Note 7), even if they start under equal initial conditions.

2.4. Software

Genetic algorithm-aided design of artificial GRNs as controllers for specific system requires a combination of three elements:

(1) the device for which a controller is required (here the CPM), (2) a GRN modeling and simulation tool, and (3) a tool that applies the GA.

Obviously, the "device" is different for each application; it may be a virtual system (such as the CPM in our running example), but could equally well be a physical apparatus, a robot of some kind, or simply a table that lists the quantities of particular PRs during a simulation. Whatever the device, it must be possible to formulate a desired behavior for it, and to somehow quantify its accomplishment in comparison with the target.

Because there is no standard way of modeling GRNs, few off-the-shelf tools are available for this purpose. Some relatively mature tools that have been designed specifically for modeling and analyzing GRNs are the Genetic Network Analyser (GNA) (37), and GINsim (38), which use qualitative simulation methods for predicting gene expression in highly abstracted GRN models. Furthermore, there are many software packages that are suitable for modeling of biochemical reaction networks and dynamical systems in general (see (39) and (40) for examples), and in principle it is possible to use almost any of these to create dynamic GRN models and simulate their dynamics. Mathematics packages such as Mathematica, MATLAB, Maple, and Octave, as well as programming languages for which mathematics libraries are available, such as FORTRAN, C/C++, Java, and Python, may facilitate the numerical simulation of GRNs. Moreover, these high-level programming packages often contain generic tools for setting up and using GAs.

In general, because application of GAs to any problem requires the integration of a number of software tools, some of which may have to be developed from scratch, and most will need to be adapted to the current problem, knowledge of a high-level programming or scripting language, or familiarity with a dedicated mathematics package is essential.

As a general rule, application of GAs is computationally intensive. In the case of our running example, an "evolutionary run" lasted for 250 generations, with each generation consisting of 250–300 individuals; that is 250 times 250–300 × 10 simulation rounds of 200 time steps each (see Subheadings 2.3 and 3.3), plus overhead for the computation involved in the fitness assessment for each individual in each generation, the selection procedure, and the application of the evolutionary operators to each generation. Although it is possible to do these computations using a single processor, they are readily split into segments that can be performed in parallel. We used a specialized workload management system called Condor (41) to distribute the parallel segments between hundreds of desktop workstations in our University (including many student lab machines), harnessing the CPU power that would have been wasted while they were standing idle,

and cutting the overall computation time by one or two orders of magnitude.

3. Methods

As explained in Subheading 1.1 and shown in Fig. 1, an evolutionary cycle includes three steps:

1. Selection: from a population of candidate solutions to the problem, encoded in "genomes," a number of individuals are selected for reproduction. Each genome is associated with a "phenotype," and selection occurs on the basis of its phenotypic characteristics.
2. Variation: the offspring of the selected genomes receive a "mutated" and "recombined" version of their parental genomes.
3. Replacement: the previous population is replaced by their offspring, after which event a new cycle begins.

An evolutionary run usually starts with a population of random solutions (genomes), and terminates either when a predefined level of satisfaction with the solution has been met, or simply after a preset number of cycles has been completed. The application of a GA to a specific problem requires tailored specification of the following:

1. A genome (see Note 8): a representation of the "solution domain" that allows modification by evolutionary operators.
2. A phenotype: a representation whose characteristics can be compared to the target of the search, and a collection of rules that stipulate how the information contained in a genome is transformed into a phenotype.
3. A fitness function to evaluate the solution contained in the genotype and manifested in the behavior of the phenotype.
4. Appropriate mechanisms for selection, mutation and recombination, and reproduction and a termination strategy.

In this section we introduce the most popular approaches to the introduction of variation and selection, and describe in detail how the genome, phenotype, and fitness concepts were tailored to our running example.

3.1. Genome Representation

In GAs, genomes are typically represented by strings (linear sequences, as in DNA itself) of characters that somehow represent a solution to the problem that needs to be solved. These strings are often just sequences of zeros and ones, but larger "alphabets" – sets of permitted characters – are also allowed. The genome

representation in our running example uses an alphabet of four characters: 0, 1, 2, and 3. Each genome consists of 20 individual "genes" (see Note 9), and also encodes some information that is global to the system. Each gene is subdivided into one "coding" area and several "CRMs." In turn, each CRM contains a number of "binding sites" for TFs. The characters 2 and 3 function as CRM and gene delimiters only, whereas the information carried by the characters 0 and 1 varies, as explained below. Figures 5b (full genome) and 2b (single gene) illustrate the encoding. The last (rightmost) seven characters in the full genome are used to set two global properties, whose significance for brevity we will not go into. The next (from the end) 16×4 positions in the string encode the decay rates for each of the 16 (see below) different PRs. The 72nd character from the end, always a 3, indicates the start of the first gene. Although genes may have different numbers of CRMs, they are structured in the same way. The gene delimiter, 3, is followed by a single character (0 or 1) that determines whether the gene is constitutive (0) or facultative (1). The next four characters, all 0 or 1, indicate, as a binary number, the PR encoded by the gene; thus 0000 simply encodes pr00, 0101 corresponds to pr05, and 1111 to pr15 (see Note 10). Any zeros or ones following this five-character area are ignored, and the regulatory region begins at the first CRM delimiter (a 2) to the left of the gene. CRM representations may have different lengths, but the character (0 or 1) that immediately follows the delimiter always indicates whether the overall effect of the TF complex that binds to the CRM is inhibitory (0) or activating (1). The characters (0 or 1) in the "TF-binding area" of the CRM, which extends up to the following CRM or gene delimiter, determine which PRs will bind to the CRM. To this aim, the TF-binding area, for instance 00111110010110, as in CRM α in Fig. 2b, is split into as many quadruplets as possible, reading from left to right along the CRM (here, 0011, 1110, and 0101), and a residual (here 10). The residual (10) is taken to be "junk," and ignored, but the quadruplets (0011, 1110, and 0101) specify that pr03, pr14, and pr05 act, in synergy, as TFs in the expression of the gene to which the CRM belongs.

3.2. Genotype–Phenotype Mapping

In the case of our running example, the rules that specify how the information contained in the genome is transformed into the phenotype are easily understood. The phenotype is formed by the whole machinery of the CPM (see Subheading 2.1) and its GRN controller (Subheading 2.2). Most of the equations that govern behavior of the phenotype are contained in the CPM and GRN model themselves, and are therefore unchangeable. However, the connectivity of the GRN is contained in the genome, as well as some of the parameter values that determine the dynamics of the PRs (namely their decay rates). The decay rates are encoded as

"words" of four characters (0 or 1), each of which can therefore represent ("address") one of 16 different values. In our example, an even narrower "mapping" was used: the numbers 0–9 (0000–1001) represent decay rates of 0.0–0.9 (in steps of 0.1), and all higher numbers (1010–1111) map on a decay rate of 1.0. Furthermore, 12 of the 16 possible PRs were given the following (predetermined, and unchangeable) control functions: color determinants: pr00 and 01; CPM constraint control: pr02, 03 (morphogen secretion), 04 (shape), 05, 06, 07 (stickiness), 08 (size), 10 (preferential division direction); and morphogen concentration sensors (input): pr12 and 13.

3.3. Fitness Evaluation

To rank the fitness of the individual genomes, a so-called fitness function, a quantitative measure for their proximity to the target, must be specified. The design of the fitness function depends entirely on the nature of the problem, and may vary from simple root mean square deviation to a sophisticated statistical analytic function.

In our example, fitness was assessed as follows. As stated in Subheading 2.3, each simulation round, in which a phenotype was given a chance to develop from a single cell into a fully grown 2D array of cells, lasted for 200 time steps. At the end of the simulation, the fitness f of each individual was quantified using a pixel-by-pixel comparison of final arrangement of colors $\boldsymbol{R}$ in the cell array (after 200 time steps in the simulation) with the target pattern $\boldsymbol{T}$ (the French flag) as expressed in Eq. 2:

$$f = 1 - \frac{4 - n_C}{w \times h} \sum_{x=1}^{w} \sum_{y=1}^{h} (R_{xy} \neq T_{xy}) \tag{2}$$

Here, x and y enumerate the width w (60 pixels) and height h (40 pixels) of the lattice, respectively, so that $0 < x \leq w$, and $0 < y \leq h$. The color T_{xy} of the pixel at position (x, y) in the target lattice is compared with the color of the pixel in the same position in the cell array, R_{xy}. If the colors are different, the statement $R_{xy} \neq T_{xy}$ is true, and if they are the same, it is false. Each "false" is converted to 0, and each "true" to 1, and the resulting zeros and ones for each position are added up, and the answer divided by the total number of pixels, $w \times h$. Thus, the fitness f will evaluate to its maximum value of 1 if all pixels in R have the same color as those in $\boldsymbol{T}$ (so that the sum of all $(R_{xy} \neq T_{xy})$ terms equals 0). The factor $4 - n_C$, where n_C is the total number of colors present in the final cell array, in the second term of the function, puts a "penalty" on the use of fewer than three colors in the final pattern, as follows. Suppose $\boldsymbol{R}$ uses all three colors $(4 - n_C = 1)$, and 25% of the pixels in $\boldsymbol{R}$ have the same color as their counterparts in T, so that the sum of all $(R_{xy} \neq T_{xy})$ terms equals 0.75. In that case, f evaluates to 0.25. However, if only one or two colors

are used in $\boldsymbol{R}$ ($4-n_C = 3$ or 2), f is much smaller, namely −1.25 or −0.5, respectively. In this way, individuals are "encouraged" to use all three colors early on in the evolution. This refinement of the fitness function was deemed to be necessary when it was found that too many evolutionary runs led to monochrome arrays, indicating that the original fitness function (without the $4-n_C$ factor) had insufficient discriminatory power to allow the search come to a satisfactory conclusion. Furthermore, the overall fitness of an individual was computed by carrying out ten separate simulation rounds (performed with different random number sequences, see Subheading 2.3) and averaging the values of f obtained in each of these rounds, to ensure that the solution was relatively robust.

3.4. Initial Population

To start an evolutionary run, an initial genome population must be created. The composition of this initial population is, again, entirely dependent on the problem itself, on the way the problem has been encoded in the genome, and also on the prior knowledge that is to be taken into account. In the case of the running example, the initial population of 250–300 individuals consisted of genomes in which each of the 20 genes had a single CRM. All mutable characters in the sequences (the zeros and ones) were randomly assigned, yielding a collection of randomly connected network. Note that the genes in these networks produced a subset, and not necessarily all, of the possible PRs.

3.5. Selection and Replacement

Once the fitness of each individual in the population has been established, a proportion of the genomes are selected to act as parents for the next generation. Typically, two individuals are selected, and allowed to produce two offspring. During this process, exchange and mutation of the genetic material occurs, so that the offspring differ somewhat from the parents. Selection of parents continues until the total number of offspring is equal to the number of individuals in the parent population, after which the older population is replaced with its offspring. Several strategies have been developed that let the fittest produce the most offspring, but leave a fair amount of genetic variability in the population. Here we introduce the selection techniques that are most widely used for the selection of the parents.

The first method is "fitness proportionate" selection, a weighted lottery in which the chance to be selected increases with fitness. This method is often depicted as a roulette wheel, as in Fig. 3a. As individuals may be selected more than once, their reproduction rate is proportional to their fitness. Another method involves a "tournament," in which a number of individuals are randomly chosen from the current population, and out of these the two fittest are selected for reproduction (Fig. 3b). Again, individuals may be selected multiple times, and the fittest may win

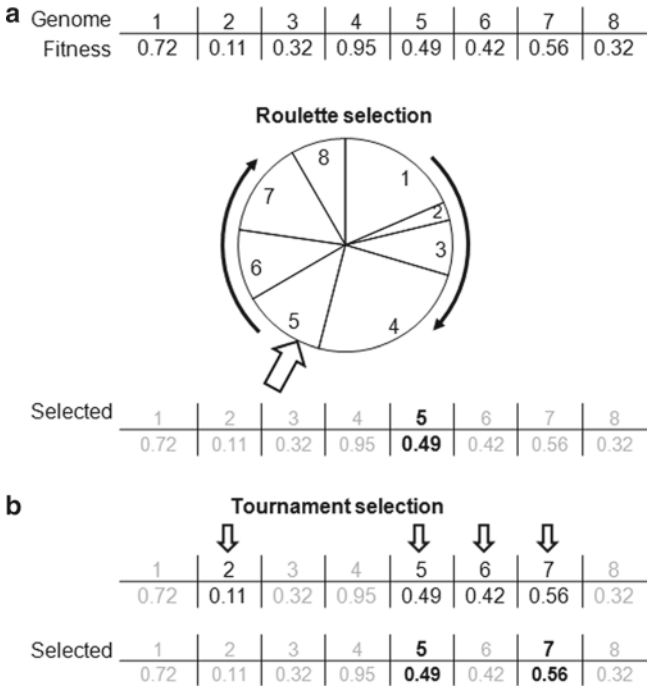


Fig. 3. Selection methods. (**a**) Roulette wheel selection: each individual is assigned a slice of the wheel whose size is proportional to its fitness. The wheel is rotated, and when it stops, the individual associated with the slice under the pointer is selected as a parent. (**b**) Tournament selection: a fixed number of individuals is randomly chosen from the pool. The couple with the highest fitness scores is selected as parents.

several tournaments, but here their reproduction rate is not necessarily proportional to their fitness. The last strategy is "elitism," in which the genomes of a number of the fittest individuals are copied exactly into the pool of offspring. The remainder of the offspring is then generated using one of the other procedures. In a strategy that uses elitism, the highest fitness value never decreases in the population, which some consider to be an advantage.

In the example, a combination of elitism with tournament selection was used: the fittest genome was kept unchanged, and the rest of the offspring were obtained from tournaments involving 15 randomly chosen individuals.

3.6. Variation

As in nature, mutations in a genome are achieved by changing a character in a random position along the length of the genome into another character from the alphabet, or by inserting or deleting a string of characters. Cross-over is realized by allowing two parents to exchange part of their genomes. Figure 4 illustrates the typical implementation of point mutation, deletion, duplication, inversion, and cross-over.

In the running example, point mutations are made by changing a 0 into a 1 or a 1 into a 0. Point mutations to delimiters (2 or 3)

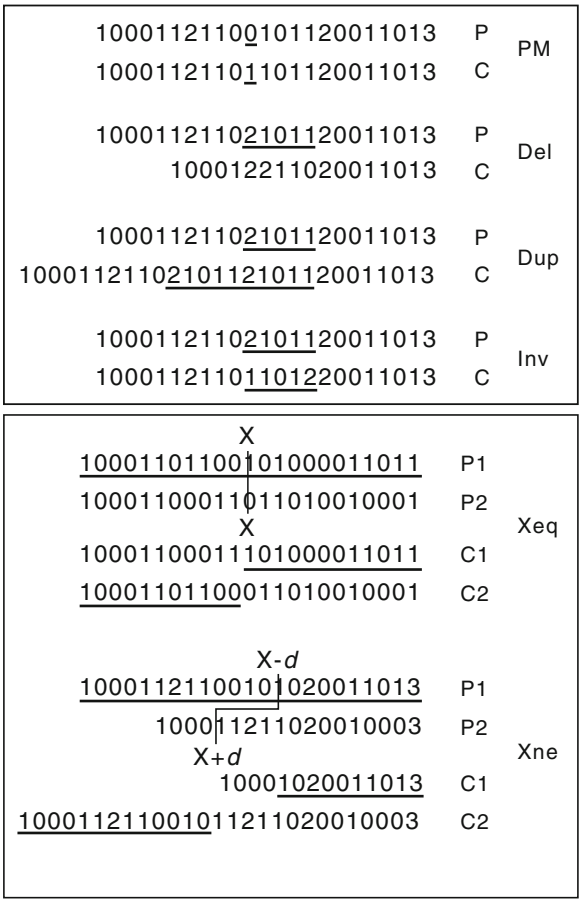


Fig. 4. Introducing genetic variation. *Top panel*: mutation mechanisms, requiring one parent (P) and producing one child (C). PM, point mutation: a single zero changes into a one or vice versa; Del, deletion: a substring of characters is deleted in the child genome; Dup, duplication: a copy of substring appears in the child genome (not necessarily adjacent to the original); Inv, inversion: a substring is inverted in the child. *Bottom panel*: Cross-over mechanisms, requiring two parents (P1, P2), and producing two children (C1, C2). Xeq, equal cross-over: the characters to the left of a position *X* on two equally sized genomes or genome parts are swapped between the parents; Xne, nonequal cross-over: the characters to the left of *X*–*d* on P1 are swapped with those left of *X*+*d* on P2. In a Gaussian-offset cross-over mechanism, the offset *d* is randomly chosen from a Gaussian distribution.

do not occur. Cross-over requires a pair of genomes, P1 and P2, and is performed as follows. Each genome is parsed into 21 segments, 20 of which containing a gene, and one containing the global parameter section. One segment is randomly selected, and a cross-over position, *X*, is chosen within this segment. If the chosen segment happens to be the global area of the genome (which has the same length in both parents), the genome parts distal (see Note 11) to *X* on P1 and P2 are swapped (Fig. 4, Xeq). If *X* is situated in one of the genes, an offset *d* is randomly selected

from a Gaussian distribution centered on 0, with a standard deviation of 4 (but cut off in cases where the end of a gene will be crossed). Then the parts distal to X1 = $X - d$ on P1 and X2 = $X + d$ on P2 are swapped, as illustrated in Fig. 4, Xne. The insertions and deletions that result from this Gaussian-cross-over mechanism may generate a reading frame shift, and thus change the gene's regulatory pattern. Moreover, it allows the total number of CRMs within a gene to change. The relatively small standard deviation in the Gaussian distribution prevents extensive perturbation of existing control patterns. Nonetheless, in the course of the evolutionary run CRMs, and as a result even whole genes, may become nonfunctional, as CRM size is allowed to drop below that of a single TF-binding site. "Degenerate" CRMs without TF-binding sites are ignored in the computation of the gene expression rate.

Associated with each genetic operator is a value that specifies the frequency with which it is to be applied. In the example, the probability that a 0 or 1 changes into a 1 or a 0 during a single evolutionary cycle is 1%, whereas the probability for a point mutation in a 2 or a 3 is zero. The deletion, duplication, and inversion parameters were not used, but the Gaussian-offset cross-over frequency per genome in one evolutionary cycle was set at 90%.

4. Results

The results of one successful evolutionary run are summarized in Fig. 5. The top panel shows that the fitness of the best individual in the population rises in a number of distinct phases. These phases are characterized by an initial sharp increase in fitness, followed by period in which the rate of increase becomes gradually smaller, sometimes dropping down to almost zero for an extended period. It would appear that an "evolutionary innovation" has been made, and a whole new set of better solutions has been uncovered. The GA then spends some time searching "close to home" for the best solution within this new area, until it hits upon a next innovation. The similarity between the target pattern in the inset in the top panel, and the achieved solution, in the bottom right panel (Fig. 5d, $t = 200$, bottom row) is approximately 90% after 250 evolutionary cycles. The genome string of the best individual after the full run and the diagram of the GRN it represents (Fig. 5b, c) demonstrate that during the course of the evolution the genome has shrunk from the initial 20 functional genes to 13, and that of these only 8 contribute to the total gene expression pattern (all 5 noncontributing genes are facultative and need activation to start producing their PR; however g04, 15, 17, and 19 have no activating TFs, and

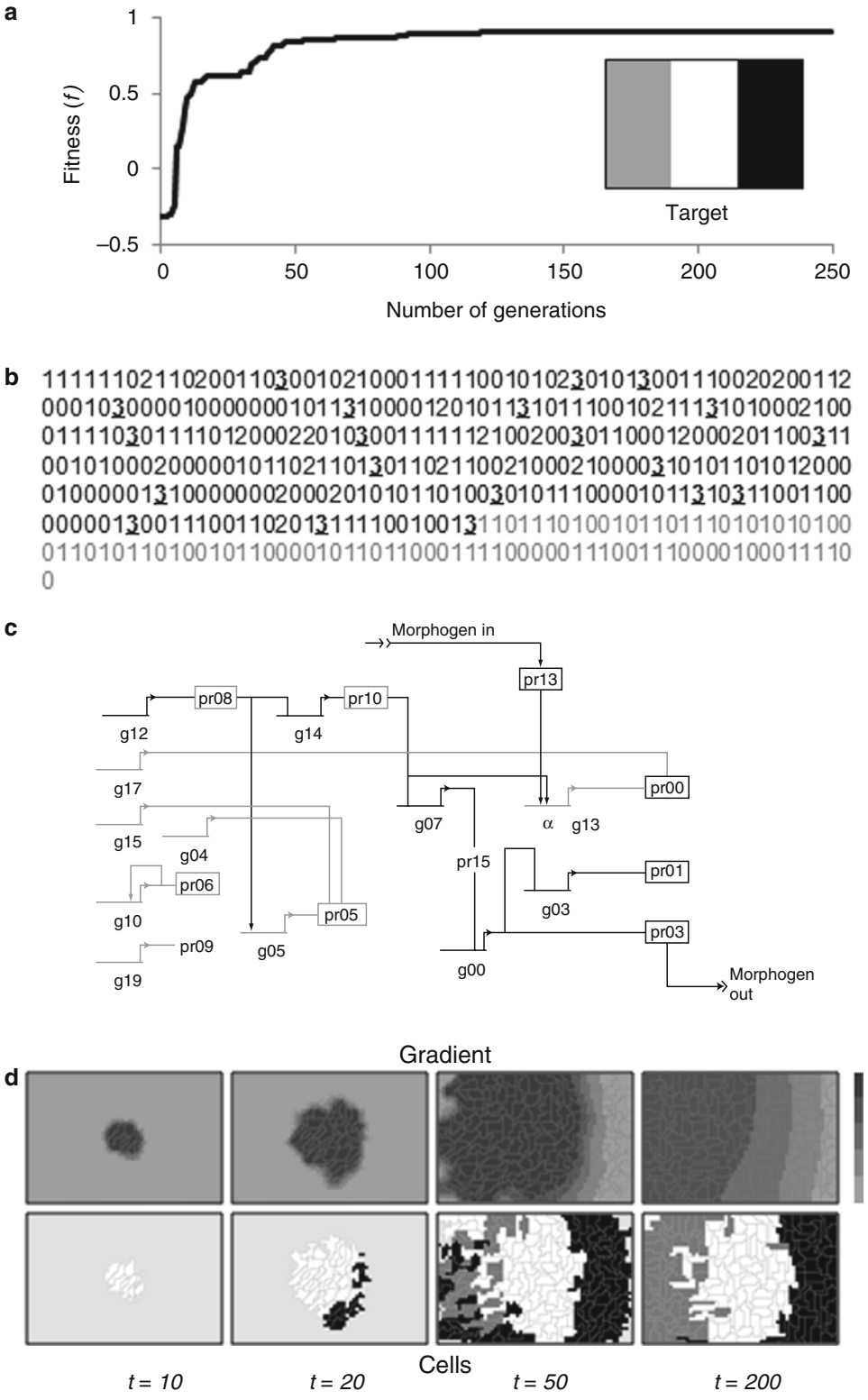


Fig. 5. Results. (**a**) Fitness value (*f*) of the fittest individual in the population over an evolutionary run of 250 generations. Inset: the "French flag" target pattern, with the *dark gray zone on the left* and the *lighter gray zone on the right* representing the *blue* and the *red areas* of the flag. (**b**) The genome of the fittest individual in the 250th generation.

g10 is activated only by its own product). Only 8 of 16 possible PRs are produced, with only one (out of a possible 4) without a predefined function: pr00 and 01 (color 1 and 2), 03 (morphogen 2 secretion), 05 and 06 (stickiness 1 and 2), 08 (size), 10 (division direction), 13 (morphogen 2 concentration sensor), and 15 (the only TF without a predefined function). Note that the system could have used two different morphogens and three different stickiness constraints, but appears to have "discovered" that it needed to use only one morphogen (morphogen 2), and that two stickiness constraints were sufficient.

Figure 5d shows snapshots of the expanding cell array and the morphogen gradients in its stratum at various developmental stages. The simulation begins with a single cell, initially consisting of one blue pixel, which grows and turns to red, then to white, and divides. After five time steps, there are two white cells, which then proliferate in the next five steps to a total of 14. After 20 time steps, there are about 50 cells, and second (red) color begins to appear to the right of the white cells. At around $t = 40$, the third color (blue) appears to the left of the pattern. The area fills up with cells in 70–80 time steps, and, while the cells remain dynamic, the final pattern stabilizes after some 110 time steps. It should be emphasized that the solution presented here is not the only one possible: other evolutionary runs have produced GRNs with an apparently entirely different connectivity and greater complexity that were equally capable of directing the development of a French flag pattern.

5. Concluding Remarks

Just as there is no single best way of modeling GRNs, there is no all-purpose GA, and not all problems lend themselves equally well to the GA approach. Even if a solution domain is easily expressed in some sort of "genome" structure, the genome may turn out not

Fig. 5. (continued) The gene delimiters are *underlined*; the area of the genome with the global information is indicated in *gray*. (**c**) *Box*-and-*arrow* GRN representation of the genome in **b** (visualized using the tool BioTapestry (49)). The symbols labeled g00, g03, etc., represent genes, and the text boxes labeled pr00, pr01, etc., represent their gene products. Constitutive and facultative genes are represented by, respectively, *black* and *gray symbols*. The *boxes* of PRs with a predefined function have a *solid outline*; *black* for input and output, and *gray* for PRs that drive a CPM constraint (see text). Connections ending in *arrows* (activating) and *bars* (inhibitory) specify regulatory interactions; the α under g13 indicates that both regulatory interactions happen on one CRM, named α (in this GRN, there are no other CRMs with multiple participants, and none of the genes have more than one CRM). Note, furthermore, that non-functional genes g01, 02, 06, 08, 09, 11, and 16 have been left out of the drawing, and that only pr00, 01, 03, 05, 06, 08, 10, 13, and 15 are produced. (**d**) Four stages (after 10, 20, 50, and 200 time steps) in the simulation of the development controlled by the GRN in **b** and **c**. The four *top panels* show the spread of the morphogen over the stratum (with the cell outlines superimposed; the stratum itself consists of 40×60 pixels with no higher-level structure); *darker grays* indicate higher morphogen concentrations. The *bottom panels* show the proliferation of cells. Blue, white, and red cells are represented in middle *gray*, *white*, and *black*, respectively, and the *light gray* background color indicates the absence of cells.

to be "evolvable." Poor evolvability may be inherent in the problem, but it may also be due to the way the solution domain is encoded, the genotype–phenotype mapping, or simply to the choice of evolutionary operators and parameter settings. As a rule, it is a good idea to ensure, if possible, that single mutations mostly lead to relatively small changes in the phenotype. It is also advised to try out different evolutionary operators and parameter settings. Techniques exist to automatically adapt parameter values during an evolutionary run: a widely used algorithm is Simulated Annealing (42, 43), but it is also possible to use other EC techniques (which are, after all, good at optimization) for this purpose (44). Furthermore, solutions to complex problems may sometimes be found by "easing" the system along by shifting the target, as was done in (45, 46). To this aim, the "evolutionary pressure" is gradually increased, by first letting the system find a good solution to a part of the problem, and then gradually changing the target to the parts for which a solution is more difficult to find.

It may have become clear from the above exposition that while GAs borrow ideas from nature, they still lack a lot of biological detail. Because basic GAs have quite a few limitations, their users – engineers, scientists – have been prompted to inject more ideas from natural evolution into GAs to try and boost their performance in optimization problems (47). It is generally believed that the introduction of "junk code" (as in the genome representation in our example) and diploidy with dominance–recessivity allows for information buffering and protection. The inclusion of developmental programs in which a single unit grows into a differentiated multiunit structure may result in improved scalability, modularity, and robustness. Because genetic diversity allows species to adapt more easily to changing conditions in nature, some GAs include strategies designed to preserve diversity. Many of these and similar extensions do indeed improve performance in particular instances, but on other occasions they may only add complexity and increase the search time (48).

6. Notes

1. For reasons of simplicity we have omitted the concept of species from this discussion.
2. Other techniques include evolutionary strategies, genetic programming, and evolutionary programming; discussion of their characteristics and application is outside the scope of this article.
3. We use PR, rather than GP, as the abbreviation for gene product, in order to avoid confusion with GP as an abbreviation for genetic

programming, another branch of evolutionary computation. Note that PR does not stand for protein, although proteins are a PR category, but could mean protein/RNA.

4. TF is often used as an abbreviation for transcription factor; however, in the definition of *trans*-regulatory factor to which we adhere here, transcription factors are a specific class of TFs.
5. In fact, a capacity for rapid breakdown contributes significantly to a cell's potential to adapt efficiently to changed conditions.
6. Note that this approach is highly simplified, and does not take into account that a complex may contain more than one molecule of a particular PR, and that a PR may take part in more than one type of TF complex.
7. Provided the sequences of random numbers produced by the RNG are different: this is achieved by using a different "seed" value for the RNG in each simulation round.
8. In this context, sometimes also indicated as genotype or chromosome.
9. The number of genes was fixed to facilitate further analysis.
10. Thus, a coding area consisting of four "bits" (whose value can be 0 or 1) may code for one of a total of $2^4 = 16$ different PRs.
11. Distal to X: the genome section from X to the end of the genome.

References

1. Holland, J. H. (1962) Outline for a logical theory of adaptive systems. *JACM* **9**, 297–314.
2. Holland, J. H. (1992) *Adaptation in natural and artificial systems, 2nd edition*, MIT Press, Cambridge, MA.
3. Mitchell, M. (1998) *An introduction to genetic algorithms*, MIT Press, Cambridge, MA.
4. Back, T., Fogel, D. B., and Michalewicz, Z. (1999) *Evolutionary algorithms*, Vol. I and II, IOP, Bristol, UK.
5. Schilstra, M. J., and Nehaniv, C. L. (2008) Bio-logic: gene expression and the laws of combinatorial logic. *Artif Life* **14**, 121–33.
6. Karlebach, G., and Shamir, R. (2008) Modelling and analysis of gene regulatory networks. *Nat Rev Mol Cell Biol* **9**, 770–80.
7. Kulasiri, D., Nguyen, L. K., Samarasinghe, S., and Xie, Z. (2008) A review of systems biology perspective on genetic regulatory networks with examples. *Curr Bioinform* **3**, 197–225.
8. Mitrophanov, A. Y., and Groisman, E. A. (2008) Positive feedback in cellular control systems. *Bioessays* **30**, 542–55.
9. Cho, K. H., Choo, S. M., Jung, S. H., Kim, J. R., Choi, H. S., and Kim, J. (2007) Reverse engineering of gene regulatory networks. *IET Syst Biol* **1**, 149–63.
10. Goutsias, J., and Lee, N. H. (2007) Computational and experimental approaches for modeling gene regulatory networks. *Curr Pharm Design* **13**, 1415–36.
11. Tomlin, C. J., and Axelrod, J. D. (2007) Biology by numbers: mathematical modelling in developmental biology. *Nat Rev Genet* **8**, 331–40.
12. Palumbo, M. C., Farina, L., Colosimo, A., Tun, K., Dhar, P. K., and Giuliani, A. (2006) Networks everywhere? Some general implications of an emergent metaphor. *Curr Bioinform* **1**, 219–34.
13. Kaznessis, Y. N. (2006) Multi-scale models for gene network engineering. *Chem Eng Sci* **61**, 940–53.
14. Facciotti, M. T., Bonneau, R., Hood, L., and Baliga, N. S. (2004) Systems biology experimental design – considerations for building predictive gene regulatory network models

for prokaryotic systems. *Curr Genomics* **5**, 527–44.

15. Welch, S. M., Dong, Z. S., Roe, J. L., and Das, S. (2005) Flowering time control: gene network modelling and the link to quantitative genetics. *Aust J Agric Res* **56**, 919–36.
16. Prusinkiewicz, P. (2004) Modeling plant growth development. *Curr Opin Plant Biol* **7**, 79–83.
17. Kaern, M., Blake, W. J., and Collins, J. J. (2003) The engineering of gene regulatory networks. *Annu Rev Biomed Eng* **5**, 179–206.
18. Thieffry, D., and Sanchez, L. (2003) Dynamical modelling of pattern formation during embryonic development. *Curr Opin Gen Dev* **13**, 326–30.
19. Bolouri, H., and Davidson, E. H. (2002) Modeling transcriptional regulatory networks. *Bioessays* **24**, 1118–29.
20. De Jong, H. (2002) Modeling and simulation of genetic regulatory systems: a literature review. *J Comput Biol* **9**, 67–103.
21. Shmulevich, I., Dougherty, E. R., and Mang, W. (2002) From Boolean to probabilistic Boolean networks as models of genetic regulatory networks. *Proc IEEE* **90**, 1778–92.
22. Hasty, J., McMillen, D., and Collins, J. J. (2002) Engineered gene circuits. *Nature* **420**, 224–30.
23. Van Someren, E. P., Wessels, L. F. A., Backer, E., and Reinders, M. J. T. (2002) Genetic network modeling. *Pharmacogenomics* **3**, 507–25.
24. Rao, C. V., and Arkin, A. P. (2001) Control motifs for intracellular regulatory networks *Annu Rev Biomed Eng* **3**, 391–419.
25. Hasty, J., McMillen, D., Isaacs, F., and Collins, J. J. (2001) Computational studies of gene regulatory networks: in numero molecular biology. *Nat Rev Genet* **2**, 268–79.
26. Smolen, P., Baxter, D. A., and Byrne, J. H. (2000) Modeling transcriptional control in gene networks – methods, recent results, and future directions. *Bull Math Biol* **62**, 247–92.
27. McAdams, H. H., and Arkin, A. (1998) Simulation of prokaryotic genetic circuits. *Annu Rev Biophys Biomol Struct* **27**, 199–224.
28. Kauffman, S. A. (1993) *The origins of order. Self-organization and selection in evolution*, Oxford University Press, New York.
29. Wolpert, L. (1969) Positional information and the spatial pattern of cellular differentiation. *J Theor Biol* **25**, 1–47.
30. Wolpert, L. (1996) One hundred years of positional information. *Trends Genet* **12**, 359–64.
31. Jaeger, J., and Reinitz, J. (2006) On the dynamic nature of positional information. *BioEssays* **28**, 1102–11.
32. Miller, J. F. (2004) Evolving a self-repairing, self-regulating, French Flag Organism, in *Proceedings of genetic and evolutionary computation conference – GECCO 2004*, (Kalyanmoy Deb, Riccardo Poli, Wolfgang Banzhaf, Hans-Georg Beyer, Edmund K.Burke, Paul J. Darwen, Dipankar Dasgupta, Dario Floreano, James A. Foster, Mark Harman, Owen Holland, Pier Luca Lanzi, Lee Spector, Andrea Tettamanzi, Dirk Thierens, Andrew M. Tyrrell, Eds.) vol 1, pp 129–139, Springer, Berlin.
33. Meinhardt, H. (1982) *Models of biological pattern formation*, Academic Press, London.
34. Knabe, J. F., Schilstra, M. J., and Nehaniv, C. L. (2008) Evolution and morphogenesis of differentiated multicellular organisms: Autonomously generated diffusion gradients for positional information, in *Artificial Life XI: Proc eleventh international conference on the simulation and synthesis of living systems* (Bullock S., Noble, J., Watson, R., and Bedau, M., Eds.), pp 321–8, MIT Press, Cambridge, MA, USA.
35. Glazier, J. A., and Graner, F. (1993) Simulation of the differential adhesion driven rearrangement of biological cells. *Phys Rev E* **47**, 2128–54.
36. Merks, R. M. H., and Glazier, J. A. (2005) A cell-centered approach to developmental biology. *Physica A* **352**, 113–30.
37. De Jong, H., Geiselmann, J., Hernandez, C., and Page, M. (2003) Genetic Network Analyzer: qualitative simulation of genetic regulatory networks. *Bioinformatics* **19**, 336–44.
38. Gonzalez, A. G., Naldi, A., Sánchez, L., Thieffry, D., and Chaouiya, C. (2006) GINsim: a software suite for the qualitative modelling, simulation and analysis of regulatory networks. *Biosystems* **84**, 91–100.
39. Schilstra, M. J., Martin, S. R., and Keating, S. M. (2008) Methods for simulating the dynamics of complex biological processes, in *Biophysical tools for the biologist* (Correia, J. J., and Dietrich, W. H., Eds.), Methods in Molecular Biology, (Wilson, L., and Matsudaira, P. T., Eds.), pp 807–841, Elsevier, San Diego.
40. Szallasi, Z., Stelling, J., and Periwal, V., (Eds.) (2006) *System modeling in cellular biology*, MIT Press, Cambridge, MA.
41. Thain, D., Tannenbaum, T., and Livny, M. (2005) Distributed computing in practice: the

Condor experience. *Concurr Comput* **17**, 323–56.

42. Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983) Optimization by simulated annealing. *Science* **220**, 671–80.
43. Cerny, V. (1985) A thermodynamical approach to the travelling salesman problem: an efficient simulation algorithm. *J Optim Theory App* **45**, 41–51.
44. Beyer, H. G., and Schwefel, H. P. (2002) Evolution strategies: a comprehensive introduction. *Natural Comput* **1**, 3–52.
45. Knabe, J. F., Nehaniv, C. L., and Schilstra, M. J. (2006) Evolutionary robustness of differentiation in genetic regulatory networks, in *Proc 7th German Workshop on Artificial Life 2006 (GWAL-7)* (Artman, S., and Dittrich, P., Eds.), pp 75–84, Akademische Verlagsgesellschaft Aka, Berlin.
46. Knabe, J. F., Nehaniv, C. L., and Schilstra, M. J. (2008) Genetic regulatory network models of biological clocks: evolutionary history matters. *Artif Life* **14**, 135–48.
47. Altenberg, L. (1994) The evolution of evolvability in genetic programming, in *Advances in genetic programming* (Kinnear, K. E., Ed.), pp 47–74, MIT Press, Cambridge, MA, USA.
48. Beer, R. D. (2004) Autopoiesis and cognition in the game of life. *Artif Life* **10**, 309–26.
49. Longabaugh, W. J. R., Davidson, E. H., and Bolouri, H. (2005) Computational representation of developmental genetic regulatory networks. *Dev Biol* **283**, 1–16.

INDEX

David Fenyö (ed.), *Computational Biology*, Methods in Molecular Biology, vol. 673,
DOI 10.1007/978-1-60761-842-3, © Springer Science+Business Media, LLC 2010

N

O

P

Q

R

S

T

U

V

X

Z