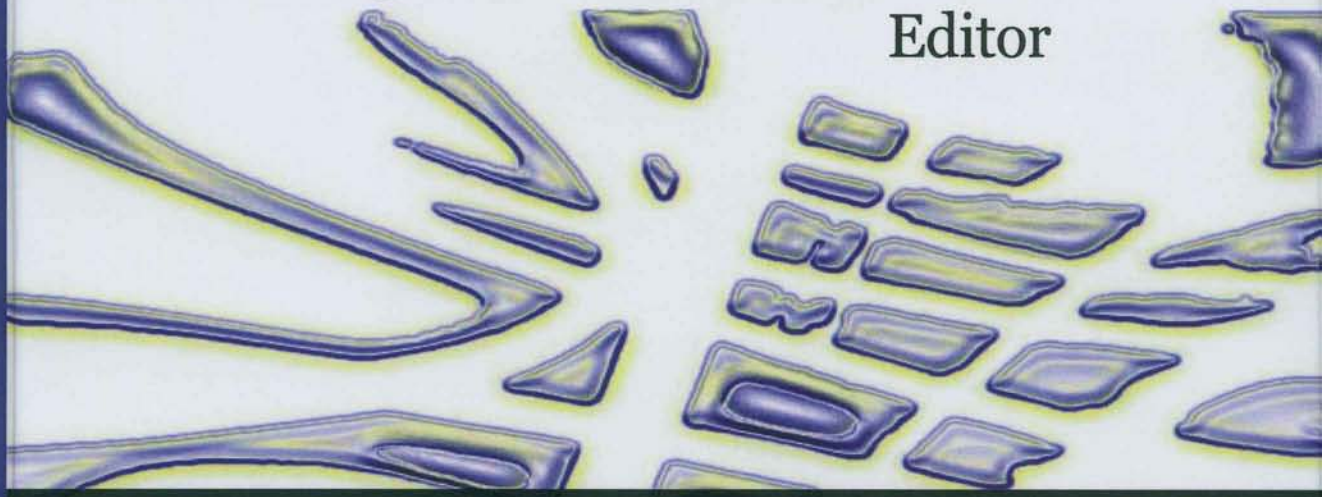
Advances in Grid Generation

Olga V. Ushakova
Editor



Contributors

Nina N. Anuchina
Nataliya A. Artyomova
Boris N. Azarenok
Tatjana N. Bronina
Aleksandr A. Charakhch'yan
Nikolay S. Es'kov
Olga B. Feodoritova
Irina A. Gasilova
Sergei K. Godunov
Viatcheslav A. Gordeychuk
Olga S. Ilyutina
Sergey A. Ivanenko
Aleksey F. Khairullin
Olga B. Khairullina
Oleg M. Kozyrev
Andrew V. Lygin
Vladimir M. Miklyukov
Genadii P. Prokopov
Alexander M. Sorokin
Olga V. Ushakova
Natalia A. Vladimirova
Vladimir I. Volkov
Victor T. Zhukov

NOVA

ADVANCES IN GRID GENERATION

OLGA V. USHAKOVA
EDITOR

Nova Science Publishers, Inc.
New York

LIBRARY OF CONGRESS CATALOGING-IN-PUBLICATION DATA

Available upon request

ISBN 1-59454-273-2

Published by Nova Science Publishers, Inc. + New York

This book is devoted to the memory of Sergey A. Ivanenko

CONTENTS

PREFACE

This book is a collection of works on grid generation. It brought together the results of some resent outstanding investigations on this subject.

The book is addressed to researchers and specialists working in the area of grid generation and numerical simulation of physical field phenomena and processes.

The book is helpful for young researchers, undergraduate and postgraduate students, and for practicing engineers working also in fields of computational and applied mathematics, mechanics and physics.

It can be interesting to the scientists of other branches of mathematics, since it concerns large variety of general mathematical problems.

Olga V. Ushakova

CONTRIBUTORS

Nina N. Anuchina
Russian Federal Nuclear Center - Zababakhin Institute of Technical Physics

Nataliya A. Artyomova
Institute of Mathematics and Mechanics
Ural Branch of the Russian Academy of Sciences

Boris N. Azarenok
Dorodnicyn Computing Center
Russian Academy of Sciences

Tatjana N. Bronina
Institute of Mathematics and Mechanics
Ural Branch of the Russian Academy of Sciences

Aleksandr A. Charakhch'yan
Dorodnicyn Computing Center
Russian Academy of Sciences

Nikolay S. Es'kov
Russian Federal Nuclear Center - Zababakhin Institute of Technical Physics

Olga B. Feodoritova
Keldysh Institute of Applied Mathematics
Russian Academy of Sciences

Irina A. Gasilova
Institute of Mathematics and Mechanics
Ural Branch of the Russian Academy of Sciences

Sergei K. Godunov
Sobolev Institute of Mathematics
Siberian Branch of the Russian Academy of Sciences

Viatcheslav A. Gordeychuk
Russian Federal Nuclear Center - Zababakhin Institute of Technical Physics

Olga S. Ilyutina
Russian Federal Nuclear Center - Zababakhin Institute of Technical Physics

Sergey A. Ivanenko
Dorodnicyn Computing Center
Russian Academy of Sciences

Aleksey F. Khairullin
Institute of Mathematics and Mechanics
Ural Branch of the Russian Academy of Sciences

Olga B. Khairullina
Institute of Mathematics and Mechanics
Ural Branch of the Russian Academy of Sciences

Oleg M. Kozyrev
Russian Federal Nuclear Center - Zababakhin Institute of Technical Physics

Andrew V. Lygin
Volgograd State University

Vladimir M. Miklyukov
Volgograd State University

Genadii P. Prokopov
Keldysh Institute of Applied Mathematics
Russian Academy of Sciences

Alexander M. Sorokin
Central Aerohydrodynamic Institute (TsAGI)

Olga V. Ushakova
Institute of Mathematics and Mechanics
Ural Branch of the Russian Academy of Sciences

Natalia A. Vladimirova
Central Aerohydrodynamic Institute (TsAGI)

Vladimir I. Volkov
Russian Federal Nuclear Center - Zababakhin Institute of Technical Physics

Victor T. Zhukov
Keldysh Institute of Applied Mathematics
Russian Academy of Sciences

ACKNOWLEDGEMENTS

It happens that this book is a collection of works devoted to the memory of Sergey A. Ivanenko, the outstanding researcher in the field of computational mathematics and grid generation. It was not the special aim of the book. S. A. Ivanenko also was going to take part in it. However the circumstances went in such a way that this book became the book devoted to the memory of Sergey. Sergey A. Ivanenko died just at that time when this book was planed. The book really became good memory of this scientist because in many chapters of it one can find his name. Many of authors were influenced by him, even if this was not reflected in their chapters. This collection of works is our last "thanks" to Sergey.

While elaborating our methods, we were influenced also by our teachers and investigations of other scientists. Their role, we tried to reflect in the text and corresponding references.

We are also grateful to Nova Science Publishers for giving us the opportunity to publish our *Advances in Grid Generation*.

We are thankful to different funds and organizations, Russian Foundation for Basic Research, Department of Mathematical Sciences of the Russian Academy of Sciences and Program of Fundamental Research of the Presidium of the Russian Academy of Sciences, International Science and Technology Center, providing us with the opportunities to conduct our investigations.

We are thankful to our institutes and universities for giving us the opportunities and facilities for our research. These are the institutes of the Russian Academy of Sciences — Dorodnicyn Computing Center, Keldysh Institute of Applied Mathematics, Institute of Mathematics and Mechanics of the Ural Branch, Sobolev Institute of Mathematics of the Siberian Branch, — Central Aerohydrodynamic Institute (TsAGI), Russian Federal Nuclear Center (Zababakhin Institute of Technical Physics) and Volgograd State University.

Olga V. Ushakova

In memory of Sergey A. Ivanenko

A talented mathematician and specialist in mechanics, senior researcher, head of the sector at the Dorodnitsyn Computing Center, Russian Academy of Sciences, Sergey Alexandrovich Ivanenko died suddenly on September 24, 2003, as a result of a fatal heart attack. Science lost an outstanding researcher, who left a notable trace in computational fluid dynamics and numerical mathematics. His entire scientific career was tied with the Computing Center, where he began working as an engineer, then a junior researcher, and finally became a sector head, where he had many friends, colleagues, and followers. The untimely death of Sergey in the prime of his creative talent has become a great grief for everyone who knew him. A relatively new field in computational mathematics – the theory of curvilinear grid generation – where Sergey was a prominent specialist, also suffered bereavement.

Sergey Ivanenko was born on June 11, 1954, in Feodosiya, Crimea. His farther, Alexander Ivanovich Ivanenko, was a chief constructor at the shipbuilding plant, mother, Maria Nikonovna, was a pharmaceutical chemist. In 1971 Sergey graduated from the school in Feodosiya (he was awarded the gold medal for the excellent knowledge) and entered the Department of Control and Applied Mathematics of the Moscow Institute of Physics and Technology. In 1977 he graduated from the Institute and was assigned to the Computing Center, USSR Academy of Sciences.

In the first years, his abilities were noted by Academician N.N. Moiseev (1917–2000), who guided many scientific directions at the institution and entrusted various important research studies to Sergey. Sergey focused on developing mathematical methods for modeling dynamics and water quality of inland water basins. He developed several mathematical models and participated in their practical implementation. For practical developing these models he applied the finite element method, a new technique at that time, which later became his specialisation along with effective grid generation. Sergey was a leading world specialist in the theory and practice of grid generation.

In 1979 Sergey began his postgraduate studies at the Computing Center and in 1985 defended a candidate dissertation in physics and mathematics [1]. This rather long period of work under the dissertation is explained by, first, the deep understanding the subject and second, reluctance to compromise concerning the scientific problems. For the preprint [2], based on the dissertation work, Sergey was awarded a yang scientist's diploma of the USSR Academy of Sciences in 1986. It is of interest that Sergey chose S.K.Godunov as a reviewer when presenting this preprint to the competition. A year later he obtained the first very important result, a variational barrier method, which now is known as a classical method in grid generation. To understand what price a scholar must pay for the real scientific results we asked Prof. Alexander Charakch'yan to confide his impressions on their shared work. This is his story.

"I met Sergey Ivanenko in the mid of 1980s. Though we both graduated the Moscow Institute of Physics and Technology and worked at the Computing Center (I since 1972, he five years later), we were not acquainted, maybe due to the difference in age or maybe because we worked in different departments on different floors, and were engaged in various problems, far from each other. A common cause in our problems was that they had required to generate regular meshes in planar domains with strongly crooked boundaries. I did not want to penetrate into the problem of grid generation and, thus, worked in the following manner. I took a paper on grid generation, found description of the algorithm, realized it and observed whether it operated properly when solving my problem or did not. So, I tried several known algorithms, which did not work with my problem.

Sergey also tried several known algorithms, which failed in his models. However, in contrast to me, he recognized the importance of the grid generation problem, for both his own modeling and as a self-maintained task of computational mathematics. At that time he went deeply into this problem, understood the reason of why well-known algorithms failed to work, and proposed a new direction in the methods of regular grid generation – variational methods with restriction of the set of admissible grids. He realized this idea in the form of an algorithm being close to the algorithm of constrained minimization for the finite-dimensional function. The algebraic relations contain the Lagrange multipliers which provide convexity of all quadrilateral grid cells provided that the mesh iterations converged. Remained summands are approximations of the terms in the Euler-Lagrange equations for the functional of smoothness. This algorithm operated properly in the problems, which Sergey was solving at that time.

In connection with defending the dissertation work, Sergey gave a talk at the seminar in our department. Therefore, the next grid algorithm, realized by me, was his method. The result turned out unsatisfactory. With time, the domain boundary distorted so much that the iterations stopped to converge and my attempts to restrict the number of iterations led, as in the case of all other mesh algorithms, the grid lines to overlap. I was sorry of time spent on implementing his algorithm, and so decided to work with it a bit more time before seeking in the literature other methods. It seemed one should apply the natural modification of his method, based on the classical problem of constrained minimization of the finite-dimensional function, approximating the functional of smoothness. I went to Sergey,

informed him of failure and my desire to modify his method. After listening to me, he took from the table a sheet of paper, where it was written the finite-dimensional function that further we suggested in many our papers. In couple of days, after realizing the algorithm of constrained minimization of that function, to my surprise I found that my problem had calculated thorough, since at every time moment the mesh consists of only convex quadrilaterals. I was more surprised observing that the Lagrange multipliers nearly did not change the initial value during iterations. And only then I saw the nice barrier property of Ivanenko's function which allows refusing the constrained minimization and introduction of the Lagrange multipliers. It is sufficient to realize the algorithm of unconstrained minimization taking as an initial guess a mesh of convex quadrilaterals. Then, after every iteration, a new mesh holds convexity of all quadrilateral cells.

The natural question appears: why did not Sergey himself see the barrier property of his function? The answer is in state of his health. Recently he defended the dissertation, spending a lot of energy on his nervous system. Later he even went to a hospital (due to overstrain), where I brought the manuscript of our first joint publication [3] so as he can read it and make improvements. Obviously, if a temporary poor health after defending the dissertation did not occur, he would do everything without my participation. He needed only to see carefully at his function or, at least, to observe the barrier property when constructing the grids."

By the time when (by virtue of age) Sergey linked to the problem of grid generation, some achievements have been already reached. Nevertheless there was a field on applying his talent. First of all, one should note the variational barrier method for curvilinear grid construction in domains with complex boundaries. At the first stage of the scientific work Sergey considered the problem of minimizing the finite difference function, approximating the well-known functional (see [5]) for the harmonic mapping of the computational physical domain onto a unit square

$$\int_0^1 \int_0^1 \frac{x_\xi^2 + y_\xi^2 + x_\eta^2 + y_\eta^2}{x_\xi y_\eta - y_\xi x_\eta} \, d\xi d\eta \, . \tag{1}$$

The Euler-Lagrange equations for the functional (1) have as a consequence the inverse Laplace equations

$$\begin{aligned} g_{22} x_{\xi\xi} - 2 g_{12} x_{\xi\eta} + g_{11} x_{\eta\eta} = 0 \, , \\ g_{22} y_{\xi\xi} - 2 g_{12} y_{\xi\eta} + g_{11} y_{\eta\eta} = 0 \, , \end{aligned} \tag{2}$$

where the coefficients g_{ij} are the metric parameters of the mapping sought

$$g_{11} = x_\xi^2 + y_\xi^2 \, , \quad g_{12} = x_\xi x_\eta + y_\xi y_\eta \, , \quad g_{22} = x_\eta^2 + y_\eta^2 \, . \tag{3}$$

The system (2) came into the grid numerical practice after appearing the paper [6]. However, soon it was found that the direct approximation of the equations (2), using the

simple finite difference relations, does not guarantee producing an unfolded mesh, despite it was substantiated in a differential formulation. Reasoning, that it is a consequence of an insufficient number of grid nodes and will be eliminated at mesh refining, was not confirmed and was refuted. Rather a simple example was suggested in the book [7], where the mesh is constructed in the square domain with a backstep [1] when consequently doubling the grid nodes number.

Note that this example does not exclude possibility that an approximation of the equations (2)–(3) with finite difference relations can guarantee that the produced mesh is nondegenerate. This example only illustrates (and confirms) the fact that the way of *discretization* is a substantial stage of the algorithm and is not reduced to formal changing the differential expressions by the finite difference relations. When performing a formal discretization, some important properties of the differential model can be lost.

The warrantee of mesh non-degeneracy with an arbitrary grid nodes number can be obtained by applying a special approximation of the variational functional (1), which uses the grids of convex quadrilaterals as an admissible set of meshes. Then the requirement of grid non-degeneracy is reduced to the condition of positivity for the area of four triangles into which a quadrilateral cell is partitioned by its two diagonals. This is a correct finite difference implementation of the conditions providing positivity of the Jacobian of the mapping sought.

On the boundary of admissible grid set, the integral sum, approximating (1), possesses an infinite barrier that guarantees producing an unfolded mesh in the result of optimization.

First this idea was realized by Sergey as a problem of constrained minimization, where the constrained for the set of admissible grids were introduced with the Lagrange multipliers. Its solution was a part of Sergey's candidate dissertation [1], defended in 1985, and it was also described in the preprint [2]. Further, in joint papers with A.A.Charakch'yan [3, 4] it was suggested an algorithm with using an unconstrained minimization of the discrete variational functional.

Within many years the problem of constructing various equations and variational functionals, allowing to produce unfolded grids, was of interest for many authors and stimulated appearing a lot of publications on this subject. In 2000, Sergey suggested to minimize the functional [9]

$$F = \frac{1}{2}\int_0^1\int_0^1 \frac{g_{11}G_{22} - 2g_{12}G_{12} + g_{22}G_{11}}{\sqrt{g_{11}g_{22} - g_{12}^2}\sqrt{G_{11}G_{22} - G_{12}^2}}\, d\xi d\eta \qquad (4)$$

for grid construction. Here G_{11}, G_{12}, G_{22} are the elements of symmetric positive definite matrices, defined at points of the unit square $Q : (0 \le \xi, \eta \le 1)$. The functional is minimized on the class of functions $x(\xi, \eta)$, $y(\xi, \eta)$, being the smooth extension of the functions x_b, y_b, given on the boundary, inwards the square Q. The latter execute the smooth one-to-one mapping of the square boundary ∂Q onto the boundary of the domain Ω, where the grid is to be generated.

[1] This example was independently and earlier considered in [8]

When defining the coefficients as

$$G_{11} = g_{11}\,, \quad G_{12} = g_{12}\,, \quad G_{22} = g_{22}\,, \tag{5}$$

the functional (4) attains its absolute minimum. Consequently, defining (5) the minimizer of the functional reproduces any invertible mapping. That is why it can be considered as an *universal* grid generator.

From the practical standpoint the assignment (5) seems fruitless (if there is a mesh, why does one need to make something else?). However, in the numerical practice of unsteady problems it plays a fundamental role, since the mesh of the preceding time step is a base for constructing the grid at the next time step (and these two meshes should differ only slightly). One of the works in the present book (see chapter 5) realizes this idea.

Capabilities, provided by various algorithms defining the metric tensor G_{ij}, are crucial for this approach. The functional (1), serving for producing the harmonic meshes, is a particular case of (4) when the metric G is Euclidean (G_{11}=G_{22}=1, G_{12}=0)

The method, suggested by Sergey, in general allows us to generate any mesh of a given structure, if the objective shapes of the grid cells are defined. The maximal advantage of this approach will appear when implementing in the interactive mode (dialogue system). It is realistic and acceptable, when constructing the grids in the domains with a given fixed boundary.

Some capabilities of controlling the mesh structure were suggested in [9]. For instance, they allow us to produce the meshes, close to orthogonal near the boundary and with a regulated cell size in the boundary cell layer, etc.

From the practical standpoint it is of importance that, when treating the set of admissible grids with convex cells, an arbitrary definition of positive definite matrices of the metric coefficients G_{ij} allows us to obtain a nondegenerate mesh delivering the minimum of the variational functional (4) at discrete implementation.

Thus, Sergey succeeded in selecting a complete class of invertible mappings of a square onto a given domain with a given mapping between boundaries, and, at the discrete level, in selecting a complete class of regular grids of convex quadrilaterals with given boundary nodes. These results were published in [10, 11].

Along with regular (structured) grids, Sergey also considered irregular (unstructured) meshes. His algorithms can be applied in this, more complicated, case as well. It is of importance in practical calculations of real industrial problems.

When solving the problems of mathematical physics in domains with a complex shape, there appears necessity to increase the accuracy of modeling owing to special grid construction. With this purpose the grid nodes should be distributed in such a way that node concentration corresponds to the subdomains where the solution sought undergoes great changing. The grids, possessing such properties, are said to be adaptive.

Beginning with 1990s, Sergey was engaged in constructing adaptive mesh algorithms, cf. [7]. The last result in this direction is an elegant solution, suggested in [9]. With purpose of adapting by a given control function f the functional (4) is written on the surface of the graph of this function. Initially, the idea to write the functional of smoothness on the graph

of a monitor (or control) function f was suggested in [12], and as a matter of fact Sergey extended this idea by applying the second metric G in the parametric domain, meanwhile in [12] the metric G is Euclidean. Besides, in numerical implementation, like in grid generation problems, the method of dicretizing the differential expressions plays the crucial role. To adapt, in the functional (4) instead of metric (3) it is introduced the new metric given on the surface of graph of f as follows

$$\begin{aligned} g_{11} &= x_\xi^2 + y_\xi^2 + (f_x x_\xi + f_y y_\xi)^2 , \\ g_{12} &= x_\xi x_\eta + y_\xi y_\eta + (f_x x_\xi + f_y y_\xi)(f_x x_\eta + f_y y_\eta) , \\ g_{22} &= x_\eta^2 + y_\eta^2 + (f_x x_\eta + f_y y_\eta)^2 . \end{aligned} \tag{6}$$

After selecting the control function f (that is a subject of a special discussion and depends on a concrete problem to be solved, cf. [13, 14]) and calculating its values in the grid nodes, the procedure of obtaining the adaptive mesh goes in the same manner that is defined in the course of the simple (without adaptation) grid generation algorithm. This is a significant advantage of this method over the methods, suggested by other authors. They, for example, suggest to include special terms (summands) into the variational functionals. These terms are responsible for adaptation. From the point of view of automation of modeling, in general, it is very difficult to define the weight coefficients of these additional terms. Unfortunate definition of the weight coefficients leads to failure in modeling. These terms can cause the resulting functional to lose ellipticity.

One more interesting result of Sergey's is the method of grid construction on surfaces [7]. Initially this work related to extension of the variational functional (1), then to the functional (4) with the second metric G, and to its adaptive variant (6) [15].

Since the mid 1990s, Sergey was engaged in developing variational barrier methods for constructing three-dimensional grids [7, 16]. This problem is always actual, because it allows us to solve real industrial problems (without idealizing, applied in the approximate 2D approach). This problem is of particular interest due to the modern powerful computers help the users to implement spatial modeling of physical problems.

First, it seemed that the scientific experience, gained when solving 2D grid generation problems, allowed to proceed immediately to 3D grid construction. However, situation proved not to be simple. Sergey's works on this subject as well as other authors have revealed the number of difficulties. In the papers [9, 15] and his last monograph [17] the matter of extending the variational method of grid generation for the n-dimensional and, in particular, three-dimensional case is considered.

Sergey referred the integrand in the functional (4) as the energy density of the mapping (whose discrete implementation delivers the mesh sought). He extended the conception of the energy density of the mapping to the case of a linear mapping of 3D Euclidean space (see also chapter 4 in this book). It turns out that, in contrast to the 2D case, in the 3D case the energy densities of the direct and inverse mapping are not equal. This causes various ways to formulate the variational barrier functionals for grid generation in the 3D case. In the paper [15] it revealed some more features which distinguish the multidimensional case (n>2) from the two-dimensional one.

The pass to the discrete variant of the 3D functional is difficult due to the following. Formulation of the mesh non-degeneracy criteria, which in the 2D case are very simple and graceful, is very complicated in the 3D case. Sergey was one of the first to execute some steps on formulating such criteria in [7]. In the present book this matter is considered in details in chapter 9.

Sergey approached the problem of constructing robust 3D grid generation methods and, undoubtedly, his ideas will be further developed.

Besides grid generation problems, since beginning his scientific career, Sergey was engaged in incompressible flow modeling, in particular in basins, lakes, rivers, sea gulfs, etc. One of the difficulties encountered in such problems is that they are characterized by complex morphometry exhibited in both bottom-shape irregularities and heavily indented coastlines. Fluids flows in geographical applications are described by nonlinear hyperbolic equations (shallow-water equations), and their solution in domains with complex geometry is a difficult task. In the 1980s, only a few attempts were made in the world to apply the finite element method (FEM) on nonuniform meshes to the unsteady problems in geophysical fluid dynamics, and Sergey was one of the pioneers in this field. First, the methods, developed by him, were applied to the simulation of tidal flows in Onega Bay, and then the were extended to other basins and applications. He faced a variety of complicated problems, such as FEM approximations of equations on curvilinear grids, the stability of approximations in time, etc. Sergey managed to overcome these difficulties in modeling the real problem of tidal flow. This made it possible to calculate and construct cotidal maps of Onega Bay, southeast Barents Sea, Chaivo and Pultin Bays in Sakhalin Island, Azov sea, etc., cf. [18, 19].

Flow simulation and pollution transport in reservoirs are another application, where Ivanenko's methods have been proved to be effective. In cooperation with Mosvodokanal ("Moscow Water Channel") Enterprise, Sergey simulated pollutant transport in the Ucha and Mozhaisk reservoir, which are important elements of the Moscow water supply system, a sophisticated hydraulic structure. Ivanenko's computational methods with a reduced numerical diffusion were applied to calculate accurately pollutant dispersion and pollutant conservations at water supply points under normal conditions and in the case of abnormally high discharges caused by malfunctions at sewage treatment plants. For solving nonlinear equations governing free-surface flows with large elevation heads. Sergey applied methods with moving curvilinear grids. He solved the complicated problem of flow through the spillway of the Rublev Dam and computed discharge ratings for various modes of Rublev reservoir operation, cf. [20, 21].

Sergey worked hard as a scientist and was a world-renowned specialist. His works are acknowledged in Russia and abroad. He was invited to lecture on grid generation at many foreign universities in USA, Finland, Mexico. Shortly before death, he gave a course of lectures at the Moscow Institute of Physics and Technology. Sergey participated in many international conferences, including those on grid generation. For example, since 1988 he was a member of the organizing committee and a constant participant of All-Union (later All-Russia) Biennial Conference on Problems of Grid Generation for Solving the Prob-

lems of Mathematical Physics. In 2001, 2002 Sergey organized and held the international seminar "Grid Generation: Theory and Applications". He was the heart of the specialists on grid generation in Russia and was in touch with many foreign scientists. Sergey could gather like-minded persons in any area of activity in which he was engaged. He initiated many studies in grid generation. For example, in the theory of mappings, Sergey stimulated a study on seeking the conditions provided that the mapping is a global homeomorphism, and attracted to this work Prof. N.A. Bobylev, see [22, 23].

He published more than 50 papers.

After Sergey's death, his last monograph was found in the electronic form on the home computer and published at the Computing Center [17].

The authors would like to thank Prof. A.A. Charakhch'an for his contribution to this text. We also used the material of the paper [24].

Bibliography

[1] Ivanenko S.A. Application of curvilinear grids in the finite element method for solving the equations of shallow-water. *Candidate's Dissertation in Physics and Mathematics.* Moscow, Computing Center of USSR Academy of Sciences, 1985. (in Russian).

[2] Ivanenko S.A. Curvilinear grid generation and its application in shallow-water modeling. Moscow, Computing Center of USSR Academy of Sciences, 1985. (in Russian).

[3] Ivanenko S.A. and Charakch'yan A.A. An algorithm for constructing curvilinear grids consisting of convex quadrangles. *Soviet Math. Dokl.*, **36**(1), 1988, p. 51.

[4] Ivanenko S.A. and Charakhch'yan A.A. Curvilinear grids of convex quadrilaterals. *USSR Comput. Maths. Math. Phys.*, **28**(2), 1988, pp. 126–133.

[5] Brackbill J.U. and Saltzman J.S. Adaptive zoning for singular problems in two dimensions. *J. Comput. Phys.*, **46**(3), 1982, pp. 342–368.

[6] Winslow A. Numerical solution of the quasi-linear Poisson equation in a nonuniform triangle mesh, *J. Comput. Phys.*, **1**(2), 1966, pp. 149–172.

[7] Ivanenko S.A. Adaptive-Harmonic Grid Generation. Moscow, Computing Center of Russian Academy of Sciences, 1997, 182 p. (in Russian).

[8] Knupp P. and Steinberg S. Fundamentals of Grid Generation, *CRC Press, Boca Raton, FL*, 1993.

[9] Ivanenko S.A. Control of cell shape in the construction of a grid. *Comp. Math Math. Phys.* **40**(11), 2000, pp. 1662–1684.

[10] Ivanenko S.A. Optimality principle for nondegenerate curvilinear grids. *Dokl. Math.*, **63**, 2001, p. 327.

[11] Ivanenko S.A. Existence of equations describing the classes of nondegenerate curvilinear coordinates in arbitrary domains. *Comp. Math Math. Phys.* **42**(1), 2002, pp. 43–48.

[12] Liseikin V.D. On generation of regular grids on n-dimensional surfaces, *USSR Comput. Maths. Math. Phys.*, **31**(11), 1991, pp. 47–57.

[13] Azarenok B.N. and Ivanenko S.A., Application of adaptive grids in numerical analysis of time-dependent problems in gas dynamics. *Comput. Math. Math. Phys.*, **40**(9), 2000, pp. 1330–1349.

[14] Azarenok B.N. and Ivanenko S.A. Application of moving adaptive grids for numerical solution of nonstationary problems in gas dynamics. *Intern. J. for Numer. Meth. in Fluids.*, **39**(1), 2002, pp. 1–22.

[15] Ivanenko S.A. Variational methids for adaptive grid generation. *Comp. Math Math. Phys.* **43**(6), 2003, pp. 793–806.

[16] Ivanenko S.A. Harmonic Mappings, *Chapt. 8 in Handbook of Grid Generation.* CRC Press, Boca Raton, Fl, 1999.

[17] Ivanenko S.A. Selected Chapters on Grid Generation and Applications. Moscow, Computing Center of Russian Academy of Sciences, 2004, 238 p.

[18] Ivanenko S.A. and Muratova G.V. Adaptive grid shallow water modeling. *Appl. Numer. Maths.* **32**(4), 2000, pp. 447–482.

[19] Ivanenko S.A., Edel'shtein K.K, and Patrik P.A., Spatial structure of wind flows in valley reservoir. *Meterol. Gidrol.* **89**(7), 2001, pp. 89–100. (in Russian).

[20] Ivanenko S.A. Simulation of water flow driven by a large free surface elevation head. *Comput. Maths. Math. Phys.* **40**(10), 2000, pp. 1509–1521.

[21] Belikov V.V., Kolesnikov Yu.M., and Ivanenko S.A. Mathematical modeling of passing the spring tide through a town pool of the river Moscow. *Water Resour.* **28**, 2001, pp. 556–572.

[22] Bobylev N.A., Ivanenko S.A., and Ismailov I.G. Some remarks on homeomorphysms. *Russian Mathematical Notes*. **60**(4), 1996, pp. 593–596.

[23] Bobylev N.A., Ivanenko S.A., and Kazunin A.V. Piecewise smooth homeomorpisms of bounded domains and their applications to the theory of grids. *Comp. Math Math. Phys.* **43**(6), 2003, pp. 772–781.

[24] Kerimov M.K. and Charakhch'yan A.A. In memory of Sergey Alexandrovich Ivanenko. *Comp. Math Math. Phys.* **44**(4), 2004, pp. 723–726.

INTRODUCTION

Grid generation tools play an extremely important role in computational technology. K. I. Babenko in his fundamental monograph *Principles of Numerical Analysis* (Nauka, Moscow, 1986) indicates that the methods of grid construction and node enumeration can be central and more important than error estimation methods.

Now, grid generation is undoubtedly an independent area of science, young and very fast developing. Already some monographs, concerning the questions of grid generation as a part of numerical modeling, and monographs devoted directly to grid generation as a subject, have appeared, e.g. *Numerical Solution of Multi-Dimensional Problems in Gas Dynamics* by S. K. Godunov (Ed.), A. V. Zabrodin, M. Ya. Ivanov, A. N. Kraiko, and G. P. Prokopov, (Nauka Press, Moscow, 1976) (French translation: Résolution Numérique des Problémes Multidimensionnels de la Dynamique des Gaz) (Mir, Moscou, 1979)), *Numerical Grid Generation: Foundation and Applications* by J. F. Thompson, Z. U. A. Warsi, C. W. Mastine (North Holland, 1985), *Fundamentals of Grid Generation* by P. M. Knupp and S. Steinberg (Springer, 1994), *Adaptive-harmonic Grid Generation* by S. A. Ivanenko (CC RAS, Moscow, 1997), *Grid Generation Methods* (Springer, 1999), and *A Computational Differential Geometry Approach to Grid Generation* (Springer, 2003) by V. D. Liseikin, *Selected Chapters on Grid Generation and Applications* by S. A. Ivanenko (CC RAS, Moscow, 2004), and some others (more full list of literature on grid generation see in the cite: http://www-users.informatik.rwth-aachen.de/roberts/literature.html).

There has appeared also the *Handbook of Grid Generation* by the edition of J. F. Thompson, B. K. Soni, and N. P. Weatherill (CRC Press, 1999), which has become the concentration of up-to-date technologies of grid generation.

This book gives the descriptions and examples of further development of some technologies presented in *Handbook of Grid Generation*. These are approaches by S. A. Ivanenko, and by A. F. Sidorov, O. B. Khairullina and O. V. Ushakova. The development of the ideas of these approaches one can find in chapter 4 by S. A. Ivanenko and B. N. Azarenok, in chapter 6 by A. F. Khairullin, O. B. Khairullina, and N. A. Artyomova, in chapter 10 by T. N. Bronina and O. V. Ushakova, and in chapter 11 by I. A. Gasilova.

Here, other approaches and methods of grid generation are also presented . These are chapter 1 by A. A. Charakch'yan, chapter 2 by S. K. Godunov, O. B. Feodoritova, and V. T. Zhukov, chapter 5 by G. P. Prokopov and chapter 7 by A. M. Sorokin and N. A. Vladimirova.

Two chapters are especially devoted to some general questions of grid generation. These are chapter 3 by V. M. Miklyukov and A. V. Lygin and chapter 9 by O. V. Ushakova.

There are also another two chapters: chapter 8 by N. N. Anuchina, V. I. Volkov, V. A. Gordeychuk, N. S. Es'kov, O. S. Ilyutina, and O. M. Kozyrev which gives brief description of a computer code for a three-dimensional simulation of the problems of the multimaterial hydrodynamics for which the grid generation algorithms described in chapter 10 were developed, and chapter 12 by B. N. Azarenok which represents the algorithm of a conservative three-dimensional interpolation of gas dynamics fields on a curvilinear grids. This algorithm and corresponding computer code represent also very important tool for a three-dimensional simulation, for examples, also just for that described in chapter 8.

The main attention of this book is paid to the problems of structured grid generation where grid nodes are organized into a regular structure (structured grids, regular grids), however two chapters concern to problems of unstructured grid generation where grid nodes do not form the regular structure and the connectivity between nodes can be particular for each node (unstructured grids, irregular grids). These are chapter 7 which considers the problems of anisotropic grid adaption for unstructured grids, and chapter 9 which considers the problem of nondegeneracy of all computational grids and, in particular, of unstructured grids.

The book is divided into two parts. The first part contains the description of different two-dimensional algorithms, and the second part three-dimensional algorithms.

Chapter 1 is devoted to a new grid generator based on minimization of a barrier-type finite difference function and its application in simulation of hydrodynamic flows with moving internal and external boundaries. The function combined a modified function of the known method by S. A. Ivanenko and A. A. Charakhch'yan with a variational form of a simple quasi-one-dimensional grid in which the grid lines of one family are straight, and the node distribution along these grid lines is prescribed.

First, both advantages and disadvantages of the method by Ivanenko and the author of this chapter are discussed, which is illustrated by numerical examples. Then, a special case of barrier-type functions considered earlier by Ivanenko is derived from another considerations. A minimization problem of quasi-one-dimensional grids and its regularization by the function are derived. Numerical examples illustrating advantages of the proposed method in comparison with the method by Ivanenko and the author for domains with deep protrusions at their boundaries are presented. The iteration procedure of the method based on a selection of an iteration parameter and of a subdomain of grid nodes for each iteration step is described. A numerical example demonstrating essential decrease in the computational cost for proposed iterations on subdomains of nodes is presented. The last section of this chapter is devoted to computations of shock wave flows in condensed matter with strong deformation of boundaries. An application package is described briefly. The considered problems are the instability of an interface between two media, which is interacted with a shock wave, shock compression of graphite in conical experimental assemblies and of a plate resting on a wedge of finite dimensions.

In chapter 2, an approach for quasi-isometric grid generation is presented. The main attention directs towards numerical features of the approach. A prime goal is to awake theoretical and computational interests to the problem of quasi-isometric parametrization of curvilinear regions. A few natural mathematical formulations of the extreme problems might be discovered and solved by using some modifications of the suggested parametrization. This parametrization is based on the fact, that under defined conditions a quadrangle on a smooth surface can be conformally mapped onto a quadrangle on a constant curvature surface.

The quasi-isometric mapping of a region (curvilinear quadrangle) onto the index unit square is introduced as inverse of the superposition of two mappings. The first mapping is quasi-isometric and conformal with respect to the metric chosen from a five-parameter

family of metrics defined in the parametric unit square. This metric represents the natural metric on the constant curvature surface (the Lobachevskian or Euclidean sphere or plane) such that geodesics in the parametric square are straight line segments. The second mapping maps the index unit square onto the parametric square. The superposition of these mappings is a unique solution of the variational problem given by a functional. The minimal value of the functional is the area the considered region.

The implementation of the quasi-isometric mapping technique is described. For minimization, it is used an iterative process consisting of three stages in according to three groups of functional parameters (unknowns).

A few numerical examples are given. To evaluate quality of the quasi-isometric grids and compare a few grids it is introduced a coefficient of quasi-isometry and a simple procedure for its computation is given.

Next chapter 3 is closely connected with the previous one. Authors give angle estimates under quasi-isometries of triangles in $\mathbb{R}^2$, study the distortion of a surface and the quasi-convexity of a domain.

Chapter 4 presents a variational approach of grid generation, based on the principle of the minimal energy density of a mapping, being an extension of the equidistribution principle from the one-dimensional case to two-dimensional. It is formulated in the both continuum and discrete approaches, and a complete proof is executed only for the discrete mapping.

This principle leads to a variational formulation of the elliptic grid generator. The main idea is the following. In every cell, it is introduced a local mapping of this cell onto another cell with a shape given in some manner or obtained using an algorithm by introducing additional information. In particular, as an objective shape one can use a shape of the corresponding grid cell, generated at the first stage. Thus, for every cell, the parameters, defining the shape which the cell should tend to, are given. For instance, in the triangle there are two parameters: two sides length ratio and angle between them. The shape measure is a quantity inverse to the energy density. Maximum of such a shape measure corresponds to the minimum of the energy density and it is attained if and only if the shape of the element is the same as that of in the corresponding objective shape. It is formulated a local objective function (i.e. energy density) and its minimization allows us to obtain a cell shape being homothetic to a given one. A global objective function (or energy density) is obtained by summing the local shapes over all grid cells. Grid correction is defined by minimizing the global energy density. This user-defined grid quality measure is implemented in the optimization-based method of grid generation and improvement. At the discrete level, this approach gives a guarantee that the mapping sought is invertible.

The energy density of grid deformation is a discrete functional with an infinite barrier at the boundary of the set of unfolded grids. The barrier property is very important in problems with moving boundaries and in moving adaptive grid technology, because such methods ensure generation of unfolded grids at every time step. The direct control of the cell shape is used to provide mesh orthogonality with a prescribed cell width near the boundary. Adaptation to the solution of the host equations can be realized as in the adaptive-harmonic

grid generator. In the 3D case, the well-known tetrahedron shape measure, such as the mean ratio, can be obtained from the energy density of the deformation of the equilateral tetrahedron.

In the chapter 5, the variational functionals for calculating two-dimensional difference grids, applied when solving unsteady problems of mathematical physics with moving boundaries, are suggested. Their coefficients are defined by the metric parameters of the grid, obtained at the preceding time step, and corrected additionally using other variational functionals taken with weight coefficients, being proportional to the time step. Discretization of the functionals and iteration procedures to the obtained system of equations are considered.

Grid generation problem is not a final purpose and only a means of solving complex physical problems. Hence, the choice of correcting variational functionals should be executed so as to produce meshes suitable for solving the main flow problem.

When solving unsteady problems, some difficulties can arise due to the high grid node velocity. This can be caused by necessity, when modeling, to vary grid generation algorithms and rather a slow convergence rate of the iteration procedures, used in solving complicated nonlinear system of equations. The present approach allows us regulating the grid node velocity.

Chapter 6 is a continuation of a part of work presented in *Variational methods of construction of optimal grids* by O. B. Khairullina, A. F. Sidorov, and O. V. Ushakova of *Handbook of Grid Generation* ed. by J. F. Thompson, B. K. Soni, and N. P. Weatherill (CRC Press, Boca Raton etc.,1999, pp. 36-1 - 36-25). In this chapter, authors consider questions related to constructing optimal block-structured curvilinear grids in two- and three-dimensional simply and multiply connected domains of complex geometries. Optimality criteria are criteria of closeness of grids to uniform ones with respect to the distance between nodes and to orthogonal ones at the intersection points of coordinate lines.

A survey of the authors works on the automatic generation of optimal multiblock curvilinear grids (dividing of domain into blocks, generation of the initial approximation without self-intersecting cells, organization of block overlapping, grid optimization) of large size (about hundreds of millions of nodes) in two-dimensional domains of an arbitrary connectivity and configuration is presented. Sequential and parallel algorithms are described and compared. Methods of the automatic grid quality control and improvement are also described. Method presented in this chapter permit to construct grid of good quality with a large number of nodes under maximal automation of calculation and using minimum initial information. The experience in using optimal two-dimensional grids shows that the grids constructed have high computational and approximative advantages. Their application allows to solve a number of complicated gas dynamics problems with high degree of accuracy.

In this chapter, new algorithm of automatic generation of three-dimensional curvilinear block-structured grids in a wide class of axisymmetric domains of any connectivity is also described in the case when the rotation angle of the directrix around the axis is equal to 360 degrees. A three-dimensional grid is constructed by rotation of an optimal grid in

two-dimensional radial section of the domain about the symmetry axis and further reconstruction of the grid in blocks adjacent to the symmetry axis. The constructed grids do not have singularities. When solving problems on these grids, connections between blocks are automatically organized. The results of three-dimensional grid generation and its using for calculation of the flows around a body by subsonic and supersonic flow are presented.

Since this chapter contains the material concerning both the problems of two and three-dimensional grid generation, it placed at the and of part I.

Chapter 7 transfers the reader into the area of unstructured grid generation. It contains two methods of anisotropic grid adaptation. Anisotropic grids consist of cells with arbitrary stretchings in different directions. This feature enables to produce the optimal grids adaptive to solutions with large gradients. The adaptive grid cell stretching in different directions is relied on the solution changes in these directions. The first method constructs the constrained Delaunay triangulations with specified cell stretching in the regions with large gradients of flow parameters. The other algorithm employs the edge refinement technique governed by interpolation based error indicators.

Next part is devoted to the problems of three-dimensional grid generation. It starts with the description of the computer coder MAH-3 for three-dimentional numerical simulation of multi-component vortex flows (chapter 8). This chapter gives an examples of mathematical modelling in which grid generation methods and their tools plays very important role. Examples of such methods and special tools were presented in next four chapters.

The area of investigation of chapter 8 is dynamics of multi-component media with strong interface deformations. This field is very important field of applied researches in many scientific areas. Strong interface deformations, which are caused by vortex, stream flows and large transfer of matters, appear in different practical problems. Limited possibilities of analytical treatment for such problems make necessary to use numerical simulation in researches.

In modern computational gas dynamics, there are several directions to develop numerical methods for calculations of multi-component flows in which the interfaces may be strong distorted up to destruction. The most perspective approach for effective mathematical modelling is a development of adaptive, i.e. adjusting to flows, computational algorithms.

The difference method implemented in the MAH-3 code allows to calculate complicated non-stationary hydrodynamic flows from wide range, in particular, multi-component vortex flows with strong interface deformations, their damage, loss of topological structure of the physical fields, and mixing of substances. In the adaptive algorithms, the grid generation must take in account of the domain geometry and character of processes there. Algorithms of the grid generation and program tools for the geometrical domain description are widely used in mathematical modelling of various mathematical physics problems. The adaptive computational method using a priori information about the studied flows, can apply algorithms of MAH-3 code in various combinations and at different stages of modelling for effective solution with the accuracy needed.

To elaborate grid generation algorithms for three-dimensional simulation, one need to

have methods for estimating quality of three-dimensional grids. The main requirement to a grid is its nondegeneracy. How to estimate the nondegeneracy of computational grids of different types and given in different ways, is considered in chapter 9. In this chapter, the concept of grid nondegeneracy is strictly formulated. Formulation of three special theorems guaranteeing nondegeneracy of grids are given. Investigations with the purpose to obtained such theorems were initiated by S. A. Ivanenko and he was one of the co-authors of the corresponding papers containing the proof of such theorems. In this chapter, we formulate only the most important theorems for the theory and practice of grid generation. On the basis of such theorems, nondegeneracy conditions are found for hexahedral grids, for grids composed of pyramids and prism. Nondegeneracy conditions are obtained also for cells given by Bernstein–Bezier polynomials. Since in some cases (especially for complicated configurations of the domains), degenerate cells are admissible, this question is also discussed. In this chapter, one can find new formulas of the volume for hexahedral cells, pyramids and prisms with ruled faces. The distinctive feature of such formulas is that the volume of complicated by their forms cells having in general case non-planar faces is calculated in terms of the volumes of tetrahedrons with planar faces. This gives the opportunity of the essential simplifications of the numerical algorithms on grids composed of such cells. A concrete example of such simplification, one can find in chapter 12.

Examples of three-dimensional grid generation algorithms used while modelling the phenomena, for which MAH-3 code was developed, are described in chapter 10. Presented investigations are also the continuation of work *Variational methods of construction of optimal grids* by O. B. Khairullina, A. F. Sidorov, and O. V. Ushakova of the book *Handbook of Grid Generation*. In chapter 10, the new algorithms for construction of three-dimensional optimal curvilinear grids are described. One can find there the variational continuous and discrete functionals for construction of three-dimensional optimal grids, the distinctive feature of the method which consists in a special way of formalization of such optimality criteria as a uniformity of a grid, the description of the numerical algorithms and examples of application of a method to the calculation of grids in the volumes of revolutions.

In chapter 11, the algorithm for generating grids in the pipeline domains is considered. The cross section of a pipeline is a simply connected domain of a star type. The plane directrix curve consists of straight line segments and arcs of circles. The idea of the algorithm consists in generating optimal grid in cross section which uniformly moves along directrix and rotates through an angle depending of the geometry of a directrix. In constructing three-dimensional grids, geometrical approach with using special R-functions and variational approaches are employed.

Next, chapter 12 is closely connected with the previous four chapters. While simulating on different hexahedral grids, one need to interpolate the gas dynamics parameters from one grid onto another. The first-order conservative remapping algorithm is described in this chapter. Conservative remapping is reduced to determining the volume of the overlapping figure between the old mesh cells and new mesh cells. A hex cell with ruled faces is substituted for two dodecahedrons with planar facets, and, thus, the problem of constructing the overlapping figure between the dodecahedrons is considered. The overlapping figure of

two dodecahedrons is reduced to constructing the polygonal intersection line $\mathcal{L}_{on}$ between the boundaries of these 12-faceted cells. The polygonal line $\mathcal{L}_{on}$ is a closed contour, every segment of which is the intersection line of two triangle facets. The overlapping figure is a polyhedron with planar facets. Every its facet is the polygon, carved by $\mathcal{L}_{on}$ on a triangle facet of the dodecahedron. A union of all such polygons is the overlapping figure boundary. The overlapping figure volume is calculated through the surface integral over its boundary. For the particular cell of the new mesh, it applied the optimal filtering algorithm for the old grid cells so as to proceed only the old cells with nontrivial intersection.

Part I

TWO-DIMENSIONAL ALGORITHMS

Chapter 1

AN ELLIPTIC BARRIER-TYPE GRID GENERATOR FOR PROBLEMS WITH MOVING BOUNDARIES

Aleksandr A. Charakhch'yan
Dorodnicyn Computing Center of the Russian Academy of Sciences

A new grid generator based on minimization of a barrier-type finite difference function and its application to simulation of hydrodynamic flows with moving internal and external boundaries are considered. The function combines a modified function of the known method by Ivanenko and the author with a variational form of a simple quasi-one-dimensional grid in which the grid lines of one family are straight, and the node distribution along these grid lines is prescribed. Numerical examples illustrating advantages of the proposed method in comparison with the method by Ivanenko and the author for domains with deep protrusions at their boundaries are presented. An iteration procedure of the method based on a selection of an iteration parameter and of a subdomain of grid nodes for each iteration step is also described. A numerical example demonstrating an essential decrease in the computational cost for proposed iterations on subdomains of nodes is presented. An application package for computing shock wave flows in condensed matter with strong deformation of boundaries is described briefly, and some recent computations of such sort of problems are discussed.

1.1 Introduction

Elliptic problems are often used as leading considerations for generating grids in domains with given boundary nodes. Only the two-dimensional case will be considered in this paper. Apart from the (x, y) plane where a connected domain Ω is given, the ancillary plane (ξ, η) with a rectangle

$$0 \le \xi \le a, \quad 0 \le \eta \le b, \tag{0.1}$$

and an elliptic problem for the map $x(\xi, \eta), y(\xi, \eta)$ from the rectangle onto Ω with the given one-to-one map of boundaries are considered. The desired grid $(x, y)_{ij}$ is an approximate numerical solution of the problem on the rectangular grid (ξ_i, η_j).

Most simple elliptic equations which are Laplace's ones

$$x_{\xi\xi} + x_{\eta\eta} = 0, \quad y_{\xi\xi} + y_{\eta\eta} = 0,$$

are unsuitable for grid generation since the map $x(\xi, \eta), y(\xi, \eta)$ is non-one-to-one even for simple domains Ω (see [1]). As a result, degenerate grids with self-intersecting cells are generated. This is not the case for Laplace's equations in x, y variables

$$\xi_{xx} + \xi_{yy} = 0, \quad \eta_{xx} + \eta_{yy} = 0, \tag{0.2}$$

which take a quasilinear form after transforming to independent variables ξ, η, and for which the map $x(\xi, \eta), y(\xi, \eta)$ is always one-to-one. A grid generator based on (0.2) was developed by Winslow [2]. Then a set of grid generators based on elliptic problems ensuring the one-to-one map was developed [3,4]. Nevertheless, if the given configuration of the boundary nodes is such that coordinate lines of the grid have to be bent strongly to generate a satisfactory grid, even the methods [3,4] give degenerate grids unsuitable for computations. The reason is that the grid is a solution of the system of algebraic equations which properties are determined by the elliptic problem only in the limit of infinitely many nodes.

In 1985 Ivanenko suggested to bound the set of available grids by the set of grids consisting of only convex quadrangles. At first his idea had the form of a constrained minimization of a finite - difference function with Lagrangian multipliers as additional independent variables [5]. Later this idea was transformed in the barrier-type grid generator [6] based on an unconstrained minimization of a function dependent only on the coordinates of the grid nodes. This method guarantees the convexity of all quadrilateral grid cells (with the natural exception of "corner" cells that cannot be convex because of the position of the boundary nodes) at each iteration step, practically for any realistic distortion of boundary lines while the number of grid nodes and the structure of the grid remain fixed.

Time-dependent hydrodynamic problems with strong deformation of boundaries between media are a natural field of application of the method [6]. Currently, problems of this kind are most often solved by "particle-in-cell" methods or by free-Lagrangian methods based on irregular grids with variable difference scheme patterns. The grid generator [6] significantly extended the possibilities of applying methods [3], which are based on regular grids with an explicit specification of boundaries between media as some of the grid lines, to the problems mentioned above. All three approaches have certain advantages and shortcomings, which are not discussed in this paper.

As an example of applying the method [6], consider simulation of a collision of a thin foil with a cone target. Fig. 1.1a shows the initial shape of the foil in cylindrical coordinates, and Fig. 1.1b shows its shape at some time after the collision. The method [6] enables one to construct, for the domain shown in Fig. 1.1b, a grid of convex quadrangles that has the same structure as the natural grid for the domain shown in Fig. 1.1a, i.e., a grid whose corner nodes are the vertices A, B, C, and D. Fig. 1.2 shows a fragment of this grid in the vicinity of the node E, where the grid lines bent most sharply. The accuracy of the solution obtained on such a grid is beyond the scope of this paper. We only mention that the very

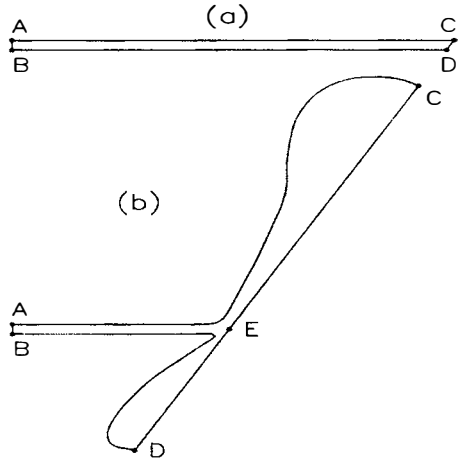


Figure 1.1: Initial (a) and typical (b) domains in simulating collision of a thin foil with a cone target.

possibility to obtain a numerical solution implies the possibility to analyze its accuracy by comparing results obtained on grids with various number of nodes.

Let us consider the possibility to use block grids for the above problem. Apparently, for the first time, block grids were described in [3]. A domain is decomposed into blocks by several lines specified *a priori*. In each block, a grid of its own structure determined by the indexing of boundary nodes is constructed. At present, block grids are widely used (see, e.g., papers on program packages for construction of structured grids in [7]). In constructing a block grid for the domain shown in Fig. 1.1b, it is natural to use the three blocks corresponding to both "tongues" and the central part. The grid lines then bend not too sharply inside each block, and not only the method [6], but also methods described in [3,4] can be used to construct the block grid. For numerical solution of a problem with unmoving boundaries in the domain from Fig. 1.1b, the block grid is possibly more suitable then the grid from Fig. 1.2. However, for the considered problem, implementation of computations on the block grid involves serious difficulties. Indeed, the number of boundary nodes inside a tongue must increase with extending the tongue extends. For this reason, the grid line that separates the blocks cannot pass through the same boundary node during the computation. This leads to the necessity of designing an algorithm for periodically changing the structure of the blocks. This algorithm involves cumbersome remap procedures between grids of different structures. At the same time, it is not clear, whether this method can give a more accurate numerical solution then that on the grid shown in Fig. 1.2, because the accuracy can be lowered by frequent use of the remap procedure. We also remind the reader of the strong competition with the free-Lagrangian methods in solving problems of this class. These considerations make us regard the prospects of applying block grids in situations where the problem requires frequently changing the structure of blocks with pru-

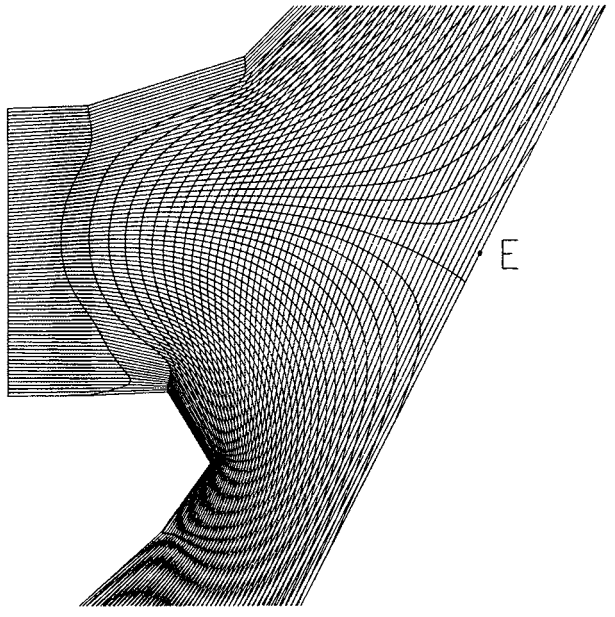


Figure 1.2: The fragment of the grid obtained by [6].

dent pessimism. This is confirmed by an analysis of papers from [7]: among all the papers dealing with block grids, this author could not find a single one where moving boundary problems were mentioned as a possible field of application.

The method [6] has disadvantages which have stimulated development of a new grid generator based on [6] and presented in this paper. One disadvantage is as follows. Inside a protrusion of the domain boundary, the nodes lying on the grid line closest to the boundary may be separated from it by a considerable distance. Accordingly, the cells generated inside the protrusion may be too large. This disadvantage played a minor role in problems such as shown in Fig. 1.1, because the protrusion (the tongue in Fig. 1.1) is bounded by two grid lines belonging to different families (with $i = const$ and $j = const$). In this case, the cell size inside the protrusion is primarily determined by the distribution of nodes along the boundary lines. In problems of a different type, such as the instability of an interface between media, the protrusion is bounded by a grid line belonging to a particular family, and the disadvantage appears. This is demonstrated in Fig. 1.3 where a domain with a protrusion and a fragment of the grid are presented. One can easily see very large cells in the protrusion.

As to the procedure recommended in [6] for eliminating this disadvantage, in which the value of a certain monitor function is specified in each cell, it was found to be inefficient, because the choice of such a monitor function for every new problem requires a substantial effort increasing with the number of grid nodes.

In 1996 V.M.Us'kov [8] noted one more disadvantage of [6] connected with grids

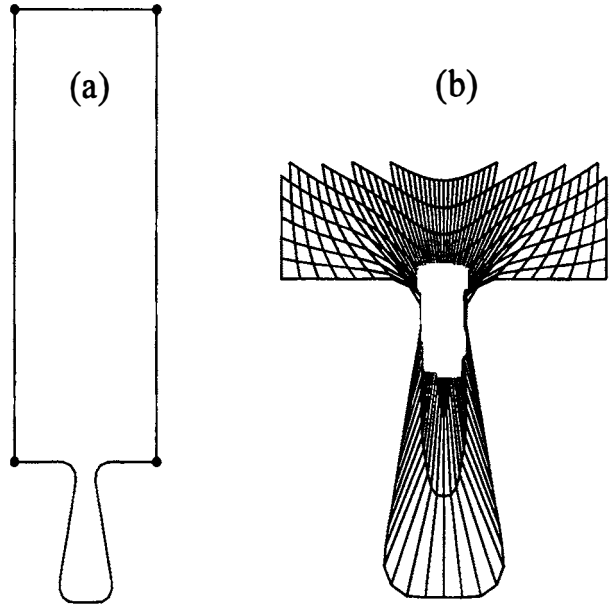


Figure 1.3: A domain (a) and the fragment of the grid (b) computed with [6].

of elongated quadrangles and suggested a new, more suitable for such grids, barrier-type method.

Another disadvantage of [6] is the law rate of convergence of the iterative procedure. As was noted in [3], this disadvantage manifests itself most explicitly in computing time-dependent problems with moving boundaries, when a grid must be generated at each time step, and a reasonable limit has to be imposed on the number of steps in an iterative grid generation cycle. On the other hand, the inner nodes of a substantially "underiterated" grid tend to lag behind the motion of flow boundaries. Very large cells may be generated near some boundary segments, in which case the numerical accuracy deteriorates. Near other boundaries, very small cells can appear, and the corresponding time step becomes unreasonably small.

We illustrate this disadvantage by the following simple example. Consider a rectangular domain with grid nodes uniformly distributed along all its boundaries, as shown in Fig. 1.4. The most suitable for such a domain is, as a rule, a rectangular grid consisting of straight lines parallel to the boundaries. Such a grid is a solution of the algebraic equations of the method [6]. Suppose that all the boundary grid nodes are moved downward by some step, while the inner grid notes remain fixed. Then the vertical grid lines do not change, while the horizontal lines move upward, and kinks arise on them near the lateral boundaries, as shown in Fig. 1.4a. This grid is taken as the initial one in the method of [6]. Fig. 1.4b shows horizontal lines of the grid after 50 iterations. It is evident that the effect of moving boundaries is still considerable, which may lead to a noticeable loss of accuracy as compared to

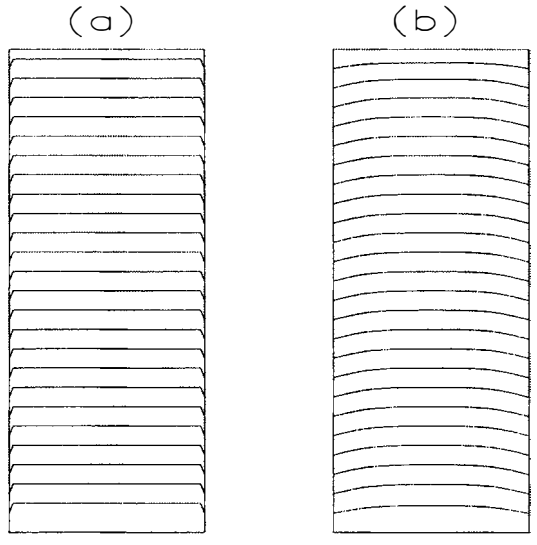


Figure 1.4: Horizontal lines of the initial grid (a) and of the grid after 50 iterations with the method [6].

computations on the rectangular grid.

In this paper we present a new grid generator based on minimization of a finite difference function. The function combined a modified function of the method [6] with a variational form of a simple quasi-one-dimensional grid in which the grid lines of one family are straight, and the node distribution along these grid lines is prescribed. Section 1.2 is devoted to a modification of the function [6] which is a special case of functions considered earlier by Ivanenko [9]. In Section 1.3, we derive algebraic equations of the method and present some numerical examples. An iteration procedure is considered in Section 1.4. In Section 1.5, we discuss some recent computations of shock wave flows in condensed matter.

1.2 Barrier-Type Function

The considered problem of grid generation is as follows. The grid

$$G = \{(x, y)_{ij},\ i = 0, \ldots, N;\ j = 0, \ldots, M\} \tag{1.1}$$

is to be constructed in the domain Ω with given coordinates $(x, y)_{i0}$, $(x, y)_{iM}$, $(x, y)_{0j}$ and $(x, y)_{Nj}$ of the boundary notes. Consider the cell $i + 1/2, j + 1/2$ consisting of the nodes $(x, y)_{i,j}$, $(x, y)_{i,j+1}$, $(x, y)_{i+1,j+1}$, $(x, y)_{i+1,j}$. The cell vertices are numbered from 1 to 4 in the clockwise direction. The node (i, j) corresponds to the vertex 1, node $(i, j + 1)$ to vertex 2 and so on. Each vertex is associated with a triangle: vertex 1 with $\triangle_{412}$, vertex 2 with $\triangle_{123}$ and so on. The doubled area J_k, $k = 1, 2, 3, 4$, of these triangles is introduced

as follows

$$J_k = \Delta x_{k+1}\Delta y_k - \Delta x_k \Delta y_{k+1}, \tag{1.2a}$$

$$\Delta x_k = x_k - x_{k-1}, \quad \Delta y_k = y_k - y_{k-1}, \tag{1.2b}$$

where one should put $(x, y)_0 = (x, y)_4$, $(x, y)_5 = (x, y)_1$. The system of inequalities

$$[J_k]_{i+1/2,j+1/2} > 0, \quad k = 1, 2, 3, 4; \; i = 1, \ldots, n-1; \; j = 1, \ldots, m-1$$

defines a grid consisting of only convex quadrangles. The set of such grids is called a convex grid set and denoted by D.

Return to the elliptic problem (0.2) on the parameter rectangle (0.1). After transforming to independent variables ξ, η, its variational form is as follows

$$\int_0^a \int_0^b \frac{x_\xi^2 + y_\xi^2 + x_\eta^2 + y_\eta^2}{x_\xi y_\eta - x_\eta y_\xi} d\xi d\eta = \min. \tag{1.3}$$

We put $a = N$, $b = M$ and introduce the rectangle grid $\xi_i = i$, $\eta_j = j$. Then, at each triangle k of the grid G, the integrand in (1.3) is evaluated as

$$\varphi_k = \frac{\Delta x_k^2 + \Delta y_k^2 + \Delta x_{k+1}^2 + \Delta y_{k+1}^2}{2J_k}, \tag{1.4}$$

where the factor 1/2 introduced to simplify formulas for derivatives of φ_k. The discrete analog of the doubled integral in (1.3) takes the form

$$I_{(6)} = \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} \sum_{k=1}^{4} [\varphi_k]_{i+1/2,j+1/2}. \tag{1.5}$$

In the method [6], the grid G is generated from an iteration process for solving the problem $I_{(6)}(G) = \min$.

The function $I_{(6)}$ has the following disadvantage. Suppose that Ω is a rectangle and the distributions of nodes along its opposite sides are the same. Then, a rectangular grid $G_\perp$ can be generated. Normally, this grid is best suited for such a domain. However, $I_{(6)}(G_\perp) = \min$ only when the nodes are distributed uniformly along the all domain boundaries.

We modify function (1.5) as follows. One can consider the grid G as two families of vectors $\Delta r_{i,j}^j = (x_{i+1,j} - x_{i,j}, y_{i+1,j} - y_{i,j})$, $\Delta r_{i,j}^i = (x_{i,j+1} - x_{i,j}, y_{i,j+1} - y_{i,j})$. Introduce two families of the parameters of the method $d_{i,j}^j$ and $d_{i,j}^i$ having the meaning of the desirable length of the corresponding vector. The parameters of the boundary lines $d_{0,j}^i$, $d_{N,j}^i$, $d_{i,0}^j$ and $d_{i,M}^j$ are computed. Other values are determined by the interpolation

$$d_{i,j}^i = ((N-i)d_{0,j}^i + id_{N,j}^i)/N, \quad d_{i,j}^j = ((M-j)d_{i,0}^j + jd_{i,M}^j)/M.$$

Using the correspondence between the indexes i, j of the grid nodes and the index k numbering the vertexes of the cell $i+1/2, j+1/2$, one can define the parameter d_k for the each

side of the cell as equal to one of the values $d^j_{i,j}$ or $d^i_{i,j}$. The function $I(G)$ is defined by (1.5), (1.4), (1.2a), but, instead of Δx_k and Δy_k from (1.2b), we use

$$\Delta\bar{x}_k = \Delta x_k/d_k, \quad \Delta\bar{y}_k = \Delta y_k/d_k.$$

Let us return to the example with the rectangle domain Ω. In contrast to the function $I_{(6)}$, $I(G_\perp) = \min$ for any $G_\perp$. Indeed, as pointed out by Ivanenko, $\varphi_k \geq 1$ for $J_k > 0$. On the other hand, $\varphi_k = 1$ for $G_\perp$ since the numerator in (1.4) is the sum of two moduli squared of unit vectors, and the denominator is the doubled vector product of two unit orthogonal vectors.

Finally, in the terms of Δx_k and Δy_k from (1.2b), the function $I(G)$ takes the form (1.5) where

$$\varphi_k = \frac{\alpha_k(\Delta x_k^2 + \Delta y_k^2) + \alpha_k^{-1}(\Delta x_{k+1}^2 + \Delta y_{k+1}^2)}{2J_k}, \alpha_k = d_{k+1}/d_k. \tag{1.6}$$

So as the function $I_{(6)}(G)$, $I(G)$ possesses the following barrier property. If $G \to \partial D$ for $G \in D$, where ∂D is the boundary of the set of convex grids D, i.e. if at least one of the quantities J_k tends to zero for some cell while remaining positive, then $I(G) \to +\infty$. Therefore, if the set D is not empty, the problem $I(G) = \min$ has at least one solution which is a convex grid. To find it, one must first obtain a certain initial grid $G_0 \in D$, and then use some method of unconstrained minimization. Due to the infinite barrier on the boundary of the set D, each step of the method can be chosen so that the grid always remains convex.

For the following (see Section 1.4), we are required the derivatives $\partial I/\partial x_{ij}$, $\partial I/\partial y_{ij}$, $\partial^2 I/\partial x_{ij}^2$, $\partial^2 I/\partial y_{ij}^2$, $\partial^2 I/\partial x_{ij}\partial y_{ij}$, and also an evaluation of the value of magnitude of the terms involved in $\partial I/\partial x_{ij}$ and $\partial I/\partial y_{ij}$. The latter is denoted by g_{ij}. Each of the derivatives is the sum of twelve terms, in accordance with the number of triangles containing the given node as a vertex. Arrays storing the derivatives and the quantity g_{ij} are first cleared, and then all grid triangles are scanned and the appropriate derivatives of (1.6) and a contribution in g_{ij} add to the relevant elements of the arrays. To compute the derivatives, we use recurrent formulas, for example

$$\frac{\partial\varphi_k}{\partial x_{k-1}} = \varphi_{kx} = -\frac{\alpha_k\Delta x_k + \varphi_k\Delta y_{k+1}}{J_k}, \quad \frac{\partial^2\varphi_k}{\partial x_{k-1}^2} = \frac{\alpha_k - 2\varphi_{kx}\Delta y_{k+1}}{J_k};$$

$$\frac{\partial\varphi_k}{\partial y_{k-1}} = \varphi_{ky} = \frac{-\alpha_k\Delta y_k + \varphi_k\Delta x_{k+1}}{J_k}, \quad \frac{\partial^2\varphi_k}{\partial y_{k-1}^2} = \frac{\alpha_k + 2\varphi_{ky}\Delta x_{k+1}}{J_k};$$

$$\frac{\partial^2\varphi_k}{\partial x_{k-1}\partial y_{k-1}} = \frac{\varphi_{kx}\Delta x_{k+1} - \varphi_{ky}\Delta y_{k+1}}{J_k}.$$

Basing on the form of $\partial\varphi_k/\partial x_{k-1}$ and $\partial\varphi_k/\partial y_{k-1}$, we use the expression

$$\frac{\alpha_k\sqrt{\Delta x_k^2 + \Delta y_k^2} + \varphi_k\sqrt{\Delta x_{k+1}^2 + \Delta y_{k+1}^2}}{J_k},$$

which is invariant under rotations in the plain (x, y), as the corresponding contribution in g_{ij}.

In practical implementation, an arbitrary set of grid nodes can be marked as movable during iterations, while all other nodes are considered as stationary. All the terms in the function (1.5) which become independent on movable nodes are excluded from computations. Since the boundary nodes are always marked as stationary, four terms in (1.5) corresponding to "corner" triangles are always excluded from computations. As a result, the method becomes applicable to those domains for which the angle between two intersecting boundaries is greater than or equal to π, despite the fact that the corresponding grid cell becomes non-convex independently on the inner nodes positions.

One is led to a natural question: does the function $I(G)$ remove the aforementioned "expulsion" of the grid out of a protrusion due to minimization of $I_{(6)}$? The answer to this question is negative, because this effect takes also place for domains with boundary segments of equal length, in which case the functions I and $I_{(6)}$ are identical since $d_k = const$. Let us return to the example presented in Fig. 1.3. The dots in Fig. 1.3a correspond to the corner nodes $(0, 0)$, $(0, M)$, $(N, 0)$, and (N, M). Boundary nodes are distributed uniformly along each of four boundaries, and the lengths of segments lying on the left, right and bottom boundaries are equal. Thus, the difference between I and $I_{(6)}$ is due only to the top boundary. By this reason, the difference between the grids obtained by minimization of the functions I and $I_{(6)}$ near the protrusion located far from the top boundary is insignificant, and the fragment of the grid shown in Fig. 1.3b is also related to the function I.

1.3 New Grid Generator

Here we use the symbol $\vec{r}_{i,j} = (x, y)_{i,j}$ for notational brevity, and the formulas are written in the vector notation.

The simplest way to construct the grid (1.1) is as follows (see [3]). Consider a family of lines, for definiteness, those corresponding to fixed values of the index j. The lines of this family are taken to be straight lines, with grid nodes distributed along them in some reasonable way. First, the quantities s_i^0 in the boundary $j = 0$ are computed:

$$s_0^0 = 0,\ s_{i+1}^0 = s_i^0 + \mid \vec{r}_{i+1,0} - \vec{r}_{i,0} \mid,\ i = 1, \ldots, N-1.$$

The monotonically increasing sequence

$$t_i^0 = s_i^0 / s_N^0,\ i = 0, \ldots, N,\ t_0^0 = 0,\ t_N^0 = 1$$

is called the arrangement law. The arrangement law t_i^M along the boundary line $j = M$ is defined analogously. The arrangement law along the other lines of the family is determined by the linear interpolation $t_i^j = [t_i^0(M - j) + t_i^M j]/M$. Then, the inner grid nodes are determined as follows:

$$\vec{r}_{i,j} = \vec{r}_{0,j} + t_i^j(\vec{r}_{N,j} - \vec{r}_{0,j}),\ i = 1, \ldots, N-1,\ \ j = 1, \ldots, M-1. \qquad (2.1)$$

We call such grids quasi-one-dimensional. Their obvious advantage is that no iteration is required for their construction, while their obvious disadvantage is a narrow range of applicability. At the same, for many problems of practical importance, quasi-one-dimensional grids fail only in relatively small subdomains. This occurs, in particular, for the domain shown in Fig. 1.3. Thus, it seems reasonable to construct elliptic grid generators that correct quasi-one-dimensional grids only (when possible) in the subdomains where the quasi-one-dimensional grid fails. Moreover, running ahead, it is worth noting the high quality of the resulting grid in the subdomains where the grid lines become curvilinear, because the given arrangement law is approximately satisfied in these subdomains.

The possibility of using quasi-one-dimensional grids in the construction of elliptic grid generators is based on the following minimization problem, which is equivalent to equations (2.1). Subtracting equations (2.1) pairwise for each two nearest values of index i yields

$$\vec{u}_{i,j} = q_{i,j}(\vec{r}_{i,j} - \vec{r}_{i+1,j}) + q_{i-1,j}(\vec{r}_{i,j} - \vec{r}_{i-1,j}) = 0, \qquad (2.2)$$
$$i = 1, \ldots, N-1, \quad j = 1, \ldots, M-1,$$

where $q_{i,j} = 1/[N(t^j_{i+1} - t^j_i)]$. Here, it was taken into account that (2.1) formally holds even on the boundary, when $i = 0$ and $i = N$. The coefficients $q_{i,j}$ are determined up to a constant factor, which is taken so that $q_{i,j} = 1$ under the uniform arrangement law $t^j_i = i/N$.

Consider the function

$$Q(G) = \sum_{i=0}^{N-1} \sum_{j=1}^{M-1} q_{i,j} \mid \vec{r}_{i+1,j} - \vec{r}_{i,j} \mid^2 /2, \qquad (2.3)$$

which depends on the coordinates of all inner grid nodes. Taking the first and second derivatives of this function, it is possible to demonstrate that equations (2.2) and, consequently, (2.1) are a solution of the problem $Q(G) = \min$ for $q_{i,j} > 0$.

Consider the function

$$J(G) = Q(G) + \varepsilon\sigma I(G), \qquad (2.4)$$

where ε is a parameter, σ is the average area grid of a grid cell equal to the domain area divided by the number of cells, $I(G)$ is defined in Section 1.2. The constant factor σ was introduce into (2.4) to obtain a dimensionless parameter ε, since $Q(G)$ has the dimensionality of length squared, and $I(G)$ is dimensionless.

Recall that the function $I(G)$ has the infinite barrier on the boundary of the set of convex grids. Clearly, the function $J(G)$ defined by (2.4) also has the barrier for any $\varepsilon > 0$. This allows one to construct a grid generator as reliable as that constructed by the minimization of $I(G)$. By analogy with the well-known approach to the solution of ill-posed problems in mathematical physics, formula (2.4) is naturally called a regularization of $Q(G)$, and ε, the regularization parameter.

A local minimum of function (2.4) is reached on a solution of the algebraic system

$$\begin{pmatrix} \partial J/\partial x_{ij} \\ \partial J/\partial y_{ij} \end{pmatrix} = \vec{R}_{i,j} = \vec{u}_{i,j} + \varepsilon\sigma \begin{pmatrix} \partial I/\partial x_{ij} \\ \partial I/\partial y_{ij} \end{pmatrix} = 0, \qquad (2.5)$$

$$i = 1, \ldots, N-1, \quad j = 1, \ldots, M-1,$$

where $\vec{u}_{i,j}$ is defined in (2.2).

An iteration procedure for solving (2.5) will be described in the following section. Here we present some examples of grids obtained by solving (2.5).

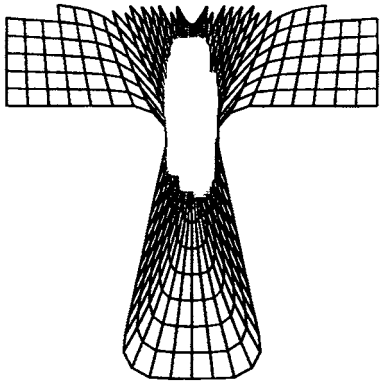


Figure 1.5: A fragment of the grid obtained by the proposed method for the domain from Fig. 1.3.

First we return to the domain with one protrusion presented in Fig. 1.3. A fragment of the grid obtained by solving (2.5) for $\varepsilon = 10^{-2}$ is shown in Fig. 1.5. The grid lines $j = const$ are directed from the lower to the upper boundaries of the domain. Comparing Fig. 1.5 with Fig. 1.3b where the same fragment of the grid obtained by the method [6] (or by minimization of $I(G)$, see Section 1.2), one can see that the grid in Fig. 1.5 becomes involved in the protrusion, and large cells presented in Fig. 1.3b are absent in Fig. 1.5. The reason is that the node distribution along the lines with $j = const$ prescribed in $Q(G)$ (uniform distribution in the present case) is approximately preserved when the grid lines are curved.

Our second example is the domain with many protrusions shown in Fig. 1.6. Here the grid lines $j = const$ are also directed from the lower to the upper boundaries of the domain. For the grid obtained by the method [6] (see Fig. 1.6b), very large grid cells are presented in the all protrusions. The proposed method gives for this domain a much more suitable grid which fragment is shown in Fig. 1.7 for $\varepsilon = 10^{-2}$ and 10^{-3}. A decrease in ε leads, as expected, to a straightening of the lines $j = const$. In the case under consideration, this causes a deterioration in the quality of the grid, since the sizes of the smallest cells become smaller. We do not expect the choice of ε in a particular case to be a hard problem.

1.4 Iteration Procedure

The initial grid G_0 in the iteration procedure for (2.5) must be convex. It is not difficult to construct initial convex grids in solving moving boundary problems, because it can always

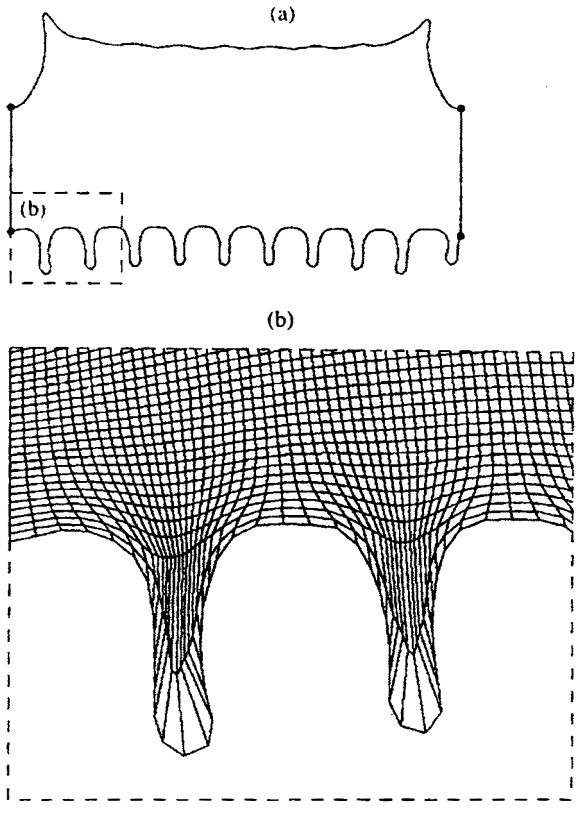


Figure 1.6: A domain with many protrusions (a) and a fragment of the grid obtained by [6] (b).

be taken from the previous time layer. The inner nodes of the initial grid are either taken directly from the grid of the previous time layer or obtained from the displacements of the boundary nodes at the current time step with the use of some interpolation procedure. As a result, the initial grid is, at worst, close to a convex one, which can be easily corrected by minimizing the function

$$I_D = \sum_{i=1}^{n-1} \sum_{j=1}^{m-1} \sum_{k=1}^{4} \left([\varepsilon_1 - J_k]_{i+1/2,j+1/2} \right)_+^2, \quad (f)_+ = \max(0, f),$$

with an appropriate value of $\varepsilon_1 > 0$.

Let us introduce the vector $\Delta\vec{r}_{ij} = (\vec{r}_{ij}^{(s+1)} - \vec{r}_{ij}^{(s)})/\tau_{ij}$, where s is the iteration number, τ_{ij} is the iteration parameter. The vector $\Delta\vec{r}_{ij}$ is determined by solving the following linear system, obtained by simplifying Newton's method:

$$\vec{R}_{ij}^{(s)} + A_{ij}\Delta\vec{r}_{ij} - q_{ij}\Delta\vec{r}_{i+1,j} - q_{i-1j}\Delta\vec{r}_{i-1,j} = 0, \tag{3.1}$$

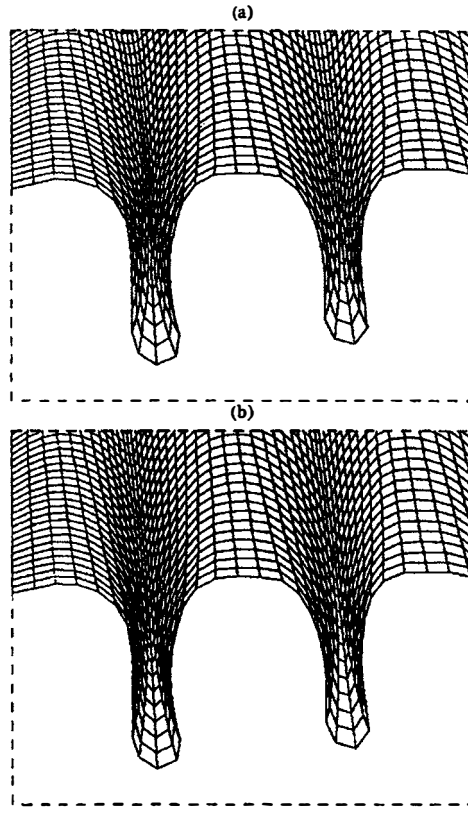


Figure 1.7: A fragment of the grid obtained by the proposed method for $\varepsilon = 10^{-2}$ (a) and 10^{-3} (b).

$$A_{ij} = \begin{pmatrix} q_{ij} + q_{i-1j} + \varepsilon\sigma\partial^2 I/\partial x_{ij}^2 & \varepsilon\sigma\partial^2 I/\partial x_{ij}\partial y_{ij} \\ \varepsilon\sigma\partial^2 I/\partial x_{ij}\partial y_{ij} & q_{ij} + q_{i-1j} + \varepsilon\sigma\partial^2 I/\partial y_{ij}^2 \end{pmatrix}.$$

Here we took into account all the second derivatives of $Q(G)$ and those of $I(G)$ that were taken into account in the quasi-Newtonian iteration procedure for the method [6]. The equations (3.1) are solved by one-dimensional Gaussian elimination for two-dimensional vectors along the lines j =const.

It follows from the derivation of (3.1), if $\varepsilon = 0$ and $\tau_{ij} = 1$, the iteration process stops after the first iteration and yields the quasi-one-dimensional grid defined by (2.1). Clearly, if $0 < \varepsilon \ll 1$, even the first iteration yields a grid that is close to a quasi-one-dimensional one in the subdomains where small distortions of lines of the selected family are enough to preserve the convexity of cells.

To reduce the computational cost we use a special choice of the iteration parameter τ_{ij} and iterations on subsets of grid nodes.

The iteration parameter is defined as $\tau_{ij} = \tau_{ij}^0 \tau$. The parameter τ_{ij}^0 is chosen so as

to prevent going beyond the set D. Clearly, such an event is fatal to the iterative process because of the above infinite barrier of the minimized function on the boundary of D. First, $\tau_{ij}^0 = 1$ is set at all grid nodes. Then, the cells are checked for convexity. If the area of any of the cell's triangles is negative, then τ_{ij}^0 is reduced by half at the three corresponding grid nodes. The reduction process terminates if the next check shows that all cells are convex. In practice, τ_{ij}^0 is decreased only at the first iteration steps if the initial grid G_0 is close to the boundary of the set D.

The scalar parameter τ is set to 1 at the first several iteration steps in order to ensure fast convergence in the subdomains where $Q(G) \gg \varepsilon\sigma I(G)$. As a result, the described lower method of iterations on subsets of grid nodes, in general, substantially reduces the number of grid nodes of the subset involved in the iteration. Then, τ is determined from a version of the method of parabolas with the squared residual of the equations (2.5)

$$W = \sum_{i,j} \mid \vec{R}_{i,j} \mid^2 \tag{3.2}$$

as the controlling quantity. The parabola $W(\tau)$ is constructed from the grids obtained for $\tau = 0$, 1/2 and 1. The parameter τ is then chosen so that $W(\tau) = \min$ in the interval $\theta \leq \tau \leq 1.0$ where the parameter $\theta \sim 0.1$ is given *a priori* and bounds the value of τ away from zero.

Now we consider a method of iterations on subsets of nodes. The selection of a subset is based on the analysis of the local relative residual of the equations (2.5)

$$\delta_{ij} = \mid \vec{R}_{ij} \mid / (q_{ij} \mid \vec{r}_{i+1,j} - \vec{r}_{ij} \mid + q_{i-1,j} \mid \vec{r}_{ij} - \vec{r}_{i-1,j} \mid + \varepsilon\sigma g_{ij}),$$

where g_{ij} is defined in Section 1.2 and is of the same order of magnitude as the terms involved in $\partial I/\partial x_{ij}$ and $\partial I/\partial y_{ij}$. Let us denote the set of all inner grid nodes by Ω_0

The computational expenditures are reduced as follows. At each iteration step s, we select a subset Ω^s and perform calculations on nodes from this subset, while all nodes $(i,j) \in \Omega_0 \setminus \Omega^s$ are assumed fixed. A maximum admissible value of the relative residual δ and an integer N are assigned. After each sth iteration, starting with the Nth one, the subset Ω^s is selected so that, at the previous iteration step, $\delta_{ij} \leq \delta$ for $(i,j) \in \Omega_0 \setminus \Omega^s$. To do this, we must take into account that the sth iteration changes the value of the δ_{ij} parameter not only at nodes from Ω^s. Let γ_{ij} denote the set of nine nodes in the neighborhood of (i,j): $(i',j') \in \gamma_{ij}$, if $|i - i'| \leq 1$ and $|j - j'| \leq 1$. Let Ω^s be a set such that $\delta_{ij} \leq \delta$ for $(i,j) \in \Omega_0 \setminus \Omega^s$. Let us find a set Ω^{s+1} that has the same property after the sth iteration step whenever all nodes except for those from Ω^s are fixed at this step. For every node (i,j), equations (2.5) include only the coordinates of nodes $(i',j') \in \gamma_{ij}$; therefore, δ_{ij} can change only at nodes from the set

$$\bar{\Omega}^s = \bigcup_{ij \in \Omega^s} \left(\gamma_{ij} \bigcap \Omega_0\right).$$

during the sth iteration. To obtain Ω^{s+1} , we must evaluate δ_{ij} for $(i,j) \in \bar{\Omega}^s$ using the coordinates computed at the sth iteration step. Although $\bar{\Omega}^s$ can be much larger than Ω^s, $\bar{\Omega}^s$ can remain small in comparison with Ω_0 if the grid has a large number of nodes.

The set Ω^{s+1} can be obtained from $\bar{\Omega}^s$ by removing all nodes for which $\delta_{ij} \leq \delta$. For the example on which the algorithm was tested, it was more expedient to leave some nodes with $\delta_{ij} \leq \delta$ in Ω^{s+1}, namely, $(i, j) \in \Omega^s$ such that $\delta_{i'j'} > \delta$ for at least one node $(i', j') \in \gamma_{ij}$.

Here we present a test of the algorithm described above in reference to the grid generator [6], which corresponds to $\varepsilon = \infty$ and changing I for $I_{(6)}$. Return to the problem of a high-speed collision of a thin foil with a cone target shown in Fig. 1.1. The shape of the test domain is shown in Fig. 1.1b. The grid in the undistorted left-hand part of the domain can easily be constructed from straight lines. For this reason, the method was only applied for constructing the inner grid points in the right-hand part, which included about 4000 nodes. The inner nodes of the initial grid were computed with the use of an interpolation procedure according to the displacements of the boundary nodes at the current time step.

As a characteristic of the computational cost, the value

$$p = \sum_s N(\bar{\Omega}^s)/N(\Omega_0),$$

is selected, where $N(\Omega)$ is the number of nodes in Ω . If iterations are performed on the set Ω_0, p coincides with the number of iterations. Fig. 1.8 shows the dependencies of the quadratic residual W (3.2) on the computational cost p for $\delta = 0.01$, 0.02, and 0.03 and for iterations on the set Ω_0, which corresponds to the case $\delta = 0$. The dots mark the data for values close to integer p. We can see that, for all three values of δ, the algorithm yields much more "completely iterated" grids than the original version of the method on the set Ω_0, while the computational cost is the same. For $\delta = 0.03$, the iterative process converges very rapidly, and the expenditures are moderate. Computation with $\delta = 0.02$ takes more time but makes the grid less "underiterated". For $\delta = 0.01$, the performance is worse, but still better than that of the original version of the method. The computational cost p of the original method in comparison with the computational cost p of the modified method for the same values of W are tabulated below:

δ	0.01	0.02	0.03
W	4.5	3.2	9.2
p	20	16	3
p_0	50	71	18

The ratio between the computational costs of the original and modified methods varies from 2.5 for $\delta = 0.01$ to 6 for $\delta = 0.03$. As for the computational expenditures required for the original method to give the same values of the maximum local residual

$$D = \max_{ij \in \Omega_0} \delta_{ij}$$

as the modified version, we were unable to determine them at all, because, even for $p_0 = 100$, the D values obtained by the original method did not go below 0.04. We made an attempt to further reduce the computational cost by using purely Newton iterations that take into account all the second derivatives of $I_{(6)}$. As expected, this considerably decreased the

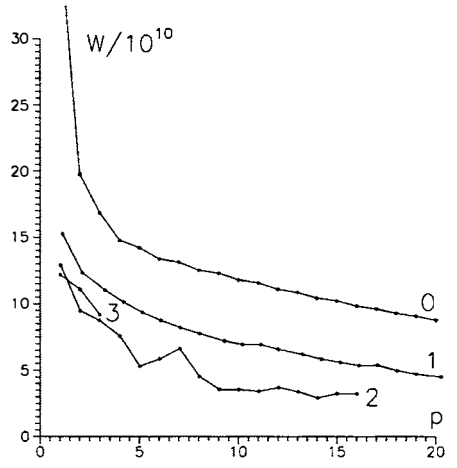


Figure 1.8: The quadratic residual W against the computational cost p for $\delta = 0.01$ (the curve 1), 0.02(2), 0.03(3) and 0(0).

residuals in the domains Ω^s at all iteration steps. However, the rate of decrease of the residual in $\bar{\Omega}^s$ changed insignificantly. As a result, the computational cost even increased to some extent because of the increased expenditures at the stage of solving the system of linear equations.

1.5 Applications

The grid generator presented above was integrated into an application package designed to compute flows with moving interior and exterior boundaries. We split the algorithm into a Lagrangian stage and a stage of remeshing from the Lagrangian grid to the grid used at the current time step [10]. Second-order accurate quasi-monotone schemes were used at each stage. The scheme employed at the Lagrangian stage was the variant of the scheme proposed in [11] that was formulated in [12] and modified in accordance with [13]. The algorithms used to compute the Riemann problem were described in [14]. At the remeshing stage, we used a variant of the method developed in [15]. When the necessary condition of stability for the method was violated, we applied the multistep algorithms discussed in [16]. In addition to the interior grid generator, we used a package of procedures for computing the motion of specific grid lines (typically, interfaces between media). The package deals with the motion of a line, the distribution of nodes along it, and the calculation of the points where the specific lines intersect. The corresponding procedures were borrowed mainly from [3] and extended in the course of the practical application of the package.

The first example was taken from computations of the instability of an interface interacting with a shock wave (see [17]). Fig. 1.9 shows an overall view of the rectangular computational domain and a fragment of the grid near the interface. The problem was posed

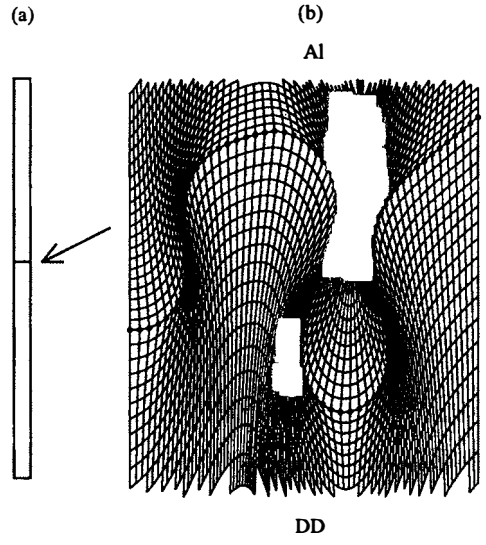


Figure 1.9: The instability of the interface between aluminum (Al) and deuterium ice (DD): the computational domain (a) and a fragment of the grid (b) near the interface (dots).

for a weakly perturbed initial interface in the middle of the rectangle shown by an arrow in Fig. 1.9. Since short-wave disturbances had to be computed, the rectangles aspect ratio was about 100, whereas the aspect ratios of grid cells lying near the interface should be closed to unity. Along the interface the grid nodes are placed uniformly. Along the lateral boundaries the distribution of the grid nodes is nonuniform. Near the interface, the grid points are located almost uniformly with a step of the same order as that along the interface. Away from the interface, the grid step along the side boundaries is gradually increased, so that the maximal ratio of the neighboring grid intervals is about 1.1, while the largest-to-smallest grid interval was about 50.

Far from the interface, the flow was almost one-dimensional, and strong deviation of the grid from a one-dimensional one would lead to a significant computational error. It was found that the function $I_{(6)}(G)$ significantly contributes to (2.4) far from the interface, even if the function is multiplied by a small parameter, which makes the grid essentially nonuniform. This disadvantage was eliminated by using the function $I(G)$, since the regularizing function would not affect the grid generation far from the interface.

The grid depicted in Fig. 1.9 consists of convex quadrilaterals only. It is clear that the grid is very fine inside the protrusion (which represents an aluminum droplet in this example). In time, the droplet may detach from the bulk of the aluminum. It is obviously impossible to compute the detachment of a droplet by tracking its boundary on a grid having an invariant structure. The areas of cells inside the bridge between the droplet and the bulk tend to zero. Since the time step determined by the stability criterion for the scheme in Lagrangian stage (see [3]) should also tend to zero, the computation would terminate.

Computation on a grid of varying structure would be a cumbersome but feasible task (cf. computations of a collapsing annular cumulative jet described in [18,19]). In the present study, it was not necessary to compute the detachment of a droplet.

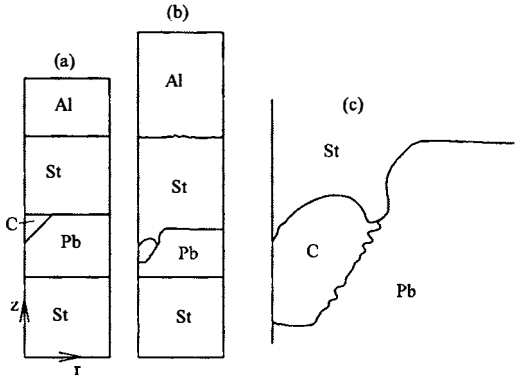


Figure 1.10: Compression of graphite in conical experimental assemblies: the aluminum striker (Al), the steel shell (St), the lead target (Pb) and the carbon volume (C); a: $t = 0$; b and c (a vicinity of the carbon volume): $t = 5\mu s$.

Some difficulties of a different nature have encountered in computations of experimental assemblies for shock compression of graphite in conical targets with strikers. A mathematical model taking into account the kinetics of graphite-to-diamond transition was presented in [20]. Fig. 1.10 shows the interface geometry in cylindrical coordinates at $t = 0$ and $t = 5\mu s$ after the impact between the striker and the assembly. Each grid block represents a certain material. The main difficulty lies in the computation of the point where the interfaces separating carbon, steel, and lead intersect. The grid generator worked well at the nonlinear stage of the evolution of an unstable carbon-lead interface, as illustrated in Fig. 1.10c. Computations of a more fully developed instability can be found in [20].

The last example discussed here simulates an experiment on a shock compression of a plate resting on a wedge. A problem of this kind was solved in [21]. The results of particular computations are illustrated by Fig. 1.11, where the plate and wedge boundaries corresponding to various instants are shown. A shock wave was initiated in a plate initially resting on a wedge (Fig. 1.11a). The lower free boundary of the plate began to move into the wedge. At the point of contact A moving with the plate boundary (Fig. 1.11b), an attached shock wave developed, preventing the formation of a cumulative jet. The compression continued without any jet formation until the plate's free boundary reached the wedge base. As a result, a narrow jet developed (Figs. 1.11c and 1.11d).

The computations were organized as follows. Both plate and wedge were discretized as independent grid blocks. The grids could slip along the boundary between the plate and the wedge. The point of contact A was represented by the same node in the plate grid block. After node A reached node B representing the base corner of the wedge, the computation

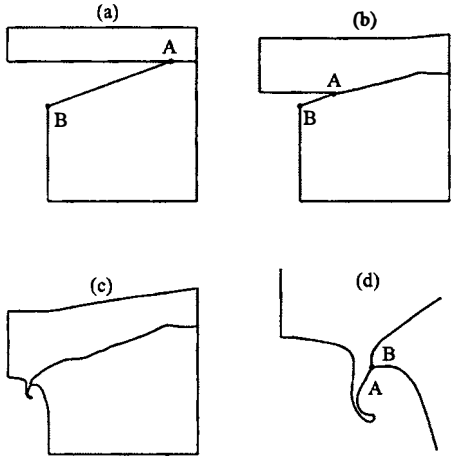


Figure 1.11: Shock compression of a plate on a wedge; *a*: $t = 0$; *b*: $t = 2\mu s$; *c,d*: $t = 4\mu s$; A: a grid node in the plate; B: in the wedge.

of the motion of A was stopped, and both nodes were treated as a single one. Note that the grid generator worked well in the computation of the narrow jet shown in Fig. 1.11c without any additional adjustment of parameters.

1.6 Concluding Remarks

We note the obvious circumstance that the suggested method is not universal. Indeed, the quasi-one- dimensional grid (2.1) is virtually insensitive to deformations of the boundary lines $j = 0$ and $j = M$, but sensitive to deformations of the other two boundary lines, $i = 0$ and $i = N$. Therefore, it is reasonable to apply this method to problems where considerable deformations take place on either one boundary or two opposite boundaries. Moreover, the lines $j = const$ must start from the deformed boundaries.

As a prospect for the construction of an universal grid generator, we point out the following approach. It is possible to supplement the function $J(G)$ (2.4) with a function of quasi-one-dimensional grids along lines of the family $i = const$. However, it should be noted that, for many problems such an universal generator will be less efficient than that suggested in this paper, because it will require a larger amount of computations, while the grid quality will not be noticeably improved. The increase in computational costs is explained by an increase in the number of iteration steps when $\varepsilon \ll 1$, rather than the computational complexity of $J(G)$, because the resulting grid generator requires an iteration procedure even if $\varepsilon = 0$.

Acknowledgments

This work was supported by the Department of Mathematical Sciences RAS, Programme No. 3.

References

[1] Prokopov G.P., Some general problems in constructing grid generation algorithms (Keldysh Institute of Applied Mathematics RAS, Preprint No. 98, Moscow, 1987) [in Russian].

[2] Winslow A.M., Numerical solution of quasilinear Poisson equation in nonuniform triangle mesh, *J. Comput. Phys.* **1**, 149 (1967).

[3] Godunov S.K., Zabrodin A.V., Ivanov M.Ya., Prokopov G.P., and Kraiko A.N., *Numerical Solution of Multidimensional Problems of Gas Dynamics* (Nauka, Moscow, 1976) [in Russian].

[4] Thompson J.F., Warsi Z.U.A., and Mastin C.W., *Numerical Grid Generation* (North-Holland, New York, 1985).

[5] Ivanenko S.A., Construction of curvilinear grids and their use in finite element method for solving shallow water equations, (Computing center of the USSR Academy of sciences, Moscow, 1985) [in Russian].

[6] Ivanenko S.A. and Charakhch'yan A.A., An algorithm for constructing curvilinear grids consisting of convex quadrangles, *Soviet. Math. Dokl.* **36**, No. 1, 51, (1988); Curvilinear grids of convex quadrilaterals, *USSR Comput. Maths Math. Phys.* **28**, No. 2, 126 (1988); A variational form of the Winslow grid generator, *J. Comput. Phys.* **136**, No. 2, 385 (1997).

[7] Numerical Grid Generation in Computational Field Simulations / Edited by B.K.Sony, J.F.Thompson, J. Hauser, and P.R.Eiseman. Proceedings of the 5th International Conference. Mississippi State University, 1996.

[8] Us'kov V.M. Curvilinear grids formed by elongated convex quadrangles, (unpublished).

[9] Ivanenko S. A., Control of cell shape in the construction of a grid, *Comput. Math. Math. Phys.* **40**, No. 11, 1596 (2000).

[10] Van Leer B., A second-order sequel to Godunov's method, *J. Comput. Phys.* **32**, No. 1, 101 (1979).

[11] Rodionov A.V., Methods of increasing the accuracy in Godunov's scheme, *U.S.S.R Comput. Math. Math. Phys.* **27**, No. 6, 164 (1987).

[12] Charakhch'yan A.A., Compound difference schemes for time–dependent equations on non–uniform nets, *Communs. Numer. Methods in Engineering* **10**, No 2, 93 (1994).

[13] Charakhch'yan A.A., Almost conservative difference schemes for the equations of gas dynamics, *Comput. Math. Math. Phys.* **33**, No. 11, 1681 (1993).

[14] Charakhch'yan A.A., On algorithms for computing the Riemann problem by means of Godunov's scheme, *Comput. Math. Math. Phys.* **40**, No. 5, 746 (2000).

[15] Van Leer B., Towards the altimate conservative difference scheme. IV. A new approach to numerical convection, *J. Comput. Phys.* **23**, No. 3, 276 (1977).

[16] Charakhch'yan A.A., Application of moving regular grids to computation of gasdynamic flows with interfaces, in *Modern Problems in Comput. Aerogasdynamics* (Mir Publishers, Moscow and CRC Press, Boca Raton, Fl. 1992), p. 189.

[17] Charakhch'yan A.A., Reshocking at the non-linear stage of Richtmyer-Meshkov instability, *Plasma Phys. Control. Fusion.* **43**, No. 9, 1169 (2001).

[18] Charakhch'yan A.A., Numerical investigation of compression of deuterium in a conical target under a strong cumulation effect, *Pril. Mekhan. i Tekhnich. Fiz. (J. Appl. Mechan. Technical Phys.)* **35**, No.4, 22 (1994).

[19] Charakhch'yan A.A., Numerical study of a spontaneous electromagnetic field induced by annular cumulative jets in conical targets, *Plasma Physics Reports* **24**, No. 4, 349 (1998).

[20] Lomonosov I.V., Fortov V.E., Frolova A.A., Khishchenko K.V., Charakhchyan A.A., and Shurshalov L.V., Numarical study o shock compression of graphite and its conversion to diamond in conical targets, *Technical Physics* **48**, No. 6, 727 (2003).

[21] Charakhch'yan A.A., Shock compression of a plate on a wedged-shaped target, *J. Appl. Mechan. Technical Phys.* **42**, No. 1, 14 (2001).

Chapter 2

ON ONE CLASS OF QUASI-ISOMETRIC GRIDS

Sergei K. Godunov[1], *Olga B. Feodoritova*[2] *and Victor T. Zhukov*[2]
[1] Sobolev Institute of Mathematics
[2] Keldysh Institute of Applied Mathematics

There is a number of publications devoted to grid construction in different engineering applications. A presented paper describes one approach for constructing quasi-isometric grids. The main attention directs towards numerical features of the suggested approach. Our basic goal is to awake theoretical and computational interest to the problem of quasi-isometric grid generation.

2.1 Introduction. On One New Extreme Problem

At that moment many researches are interested in the problem of parametrizations of various multidimensional domains. Such parametrizations are naturally used to solve partial differential equations boundary problems. Numerous, often very witty, procedures are suggested to implement such parametrizations in regions of different dimensions.

Numerical grids constructed on the basis of selected parametrizations are widely utilized in the modern computational practice. At the same time the discussions about advantages (disadvantages) of the different variants take place. This is natural for the numerous practical situations. But, it makes sense to discover the rigorous mathematical formulations of the extreme problems, which can be solved by using some modifications of the suggested parametrizations.

During many years, the authors being in various groups worked on different specific problems and solved the problem of the parametrization of the two-dimension domains, namely, the curvilinear quadrangles with the smooth arcs. These quadrangles should be subordinated to one additional, but not very strict, constrain. The minimal angle $\varphi_{\min} = \min\{\varphi_1, \varphi_2, \varphi_3, \varphi_4\}$ among apical angles needs to satisfy the inequality $\varphi_{\min} \geq \frac{1}{2}(\varphi_1 + \varphi_2 + \varphi_3 + \varphi_4 - 2\pi)$. For this condition, a quadrangle defined on some smooth surface can be mapped conformally into the interior of a quadrangle restricted by geodesics on

some constant curvature surface (the Lobachevskian or Euclidean sphere). As the parameters, the quantities that are constant along some geodesics on a surface should be selected [4, 5]. It is shown that such parametrization provides the quasi-isometry of parametric and parametrized quadrangles. But, there is no procedure to find the geodesics and to link the specific parameters for them to ensure the parametrization extremeness. Does the best and unique parametrization exist? Even, if the positive answer exists, there is possibility to select other constructions with less metric distortion. The solution of the pointed extreme problem gives the best quasi-isometric parametrization and is the classical P.L.Chebychev problem formulated in 1856 [1]. This problem is devoted to selection of a conformal parametrization on the sphere (of the globe) to draw the geographical maps. The Chebychev problem was completely solved by D.A.Grave in his papers [2, 3]. To provoke the interest to the problem described, we give below the review of statements and algorithms elaborated for quasi-isometric parametrizations and illustrate them by a few examples.

One emphasizes that the solution of the problem about quasi-isometric mappings should be obtained by specialists in geometry and mathematical analysis but not in numerical mathematics. On the other hand, this solution gives the criterion to evaluate the quality of the specific algorithm and is useful for engineers.

2.2 Problem Setting. Basic Principles

We describe the quasi-isometric mapping technique for a curvilinear quadrangle $\Omega \subset \mathbb{R}^2$ with four sides and apical angles $\varphi_1, \varphi_2, \varphi_3, \varphi_4$ (Fig.2.1). Let us mention some definitions and results from [4, 5].

A curvilinear quadrangle Ω is called smooth if each of its sides is sufficiently smooth (Liapunov arc with Hölder exponent μ). A mapping $u = u(s,t), v = v(s,t)$ of the unit square $K_0 = \{(s,t) : 0 \le s,t \le 1\}$ onto Ω is called quasi-isometric if the ration between two (sufficiently close) points (s_1,t_1) and (s_2,t_2) to the distance between their images (u_1,v_1) and (u_2,v_2) is bounded:

$$0 < \sigma_1 \le \frac{\sqrt{(u_2-u_1)^2+(v_2-v_1)^2}}{\sqrt{(s_2-s_1)^2+(t_2-t_1)^2}} \le \sigma_2.$$

A mapping $u(s,t), v(s,t)$ of the domain Ω is called the $C^1_\mu(\Omega)$-mapping if the partial derivatives u_s, u_t, v_s, v_t are continuous and satisfy the Hölder condition with exponent μ. We can also say that a mapping is called a quasi-isometric one if $|u_s|, |u_t|, |v_s|, |v_t| \le H$, $u_s v_t - u_t v_s \ge h > 0$.

Note that a quasi-isometric mapping is a quasi-conformal one, i.e. being conformal with respect to some metric. A conformal mapping may not be quasi-isometric. A mapping of a smooth quadrangle onto another one is conformal and quasi-isometric if and only if the corresponding angles and conformal modules of these quadrangles are equal. The main result of [4] is the following:

Theorem 1. Let $\Omega \subset \mathbb{R}^2$ be a smooth quadrangle with boundary $\partial\Omega$ and angles $\varphi_1, \varphi_2, \varphi_3, \varphi_4$ such that $0 < \varphi_j < \pi$, $\delta \equiv \varphi_1+\varphi_2+\varphi_3+\varphi_4-2\pi < 2\varphi_j$, $(j = 1,2,3,4)$.

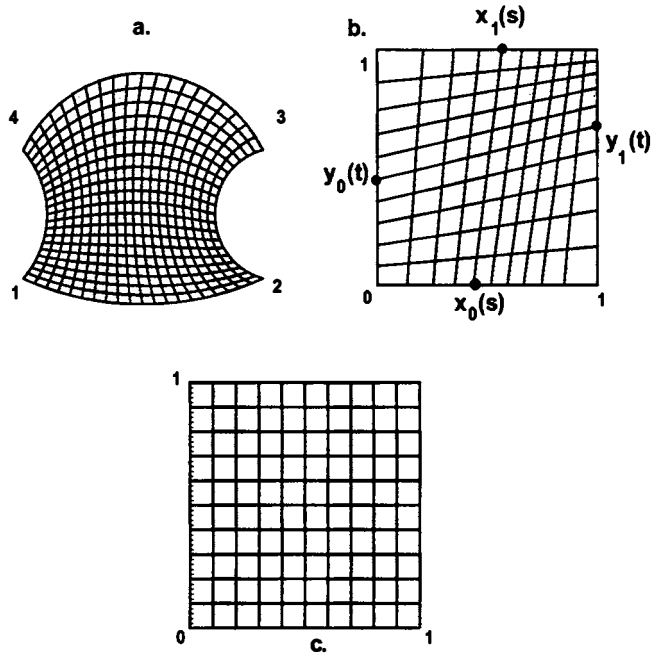


Figure 2.1: a. curvilinear quadrangle Ω; indices 1,2,3,4 denote apical angles φ_1, φ_2, φ_3, φ_4; b. unit square $K = \{(x,y) : 0 \le x, y \le 1\}, x = x(s,t), y = y(s,t)$; c. unit square $K_0 = \{(s,t) : 0 \le s, t \le 1\}$

Then any quasi-isometric C^1_μ-mapping of the boundary ∂K_0 of the unit square K_0 onto $\partial\Omega$ can be extended to quasi-isometric $C^1_\mu(K_0)$-mapping of K_0 onto Ω.

The quality of a quasi-isometric mapping is defined by the ratio σ_1/σ_2 and a mapping is optimal if this ratio is maximal among all quasi-isometric mappings. There is a hypothesis that the mapping is close to optimal one under the assumption that a curvature of any image of a straight line segment is bounded [4, 5].

The considered quasi-isometric mapping $u = u(s,t), v = v(s,t)$ of K_0 onto Ω is constructed as the superposition of two mappings (see Fig.2.1): $u = u(x,y), v = v(x,y)$ maps the rectangle $K = \{(x,y) : 0 \le x \le \sqrt{\nu}, 0 \le x \le \sqrt{1/\nu}\}$ onto Ω; $x = x(s,t), y = y(s,t)$ maps K_0 onto K [1]. The first mapping u, v is quasi-isometric and conformal with respect to the metric g_{ij} chosen from a five-parameter family of metrics defined in the unit

[1]We follow papers [4, 5] in our description and suppose $\nu = 1$, i.e. $u = u(x,y), v = v(x,y)$ maps the unit square K onto Ω. The transition to rectangle bases on the identity

$$\int_0^1 \int_0^1 [A(u_x^2 + v_x^2) - 2B(u_x u_y + v_x v_y) + C(u_y^2 + v_y^2)]dxdy =$$

square K by:

$$\begin{aligned} g_{11}(y) &= 1 + k + (p_1 + p_2 - p_3 - p_4)\bar{y} - (p_1 + p_2 + p_3 + p_4)\bar{y}^2, \\ g_{22}(x) &= 1 - k + (p_1 - p_2 - p_3 + p_4)\bar{x} - (p_1 + p_2 + p_3 + p_4)\bar{x}^2, \\ g_{12}(x,y) &= \tfrac{1}{4}(p_1 - p_2 + p_3 - p_4) - \tfrac{1}{2}(p_1 + p_2 - p_3 - p_4)\bar{x} - \\ &-\tfrac{1}{2}(p_1 - p_2 - p_3 + p_4)\bar{y} + (p_1 + p_2 + p_3 + p_4)\bar{x}\bar{y}, \end{aligned} \tag{2.1}$$

(here $\bar{x} = x - \frac{1}{2}$ and $\bar{y} = y - \frac{1}{2}$), and g_{ij} satisfy the following relations at the vertices of K:

$$\begin{gathered} p_i = \cos\varphi_i \sqrt{g_{11}(x_i, y_i) g_{22}(x_i, y_i)}, \quad i = \overline{1,4} \\ (x_1, y_1) = (0,0), \quad (x_2, y_2) = (1,0), \\ (x_3, y_3) = (1,1), \quad (x_4, y_4) = (0,1). \end{gathered} \tag{2.2}$$

The metrics (2.1) depends on five parameters $\mathbf{p} = (p_1, p_2, p_3, p_4, k)$. The desired metric g_{ij} represents the natural metric on the constant curvature surface (the Lobachevskian or Euclidean sphere or plane) such that geodesics in K are straight line segments; the general form of such metrics is described in [6].

The mapping $u(x,y), v(x,y)$ can be constructed by the minimization of the functional

$$F = \frac{1}{2} \iint\limits_K \frac{\{g_{22}(u_x^2 + v_x^2) - 2g_{12}(u_x u_y + v_x v_y) + g_{11}(u_y^2 + v_y^2)\}}{\sqrt{g_{11}g_{22} - g_{12}^2}} dxdy, \tag{2.3}$$

with respect to u, v and the metric parameters $\mathbf{p}$ under the conditions (2.2) and the free boundary condition.

The second mapping $x = x(s,t), y = y(s,t)$ of K_0 onto K is defined by four functions $x_0(s), x_1(s), y_0(t), y_1(t)$, $0 \le s \le 1$, $0 \le t \le 1$, $x_i(0) = y_i(0) = 0$, $x_i(1) = y_i(1) = 1$, $i = 1, 2$. The mapping of the lower and upper sides of the square K_0 are defined by the functions $x_0(s)$ and $x_1(s)$, and the mapping of the left and right sides are defined by the functions $y_0(t)$ and $y_1(t)$. The functions x_0, x_1, y_0, y_1 are called the control functions, they are assumed to be smooth (C_μ^1) and strictly monotonically increasing, and their derivatives are bounded:

$$0 < \delta \le x_0'(s), x_1'(s), y_0'(t), y_1'(t) \le \Delta, \ (\delta \le 1).$$

The mapping of K_0 onto K is the algebraic mapping given by the formulas:

$$x = \omega^{-1}\{[1 - y_0(t)]x_0(s) + y_0(t)x_1(s)\},$$

$$y = \omega^{-1}\{[1 - x_0(s)]y_0(t) + x_0(s)y_1(t)\},$$

$$= \int\limits_{\eta=0}^{\eta=\sqrt{\frac{1}{\nu}}} \int\limits_{\xi=0}^{\xi=\sqrt{\nu}} \left[(\nu A)\,(u_\eta^2 + v_\xi^2) - 2B(u_\xi u_\eta + v_\xi v_\eta) + \left(\frac{1}{\nu} C\right)(u_\eta^2 + v_\eta^2) \right] d\xi d\eta.$$

where

$$\omega = 1 - [x_1(s) - x_0(s)][y_1(t) - y_0(t)],$$

$$x_i(0) = y_i(0) = 0, x_i(1) = y_i(1) = 1,\ i = 0, 1.$$

This mapping transforms the rectangular grid on K_0 into the grid on K consisting of the straight line segments connecting the images of the corresponding boundary points of K_0, which lie on the opposite sides of K.

The superposition of the mappings $u(x, y), v(x, y)$ and $x(s, t), y(s, t)$ is the resulting mapping $u(s, t), v(s, t)$ of K_0 onto Ω, which is defined in [4, 5] as a unique solution of the variational problem given by the functional:

$$F = \frac{1}{2}\int_0^1\int_0^1 \frac{h_{22}(u_s^2 + v_s^2) - 2h_{12}(u_s u_t + v_s v_t) + h_{11}(u_t^2 + v_t^2)}{\sqrt{h_{11}h_{22} - h_{12}^2}} dsdt, \qquad (2.4)$$

where $h_{ij} = h_{ij}(s, t, \mathbf{p}, x_0, x_1, y_0, y_1)$ satisfy the condition $H = Q^*GQ$,

$$H = \begin{bmatrix} h_{11} & h_{12} \\ h_{12} & h_{22} \end{bmatrix}, \quad Q = \begin{bmatrix} x_s & x_t \\ y_s & y_t \end{bmatrix}, \quad G = \begin{bmatrix} g_{11} & g_{12} \\ g_{12} & g_{22} \end{bmatrix}.$$

The minimal value $F = S$, where S is the area of Ω, is achieved at the quasi-isometric mapping u, v.

A presented metric is obtained from the Beltrami metric by means of a projective transformation (see [4] for details)

$$ds^2 = \frac{(1 + k_0\eta^2)d\xi^2 - 2k_0\xi\eta d\xi d\eta + (1 + k_0\xi^2)d\eta^2}{[1 + k_0(\xi^2 + \eta^2)^2]}.$$

Let us consider one more approach to find the numerical characteristics of mapping regions that should evidence of "unremovable defects" of these regions. If there are more such defects, it is more difficult to opt for a region parametrization. Again we restrict ourself by the problem of a parametrization of "curvilinear quadrangles". As was already marked, each such quadrangle along with some restrictions on its boundary smoothness and on apical angles can be mapped conformally and quasi-isometrically on constant curvature surface with geodesic boundaries. Such a mapping can be normalized and provides an invariability of the whole quadrangle area. If a sign of a curvature surface is defined only by angle values, then a numerical value of a curvature is defined uniquely with the pointed normalization. The geodesic quadrangle on a constant curvature surface is also uniquely defined accurate to movement in this surface. Obviously, the one characteristic of an origin curvilinear quadrangle is the isometry coefficient with its conformal mapping on a constant curvature surface, i.e. a mapping is not changed its area.

If we want to map a quadrangle onto a plane quadrangle with straightforward boundaries, we need to map geodesic quadrangles from a constant curvature surface onto a plane. Applying the Kely–Klein model for constant curvature surfaces with plane geodesics, we implement a projective mapping which gives us some metric distortion. As a result, the

Beltrami metric will be obtained on a parametrical plane. This fact was mentioned above. For a mapping of a constant curvature surface onto a final parametrical plane, a distortion (isometry coefficient) is closer to unit if this point is closer to a coordinate origin. The most distortion is attained in that quadrangle vertex which is the most removed from the origin of a coordinate system. The extreme mapping with the least quadrangle distortion from a constant curvature surface onto the parametrical Beltrami plane is constructed by the following way. At first, a circle with the minimal area is constructed on a constant curvature surface, which contains all points of a geodesic quadrangle. At least two quadrangle vertices lie on the boundary circle but others are removed from the origin on the same or least distance.

The projection mapping on the Beltrami plane should be selected in such a way that a boundary circle is a circle again but its origin maps into the origin of a coordinate system. Having such mapping, the largest and smallest distance will be obtained at image points of boundary circle, i.e. at some quadrangle vertices and at quadrangle points nearest to the origin of a coordinate system (or coincides with it). A relation of these extreme stretchings is an isometry coefficient of a constant projective transformation (this coefficient is the least from possible one).

So, our parametrization type of curvilinear quadrangles is characterized by two invariants: an isometry coefficient for a parametrization on a constant curvature surface and an isometry coefficient of the projective mapping of this surface onto the Beltrami plane. Obviously, that the resultant transformation is evaluated from above as multiplication of these coefficients.

One more important fact should be remarked. The quadrangle obtained after all described constructions can be very unlike a parallelogram. In that case, a coordinate system with noncrossing straight lines can be inconvenient and leads to big distortion isometry. Therefore, it is natural to point out one more invariant — a shape of this of rectilinear quadrangle on the Beltrami plane.

2.3 General Computational Scheme

The implementation of the quasi-isometric mapping technique requires some special efforts described in this section.

For minimization (2.3), it is used an iterative process consisting of three stages according to three groups of unknowns. Each stage is a minimization procedure with respect to one of these groups provided that the others are fixed at this stage; in general, a described scheme presents the Seidel iterative process (coordinate descent).

- Calculation of $\mathbf{p}$; a grid (u, v) and control functions are known;
- Calculation of control functions and (or) movement of points along boundaries; a grid (u, v) and parameters $\mathbf{p}$ are known; [2]

[2] At each side of K_0 the Dirichlet or free (natural) boundary condition for u, v must be specified; in the case of a free boundary condition the corresponding control function must be given and the point distribution is to be found in the minimization process.

- Calculation of a grid (u, v); control functions and parameters $\mathbf{p}$ are known.

Computation of the metric parameters $\mathbf{p}$. To provide a robustness of the computational procedure one needs to modify the original functional (2.3), since there is no guarantee that the functional is convex far from an unknown solution. The modification and the local regularization gives a researcher a possibility not to care about an initial guess of a solution and start, for instance, from zero values. Let us denote $G_{11} = u_x^2 + v_x^2$, $G_{22} = u_y^2 + v_y^2$, $G_{12} = u_x u_y + v_x v_y$. One writes out the functional for calculating of the metric parameters in the modified form:

$$F = \int\int \frac{g_{11}G_{22} + g_{22}G_{11} - 2g_{12}G_{12} - 2\sqrt{g_{11}g_{22} - g_{12}^2}\sqrt{G_{11}G_{22} - G_{12}^2}}{2\sqrt{g_{11}g_{22}}\sqrt{G_{11}G_{22}}} dxdy. \tag{2.5}$$

At each point, g_{ij} and G_{ij} can be rewritten in the form

$$g_{11} = \rho e^p, \quad g_{22} = \rho e^{-p}, \quad g_{12} = \rho \sin\varphi,$$

$$G_{11} = Re^P, \quad G_{22} = Re^{-P}, \quad G_{12} = R\sin\Phi.$$

So, the integrand in (2.5) may be written as

$$f = ch(p - P) - \cos(\varphi - \Phi)$$

or

$$f = 2sh^2\frac{p - P}{2} + 2\sin^2\frac{\varphi - \Phi}{2}$$

and it is nonnegative and convex under the additional condition $|\varphi - \Phi| \le \frac{\pi}{2}$.

This condition can be destroyed through the initial stage of a computational procedure, therefore the following regularization is implemented at each grid point (u, v)

$$\sin\hat{\Phi} = \begin{cases} max\{\sin\Phi, -|\cos\varphi|\} & , \text{ if } \sin\varphi \ge 0, \\ max\{\sin\Phi, |\cos\varphi|\} & , \text{ if } \sin\varphi < 0. \end{cases}$$

The minimization of this modified nonlinear functional is based on the quadratic approximation:

$$f = f_0 + (\mathbf{T}, \delta\tilde{\mathbf{p}}) + \frac{1}{2}(\mathbf{H}\delta\tilde{\mathbf{p}}, \delta\tilde{\mathbf{p}}). \tag{2.6}$$

The local Jacobian $\mathbf{T}$ and Hessian $\mathbf{H}$ contain the complete information about all utilized transformations:

$$(\delta p, \delta\varphi) \xrightarrow{Q_1} (\delta g_{11}, \delta g_{12}, \delta g_{22}) \xrightarrow{Q_2} (\delta\mathbf{p}).$$

Denoting the sums of the expression (2.6) over all grid cells

$$F = \sum f, \quad F_0 = \sum f_0, \quad \vec{G}_0 = \sum \mathbf{T}, \quad H_0 = \sum \mathbf{H},$$

we may write the quadratic approximation of the functional (2.5) in the form:

$$F = F_0 + (\vec{G}_0, \delta\mathbf{p}) + \frac{1}{2}(H_0\delta\mathbf{p}, \delta\mathbf{p}).$$

Formally, the problem of minimization may be solved easily: $\delta\mathbf{p} = -H_0^{-1}\vec{G}_0$ and the new set of parameters is $\mathbf{p}^{n+1} = \mathbf{p}^n + \delta\mathbf{p}$. However, (p_1, p_2, p_3, p_4, k)-parametrization of the metric g_{ij} is not successful for solving the problem of minimization. There are such situations when the great metric disturbances correspond to the small disturbance of $\delta\mathbf{p}$ (possibly, at some points only, for instance, at region corners). It leads up to very small step along the Newton direction, otherwise the metric degenerates, i.e. $\theta = g_{11}g_{22} - g_{12}^2 \le 0$. Hence, the new parametrization is chosen: the vector of variations $\vec{\xi} = (\xi_1, \xi_2, \xi_3, \xi_4, \xi_5)$ instead of $\delta\mathbf{p}$, where $\xi_l = \delta(\bar{\omega}_l)$, $l = \overline{1,4}$, and $\bar{\omega}_l$ is the relative deviation of l-th angle from the right one

$$\bar{\omega}_l = \frac{\omega_l - 0.5\pi}{0.5\pi},$$

$$\cos\omega_l = (-1)^{l+1}\frac{g_{12}}{\sqrt{g_{11}g_{22}}}\Big|_{(x_l,y_l)}, \qquad l = \overline{1,4},$$

$$\xi_5 = \sqrt{\frac{g_{22}}{g_{11}}}\delta\Big(\sqrt{\frac{g_{11}}{g_{22}}}\Big)\Big|_{(x^*,y^*)}.$$

The point $(x^*,\ y^*)$ is the point of a minimum of the function $\theta = g_{11}g_{22} - g_{12}^2$. Therefore, the final relations are the following:

$$\xi_l = \frac{(-1)^l}{0.5\pi d}\left\{-\frac{g_{12}}{2g_{11}}\delta g_{11} - \frac{g_{12}}{2g_{22}}\delta g_{22} + \delta g_{12}\right\}\Big|_{(x_l,\ y_l)},$$

$$\xi_5 = \left(\frac{\delta g_{11}}{2g_{11}} - \frac{\delta g_{22}}{2g_{22}}\right)\Big|_{(x^*,\ y^*)},$$

here $d = \sqrt{g_{11}g_{22} - g_{12}^2}$. The final quadratic form of the functional used in the computational practice is

$$F = F_0 + (grad\, F, \vec{\xi}) + \frac{1}{2}(F''\vec{\xi}, \vec{\xi}),$$

where F'' is an approximation of the Hesse matrix.

This quadratic form is used for the minimization of the original functional with respect to the metric parameters. Solving the system of linear equations

$$F''\vec{\xi} = -grad\, F \tag{2.7}$$

with conditions

$$\xi_l = \bar{\varphi}_l - \bar{\omega}_l,\ l = \overline{1,4}, \tag{2.8}$$

one can found $\delta\mathbf{p}$, and the new iteration $\mathbf{p} = \mathbf{p} + \delta\mathbf{p}$.

The linear system (2.7) can be solved without conditions (2.8); they are satisfied (with an accuracy of approximation) in the process of minimization of the functional with respect to all variables.

Calculation of control functions x_0, x_1, y_0, y_1 begins with the proper modification of original functional (2.4) as well:

$$\tilde{\Phi} = \frac{1}{2}\int\int \frac{h_{22}H_{11} - 2h_{12}H_{12} + h_{11}H_{22} - 2\sqrt{h_{11}h_{22} - h_{12}^2}H_0}{\sqrt{H_{11}H_{22}}\sqrt{h_{11}h_{22}}}dsdt,$$

where

$$H_{11} = u_s^2 + v_s^2, \quad H_{22} = u_t^2 + v_t^2,$$
$$H_{12} = u_s u_t + v_s v_t, \quad H_0 = u_s v_t - u_t v_s.$$

The considered functional is modified and regularized in a manner similar to above presented to provide a nonnegativity and convexity of the functional integrand

$$f = 2sh^2\frac{p-P}{2} + 2\sin^2\frac{\varphi - \Phi}{2}.$$

At each cell (s,t), the function $f(p,\varphi)$ is approximated by three first terms of Taylor series. The arguments p, φ are complicated functions of independent quantities x_0, x_0', x_1, x_1' [3].

To write the quadratic form of the functional, the following superposition of transformations is used

$$(p,\varphi) \xrightarrow{Q_1} (h_{11}, h_{22}, h_{12}) \xrightarrow{Q_2} (x, y, x_s, x_t, y_s, y_t) \xrightarrow{Q_x} (x_0, x_0', x_1, x_1')$$

and it is slightly changed for vertical boundaries

$$(p,\varphi) \xrightarrow{Q_1} (h_{11}, h_{22}, h_{12}) \xrightarrow{Q_2} (x, y, x_s, x_t, y_s, y_t) \xrightarrow{Q_y} (y_0, y_0', y_1, y_1').$$

As a result of calculations at each point (s,t), a quadratic approximation of function f is found

$$f \cong f_0 + (grad\, f, \delta\vec{z}) + \frac{1}{2}(H\delta\vec{z}, \delta\vec{z}), \quad \vec{z} = (z_1, z_2, z_3, z_4) \equiv (x_0, x_0', x_1, x_1').$$

After summation by argument t, the following relation is obtained for each argument s

$$\bar{f} = \bar{f}_0 + (\overline{grad\, f}, \delta\vec{z}) + \frac{1}{2}(\bar{A}\delta\vec{z}, \delta\vec{z}),$$

[3]Calculating each pair of functions (x_0, x_1) and (y_0, y_1) is performed independently. So, only the pair of horizontal functions is considered in this study.

where $\bar{f}$ denotes a sum by t (over vertical set of cells).

Searching functions $x_0(s)$ and $x_1(s)$ are defined on the grid $\{s_i = (i-1)h, i = 1, ..., N+1, h = 1/N\}$. Their variations $\delta\vec{z} = (\delta x_0, \delta x_0', \delta x_1, \delta x_1')$ is defined in the midpoint of each interval $(i, i+1)$ of a grid

$$\delta x_0 = \frac{\delta x_0(i+1) + \delta x_0(i)}{2}, \delta x_0' = \frac{\delta x_0(i+1) - \delta x_0(i)}{h}.$$

This transformation is linear, so the quadratic form is not changed. The minimum condition of a quadratic functional relative to unknown functions $\vec{\delta}_i = (\delta x_0(i), \delta x_1(i), i = 1, ..., N+1)$ has the form of the vector equations

$$\hat{A}_i\vec{\delta}_{i-1} + \hat{B}_i\vec{\delta}_i + \hat{C}_i\vec{\delta}_{i+1} = \vec{G}_i, \ i = 2, 3, ..., N$$

with uniform boundary conditions $\vec{\delta}_1 = \vec{0}$, $\vec{\delta}_{N+1} = \vec{0}$, where $\hat{A}_i, \hat{B}_i, \hat{C}_i$ are (2×2)-matrices. These equations are solved by the matrix sweep method.

A described algorithm is the Newton method for a minimization with respect to a pair of functions x_0, x_1 (or y_0, y_1).

If the position of boundary points is unknown [4], the alternating *algorithm of point movement along boundaries* should be utilized by the following procedure.

Let $l(s)$ be the distance from the initial point at the boundary of the region Ω, normalized to the boundary length, where s is the coordinate on the boundary of the unit square K_0, $0 \le s \le 1, 0 \le t \le 1$.

Our goal is to find a small variation from a minimization of functional (2.5) $\delta x_0(s) = x_0^{new}(s) - x_0(s)$ with the help of the algorithm given above ($x_0(s)$ is known). Then, $\delta l = l^{(\nu+1)} - l^{(\nu)} = l'_{x_0}\delta x_0$, $l^{(\nu)}$ is an iterative approximation. Take into account the following relations for a point with the coordinate s:

$$l^{(\nu+1)} = l^{(\nu)} + \frac{l'_s}{x_0'}(x_0^{new} - x_0),$$

where

$$l'_s = \frac{l(s+\Delta s) - l(s-\Delta s)}{2\Delta s},$$

$$x_0' = \frac{x_0(s+\Delta s) - x_0(s-\Delta s)}{2\Delta s}.$$

So, a new position $l^{(\nu+1)}$ of boundary points is found.

The final stage of the presented algorithm is *the calculation of the functions* $u(s,t)$, $v(s,t)$. The algorithm is based on solving the system of difference elliptic equations by the multigrid method. The generation of a quasi-isometric coordinate system is based on the solution of variational problem (2.3). The grid is obtained as a solution of Euler–Lagrange variational equations:

[4] Control functions are known and functional (2.4) is minimized

$$\frac{\partial}{\partial s}(A\frac{\partial u}{\partial s}) + \frac{\partial}{\partial t}(B\frac{\partial u}{\partial t}) - \frac{\partial}{\partial s}(C\frac{\partial u}{\partial t}) - \frac{\partial}{\partial t}(C\frac{\partial u}{\partial s}) = 0,$$

$$\frac{\partial}{\partial s}(A\frac{\partial v}{\partial s}) + \frac{\partial}{\partial t}(B\frac{\partial v}{\partial t}) - \frac{\partial}{\partial s}(C\frac{\partial v}{\partial t}) - \frac{\partial}{\partial t}(C\frac{\partial v}{\partial s}) = 0.$$

Here A, B, C are the coefficients of equations given as functions with respect to s, t:

$$A = \frac{h_{22}}{d}, \quad B = \frac{h_{11}}{d}, \quad C = \frac{h_{12}}{d},$$

$$d = \sqrt{g_{11}g_{22} - g_{12}^2}.$$

2.4 Numerical Experiments

Firstly, we show the dependence of the functional from metric parameters. One considers a region showed in Figure 2.2.

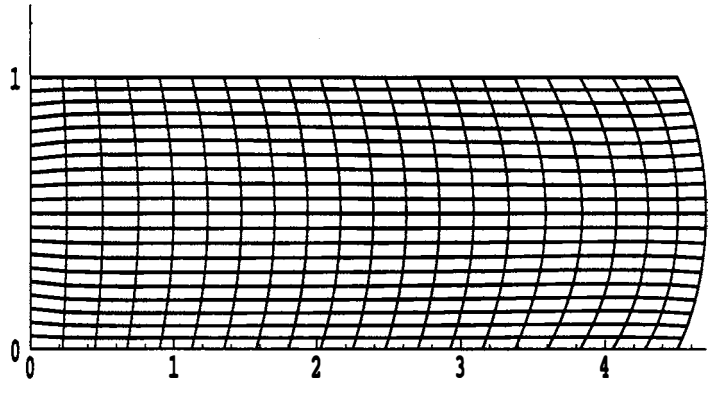


Figure 2.2: Region and quasi-isometric grid

As a result of computation of a quasi-isometric mapping, the metric parameters $s_1^* = s_4^* = 0, s_2^* = s_3^* = -0.893, k^* = 0.551$ are found. Let other parameters (grid and control function) be optimal and fixed. We study the functional as the function $f(s_2, k)$ in a small vicinity of the minimum by a variation of the parameters s_2 and k. An exact minimal value is $f = 0$. The profiles $f(s_2^*, k)$ and $f(s_2, k^*)$ are shown in Figure 2.3. In 2.4, the plot of $f(s_2, k)$ in the vicinity of a minimum is given. One can see that near the minimum the metric is degenerated (at any point (s_2, k), where the metric does not exist, we put $f = 10$). These pictures give a hint on computational difficulties that we collide with in the process of the algorithm development for the solution of the minimization problem.

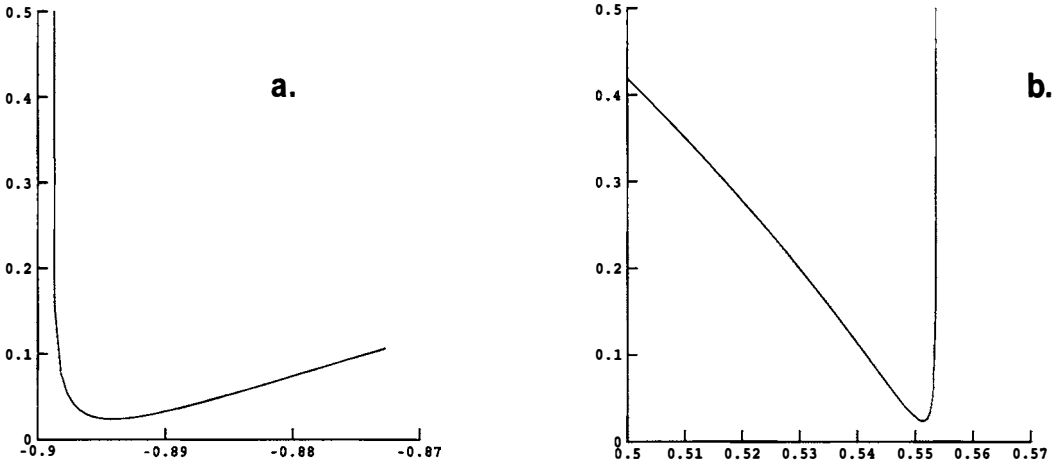


Figure 2.3: Functional behavior versus parameter variation: a). $f(s_2^*, k)$; b). $f(s_2, k^*)$

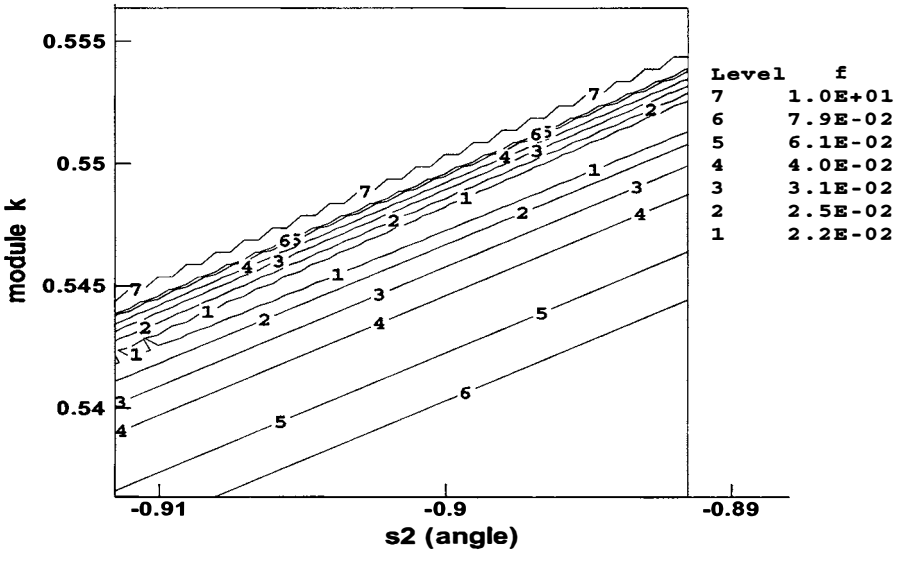


Figure 2.4: Isolines of f

Figure 2.5 presents an example of computations: a region with quasi-isometric grid and their images on a constant curvature surface (in the Poincare model) and in the parametric square K.

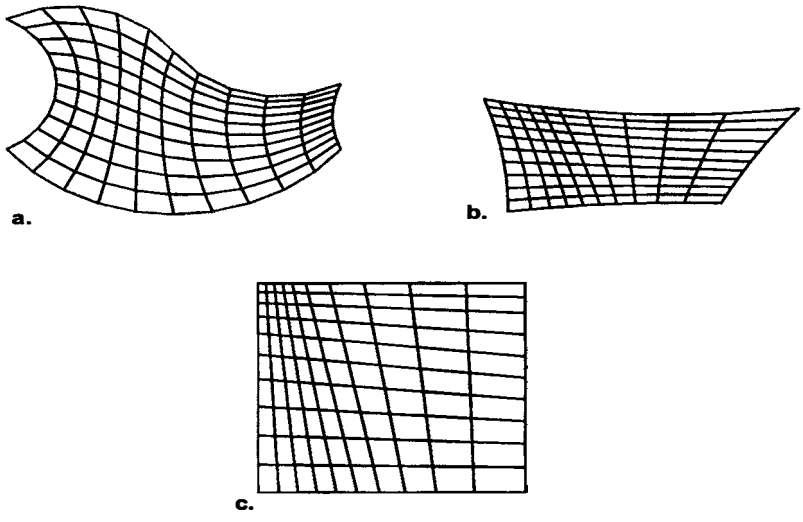


Figure 2.5: a). region ($\varphi_1 = 0.45\pi$, $\varphi_2 = 0.67\pi$, $\varphi_3 = 0.21\pi$, $\varphi_4 = 0.32\pi$) and grid; b). image on constant curvature surface; c). image in parametric region K

To evaluate quality of quasi-isometric mapping and compare a few mappings we introduce the coefficient of isometry Q that can be computed by a few procedures.

There is a simple geometric procedure to compute the isometry coefficient distribution by a grid, constructed in a region. It is naturally for each grid cell to construct an approximation mapping of parametrization gridding rectangle with the help of an affine mapping and find isometry coefficients as singular values of matrix, defining this mapping.

Let a grid cell in the parametric rectangle be a rectangle with sides a and b, see Figure 2.6; and the ratio of its sides denotes $\nu = a/b$. Let it be mapped onto a quadrangle on physical plane with vertices x_1, y_1; x_2, y_2; x_3, y_3; x_4, y_4. We construct in each these cells new quadrangle restricted by lengths connecting the midpoints of boundary edges, see Figure 2.7.

It is easy to verify that both these quadrangles (these are shaded in the pictures) are parallelograms and the mapping of the first one onto the second one is affine. This affine transformation is defined by the matrix

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} = \begin{pmatrix} \dfrac{(x_2+x_3)-(x_1+x_4)}{2a} & \dfrac{(x_3+x_4)-(x_2+x_1)}{2b} \\ \dfrac{(y_2+y_3)-(y_1+y_4)}{2a} & \dfrac{(y_3+y_4)-(y_2+y_1)}{2b} \end{pmatrix}.$$

The singular values λ_{max}, λ_{min} of matrix A are the needed approximation of isometry coefficient in a cell under investigation. The coefficient of isometry is

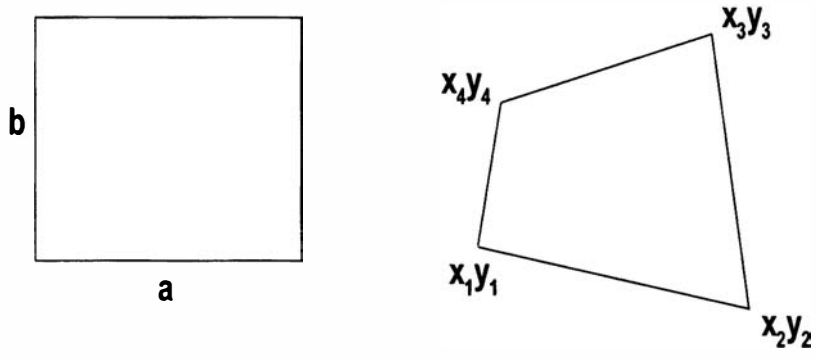


Figure 2.6: Parametric cell (left) and its image (right)

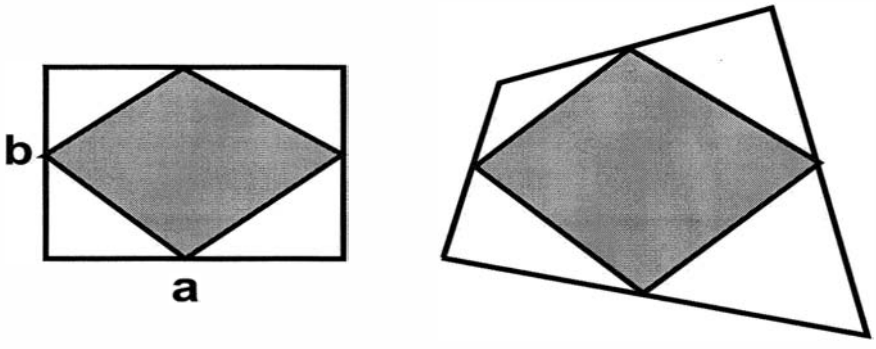


Figure 2.7: Affine equivalent parallelograms

$$Q = \frac{\max_{(x,y)} \lambda_{max}}{\min_{(x,y)} \lambda_{min}}.$$

For each ratio $\nu = a/b$, the isometry coefficient $Q(\nu)$ can be computed easily and, therefore, the extremal value ν can be evaluated by a numerical minimization of $Q(\nu)$. The plot

$Q(\nu)$ is given in Figure 2.8 for a region shown in Figure 2.9. For extreme value $\nu = 5$ in Figure 2.9 the isolines of the ratio $\lambda_{max} \setminus \lambda_{min}$ (local distortion) is presented.

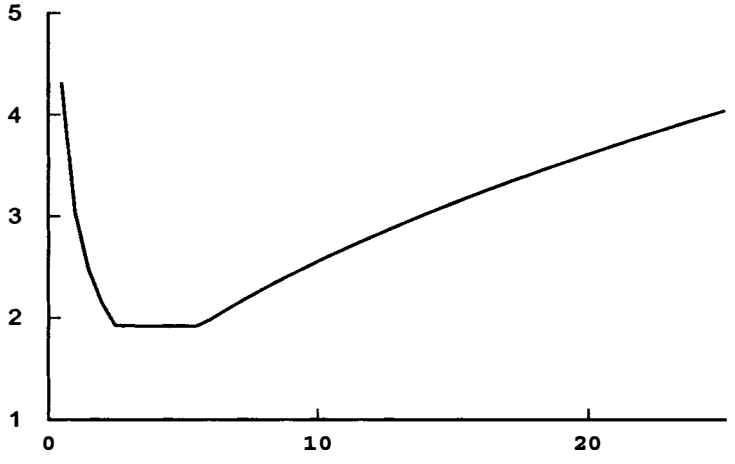


Figure 2.8: Isometry coefficient $Q(\nu)$ versus ν

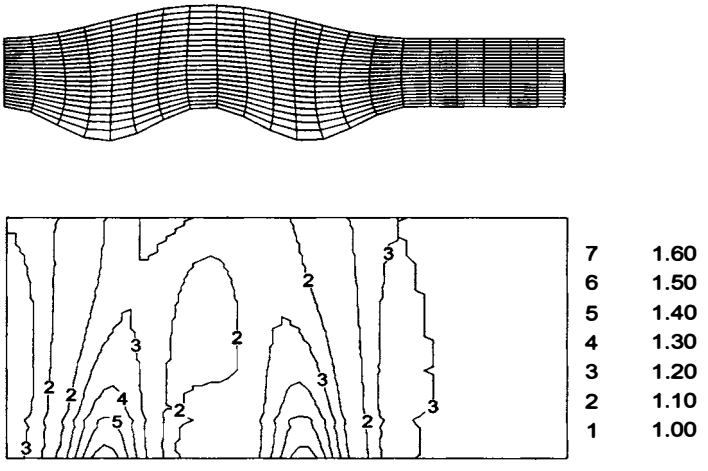


Figure 2.9: Grid and local isometry coefficient

2.5 Conclusion

The quasi-isometric mapping technique presents an approach based on the rigorous mathematical formulation. Numerical validation in geometrically complex regions is performed and demonstrated the possibility to construct efficient tools for quasi-isometric grid generation. This technique can be extended for multi-block grids. There are a few open questions and future investigations in geometry and mathematical analysis are needed to obtain formulation and solution of the problem of optimal quasi-isometric mapping.

Bibliography

[1] P.L.Chebyshov, On construction of geographical maps. Writing,Vol.I,pp. 233–236, Sur la consturtion des geographiques, Bull. Acad. Sci. Petersb., XIV, 1856.

[2] D.A.Grave, On base problem of mathematical theory of geographical map construction, 1896.

[3] D.A.Grave, Demonstration d'un theoreme de Tchebycheff generalise, Crelle Journ.,140, 1911.

[4] S.K.Godunov, V.M.Gordienko, G.A.Chumakov, Quasi-isometric Parameterization of a Curvilinear Quadrangle and a Metric of Constant Curvature, Siberian Advances in Mathematics, 1995, V. 5, N 2, pp. 1–20.

[5] S.K.Godunov, V.M.Gordienko, G.A.Chumakov, Variational principle for 2D regular quasi-isometric grid generation, Int. J. Comput. Fluid Dynamics, 1995, V. 5, pp. 99–118.

[6] F.Schur, Ueber den Zusammenhang den Räume Konstanten Riemannschen krümmungsmasses mit den projectiven Raümen, Math.Ann., N 27, 1848 (German).

Chapter 3

TRIANGLE DISTORTION UNDER QUASIISOMETRIES

Andrew V. Lygin and Vladimir M. Miklyukov
Volgograd State University

We bring some estimates of a triangle distortion under quasiisometric mappings between subdomains of $\mathbb{R}^2$ with given Riemannian metrics.

3.1 Introduction

Below we consider a problem that is appeared in the well-known Godunov method to construct two-dimensional grids on surfaces [1]. Namely, we decompose a standard domain in the w-plane with a Riemannian metric and form a grid in a given subdomain of the z-plane by an auxiliary quasiconformal mapping $z = \varphi(w)$ (see also [2], [3]). The existence and quality of the grid are connected with the angle distortions under quasiconformal mappings φ and, in particular, with the orientation preservation of triples of points.

One of the quality characteristics is the smallest angle among the triangle grid. It is clear that too sharp angles can be obstructions for the numerical solution of the physical problem on the considered grid with the necessary accuracy. Thus it is important to evaluate distortion of angles under different mappings.

The following statement is the key for this approach [4, Section D, Chapter III]: *Let φ be a K-quasiconformal mapping of $\mathbb{R}^2$ to itself with $K < \sqrt{3}$. Then the vertices of each equilateral triangle are transformed to the vertices of some triangle of the same orientation.*

The most difficulty in the test of these conditions is evoked by the assumption that the map φ must by of $\mathbb{R}^2$ to $\mathbb{R}^2$. In the general case the map is given on a proper subdomain of $\mathbb{R}^2$ and the problem of its K-quasiconformal extentions up to $\mathbb{R}^2 \to \mathbb{R}^2$ with $K < \sqrt{3}$ is highly complicated

This paper is concerned with a direct approach based on estimates of a triangle under φ. However, we essentially reduce the class of mappings φ. Namely, we study angle distortion under quasiisometries of quasiconvex subdomains of $\mathbb{R}^2$.

In particular, we give an estimate of triangle angles, prove some criteria of the preservation of their angles, and employ some conditions for mappings from surfaces to $\mathbb{R}^m$ to

satisfy the bilipschitz condition.

3.2 Surfaces

We start with the terminology. By [11], we shall use the following conception.

Definition 1. The *dilatation* of a mapping f between metric spaces X, Y is the (possibly infinite) number

$$\operatorname{dil}(f) = \sup_{\substack{x,x' \in X \\ x \neq x'}} \frac{r(f(x), f(x'))}{r(x, x')},$$

where "r" denotes the metrics (distances) dist_X in X and dist_Y in Y.

A mapping f is called *Lipschitz* if $\operatorname{dil}(f) < \infty$.

If every point $x \in X$ has a neighbourhood U such that the restriction $f^* = f|_U$ satisfies $\operatorname{dil}(f^*) < \infty$, then f is called *locally Lipschitz*.

Let $D \subset \mathbb{R}^2$ be a domain and let Ω be a surface in $\mathbb{R}^m$, $m \geq 2$, given by a locally Lipschitz vector-function

$$x = f(u, v) = (f_1(u, v), \ldots, f_m(u, v)) : D \to \mathbb{R}^m, \quad x = (x_1, \ldots, x_m). \tag{3.1}$$

In the general case the surface Ω can have self-intersections. We shall say that a surface Ω is *embedded* in $\mathbb{R}^m$ if the vector function f realizes a homeomorphic mapping from a domain D onto a set $f(D)$ with a metric (and hence a topology !) induced by $\mathbb{R}^m$. A surface Ω is *immersed* in $\mathbb{R}^m$ if a vector-function f has the described property locally in D.

Because the vector-function f is locally Lipschitz, then by Rademacher Theorem [12, 3.1.6] almost everywhere on D there exists a total differential $df(u, v)$.

Let $(u, v) \in D$ be a point in which f has the differential. By the symbol

$$f' = \begin{pmatrix} f'_{1u} f'_{2u} \cdots f'_{mu} \\ f'_{1v} f'_{2v} \cdots f'_{mv} \end{pmatrix}$$

we denote the derivative of f at the point (u, v) in which such derivative exists. Using standard designations

$$g_{11} = \left|\frac{\partial f}{\partial u}\right|^2, \quad g_{12} = g_{21} = \left\langle \frac{\partial f}{\partial u}, \frac{\partial f}{\partial v} \right\rangle, \quad g_{22} = \left|\frac{\partial f}{\partial v}\right|^2,$$

we define the first quadratic form of the surface Ω on the domain D

$$ds^2_\Omega = g_{11}\, du^2 + 2g_{12}\, du\, dv + g_{22}\, dv^2.$$

The simplest view of the first quadratic form of the surface Ω is in isothermal coordinates. However, the question – what are isothermal coordinates on a non-smooth surface – has no a sufficiently good answer today. We shall use the following definition.

Definition 2. Let Ω be a surface given over a domain $D \subset \mathbb{R}^2$ by a locally Lipschitz vector-function (3.1). The variables (u, v) are called *isothermal coordinates on the surface* Ω if

$$g_{11}(u,v) = g_{22}(u,v)\,, \quad g_{12}(u,v) = 0 \quad \text{a.e. on} \quad D\,. \tag{3.2}$$

Moreover, if (u, v) are isothermal coordinates on the surface Ω, we have

$$ds^2_\Omega = \lambda(u,v)\,(du^2 + dv^2)$$

almost everywhere on the domain D. Here, $\lambda = g_{11} = g_{22}$.

The first among the conditions (3.2) means that the tensions of f along lines u, $v =$ const coincide at points in which df exists. The second condition implies the mutually orthogonality of the images of these lines at the corresponding points $x = f(u, v)$. Thus, at every point $(u, v) \in D$ in which the total differential df exists and simultaneously the relations (3.2) hold, the mapping $f : D \to \Omega$ is conformal, that is, it conserves angles between curves. The existence of isothermal coordinates on non-smooth surfaces is studied incompletely. For surfaces of the class $W^{l,p}$ with $l \geq 3$ and $p > 2$, the existence of isothermal coordinates can be obtained from well-known results of Bojarski and Vekua (see. [6, §6, Chapter II]). Similar results for Lipschitz graphs belong Morrey [5], and for locally Lipschitz graphs can be obtained (under different assumptions on domains of graph definitions) from results Bers [7], Belinski [8, Theorem 9], Martio and Miklyukov [9].

The general case of non-parametric locally Lipschitz surfaces was considered in the recent article of Miklyukov [10].

3.3 Quasiisometry

A mapping $f : D \to \mathbb{R}^m$ is called (q, Q)-*quasiisometric* if there exist constants $q > 0$ and $Q > 0$ such that for arbitrary points $w = (u, v)$ and $w' = (u', v')$ in D, we have

$$q\,|w - w'| \leq |f(w) - f(w')| \leq Q\,|w - w'|\,. \tag{3.3}$$

Here and below, the quantity $|f(w) - f(w')|$ means the Euclidean distance between points $f(w)$, $f(w') \in \mathbb{R}^m$.

The following condition is useful in the testing of the assumption that (3.3) holds. We follow [13].

Denote $M_{2,m}$ the linear space of the $2 \times m$–matrices with natural coefficients.

Let $f(u, v)$ be a locally Lipschitz vector-function defined on a domain $D \subset \mathbb{R}^2$ and let (u, v) be a point of the differentiability of f.

For an arbitrary matrix $C \in M_{2,m}$, we set

$$|C| = \max_{|h|=1} |h\,C|\,, \quad h = (h_1, h_2)\,.$$

In particular, for the derivative f' of the vector-function $f : D \to \mathbb{R}^m$, it is easy to see that

$$|f'| = \sqrt{m} \max_{i=1,2,\dots,m} |\nabla f_i|\,,$$

where the symbol ∇f_i means the formal gradient (f'_{iu}, f'_{iv}).

If $C(u,v) : D \subset \mathbb{R}^2 \to M_{2,m}$ is a matrix-function, then let

$$\|C\|_D = \operatorname{ess\,sup}_{(u,v)\in D} |C(u,v)|\,.$$

Theorem 1. *Let $D \subset \mathbb{R}^2$ be a convex domain and let $f : D \to \mathbb{R}^m$ be a locally Lipschitz mapping, $m \geq 2$. Suppose that there exists a (q,Q)–quasiisometry $h : U \to \mathbb{R}^m$ such that*

$$\|f' - h'\|_D \leq A < \infty\,. \tag{3.4}$$

Then for arbitrary points $w = (u', v')$, $w'' = (u'', v'') \in D$ we have

$$|(f(w'') - h(w'')) - (f(w') - h(w'))| \leq (Q + A)\,|w'' - w'|\,. \tag{3.5}$$

Moreover, if $A < q$ then f is homeomorphic on D and

$$(q - A)\,|w'' - w'| \leq |f(w'') - f(w')| \leq (Q + A)\,|w'' - w'|. \tag{3.6}$$

Proof. Denote P the set of the points $w = (u, v) \in D$, in which the vector-function $f - h$ is non-differentiable. By the Rademacher theorem, the two-dimensional measure $H^2(P) = 0$.

The relation (3.4) implies the estimate

$$|f'(w) - h'(w)| \leq A\,, \tag{3.7}$$

which holds almost everywhere on D.

Fix arbitrarily points $w', w'' \in D$ and denote $l(w', w'')$ the line segment joining w' and w''. Since the domain D is convex then $l(w', w'')$ lies in D. Let $l(\tilde{w}', \tilde{w}'')$ be a line segment with the endpoints $\tilde{w}'$ and $\tilde{w}''$ achieved by a parallel translation of the segment $l(w', w'')$. For almost everywhere such line segments which are sufficiently close to $l(w', w'')$, we have

$$H^1(l(\tilde{w}', \tilde{w}'') \setminus P) = 0. \tag{3.8}$$

Let $l(\tilde{w}'_k, \tilde{w}''_k)$ be a sequence of segments satisfying (3.8) and such that

$$\tilde{w}'_k \to w'\,, \quad \tilde{w}''_k \to w''\,.$$

Because f and h are locally Lipschitz, then the vector-function $f-h$ is absolutely continuous along $l(\tilde{w}'_k, \tilde{w}''_k)$ and almost everywhere on $l(\tilde{w}'_k, \tilde{w}''_k)$ there exists the derivative $f'-h'$. Integrating, we have

$$|(f(w''_k)-h(w''_k))-(f(w'_k)-h(w'_k))|$$

$$=\left|\int_0^1 (w''_k-w'_k)\,(f'(w'_k+t(w''_k-w'_k))-h'(w'_k+t(w''_k-w'_k)))\,dt\right|$$

$$\leq \int_0^1 |f'(w'_k+t(w''_k-h'(w'_k+t(w''_k-z'_k))))|\,|w''_k-w'_k|\,dt.$$

Using (3.7), we obtain

$$|(f(w''_k)-h(w''_k))-(f(w'_k)-h(w'_k))| \leq A\,|w''_k-w'_k|\,.$$

Setting $k\to\infty$, we arrive at the estimate

$$|(f(w'')-h(w''))-(f(w')-h(w'))| \leq A\,|w''-w'| \qquad (w',\, w''\in D). \tag{3.9}$$

Let $\psi(w)=f(w)-h(w)$. For an arbitrary pair of points $w',\ w''\in D$ we have

$$f(w'')-f(w')=(\psi(w'')-\psi(w'))+(h(w'')-h(w'))\,.$$

Thus,

$$|f(z'')-f(z')| \leq |\psi(w'')-\psi(w')|+|h(w'')-h(w')|\,.$$

Using (3.9) and the (q, Q)-quasiisometry of h, we obtain

$$|f(w'')-f(w')| \leq (Q+A)\,|w''-w'|\,.$$

Analogously,

$$|f(w'')-f(w')| \geq |h(w'')-h(w')|-|\psi(w'')-\psi(w')|$$

and further,

$$|f(w'')-f(w')| \geq (q-A)\,|w''-w'|\,.$$

Thus the estimates (3.5), (3.6) are proved.

In the case $h=w$ this statement is proved in [13].

3.4 Distortion and Quasiconvexity

Let D be a subdomain of $\mathbb{R}^2$ and let $\Omega \subset \mathbb{R}^m$ be a locally Lipschitz surface given by a vector-function (3.1) realizing an enclosure.

Define an inner distance $r_\Omega(x', x'')$ between points x' and x'' on the surface Ω. Since the surface Ω is embedded in $\mathbb{R}^m$, then by the unique way in the domain D, there exist points $w' = (u', v')$ and $w'' = (u'', v'')$, for which $x' = f(w')$ and $x'' = f(w'')$. We put

$$r_\Omega(x', x'') = \inf_\gamma \int_\gamma ds_\Omega\,,$$

where the infimum is taken over the rectifiable arcs $\gamma \subset D$ joining points w' and w''.

Because the surface Ω is locally Lipschitz, then the distance $r_\Omega(x', x'') < \infty$ for all x', $x'' \in \Omega$.

Definition 3. [11, Section 1.14] The *distortion* of a surface $\Omega \subset \mathbb{R}^m$ is the quantity

$$\operatorname{distort}(\Omega) = \sup_{\substack{x', x'' \in \Omega \\ x' \neq x''}} \frac{r_\Omega(x'', x')}{|x'' - x'|}\,.$$

It is clear, that for every non-parametric surface $\Omega \subset \mathbb{R}^m$ we have $\operatorname{distort}(\Omega) \geq 1$.

If the surface Ω is a domain $D \subset \mathbb{R}^2$ with the Euclidean metric $|dw|$, then the quantity $r_D(w', w'')$ is the well-known Mazurkiewicz distance between w' and w'' in D. The assumption $\operatorname{distort}(\Omega) < \infty$ implies that

$$r_D(w'', w') \leq C\,|w'' - w'|\,, \quad C = \operatorname{distort}(\Omega)\,.$$

Such domains $D \subset \mathbb{R}^2$ are called $C-$ *quasiconvex* (see [11, p. 393]).

It is easy to see, that every convex domain is 1-quasiconvex.

Theorem 2. *If a domain $D \subset \mathbb{R}^2$ is convex and a mapping $\varphi : D \to \mathbb{R}^2$ satisfies the property $\|\varphi' - I\| = A < 1$, where $I \in M_{2,2}$ is a unit matrix, then the set $D' = \varphi(D)$ is a quasiconvex domain with the constant $C = (1 + A)/(1 - A)$.*

Proof. Indeed, consider a pair of mappings

$$x = \varphi(w) : D \to \mathbb{R}^2 \quad \text{and the identity mapping} \quad x = w\,.$$

Since the identity mapping is a $(1, 1)$-quasiisimetry and its derivative I differs from φ' neither more then $A < 1$, then by Theorem 1 we conclude that φ is homeomorphic and $D' = \varphi(D)$ is a domain.

Let x', x'' be a pair of points in the domain D' and let w', w'' be a pair of their preimages. According to Theorem 1, we may write

$$|x'' - x'| = |f(w'') - f(w')| \leq (1 + A)\,|w'' - w'|\,.$$

Let $l(w', w'')$ be a segment joining the points w' and w''. The domain D is convex and hence $l(w', w'') \subset D$. Suppose that the points $w', w_1, w_2, \ldots, w_k, w''$ follow one after another along the segment $l(w', w'')$. Then

$$|\varphi(w') - \varphi(w_1)| + \ldots + |\varphi(w_k) - \varphi(w'')| \leq$$

$$\leq (1 + A)\,|w' - w_1| + \ldots + (1 + A)\,|w_k - w''| = (1 + A)\,|w' - w''|\,.$$

Choosing the partition $w', w_1, w_2, \ldots, w_k, w''$ of the segment $l(w', w'')$ arbitrarily small and observing that the left side of this relation can be arbitrarily close to length $f(l(w', w''))$, we obtain

$$\text{length}\, f(l(w', w'')) \leq (1 + A)\,|w' - w''|\,.$$

Thus,

$$r_{D'}(x', x'') \leq (1 + A)\,|w' - w''|\,.$$

However by the relation (3.6) we have

$$(1 - A)\,|w' - w''| \leq |x' - x''|\,,$$

and hence,

$$r_{D'}(x', x'') \leq \frac{1 + A}{1 - A}\,|x' - x''|\,.$$

It means the quasiconvexity of D' and the theorem is proved.

The following statement is useful for applications.

Theorem 3. *Suppose that a domain $D \subset \mathbb{R}^2$ is C-quasiconvex.*

If a mapping $f : D \to \mathbb{R}^m$ is (q, Q)-quasiisometric, then the surface $\Omega = f(D)$ has a finite distortion, moreover

$$\text{distort}\,(\Omega) \leq C\,\frac{Q}{q}\,. \tag{3.10}$$

In particular, we have:

i) if the surface Ω is given in some isothermal coordinates such that the relations (3.2) *hold and moreover*

$$0 < \lambda' = \operatorname{ess\,inf}_{(u,v)\in D}\lambda(u, v) \leq \operatorname{ess\,sup}_{(u,v)\in D}\lambda(u, v) = \lambda'' < \infty\,,$$

then

$$\text{distort}\,(\Omega) \leq C\,\frac{\lambda''}{\lambda'}\,. \tag{3.11}$$

ii) if the domain D is convex and there exists a (q, Q)-quasiisometry $h : D \to \mathbb{R}^m$, for which the assumption (3.4) holds with $A < q$, then

$$\text{distort}\,(\Omega) \leq C\,\frac{Q + A}{q - A}\,. \tag{3.12}$$

Proof. For arbitrary pair of points x', $x'' \in \Omega$ by the assumption of f to be Lipschitz with the constant Q, we may write

$$r_\Omega(x', x'') \le Q\, r_D(w', w'')\,,$$

where $w' = f^{-1}(x')$, $w'' = f^{-1}(x')$.

Indeed, let $\gamma \subset D$ be an arbitrary rectifiable arc joining the points w', $w'' \in D$. Fix $\epsilon > 0$ and choose an arbitrary broken line $\overline{w'w_1 \dots w_k w''}$ lying in D such that

$$|\text{length}\,(\overline{w'w_1 \dots w_k w''}) - r_D(w', w'')| < \epsilon\,.$$

Then

$$|f(w') - f(w_1)| + |f(w_1) - f(w_2)| + \dots + |f(w_k) - f(w'')| \le Q\,\text{length}\,(\overline{w'w_1 \dots w_k w''})\,.$$

Approximating γ by broken lines, we easily obtain the necessary statement.

Now we use the property f^{-1} to be $\frac{1}{q}$-Lipschitz and the C-quasiconvexity of the domain D. We have

$$\begin{aligned} |x' - x''| &= |f(w') - f(w'')| \ge q\,|w' - w''| \ge \\ &\ge \tfrac{q}{C}\, r_D(w', w'') \ge \tfrac{q}{CQ}\, r_\Omega(x', x'')\,. \end{aligned}$$

From this inequality, it follows (3.10).

In the case i), if the surface Ω is given in some isothermal coordinates by a vector-function $f : D \to \mathbb{R}^m$, then for every arc $\gamma \subset D$ it is realized that

$$\lambda'\,\text{length}\,(\gamma) \le \int\limits_\gamma ds_\Omega \le \lambda''\,\text{length}\,(\gamma)\,.$$

Thus, for every pair of points w', $w'' \in D$ we obtain

$$\lambda'\, r_D(w', w'') \le r_\Omega(f(w'), f(w'')) \le \lambda''\, r_D(w', w'')\,.$$

The estimate (3.11) follows now from the relation (3.10), which was proved above.

In the case ii), for the proof of (3.12) it is sufficient to combine the estimates (3.10) and (3.6).

3.5 Distortion of Triangles

Let Δ be a triangle in a domain U of the Eucliadean plane $\mathbb{R}^2$ with the coordinates $\zeta = (\xi, \eta)$. Let $\Delta' \subset D$ be a triangle in the $w = (u, v)$-plane with a metric ds_Ω, the vertices of which are the preimages of the vertices of the triangle Δ under a mapping $\varphi : U \to D$.

Theorem 4. *Suppose that a mapping $\varphi : U \to D$ is a (q, Q)-quasiisometry. Let $D = \varphi(U)$ be a quasiconvex domain and let α be the smallest angle of the triangle $\Delta \subset U$, and let α' be the correspondent angle of the triangle Δ'. Let*

$$p = \frac{q}{Q},\ r_1 = \frac{p}{2\cos\alpha},\ R_1 = p^{-1},\ r_2 = p\sin\alpha\,.$$

Then the following statements hold:
i) if $r_1 + r_2 > 1$ and $R_1 - r_2 < 1$, then

$$\tan\alpha' \geq \min\left\{\frac{\sqrt{4r_1^2 - (r_1^2 - r_2^2 + 1)^2}}{r_1^2 - r_2^2 + 1}, \frac{\sqrt{4R_1^2 - (R_1^2 - r_2^2 + 1)^2}}{R_1^2 - r_2^2 + 1}\right\}; \tag{3.13}$$

ii) if $r_1 + r_2 \leq 1$ or $R_1 - r_2 \geq 1$, then the quantity α' can equal to 0;
iii) for an arbitrary another angle β of the triangle Δ we have

$$\frac{\sin\beta'}{\sin\beta} \geq \frac{q}{Q}\frac{\sin\alpha'}{\sin\alpha}. \tag{3.14}$$

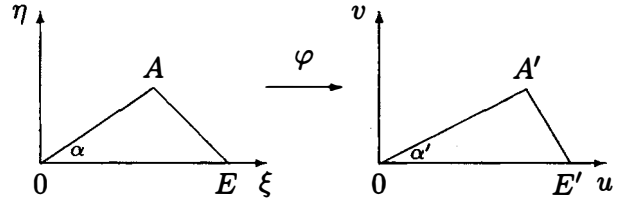
Proof. We may write

$$q\,|\zeta_1 - \zeta_2| \leq |\varphi(\zeta_1) - \varphi(\zeta_2)| \leq Q\,|\zeta_1 - \zeta_2|\,. \tag{3.15}$$

Denote the length of the largest side of the triangle Δ by E. We denote the length of the suitable side of the triangle Δ' by E'.

We locate the triangle Δ such that the vertex of the angle α lies in the origin and the side with length E lies on the axis 0ξ. Because the angle α is minimal and, hence, the opposite side is minimal, then we may define the set, in which the third vertex can lie. (We denote this set by A). It will be a segment of the straight line $\eta = \xi\tan\alpha$ in the interval $\xi \in [E/2,\ E\cos\alpha]$. The distance from the point $(0, 0)$ to the vertex A will be in the bounds between $(E/2)\cos\alpha$ and E (this is not the best estimate but it is sufficient for our aims), and the distance from the point $(E, 0)$ will be in the bounds between $E\sin\alpha$ and E.

Locate the triangle Δ' in the Euclidean plane such that the angle α' lies in the origin and the side of the length E' lies on the axis 0ξ. By (3.15), the distance from the point $(0, 0)$ to the point $A' = \varphi(A)$ will lie in the bounds between $r_1' \equiv q(E/2)\cos\alpha$ and $R_1' \equiv QE$ and the distance from the point $(E', 0)$ lies in the bounds between $r_2' \equiv qE\sin\alpha$ and $R_2' = R_1'$.



As the result, we obtain the following system of inequalities described the set, in which the point A' can lie:

$$\begin{cases} \xi^2+\eta^2 \geq r_1' \\ \xi^2+\eta^2 \leq R_1' \\ (\xi-E')^2+\eta^2 \geq r_2' \\ (\xi-E')^2+\eta^2 \leq R_2' \\ \xi \geq 0 \\ \eta \geq 0. \end{cases} \tag{3.16}$$

The similarity transformations preserve magnitude of the angles. We decrease then the triangle Δ' for the E' times. Then we obtain

$$r_1 = \frac{r_1'}{E'} = \frac{qE}{2E'\cos\alpha} \geq \frac{q}{2Q\cos\alpha},\ R_1 = \frac{R_1'}{E'} = \frac{QE}{E'} \leq \frac{Q}{q},$$

$$r_2 = \frac{r_2'}{E'} = \frac{qE\sin\alpha}{E} \geq \frac{q\sin\alpha}{Q},\ R_2 = \frac{R_2'}{E'} = \frac{QE}{E'} \leq \frac{Q}{q}.$$

Now (3.16) implies the system

$$\begin{cases} \xi^2+\eta^2 \geq r_1 \\ \xi^2+\eta^2 \leq R_1 \\ (\xi-1)^2+\eta^2 \geq r_2 \\ (\xi-1)^2+\eta^2 \leq R_2 \\ \xi \geq 0 \\ \eta \geq 0. \end{cases} \tag{3.17}$$

Under done assumptions, the minimum of the quantity α' can be reached only at two points, first, at the point of the intersection of the circle $R_{r_1}(0,0)$ and the circle $R_{r_2}(1,0)$, or at the point of the intersection of the circles $R_{R_1}(0,0)$ and $R_{r_2}(1,0)$. The values

$$\tan\alpha' = \frac{\sqrt{4r_1^2-(r_1^2-r_2^2+1)^2}}{r_1^2-r_2^2+1}$$

and

$$\tan\alpha' = \frac{\sqrt{4R_1^2-(R_1^2-r_2^2+1)^2}}{R_1^2-r_2^2+1}$$

are corresponding to these points, respectively.

If these circles are not intersecting (it is possible in the case, if $r_1+r_2 \leq 1$ or $R_1-r_2 \geq 1$), then the point A' can lie on the axis $O\xi$, and hence, the angle α' can equal to 0.

The statement iii) follows from the sine theorem for triangles and (3.15).

3.6 Orientation

As above, let U be a domain in the ζ-plane and let $w = \varphi(\zeta)$ be a homeomorphic mapping from the domain U onto the domain D.

The quasiisometry does not guarantee that the mapping φ preserves the orientation of the triples of points. We will also need an information on a deviation between $\varphi(\zeta)$ and ζ.

Theorem 5. *Suppose that there exists a constant $P > 0$ such that for all points $\zeta \in U$*

$$|\varphi(\zeta) - \zeta| < \frac{P}{T}, \tag{3.18}$$

where T is length of the largest side of the triangle.

If a triangle $\Delta \subset U$ has angles not less, than α and

$$\alpha > \arctan \frac{4P}{T^2}, \tag{3.19}$$

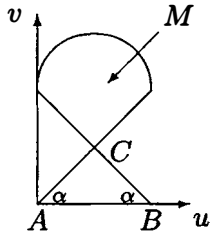
then the triple of the vertices of Δ preserves the orientation under the mapping φ.

In particular, suppose that the domain U is convex, $0 \in U$, and the mapping φ preserves the fixed point $\zeta = 0$ and satisfies the condition

$$\|\varphi' - I\| < A, \tag{3.20}$$

where $I \in M_{2,2}$ is the unit matrix. Suppose, in addition, that the triangle Δ has one of the vertices at the point $\zeta = 0$ and its angles are greater than $\arctan(4(1 + A)/T)$. Then the triple of the vertices of Δ preserves its orientation.

Proof. Denote Δ' the triangle which is formed by the images of the triple of the vertices of Δ. Locate the triangle Δ in the w-plane as on the drawing. Let M be a set in which the third vertex C can lie.



If $\varphi(C)$ lies higher than the base of Δ', then φ preserves the orientation of the triangle. Thus in the worst case, vertices A and B are displaced up at the distance P/T, and the vertex C lies in the down part of M and is displaced at the distance P/T. If the distance from the axis $0u$ to the set M greater that $2P/T$, then $\varphi(C)$ does not intersect the base. It follows from (3.19).

The second part of the theorem follows from its first part and Theorem 1. Here it is sufficient to observe that assumptions (3.20) and $\varphi(0) = 0$ imply

$$\|\varphi(\zeta) - \zeta\| < (1 + A)\, T$$

everywhere in Δ'.

3.7 Conclusion

We pointed to a few of distortion estimates under qusiisometric mappings useful for the construction of grids on surfaces. The set of such estimates does not become exhausted by them. Clearly, that there are many other helpful estimates connected with distortion triangles on surfaces. The construction of the geometric theory of quasiisometric mappings on surfaces in $\mathbb{R}^3$ is a certain primary task. Unconditionally, tests for mappings between surfaces to be quasiisometries are also valuable.

Acknowledgements

The authors are indepted to thank S.K. Godunov. His conversations and opinions have been stimulated our interest to this problem.

We would like to thank V.A. Klyachin for very useful remarks.

Bibliography

[1] S.K. Godunov, E.I. Romenskii, G.A. Chumakov, Construction of difference schemes in compliced domains by quasiconformal mappings, Computing problems in mathematical physics, *Proc. AN SSSR. Math. institute*; v. 18, Novosibirsk: Nauka, 1990, 75-84.

[2] V.M. Gordienko, On quadrangles on surfaces of constant curvature, Computing problems in mathematical physics, *Proc. RAN. Math. institute*; v. 22, Novosibirsk: Nauka, 1992, 124-133.

[3] G.A. Chumakov, Riemannian metric of harmonic parametrization of geodesic quadrangles on surfaces of constant curvature, Computing problems in mathematical physics, *Proc. RAN. Math. institute*; v. 22, Novosibirsk: Nauka, 1992, 133-151.

[4] L.V. Ahlfors, Lectures on quasiconformal mappings, - Van Nostrand Mathematical Studies, Toronto – New York – London, 1966.

[5] C.B. Morrey, On the solutions of quasi-linear elliptic partial differential equations, *Trans. Amer. Math. Soc.*, v. 43, 1938, 126-186.

[6] I.N. Vekua, *Generalized analytic functions*, Moscow: GIFML, 1959.

[7] L. Bers, Uniformization by Beltrami equation, *Comm. Pure Appl. Math.* v. 14, 1961, 215-228.

[8] P.P. Belinskii, General properties of quasiconformal mappings, "Nauka", Sibirskoe Otdelenie, Novosibirsk, 1974.

[9] O. Martio, V. Miklyukov, On existence and uniqueness of degenerate Beltrami equation, Reports of the Department of Mathematics, University of Helsinki, Preprint 347, 2003, 1-12,

[10] V.M. Miklyukov, Isothermal coordinates on surfaces with singularities, *Mathem. sbornik*, v. 195, n. 1, 2004, 69-88.

[11] M. Gromov, *Metric Structures for Riemannian and Non-Riemannian Spaces* , Boston-Basel-Berlin, Birkhäuser, 1999.

[12] H. Federer, *Geometric Measure Theory*, Springer-Verlag, Berlin, 1969.

[13] I.V. Zhuravlev, A.Yu. Igumnov, and V.M. Miklyukov, An implicit function theorem, Reports of the Depart. of Math., Preprint 346, February 2003, Univ. of Helsinki, 1-8 (submitted to The Rocky Mountain J. of Mathematics).

Chapter 4

GRID OPTIMIZATION AND ADAPTATION

Sergey A. Ivanenko and Boris N. Azarenok[1]
Dorodnicyn Computing Center of the Russian Academy of Sciences

A variational approach of two-dimensional grid generation, based on the principle of minimal energy density of a mapping, is presented. Cell shape control allows us to construct grids, being orthogonal near the boundary with prescribed spacing, as it can be done by a hyperbolic grid generator. At the same time the method is elliptic with immanent guarantee to produce grids free of folding. To adapt the mesh towards singularities in the solution of a flow problem, the energy density functional, written on the surface of graph of control function, is used. The problem of the boundary nodes redistribution is considered. A variational principle, allowing for determining classes of one-to-one mappings, is formulated. A consequence is that there exist equations describing all possible one-to-one mappings. A discrete analogue of this variational principle is formulated and proved. Method capabilities are illustrated in a number of examples. The energy density in the 3D case is introduced.

4.1 Introduction

In a one dimension space for grid generation, it was developed the equidistribution principle, cf. [25]. The idea is that the Jacobian of the mapping, multiplied by some positive weight function, is set equal to a positive constant

$$\omega x_\xi = const > 0\,, \quad \omega(\xi) > 0\,, \quad \xi \in (0,1)\,, \tag{4.1}$$

and the mapping $x(\xi)$ satisfies the boundary conditions

$$x(0) = 0\,, \quad x(1) = 1\,. \tag{4.2}$$

[1]This paper was planed by the both authors. After Sergey Ivanenko's death the second author wrote it on the basis of the material in hand.

Instead of (4.1) we may consider the minimization problem for the functional

$$F = \int_0^1 \omega x_\xi^2 d\xi \,, \tag{4.3}$$

on the class of the functions $x(\xi)$, satisfying the boundary conditions (4.2). Differentiating (4.1) with respect to ξ, we get the Euler-Lagrange equation for the functional (4.3)

$$(\omega x_\xi)_\xi = 0 \,. \tag{4.4}$$

In general all one-dimensional adaptive methods are reduced to the equidistribution principle. This is due to the function $x(\xi)$, being the solution of (4.4) with the boundary conditions (4.2), is the one-to-one mapping $\xi \to x$, since from (4.1) it follows that the Jacobian is positive. On the other hand, for any smooth one-to-one mapping $x^*(\xi)$ with a positive Jacobian we can define the weight function $w=const(x_\xi^*)^{-1}$ and then $x^*(\xi)$ will satisfy the equation (4.4). Therefore, both the variation formulation of minimizing the functional (4.3) and the Euler-Lagrange equation (4.4) include, as solutions, all smooth one-to-one mappings with a positive Jacobian. A mapping, being not one-to-one, can not satisfy the equation (4.4) with a positive weight function $\omega(\xi)$.

Simplicity and geometrical clearness of the equidistribution principle have simulated numerous attempts to extend it in the multidimensional case (cf. [10], Chapts. 1,34). An analogous reasoning with purpose to obtain a mapping of the unit square in plane (ξ, η) onto the domain Ω in plane (x, y) suggests to apply linear elliptic equations, where the coefficients are defined as that in the one-dimensional case, for instance, in the following way

$$(\omega_1 x_\xi)_\xi + (\omega_2 x_\eta)_\eta = 0 \,,$$
$$(\omega_1 y_\xi)_\xi + (\omega_2 y_\eta)_\eta = 0 \,.$$

However, the class of all solutions of these equations with positive ω_1 and ω_2 contains functions $x(\xi, \eta)$, $y(\xi, \eta)$ which cannot execute a one-to-one mapping of the unit square onto Ω. A similar result is observed to many other differential and variational formulations, employed at present time.

In this work we present a variational approach, based on the principle of the minimal energy density of a mapping [15, 16], being a generalization of the equidistribution principle in the 2D case. It is formulated in the both continuum and discrete case in Sections 4.10,4.11. A complete proof is executed only for the discrete mapping in Section 4.11.

This principle leads to a variational formulation of the elliptic grid generator. The main idea is the following. In every cell, it is introduced a local mapping of this cell onto another cell with a shape given in some manner or obtained using an algorithm by introducing additional information. In particular, as an objective shape one can use a shape of the corresponding grid cell, generated at the first stage. Thus, for every cell the parameters, defining the shape which the cell should tend to, are given. For instance, in the triangle there are two parameters: two sides length ratio and angle between them. The shape measure is a quantity

inverse to the energy density. Maximum of such a shape measure corresponds to the minimum of the energy density and it is attained if and only if the shape of the element is the same as that in the corresponding objective shape. We formulate a local objective function (i.e. energy density) and its minimization allows us to obtain a cell shape being homothetic to a given one. A global objective function (or energy density) is obtained by summing the local shapes over all grid cells. Grid correction is defined by minimizing the global energy density. This user-defined grid quality measure is implemented in the optimization-based method of grid generation and improvement. The approach can be considered as extension of methods, based on constructing a mapping being a composition of the algebraic mapping and inverse to the harmonic one (see [8, 23] and chapter 5), giving a guarantee that the composite mapping is invertible in the continuous approach. The present approach gives such a guarantee at the discrete level.

The energy density of grid deformation is a discrete functional with an infinite barrier at the boundary of the set of unfolded grids and is an extension of the barrier function, introduced in [11] (see also [5]). Barrier functions for grid generation and optimization have been also considered in [24], [19]. The barrier property is very important in problems with moving boundaries and in moving adaptive grid technology [1, 2, 3] because such methods ensure generation of unfolded grids at every time step. The direct control of the cell shape is used to provide mesh orthogonality with a prescribed cell width near the boundary. Adaptation to the solution of the host equations can be realized as in the adaptive-harmonic grid generator [14]. In the 3D case the well known tetrahedron shape measure, such as the mean ratio, can be obtained from the energy density of the deformation of the equilateral tetrahedron. The condition number of the Jacobian matrix, measured in the Frobenius norm [18], can be obtained as well.

4.2 Energy Density of a Linear Transformation in 2D

The energy density of a mapping is introduced for the linear transformations of a plane. This quantity characterizes the underlying linear mapping and is an orthogonal invariant, i.e., it does not change under orthogonal transformations. Consider a linear transformation of the coordinates

$$x = c_{11}\xi + c_{12}\eta + c_1\,, \quad y = c_{21}\xi + c_{22}\eta + c_2\,. \tag{4.5}$$

This transformation is defined uniquely by the mapping of three points, not located along a straight line, in plane (ξ, η) into three points, not located along a straight line, in plane (x, y), see Fig. 4.1. The matrix $c=(c_{ij})$ is the Jacobi matrix of the mapping (4.5). If the Jacobian of the mapping, equal to the determinant of the Jacobi matrix, is not equal to zero, i.e.

$$J = c_{11}c_{22} - c_{21}c_{12} \neq 0\,,$$

then this mapping is invertible (nondegenerate).

Consider in what way the distance squared between two points changes under the transformation (4.5). First, the coordinates are shifted so that c_1=0 and c_2=0. Then, under

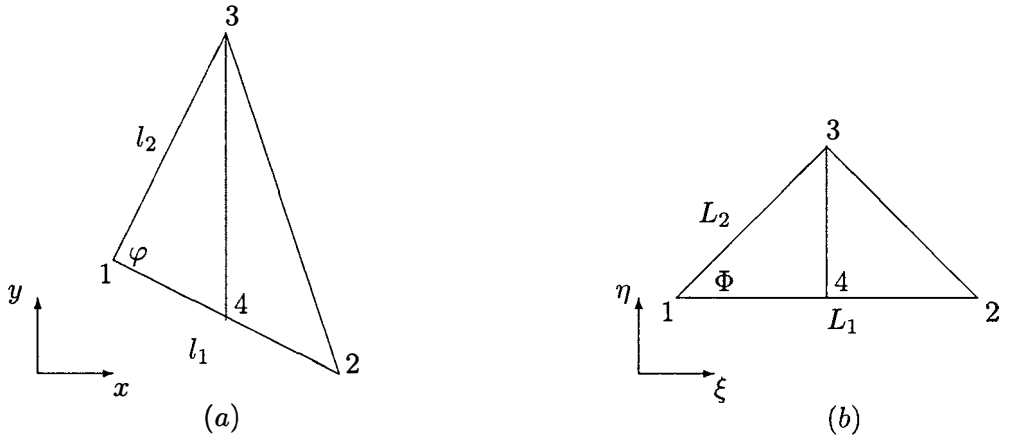


Figure 4.1: Correspondence of vertices under the mapping of a triangle in plane (ξ, η) into a triangle in plane (x, y).

the invertible mapping, the radius vector $r=(x, y)$ length squared can be represented as a positive definite quadratic form

$$x^2 + y^2 = g_{11}\xi^2 + 2g_{12}\xi\eta + g_{22}\eta^2 > 0\,. \tag{4.6}$$

The elements of the symmetric and positive definite matrix $g=(g_{ij})=c^\top c$ ($c^\top$ is a transposed matrix) are given by formulas

$$g_{11} = c_{11}^2 + c_{21}^2\,, \quad g_{12} = g_{21} = c_{11}c_{12} + c_{21}c_{22}\,, \quad g_{22} = c_{12}^2 + c_{22}^2\,.$$

One can readily check that

$$g_{11}g_{22} - g_{12}^2 = J^2\,. \tag{4.7}$$

In analytical geometry, there is a theorem stating that by rotating the ξ, η-coordinates through some angle α

$$\xi = \xi' \cos\alpha - \eta' \sin\alpha\,, \quad \eta = \xi' \sin\alpha + \eta' \cos\alpha\,,$$

the quadratic form (4.6) can be reduced to the canonical form

$$x^2 + y^2 = g'_{11}\xi'^2 + g'_{22}\eta'^2\,, \tag{4.8}$$

where the coefficients g'_{11} and g'_{22} are the roots of the quadratic equation

$$\lambda^2 - (g_{11} + g_{22})\lambda + g_{11}g_{22} - g_{12}^2 = 0\,. \tag{4.9}$$

The equation (4.9) is a characteristic equation of the quadratic form (4.6). Its roots are always real, since the discriminant is positive

$$(g_{11} + g_{22})^2 - 4(g_{11}g_{22} - g_{12}^2) = (g_{11} - g_{22})^2 + 4g_{12}^2 \geq 0\,. \tag{4.10}$$

Since the matrix $g=(g_{ij})$ is positive definite, the roots are positive: $\lambda_1, \lambda_2 > 0$.

Let λ_1, λ_2 be two solutions of the equation (4.9). The coefficients of this equation

$$tr(g) = (g_{11} + g_{22}) = \lambda_1 + \lambda_2 , \quad det(g) = g_{11}g_{22} - g_{12}^2 = J^2 = \lambda_1\lambda_2 ,$$

are said to be the orthogonal invariants of the quadratic form (4.6), since they do not change under the orthogonal transformations of plane (ξ, η), i.e., under the superposition of a rotation, reflection, and shift.

The discriminant of (4.9) depends only on invariants and equals zero if and only if

$$g_{11} = g_{22} , \quad g_{12} = 0 . \tag{4.11}$$

If the conditions (4.11) are satisfied, the linear transformation (4.5) is conformal, i.e., it is a superposition of the orthogonal transformation and dilatation with the same coefficient along the both axes. From (4.10) it follows that

$$\frac{1}{2}\frac{g_{11} + g_{22}}{\sqrt{g_{11}g_{22} - g_{12}^2}} \geq 1 , \tag{4.12}$$

moreover it turns into the equality if and only if the conditions (4.11) are satisfied.

Accordingly to the theorem on reducing the quadratic form (4.6) to the canonical form (4.8) it follows that any invertible linear mapping is a superposition of an orthogonal transformation and dilatation along the ξ', η'-axes with the coefficients $\sqrt{\lambda_1}$ and $\sqrt{\lambda_2}$, respectively. Under this mapping the unit square is transformed into the rectangle with the sides $\sqrt{\lambda_1}$ and $\sqrt{\lambda_2}$.

In the physical sense the expression in the left-hand side of (4.12) is the (internal) energy density. Actually, the numerator in (4.12), equal $\lambda_1+\lambda_2$, is the sum of the lengths squared of the rectangle sides, i.e., proportional to the elastic deformation energy of two springs with zero length oriented along the ξ', η'-axes. The denominator is the rectangle area $\sqrt{\lambda_1\lambda_2}$. Thus, this expression (in the left-hand side of (4.12)) is the orthogonal invariant of the mapping (4.5) and can be written in the form

$$e(x,y)(\xi,\eta) = \frac{1}{2}\frac{tr(g)}{\sqrt{det(g)}} = \frac{1}{2}\frac{\lambda_1 + \lambda_2}{\sqrt{\lambda_1\lambda_2}} = \frac{1}{2}\left(\sqrt{\frac{\lambda_1}{\lambda_2}} + \sqrt{\frac{\lambda_2}{\lambda_1}}\right) . \tag{4.13}$$

We refer the function $e(x, y)$ to as the (internal) energy density of the mapping $(\xi, \eta) \rightarrow (x, y)$. Its minimum is achieved under the mapping, satisfying the condition $\lambda_1=\lambda_2$, which is equivalent to (4.11). Note that in the 2D case the energy density is also a conformal invariant, since it does not change under a linear conformal transformation $(\xi, \eta) \rightarrow (\xi', \eta')$.

We extend the formula (4.13) to the case of a linear mapping between two planes, defined by different linear transformations of the same plane (ξ, η). Consider a mapping $(X, Y) \rightarrow (x, y)$, where $x(\xi, \eta), y(\xi, \eta)$ and $X(\xi, \eta), Y(\xi, \eta)$ are two linear transformations with the matrices c and C, respectively. The vector length squared x^2+y^2 and X^2+Y^2 is calculated using positive definite quadratic forms with matrices

$$g = (g_{ij}) = c^\top c , \quad \text{and} \quad G = (G_{ij}) = C^\top C .$$

The quadratic form, corresponding to the mapping $(X,Y)\rightarrow(x,y)$, is defined by the matrix $(C^{-1})^{\top}c^{\top}cC^{-1}$. The characteristic equation is of the form

$$det[(C^{-1})^{\top}c^{\top}cC^{-1}-\lambda I]=det(C)^{-2}det(c^{\top}c-\lambda C^{\top}C)=0\,,$$

where I is the identity matrix. Hence, subject to the matrix C is invertible, the characteristic equation for the matrix $(C^{-1})^{\top}c^{\top}cC^{-1}$ possesses the same solutions λ_1, λ_2 as the characteristic equation of the pair of quadratic forms

$$det(g-\lambda G)=det(G)det(G^{-1}g-\lambda I)=0\,. \tag{4.14}$$

The orthogonal invariants of the mapping $(X,Y)\rightarrow(x,y)$ depend upon the eigenvalues $\lambda_{1,2}$ of the matrix $(G^{-1}g)$

$$\lambda_1+\lambda_2=tr(G^{-1}g)=g_{11}G^{11}+2g_{12}G^{12}+g_{22}G^{22}=\frac{g_{11}G_{22}-2g_{12}G_{12}+g_{22}G_{11}}{det(G)}\,,$$

$$\lambda_1\lambda_2=det(G^{-1}g)=\frac{det(g)}{det(G)}\,.$$

Here G^{ij} are the elements of the matrix G^{-1} inverse to G. Substituting them in (4.13), we obtain

$$e(x,y)(X,Y)=\frac{1}{2}\frac{tr(G^{-1}g)}{\sqrt{det(G^{-1}g)}}=\frac{1}{2}\frac{g_{11}G_{22}-2g_{12}G_{12}+g_{22}G_{11}}{\sqrt{det(g)}\sqrt{det(G)}}\,. \tag{4.15}$$

Note that the formula (4.15) is symmetric with respect to g and G, and, consequently,

$$e(x,y)(X,Y)=e(X,Y)(x,y)\,.$$

We formulate the properties of the energy density of the linear mapping.

1. The energy density is an orthogonal invariant, which remains unchanged under orthogonal transformations of planes (X,Y) and (x,y). Moreover, it is also a conformal invariant, since $e(x,y)(X,Y)=e(\alpha x,\alpha y)(\beta X,\beta Y)$ with any $\alpha,\beta>0$. In the 2D case the energy density is identical to the direct and inverse linear transformations, i.e., $e(x,y)(X,Y)=e(X,Y)(x,y)$.

2. If a mapping $x(X,Y)$, $y(X,Y)$ becomes singular, i.e., $J\rightarrow 0$, and if at least one of the quantities λ_1 or λ_2 does not tend to 0, then the energy density $e\rightarrow\infty$. If the energy density does not tend to ∞ while $J\rightarrow 0$, the transformation degenerates into a point: $(X,Y)\rightarrow(0,0)$, and in this case $\lambda_1\rightarrow 0$ and $\lambda_2\rightarrow 0$ simultaneously.

3. Optimality principle. Minimum of the energy density $e(x,y)(X,Y)$ of a linear transformation $(X,Y)\rightarrow(x,y)$ is equal to 1 and is attained if and only if $\lambda_1=\lambda_2$, i.e., provided that the transformation is conformal.

4.3 Shape Measure for Triangular and Quadrilateral Cells

Consider a linear transformation of a given triangle in plane (ξ, η) into a triangle in plane (x, y). The sides and angles L_1, L_2, Φ and l_1, l_2, φ of the triangles are shown in Fig. 4.1. A correspondence between the vertices of the triangles defines a unique linear transformation (4.5). The Jacobian of the transformation is equal to the areas ratio

$$J = det(c) = c_{11}c_{22} - c_{12}c_{21} = \frac{l_1 l_2 \sin\varphi}{L_1 L_2 \sin\Phi} > 0\,. \tag{4.16}$$

Note that the conformality condition is the similitude conditions for the triangles

$$\frac{l_1}{l_2} = \frac{L_1}{L_2}\,, \quad \varphi = \Phi\,. \tag{4.17}$$

In the 1D case, the shape measure has no sense, since any two segments are homothetic.

We aim to construct a shape measure $e^t(l_1, l_2, \varphi, L_1, L_2, \Phi)$ so that its minimum provides the conditions (4.17) to be fulfilled. Clearly that it should be "dimensionless", i.e., not change at substitution

$$(l_1, l_2, L_1, L_2) \rightarrow (\beta l_1, \beta l_2, \beta L_1, \beta L_2)\,,$$

where $\beta \neq 0$ is a real number. The shape measure is a characteristic of a linear transformation, induced by a mapping of a given triangle in plane (ξ, η) into a triangle in plane (x, y). Hence, it should not change under an orthogonal transformation of plane (ξ, η), and, therefore, depend only on the orthogonal invariants of the quadratic form (4.17). For instance, the function

$$F = \left(\frac{l_1}{l_2} - \frac{L_1}{L_2}\right)^2 + (\varphi - \Phi)^2\,, \tag{4.18}$$

is not invariant under the orthogonal transformations, and, consequently, can not be used as a shape measure. We demonstrate it in the case depicted in Fig. 4.2, where it is presented the linear transformation of the figure (grid cell), consisting of two triangles: equilateral and right-angled isosceles. The linear transformation is a dilatation along the ξ-axis: $x=\beta\xi$, $y=\eta$. Let l_{ij} be a length of the side joining the points with numbers i,j, and φ_{ijk} be the angle between the sides l_{ij} and l_{jk}. Similarly, we introduce L_{ij} and Φ_{ijk}. Note that the shape measure for the equilateral triangle should not depend upon numeration, i.e., the following equality is to be hold

$$e^t(l_{23}, l_{24}, \varphi_{324}, 1, 1, \frac{\pi}{3}) = e^t(l_{34}, l_{23}, \varphi_{432}, 1, 1, \frac{\pi}{3}) = e^t(l_{24}, l_{34}, \varphi_{243}, 1, 1, \frac{\pi}{3})\,.$$

This equality does not hold for the function (4.18), since it is not invariant under the orthogonal transformations. Actually, the shape measures of the right-angled and equilateral triangles are known and used in smoothing irregular grids (cf. [13])

$$e^t(l_{12}, l_{14}, \varphi_{214}, 1, 1, \frac{\pi}{2}) = \frac{l_{12}^2 + l_{14}^2}{2l_{12}l_{14}\sin\varphi_{214}},$$

$$e^t(l_{23}, l_{24}, \varphi_{324}, 1, 1, \frac{\pi}{3}) = \frac{l_{23}^2 + l_{34}^2 + l_{24}^2}{4l_{23}l_{24}\sin\varphi_{324}\sin(\pi/3)}.$$

It may be of interest to note that in this example the above functions take the same value for the both triangles, equal to the energy density of the mapping.

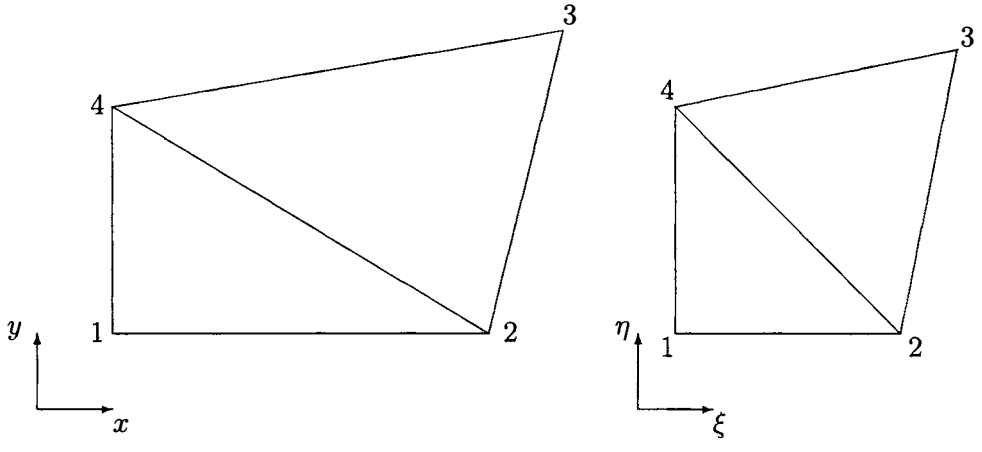


Figure 4.2: Dilatation of figure consisting of two triangles.

So as to derive the shape measure in a general case we rewrite (4.13) in terms of the sides length and angles of the triangle. Since the energy density (4.13) does not change under a coordinate system rotation in plane (ξ, η), we assume that the side L_1 is parallel to the ξ-axis. Draw a perpendicular from vertex 3 to this side. Point 4 is intersection of them, see Fig. 4.1 *b*. Due to linearity of the mapping it follows that the image of this point divides the side l_1 with the same ratio as that for the side L_1 as for the side L_1. It follows that

$$g_{11} = \left(\frac{\partial x}{\partial \xi}\right)^2 + \left(\frac{\partial y}{\partial \xi}\right)^2 = l_1^2\left(\frac{L_2\cos\Phi}{L_1}\right)^2 (L_2\cos\Phi)^{-2} = \frac{l_1^2}{L_1^2},$$

$$g_{22} = \left(\frac{\partial x}{\partial \eta}\right)^2 + \left(\frac{\partial y}{\partial \eta}\right)^2$$

$$= \left[l_1^2\left(\frac{L_2\cos\Phi}{L_1}\right)^2 + l_2^2 - 2l_1l_2\left(\frac{L_2\cos\Phi}{L_1}\right)\cos\varphi\right](L_2\sin\Phi)^{-2}.$$

Recalling (4.16) the function (4.13) takes the form

$$e^t = \frac{1}{2}\frac{l_1^2 L_2^2 + l_2^2 L_1^2 - 2 l_1 l_2 L_1 L_2 \cos\varphi \cos\Phi}{l_1 l_2 \sin\varphi L_1 L_2 \sin\Phi} . \tag{4.19}$$

This function is said to be a triangle shape measure. Note, when deriving (4.19), we assume that $l_1, l_2, L_1, L_2 \neq 0$, $\varphi, \Phi \neq 0$, and $\varphi, \Phi \neq \pi$. The formula (4.19) can be obtained from (4.15) if to define the mappings with $g_{11}=l_1^2$, $g_{22}=l_2^2$, $g_{12}=l_1 l_2 \cos\varphi$, $G_{11}=L_1^2$, $G_{22}=L_2^2$, $G_{12}=L_1 L_2 \cos\Phi$. The shape measure can be also interpreted as the energy density.

To check whether the conditions (4.17) are satisfied, we introduce notations from [9] (see also chapter 2)

$$l_1^2 = \rho \exp(a) , \quad L_1^2 = R \exp(A) , \quad l_2^2 = \rho \exp(-a), \quad L_2^2 = R \exp(-A) .$$

With these notations the expression (4.19) takes the form independent on ρ and R

$$e^t = 2\frac{sh^2\frac{\beta - B}{2} + \sin^2\frac{\varphi - \Phi}{2}}{\sin\varphi \sin\Phi} + 1 . \tag{4.20}$$

It is clear that the minimum of (4.20) is attained if $\beta=B$ and $\varphi=\Phi$, i.e., the conditions (4.17) are satisfied. Note that $1/e^t$ is a triangular shape measure according to the definition given in [7] and applied for triangles. Note that in [9] (see also chapter 2) it was suggested a functional with an integrand equal to the numerator in (4.20). Below we will show that it is the denominator $\sin\varphi \sin\Phi$ that provides the "barrier property" guaranteeing unfolded grid generation. The barrier property satisfies the conditions of invertibility for the global mapping.

From (4.20) and property 3 of the energy density (see Section 4.2) it follows

Property 1. *If two triangle vertices are fixed in plane (x, y), then the function e^t possesses a unique minimum being attainable if and only if $a=A$ and $\varphi=\Phi$, i.e., when the conditions (4.17) are satisfied.*

Consider the shape measure (4.19) when the mapping degenerates, i.e., at $J \to 0$. We assume that the triangle in plane (ξ, η) does not degenerate.

Since $J=l_1 l_2 \sin\varphi/(L_1 L_2 \sin\Phi)$, there are possible two cases:

a) The triangle in plane (x, y) degenerates into a segment. It is possible subject to $\sin\varphi \to 0$ or only one of values l_1, l_2 tends to zero.

b) The triangle degenerates into a point and then $l_1, l_2 \to 0$ simultaneously.

From property 2 of the energy density (see Section 4.2) it follows a local barrier property of the triangle shape measure.

Property 2. *If the triangle in plane (x, y) degenerates, i.e., $J \to 0$, and at least one of the values l_1, l_2 does not tend to zero, then $e^t \to +\infty$.*

Note that if $J \to 0$ at $J<0$ and at least one of the values l_1, l_2 does not tend to zero, then $e^t \to -\infty$.

The sign of the Jacobian coincides with the sign of the term $\sin\varphi \sin\Phi$ in the denominator of (4.20). Hence, the Jacobian can be positive ($J>0$) in two cases:
a) $0 < \varphi, \Phi < \pi$, and b) $-\pi < \varphi, \Phi < 0$.

The energy density of a quadrilateral element can be expressed as

$$e^q = \frac{1}{4}(e_1^t + e_2^t + e_3^t + e_4^t)\,, \tag{4.21}$$

where, for example, e_1^t is the function (4.20) for the triangle corresponding to vertex 1 of the quadrilateral element (see Fig. 4.2)

$$e_1^t = \frac{1}{2}\frac{l_{12}^2 L_{14}^2 + l_{14}^2 L_{12}^2 - 2l_{12}l_{14}L_{12}L_{14}\cos\varphi_{214}\cos\Phi_{214}}{l_{12}l_{14}\sin\varphi_{214}L_{12}L_{14}\sin\Phi_{214}}\,. \tag{4.22}$$

The quantities e_2^t, e_3^t, e_4^t for the remain vertices are calculated similarly.

If to join the quadrilateral vertices with its center (centroid), we can define another quadrilateral shape measure, based on dividing the quadrilateral into four triangles. It is possible to define another shape measures, as a sum of triangle shape measures at some partitioning of the quadrilateral.

The minimum of e^q, equal to 1, is attained for a quadrilateral, being homothetic to a given one. The properties of e^q outflow from that of e^t, set forth above.

4.4 Energy Density of Grid Deformation

Consider an unstructured grid, formed by quadrilateral elements. A correspondence between local and global enumerations is defined as

$$n = n(N,k),\ n = 1,2,\ldots,N_n,\ N = 1,2,\ldots,N_e,\ k = 0,1,2,3,$$

where n is the global node number, N_n is the total number of grid nodes, N is an element number, N_e is the number of elements, k is the local node number in the element.

The energy density of the grid deformation e^h is obtained by averaging over all grid cells

$$e^h = \frac{1}{N_e}\sum_{N=1}^{N_e}\left[e_k^q\right]_N = \frac{1}{N_e}\sum_{N=1}^{N_e}\sum_{k=0}^{3}\frac{1}{4}\left[e_k^t\right]_N\,, \tag{4.23}$$

where e_k^t are calculated via (4.22).

There is an interesting property of the function (4.23). Consider a hypothetical case when the grid structure and objective shape for every cell, identical to the shapes of an existing grid, are known. If we specify positions of two arbitrary neighboring nodes of the grid, then the positions of all other nodes can be obtained sequentially from the condition e^h=1. Similarly, if the positions of the internal boundary nodes are specified, then all other nodes can be obtained layer by layer as that in a hyperbolic method, meantime the present method is elliptic.

Obviously that any grid perturbation, when at least two nodes are fixed, leads e^h to increase, if the shape measure in the cells corresponds to the one in the initial grid.

The main property of the energy density of grid deformation, resulting from local properties of the shape measure (see Section 4.3), can be formulated as

Theorem. *Every unfolded mesh, constructed in the domain by defined coordinates of the boundary nodes, generates a class of grids with the same structure and with the same values of* $\sin\varphi$ *in the corresponding internal cell angles. Thus, the set of all grids with identical structure and coordinates of the boundary nodes can be divided into classes, separated by the infinite barrier of functions* e^h*, where the cell objective shape corresponds to a cell of a grid belonging to this class.*

4.5 Specification of Objective Shapes near the Boundary

By virtue of the above theorem on the barrier property of the energy density, it follows that a deformation of a grid, belonging to some class (e.g. to a class of convex grids, i.e., grids consisting of convex quadrilaterals), can be executed by defining a cell objective shape so that minimization of e^h produces a mesh from this class. Meanwhile, a grid with the shape in every cell equal identically to the objective shape may not exist. As an example we specify a harmonic mesh, whose objective shape in every cell is a square.

To achieve grid orthogonality near the boundary we should specify the corresponding cell objective shape. First, an "orthogonal" direction is to be specified in every boundary node. With this purpose one determines an intersection point of two lines, parallel to the boundary segments $(i-1,i)$ and $(i,i+1)$ (see Fig. 4.3 *a*). The vector $\boldsymbol{r}_1$, orthogonal to the interval $(i-1,i)$, is given by

$$x_1 = d_0(-y_i + y_{i-1})/\sqrt{(x_{i-1}-x_i)^2 + (y_{i-1}-y_i)^2}\,,$$
$$y_1 = d_0(x_i - x_{i-1})/\sqrt{(x_{i-1}-x_i)^2 + (y_{i-1}-y_i)^2}\,.$$

Here d_0 is the width of the boundary cell layer. Coordinates of $\boldsymbol{r}_2$ are computed similarly. The vector $\boldsymbol{r}$ is determined from the conditions

$$\boldsymbol{r}_1(\boldsymbol{r}-\boldsymbol{r}_1) = 0\,, \quad \boldsymbol{r}_2(\boldsymbol{r}-\boldsymbol{r}_2) = 0.$$

Solving this system of linear equations, we obtain

$$x = \frac{y_2|\boldsymbol{r}_1|^2 - y_1|\boldsymbol{r}_2|^2}{x_1y_2 - x_2y_1}\,, \quad y = -\frac{x_2|\boldsymbol{r}_1|^2 - x_1|\boldsymbol{r}_2|^2}{x_1y_2 - x_2y_1}\,.$$

The cell shapes in the next layer can be obtained similarly. To provide a smooth transformation of the cell shape from the boundary cell layer to the internal cell layer, a linear interpolation of the objective shape is used along the corresponding curvilinear coordinate lines. The shape of two cells at the ends of the interpolation layer is known. These cells are scaled and rotated in such a way that their bottom sides coincide and interpolation is performed along lines, connecting two top vertices of the cells (see Fig. 4.3 *b*)

$$\tilde{\boldsymbol{r}}_3(j) = (1-s)\boldsymbol{r}_3 + s\boldsymbol{r}_{3'}\,, \quad \tilde{\boldsymbol{r}}_4(j) = (1-s)\boldsymbol{r}_4 + s\boldsymbol{r}_{4'}\,.$$

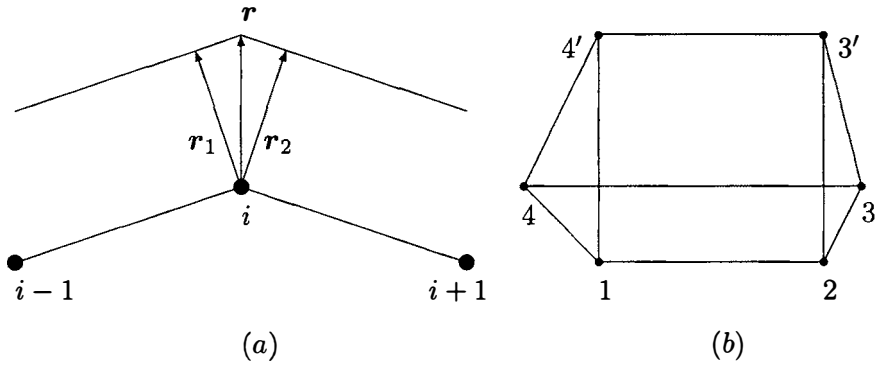


Figure 4.3:

The interpolation coefficient s is determined by a cell location in the interpolation layer

$$s = (j-1)/(N_j - 1)\,,$$

where N_j is the total number of cells in the interpolation layer along the j-direction, j is the cell number in the layer. If necessary a polynomial (quadratic, etc.) interpolation can be used here.

Coordinates of the objective cell vertices (X_k, Y_k), k=1, 2, 3, 4 are specified as follows

$$(X_1, Y_1) = (0,0),\ (X_2, Y_2) = (1,0),\ (X_3, Y_3) = (\tilde{x}_3, \tilde{y}_3),\ (X_4, Y_4) = (\tilde{x}_4, \tilde{y}_4).$$

Note that many other algorithms can be formulated to specify the objective shape. The conception "orthogonality near the boundary" can be introduced by the user himself. Besides, the objective cell can be a non-planar quadrilateral. It may be of use to apply an arbitrary quadrilateral in space $\mathbb{R}^3$. For instance, if we want that all four triangles, into which the objective cell is divided by every of two diagonals, to be equilateral triangles, then the regular tetrahedron should be used as the objective cell.

4.6 Energy Density Functional

Consider a mapping $(M, G) \to (m, g)$ of two-dimensional manifolds M and m with metrics G and g, respectively. We work with the same local system of coordinate ξ, η to the both manifolds. The problem of minimizing the energy density functional is formulated as

$$I = \frac{1}{2}\int\int \frac{tr(G^{-1}g)}{\sqrt{det(G^{-1}g)}}\,d\xi d\eta = \frac{1}{2}\int\int \frac{tr(g^{-1}G)}{\sqrt{det(g^{-1}G)}}\,d\xi d\eta \qquad (4.24)$$
$$= \frac{1}{2}\int\int \frac{g_{11}G_{22} - 2g_{12}G_{12} + g_{22}G_{11}}{\sqrt{det(g)}\sqrt{det(G)}}\,d\xi d\eta\,.$$

Note that the harmonic functional is

$$\tilde{I} = \frac{1}{2}\int\int \frac{g_{11}G_{22} - 2g_{12}G_{12} + g_{22}G_{11}}{\sqrt{det(g)}}\,d\xi d\eta\,.$$

Consequently, the mapping, delivering a minimum for the functional I, will be harmonic if instead of the metric G_{ij} the "scaled" metric

$$\tilde{G}_{ij} = G_{ij}/\sqrt{det(G)}$$

is used. In this case $det(\tilde{G})$=1. The Euler-Lagrange equations for the functional (4.24) are elliptic and, besides, subject to a non-positive curvature of M and convexity of its boundary (with respect to the metric $\tilde{G}_{ij}$) the theorem on existence and uniqueness of the one-to-one harmonic mapping $(m, g) \to (M, \tilde{G})$ is correct (cf. [22]).

When adapting to a control (in other words monitor) function $f(x, y)$ the metric g is defined as follows (for details see [3, 5])

$$g_{11} = x_\xi^2 + y_\xi^2 + (f_x x_\xi + f_y y_\xi)^2 \,, \quad g_{22} = x_\eta^2 + y_\eta^2 + (f_x x_\eta + f_y y_\eta)^2 \,,$$

$$g_{12} = x_\xi x_\eta + y_\xi y_\eta + (f_x x_\xi + f_y y_\xi)(f_x x_\eta + f_y y_\eta) \,.$$

Substituting them in (4.24) we have

$$I = \frac{1}{2} \iint \frac{D_1(1 + f_x^2) + D_2(1 + f_y^2) + 2D_3 f_x f_y}{(x_\xi y_\eta - x_\eta y_\xi)\sqrt{1 + f_x^2 + f_y^2}\sqrt{G_{11}G_{22} - G_{12}^2}} \, d\xi d\eta \,, \tag{4.25}$$

where

$$D_1 = x_\xi^2 G_{22} - 2x_\xi x_\eta G_{12} + x_\eta^2 G_{11} \,, \quad D_2 = y_\xi^2 G_{22} - 2y_\xi y_\eta G_{12} + y_\eta^2 G_{11} \,,$$

$$D_3 = x_\xi y_\xi G_{22} - (x_\xi y_\eta + x_\eta y_\xi) G_{12} + x_\eta y_\eta G_{11} \,.$$

The minimization problem for the functional (4.25) is equivalent to the problem of constructing such a curvilinear coordinate system (i.e. structured grid) on the surface $z=f(x, y)$ that the averaged energy density of the mapping $(X, Y) \to (x, y, f(x, y))$ is minimized. The mapping $(\xi, \eta) \to (X, Y)$ is assumed to be known, and the metric tensor components G_{ij} are calculated as

$$G_{11} = X_\xi^2 + Y_\xi^2 \,, \quad G_{12} = X_\xi X_\eta + Y_\xi Y_\eta \,, \quad G_{22} = X_\eta^2 + Y_\eta^2 \,. \tag{4.26}$$

In the case of an unstructured grid, the problem is posed for the energy density of the mapping of a target cell in plane (X, Y) onto the corresponding cell on the surface. Finally, a minimization problem is posed to the sum of functionals written for every cell, i.e., for the averaged energy density.

To control the number of cells contained in a subdomain of high gradient of f, it was previously proposed to replace the function $f(x, y)$ (cf. [10], Chapt. 8) by the function $c_a f(x, y)$, where c_a is the coefficient of adaptation. The function $f(x, y)$ is scaled so that the difference of the maximal and minimal values of f equals to the diagonal of the rectangle circumscribed about the domain in plane (x, y)

$$f_{max} - f_{min} = \sqrt{(x_{max} - x_{min})^2 + (y_{max} - y_{min})^2} \,.$$

The relative number of grid points, contained in a steep-gradient layer, increases with rise of the coefficient of adaptation c_a, whose value is usually set within the interval $(0.05, 0.5)$. In the case of an elliptic problem, the relative number of grid points, contained in the layer, is approximately equal to $c_a/(c_a+1)$ [13], i.e., about one-third of the points belongs to the steep-gradient layer when $c_a=0.5$.

Practical computations show that when a grid is adapted to a smooth function, an increase in c_a makes the resulting grid "sensitive" to a larger domain of a sufficiently steep gradient. Furthermore, as $c_a \to \infty$ the grid, adapted to a continuously differentiable function f of one variable, is optimal in the C norm in the sense that the C-norm error of piecewise constant interpolation is minimized. Indeed, with f replaced by $c_a f(x)$, the one-dimensional functional (with $G_{ij}=\delta_{ij}$) has the form

$$I = \int \frac{1}{x_\xi \sqrt{1 + c_a^2 f_x^2}} d\xi ,$$

and the corresponding Euler-Lagrange equation, which determines the asymptotic behavior of the Jacobian of the adaptive harmonic grid, can be written as

$$x_\xi = const(1/c_a^2 + f_x^2)^{-1/2} .$$

At the same time, a grid, being optimal in the C norm under a piecewise constant interpolation, is characterized by a Jacobian asymptotically expressed as

$$x_\xi = const|f_x|^{-1} .$$

Therefore, the adaptive harmonic grid is optimal in the C norm as $c_a \to \infty$.

When an adaptive grid is to be generated for a vector-valued control function with components $f_i(x, y)$, rather than for a scalar $f(x, y)$, one can write the functional I in the following generalized form:

$$I = \frac{1}{2} \int\int \frac{\alpha \tilde{g}_{11} + 2\beta \tilde{g}_{12} + \gamma \tilde{g}_{22}}{(x_\xi y_\eta - x_\eta y_\xi)\sqrt{\tilde{g}_{11}\tilde{g}_{22} - \tilde{g}_{12}^2}\sqrt{G_{11}G_{22} - G_{12}^2}} d\xi d\eta ,$$

where

$$\tilde{g}_{11} = 1 + \sum_i c_i^2 (f_i)_x^2 , \quad \tilde{g}_{12} = \sum_i c_i^2 (f_i)_x (f_i)_y , \quad \tilde{g}_{22} = 1 + \sum_i c_i^2 (f_i)_y^2 .$$

The minimization problem for this functional is equivalent to the problem of constructing such a coordinate system on the surface, described by the vector function

$$\boldsymbol{r}(x, y) = (c_1 f_1, c_2 f_2 ...)(x, y)$$

that the averaged energy density for the mapping $(X, Y) \to (x, y, r(x, y))$ is minimized. As a result one obtains a surface parametrization in the form $(x, y, c_1 f_1, c_2 f_2 ...)(\xi, \eta)$, which corresponds to a structured grid. In the case of an unstructured grid, the problem is posed for the energy density of the mapping of a given cell in plane (X, Y) onto the corresponding surface cell. When a problem of computational fluid dynamics is to be solved, $f_i(x, y)$ denote either the flow parameters u, v, p, ρ or their functions, e.g. $|V| = \sqrt{u^2 + v^2}$ (cf. [1, 2]).

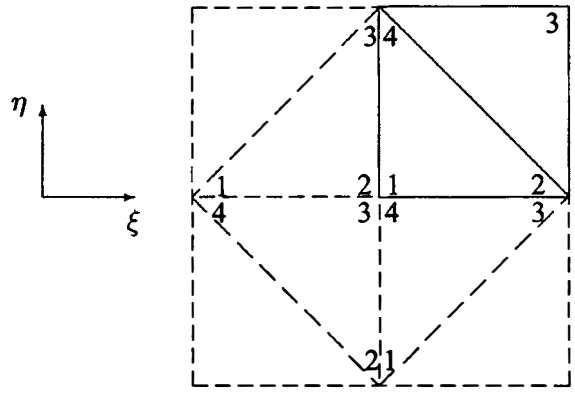


Figure 4.4: The cell in plane (ξ, η) is a unit square. Solid lines indicate the underlying cell, dash lines show the adjacent cells. The underlying cell is split into two triangles by the diagonal 24 (and analogously by the diagonal 13).

4.7 Discretization

A finite-difference analogue of the functional I can be derived as follows. A square cell (with the side equal to 1) in plane (ξ, η) is split into triangles by the diagonals 24 and 13, see Fig. 4.4. The mapping of the square onto a quadrilateral cell in plane (x, y) is approximated by the half-sum of two functions, each is a linear function in the corresponding triangle. Denote the half-sum of these functions by $x^h(\xi, \eta)$ and $y^h(\xi, \eta)$. They define a piecewise linear mapping of the parametric square onto a quadrilateral cell. Unlike a bilinear mapping, the inverse mapping of the quadrilateral onto the parametric square has derivatives, being discontinuous across the diagonals. Furthermore, the mapping of the square onto the quadrilateral cell in plane (X, Y) with vertices $X_1, Y_1, X_2, Y_2, X_3, Y_3, X_4, Y_4$ is also approximated by the half-sum of two functions, being linear in the corresponding triangle. All derivatives in the integrand of I are readily calculated, and the integral over the square cell in plane (ξ, η) is approximated by the half-sum of the integrals of piecewise linear approximations for the underlying functions. As a result one obtains an approximation of the integral (4.25) in the form

$$I^h = \frac{1}{2} \sum_{N=1}^{N_e} \sum_{k=1}^{4} \frac{1}{4} [F_k]_N \ , \tag{4.27}$$

where N_e is the number of grid cells, k specifies the vertex number in the cell, see Fig. 4.4, and the integrand is

$$F_k = \frac{\alpha D_1 + \beta D_2 + \gamma D_3}{J} \ . \tag{4.28}$$

For brevity we omit the subscript k in the right-hand side of (4.28). These quantities are defined as

$$\alpha = \frac{1+f_x^2}{B\sqrt{1+f_x^2+f_y^2}}\,,\quad \beta = \frac{1+f_y^2}{B\sqrt{1+f_x^2+f_y^2}}\,,\quad \gamma = \frac{2f_xf_y}{B\sqrt{1+f_x^2+f_y^2}}\,,$$

$$D_1 = (x_{k+1}-x_k)^2G_{22} - 2(x_{k+1}-x_k)(x_{k-1}-x_k)G_{12} + (x_{k-1}-x_k)^2G_{11}\,,$$

$$D_2 = (y_{k+1}-y_k)^2G_{22} - 2(y_{k+1}-y_k)(y_{k-1}-y_k)G_{12} + (y_{k-1}-y_k)^2G_{11}\,,$$

$$D_3=(x_{k+1}-x_k)(y_{k+1}-y_k)G_{22}-[(x_{k+1}-x_k)(y_{k-1}-y_k)+(x_{k-1}-x_k)(y_{k+1}-y_k)]G_{12} + (x_{k-1}-x_k)(y_{k-1}-y_k)G_{11}\,,$$

$$J = (x_{k+1}-x_k)(y_{k-1}-y_k) - (x_{k-1}-x_k)(y_{k+1}-y_k)\,,$$

and approximation of (4.26) gives

$$G_{11} = (X_{k+1}-X_k)^2 + (Y_{k+1}-Y_k)^2\,,\quad G_{22} = (X_{k-1}-X_k)^2 + (Y_{k-1}-Y_k)^2\,,$$

$$G_{12} = (X_{k+1}-X_k)(X_{k-1}-X_k) + (Y_{k+1}-Y_k)(Y_{k-1}-Y_k)\,,$$

$$B = \sqrt{G_{11}G_{22}-G_{12}^2} = (X_{k+1}-X_k)(Y_{k-1}-Y_k) - (X_{k-1}-X_k)(Y_{k+1}-Y_k)\,.$$

4.8 Minimization Procedure

Recall that under a convex mesh we mean the grid, every cell of which is a convex quadrilateral. If the set of convex meshes is not empty, the system of algebraic equations written for every interior node (here i is a global node number)

$$R_x = \frac{\partial I^h}{\partial x_i} = 0\,,\quad R_y = \frac{\partial I^h}{\partial y_i} = 0\,, \tag{4.29}$$

has at least one solution being a convex mesh. To find it one should have an initial convex mesh and then to use a method of unconstrained minimization.

The discrete functional I^h is minimized by applying a quasi-Newton procedure. Assuming the grid to be convex at the pth step of the iteration procedure we find the coordinates of the ith node at the $p+1$st step using the quasi-Newton method in the sense that the Hessian is a diagonal matrix

$$x_i^{p+1} = x_i^p - \tau\left(R_x\frac{\partial R_y}{\partial y_i} - R_y\frac{\partial R_x}{\partial y_i}\right)\left(\frac{\partial R_x}{\partial x_i}\frac{\partial R_y}{\partial y_i} - \frac{\partial R_y}{\partial x_i}\frac{\partial R_x}{\partial y_i}\right)^{-1}, \tag{4.30}$$

$$y_i^{p+1} = y_i^p - \tau\left(R_y\frac{\partial R_x}{\partial x_i} - R_x\frac{\partial R_y}{\partial x_i}\right)\left(\frac{\partial R_x}{\partial x_i}\frac{\partial R_y}{\partial y_i} - \frac{\partial R_y}{\partial x_i}\frac{\partial R_x}{\partial y_i}\right)^{-1};$$

here the iteration parameter τ is within the interval $0 < \tau < 1$.

To demonstrate, how to find R_x, R_y and their derivatives, we consider one of four triangles, into which the cell is partitioned by two diagonals, e.g., with vertices 1,2 and 4, see Fig. 4.4. The integrand (4.28) does not depend on rotation of the coordinates ξ, η, and as a result the formulas below hold for the remaining three triangles. For the triangle 124 setting k=1, k+1=2, k−1=4 in (4.28) gives (we omit the subscript of F_1)

$$F = \frac{\alpha D_1 + \beta D_2 + \gamma D_3}{J}, \tag{4.31}$$

where

$$\alpha = \frac{1 + f_x^2}{B\sqrt{1 + f_x^2 + f_x^2}}, \quad \beta = \frac{1 + f_x^2}{B\sqrt{1 + f_x^2 + f_x^2}}, \quad \gamma = \frac{2 f_x f_y}{B\sqrt{1 + f_x^2 + f_x^2}},$$

$$D_1 = (x_2 - x_1)^2 G_{22} - 2(x_2 - x_1)(x_4 - x_1) G_{12} + (x_4 - x_1)^2 G_{11},$$

$$D_2 = (x_2 - x_1)(y_2 - y_1) G_{22} - [(x_2 - x_1)(y_4 - y_1) + (x_4 - x_1)(y_2 - y_1)] G_{12} + (x_4 - x_1)(y_4 - y_1) G_{11},$$

$$D_3 = (y_2 - y_1)^2 G_{22} - 2(y_2 - y_1)(y_4 - y_1) G_{12} + (y_4 - y_1)^2 G_{11},$$

$$G_{11} = (X_2 - X_1)^2 + (Y_2 - Y_1)^2, \quad G_{22} = (X_4 - X_1)^2 + (Y_4 - Y_1)^2,$$

$$G_{12} = (X_2 - X_1)(X_4 - X_1) + (Y_2 - Y_1)(Y_4 - Y_1),$$

$$B = (X_2 - X_1)(Y_4 - Y_1) - (X_4 - X_1)(Y_2 - Y_1),$$

$$J = (x_2 - x_1)(y_4 - y_1) - (x_4 - x_1)(y_2 - y_1).$$

Here the derivatives f_x, f_y are calculated at point 1.

For the sake of convenience we introduce new variables

$$U = \alpha D_1 + \beta D_2 + \gamma D_3, \quad V = J,$$

and by virtue of (4.31) obtain

$$F = \frac{U}{V}.$$

Using the chain rule gives

$$\begin{gathered} F_x = (U_x - F V_x)/V, \quad F_y = (U_y - F V_y)/V, \\ F_{xx} = (U_{xx} - 2F_x V_x - F V_{xx})/V, \quad F_{yy} = (U_{yy} - 2F_y V_y - F V_{yy})/V, \\ F_{xy} = (U_{xy} - F_x V_y - F_y V_x - F V_{xy})/V, \quad F_{yx} = F_{xy}, \end{gathered} \tag{4.32}$$

where

$$U_x = \alpha (D_1)_x + \beta (D_2)_x + \gamma (D_3)_x, \quad U_y = \alpha (D_1)_y + \beta (D_2)_y + \gamma (D_3)_y, \text{ etc.}$$

When differentiating, we assume f_x, f_y are independent on the variables x, y (frozen metric, see [3]).

When varying triangle vertex 1, the variables are x_1, y_1. Substituting them instead of x, y, we have

$$(D_3)_{x_1} = (y_1 - y_2)(G_{22} - G_{12}) + (y_4 - y_1)(G_{11} - G_{12}) ,$$

$$(D_3)_{y_1} = (x_1 - x_2)(G_{22} - G_{12}) + (x_4 - x_1)(G_{11} - G_{12}) ,$$

$$(D_1)_{x_1} = 2(D_3)_{y_1} , \quad (D_1)_{y_1} = 0 , \quad (D_2)_{y_1} = 2(D_3)_{x_1} , \quad (D_2)_{x_1} = 0 ,$$

$$(D_3)_{x_1x_1} = (D_3)_{y_1y_1} = 0 , \quad (D_3)_{x_1y_1} = G_{22} - 2G_{12} + G_{11} ,$$

$$(D_1)_{x_1x_1} = (D_2)_{y_1y_1} = 2(D_3)_{x_1y_1} , \quad (D_1)_{y_1y_1} = (D_1)_{x_1y_1} = (D_2)_{x_1x_1} = (D_2)_{x_1y_1} = 0 ,$$

$$V_{x_1} = y_2 - y_4 , \quad V_{y_1} = x_4 - x_2 , \quad V_{x_1x_1} = V_{x_1y_1} = V_{y_1y_1} = 0 ,$$

and via the formulas (4.32) we find the derivatives of F.

When varying vertex 2, the variables are x_2, y_2. Analogously we have

$$(D_3)_{x_2} = (y_2 - y_1)G_{22} - (y_4 - y_1)G_{12} , \quad (D_3)_{y_2} = (x_2 - x_1)G_{22} - (x_4 - x_1)G_{12} ,$$

$$(D_1)_{x_2} = 2(D_3)_{y_2} , \quad (D_1)_{y_2} = 0 , \quad (D_2)_{y_2} = 2(D_3)_{x_2} , \quad (D_2)_{x_2} = 0 ,$$

$$(D_1)_{x_2x_2} = (D_2)_{y_2y_2} = 2G_{22} , \quad (D_3)_{x_2y_2} = G_{22} ,$$

$$(D_1)_{y_2y_2} = (D_1)_{x_2y_2} = (D_2)_{x_2x_2} = (D_2)_{x_2y_2} = (D_3)_{x_2x_2} = (D_3)_{y_2y_2} = 0 ,$$

$$V_{x_2} = y_4 - y_1 , \quad V_{y_2} = x_1 - x_4 , \quad V_{x_2x_2} = V_{x_2y_2} = V_{y_2y_2} = 0 .$$

When varying vertex 4, the variables are x_4, y_4. We have

$$(D_3)_{x_4} = (y_4 - y_1)G_{11} - (y_2 - y_1)G_{12} , \quad (D_3)_{y_4} = (x_4 - x_1)G_{11} - (x_2 - x_1)G_{12} ,$$

$$(D_1)_{x_4} = 2(D_3)_{y_4} , \quad (D_1)_{y_4} = 0 , \quad (D_2)_{y_4} = 2(D_3)_{x_4} , \quad (D_2)_{x_4} = 0 ,$$

$$(D_1)_{x_4x_4} = (D_2)_{y_4y_4} = 2G_{11} , \quad (D_3)_{x_4y_4} = G_{11} ,$$

$$(D_1)_{y_4y_4} = (D_1)_{x_4y_4} = (D_2)_{x_4x_4} = (D_2)_{x_4y_4} = (D_3)_{x_4x_4} = (D_3)_{y_4y_4} = 0 ,$$

$$V_{x_4} = y_1 - y_2 , \quad V_{y_4} = x_2 - x_1 , \quad V_{x_4x_4} = V_{x_4y_4} = V_{y_4y_4} = 0 .$$

If vertex 1 corresponds to the ith node of global numeration, then the contributions of four adjacent cells (see Fig. 4.4) are summarized by

$$[R_x]_i = \sum_{l=1}^{4} [F_x]_l , \quad [R_y]_i = \sum_{l=1}^{4} [F_y]_l , \quad [R_{xx}]_i = \sum_{l=1}^{4} [F_{xx}]_l ,$$

$$[R_{xy}]_i = \sum_{l=1}^{4} [F_{xy}]_l , \quad [R_{yy}]_i = \sum_{l=1}^{4} [F_{yy}]_l .$$

Here l=1 corresponds to the triangle 124 of the underlying cell, l=2 corresponds to the triangle 231 of the left cell, l=3 corresponds to the triangle 342 of the left-down cell, l=4 corresponds to the triangle 413 of the down cell.

To find the derivatives $(f_x)_i$ and $(f_y)_i$, we discretize the following relations

$$f_x = (y_\eta f_\xi - y_\xi f_\eta)/J, \quad f_y = (-x_\eta f_\xi + x_\xi f_\eta)/J,$$

and calculate them as average values over the four adjacent cells attached to the ith node

$$[f_x]_i = \sum_{l=1}^{4}[f_x]_l \Big/ \sum_{l=1}^{4} J_l, \quad [f_y]_i = \sum_{l=1}^{4}[f_y]_l \Big/ \sum_{l=1}^{4} J_l \ .$$

Here the index l corresponds to the same triangles as described above. For example, if l=1 then for the underlying cell we have

$$[f_x]_1 = [(y_4-y_1)(f_2-f_1) - (y_2-y_1)(f_4-f_1)],$$
$$[f_y]_1 = [(x_2-x_1)(f_4-f_2) - (x_4-x_1)(f_2-f_1)],$$

and J_1 is equal to J in the formula (4.31).

If a flow solver gives the values of the control function in the cell center, it is required to update them to the nodes. Since every ith node (except boundary nodes) is surrounded by four cells, it can be done by

$$f_i = c_a \sum_{l=1}^{4} f_l^c J_l \Big/ \sum_{l=1}^{4} J_l \ ,$$

where i is a global node number, f_l^c is the value in the cell center, c_a is the coefficient of adaptation which may depend on the node position, i.e., c_a=$c_a(x, y)$.

Remark 1. The method of untangling the initially prepared folded mesh was suggested in [12] (see also [11] p. 389). When treating a folded mesh the computational formulas are modified so that the initial greed need not belong to the set of convex grids. The quantities J_k in the denominator of (4.31),(4.32) are replaced with new quantities

$$\tilde{J}_k = \begin{cases} J_k & \text{if} \quad J_k > \varepsilon \ , \\ \varepsilon & \text{if} \quad J_k \le \varepsilon \ , \end{cases}$$

where ε>0 is a small quantity.

Remark 2. Numerical experiments have shown that employing the standard Newton procedure, when it is used the full matrix of the second derivatives (it consists of seven blocks, each block is a three-diagonal matrix), does not provide gainings when executing the iteration procedure analogous to (4.30). This is because the gainings in decreasing the mesh iterations are insignificant and additional computational operations (necessary to invert this matrix) reduce these gainings to zero and even leads to increasing the running time.

4.9 Boundary Nodes Redistribution

The algorithm of redistributing the boundary nodes, consisting of using constrained minimization, has been suggested in [3]. Such an approach leads to consistent redistribution of the grid nodes inside the countable domain Ω and on its boundary $\partial\Omega$, which increases the reliability of grid generation and modeling the main flow problem. In this approach, we minimize the following discrete functional

$$\tilde{I}^h = \frac{1}{2}\sum_{N=1}^{N_e}\sum_{k=1}^{4}\frac{1}{4}\,[F_k]_N + \sum_{l\in\mathcal{L}}\lambda_l H_l = I^h + \sum_{l\in\mathcal{L}}\lambda_l H_l\,, \tag{4.33}$$

where the constraints $H_l{=}H(x_l, y_l){=}0$ define $\partial\Omega$, λ_l are the Lagrange multipliers, $\mathcal{L}$ is the set of the boundary nodes. The function $H(x, y)$ is assumed piecewise differentiable.

If the set of convex grids is not empty, the system of algebraic equations has at least one solution being the convex mesh

$$R_x = \frac{\partial I^h}{\partial x_i} + \lambda_i\frac{\partial H_i}{\partial x_i} = 0\,,\quad R_y = \frac{\partial I^h}{\partial y_i} + \lambda_i\frac{\partial H_i}{\partial y_i} = 0\,,\quad H_i = 0\,. \tag{4.34}$$

Here $\lambda_i{=}0$, if $i \notin \mathcal{L}$ and constraints are defined for the boundary nodes $i \in \mathcal{L}$.

Consider the method of minimizing the functional (4.33) assuming the grid to be convex at the pth step of the iteration procedure. We use the quasi-Newton procedure to find the coordinates x_i^{p+1}, y_i^{p+1} of the ith node from the system (4.34)

$$\begin{aligned}
\tau R_x + \frac{\partial R_x}{\partial x_i}(x_i^{p+1} - x_i^p) + \frac{\partial R_x}{\partial y_i}(y_i^{p+1} - y_i^p) + \frac{\partial R_x}{\partial \lambda_i}(\lambda_i^{p+1} - \lambda_i^p) &= 0\,,\\
\tau R_y + \frac{\partial R_y}{\partial x_i}(x_i^{p+1} - x_i^p) + \frac{\partial R_y}{\partial y_i}(y_i^{p+1} - y_i^p) + \frac{\partial R_y}{\partial \lambda_i}(\lambda_i^{p+1} - \lambda_i^p) &= 0\,,\\
\tau H_i + \frac{\partial H_i}{\partial x_i}(x_i^{p+1} - x_i^p) + \frac{\partial H_i}{\partial y_i}(y_i^{p+1} - y_i^p) &= 0\,,
\end{aligned} \tag{4.35}$$

where

$$\begin{aligned}
\frac{\partial R_x}{\partial x_i} &= \frac{\partial^2 I^h}{\partial x_i^2} + \lambda_i\frac{\partial^2 H_i}{\partial x_i^2}\,,\quad \frac{\partial R_x}{\partial y_i} = \frac{\partial^2 I^h}{\partial x_i \partial y_i} + \lambda_i\frac{\partial^2 H_i}{\partial x_i \partial y_i}\,,\quad \frac{\partial R_x}{\partial \lambda_i} = \frac{\partial H_i}{\partial x_i}\,,\\
\frac{\partial R_y}{\partial x_i} &= \frac{\partial^2 I^h}{\partial x_i \partial y_i} + \lambda_i\frac{\partial^2 H_i}{\partial x_i \partial y_i}\,,\quad \frac{\partial R_y}{\partial y_i} = \frac{\partial^2 I^h}{\partial y_i^2} + \lambda_i\frac{\partial^2 H_i}{\partial y_i^2}\,,\quad \frac{\partial R_y}{\partial \lambda_i} = \frac{\partial H_i}{\partial y_i}\,.
\end{aligned}$$

Resolving the last equation of (4.35) about $y_i^{p+1} - y_i^p$ and substituting it in the two remaining equations, we get the system

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}\begin{pmatrix} x_i^{p+1} - x_i^p \\ \lambda_i^{p+1} - \lambda_i^p \end{pmatrix} = \begin{pmatrix} a_{13} \\ a_{23} \end{pmatrix}\,,$$

where

$$a_{11} = \frac{\partial R_x}{\partial x_i} - \frac{\partial R_x}{\partial y_i}\frac{\partial H_i}{\partial x_i}\Big/\frac{\partial H_i}{\partial y_i}, \quad a_{12} = \frac{\partial H_i}{\partial x_i}, \quad a_{13} = \tau\left[\frac{\partial R_x}{\partial y_i} H_i \Big/ \frac{\partial H_i}{\partial y_i} - R_x\right],$$

$$a_{21} = \frac{\partial R_y}{\partial x_i} - \frac{\partial R_y}{\partial y_i}\frac{\partial H_i}{\partial x_i}\Big/\frac{\partial H_i}{\partial y_i}, \quad a_{22} = \frac{\partial H_i}{\partial y_i}, \quad a_{23} = \tau\left[\frac{\partial R_y}{\partial y_i} H_i \Big/ \frac{\partial H_i}{\partial y_i} - R_y\right].$$

Since $H_i = 0$, the terms a_{13}, a_{23} are simplified. Denoting

$$\Delta = a_{11}a_{22} - a_{12}a_{21}\,, \quad \Delta_1 = a_{13}a_{22} - a_{23}a_{12}\,, \quad \Delta_2 = a_{11}a_{23} - a_{21}a_{13}\,,$$

we obtain

$$x_i^{p+1} = x_i^p + \Delta_1/\Delta\,, \quad \lambda_i^{p+1} = \lambda_i^p + \Delta_2/\Delta\,, \tag{4.36}$$

and y_i^{p+1} is determined from the third equation of (4.35). If the constraints are resolved about y in the form $H(x,y) = y - h(x) = 0$, then

$$\frac{\partial H_i}{\partial x_i} = -\frac{\partial h_i}{\partial x_i}\,, \quad \frac{\partial H_i}{\partial y_i} = 1\,,$$

and above formulas are simplified. Constraints can be resolved about x in the form $H(x,y) = x - \tilde{h}(y) = 0$ and then (here it is better to resolve the third equation of (4.35) about $x_i^{p+1} - x_i^p$)

$$\frac{\partial H_i}{\partial x_i} = 1\,, \quad \frac{\partial H_i}{\partial y_i} = -\frac{\partial \tilde{h}_i}{\partial y_i}\,.$$

These two forms of $H(x,y)$ can substitute for each other. For example, on the part of $\partial\Omega$ that is nearly parallel to the axis x the boundary should be defined in the form $y = h(x)$, and where $\partial\Omega$ is nearly parallel to the axis y it should be defined as $x = \tilde{h}(y)$.

If $\partial\Omega$ is given by parametric functions $x{=}x(t)$, $y{=}y(t)$ or tabular values $(x,y)_i$, the following algorithm can be used. When calculating the coordinates of the ith node, in the interval (x_{i-1}, x_{i+1}) we construct an interpolating parabola $t{=}t(x)$ using the values in three nodes $i{-}1, i, i{+}1$. From (4.36) we compute an intermediate value $\tilde{x}_i^{p+1}$, further from the interpolation formula we determine $t_i{=}t(\tilde{x}_i^{p+1})$ and final values x_i^{p+1}, y_i^{p+1} from the parametric formulas.

Another way of redistributing the nodes along $\partial\Omega$, given as parametric functions or by tabular values, employs an unconstrained minimization of the functional in parametric form and is based on solving the following system of algebraic equations, referred to as parametric minimization,

$$R_t = R_x\frac{\partial x_i}{\partial t_i} + R_y\frac{\partial y_i}{\partial t_i} = 0\,,$$

via the quasi-Newton procedure

$$\tau R_t + \frac{\partial R_t}{\partial t_i}\,(t_i^{p+1} - t_i^p) = 0\,. \tag{4.37}$$

Here

$$\frac{\partial R_t}{\partial t_i} = \frac{\partial R_x}{\partial x_i}\left(\frac{\partial x_i}{\partial t_i}\right)^2 + \frac{\partial R_y}{\partial y_i}\left(\frac{\partial y_i}{\partial t_i}\right)^2 + \left(\frac{\partial R_x}{\partial y_i} + \frac{\partial R_y}{\partial x_i}\right)\frac{\partial x_i}{\partial t_i}\frac{\partial y_i}{\partial t_i}$$
$$+ R_x\frac{\partial^2 x_i}{\partial t_i^2} + R_y\frac{\partial^2 y_i}{\partial t_i^2}, \quad R_x = \frac{\partial I^h}{\partial x_i}, \quad R_y = \frac{\partial I^h}{\partial y_i}.$$

Remark 1. For the analytical control function f, constrained and parametric minimization provide similar results. Real-world 2D flow computations have shown that it is better to perform adaptation along the boundary using constrained minimization (4.35), (4.36) since the procedure (4.37) does not always ensure consistent redistribution of the nodes in Ω and on $\partial\Omega$.

Remark 2. Using the constrained minimization without adaptation (in (4.25) we set f=*const.*) and with Euclidean metric G (G_{11}=G_{22}=1, G_{12}=0) implies that we seek the conformal mapping $x(\xi,\eta)$, $y(\xi,\eta)$ of the parametric square onto the domain Ω with an additional parameter, so-called conformal modulus. This is because according to the Riemann theorem under the conformal mapping we can define correspondence only between three points on the boundary contour of the physical and parametric domains, and in our case there is a correspondence between four corner points on the boundary. That is why such a mesh is said to be rather a quasi-conformal grid, and, therefore, the mapping is quasi-conformal. For the first time the quasi-conformal mapping in grid generation was applied in [8].

To demonstrate it we write the Dirichlet functional for the direct mapping $x(\xi,\eta)$, $y(\xi,\eta)$

$$I(x,y) = \int (x_\xi^2 + y_\xi^2 + x_\eta^2 + y_\eta^2)\, d\xi d\eta, \tag{4.38}$$

for which the Euler-Lagrange equations are the system of the Laplace equations

$$\triangle x = 0, \quad \triangle y = 0,$$

where the Laplace operator is expressed in terms of the variables ξ, η. Applying the identity

$$x_\xi^2 + y_\xi^2 + x_\eta^2 + y_\eta^2 = (x_\xi - y_\eta)^2 + (x_\eta + y_\xi)^2 + 2(x_\xi y_\eta - x_\eta y_\xi),$$

we have

$$I(x,y) = \int [(x_\xi - y_\eta)^2 + (x_\eta + y_\xi)^2] d\xi d\eta + 2A,$$

where A is the area of the domain Ω. Hence, the minimum of the functional (4.38), equal to the double area of Ω, is attained under the conformal mapping obeying the Cauchy-Riemann relations

$$x_\xi = y_\eta, \quad x_\eta = -y_\xi,$$

when it is defined the correspondence between three boundary points on the parametric square contour (three corner points) and three points on $\partial\Omega$. That is why, in general case, the Diriclet functional is the *measure of non-conformality* of the mapping.

Before proceeding to numerical examples, in the next two sections we will present a variational principle, being an extension of the equidistribution principle in the 2D case. It is formulated in the both continuum and discrete case. Complete proof is executed only for the discrete mapping. The reader, interested in practical applications, may omit these sections.

4.10 Formulation of Variational Principle

Consider the problem of constructing a smooth mapping of the parametric square $Q=\{(\xi,\eta)$ $0<\xi,\eta<1\}$ onto an arbitrary domain Ω with given transformation between the boundaries $(x_b, y_b) : \partial Q \to \partial\Omega$, where x_b, y_b are given functions, so that the resulting mapping $x(\xi,\eta), y(\xi,\eta)$ is invertible.

With the metric g defined in the domain Ω as

$$g_{11} = x_\xi^2 + y_\xi^2\,,\quad g_{12} = g_{21} = x_\xi x_\eta + y_\xi y_\eta\,,\quad g_{22} = x_\eta^2 + y_\eta^2\,, \tag{4.39}$$

the energy density functional (4.24) takes the form

$$I = \frac{1}{2}\int_0^1\int_0^1 \frac{(x_\xi^2 + y_\xi^2)G_{22} - 2(x_\xi x_\eta + y_\xi y_\eta)G_{12} + (x_\eta^2 + y_\eta^2)G_{11}}{(x_\xi y_\eta - x_\eta y_\xi)\sqrt{G_{11}G_{22} - G_{12}^2}}\,d\xi d\eta\,. \tag{4.40}$$

Here $\{G_{lm},\ l,m=1,2\}$ are the elements of the symmetric and positive definite matrix $G(\xi,\eta) = G^\top > 0$.

The integrand in (4.40) can be written as

$$\frac{1}{2}\frac{tr(G^{-1}g)}{\sqrt{det(G^{-1}g)}} = \frac{1}{2}\frac{\lambda_1+\lambda_2}{\sqrt{\lambda_1\lambda_2}} \geq 1\,, \tag{4.41}$$

where λ_1 and λ_2 are the eigenvalues of the matrix $G^{-1}g$. The equality is attained if and only if $\lambda_1=\lambda_2$. Then it follows that $I\geq 1$ and $I=1$ if and only if

$$G_{lm} = \alpha(\xi,\eta)g_{lm}\,,$$

where $\alpha(\xi,\eta)$ is an arbitrary smooth function defined in the unit square. Hence, for every one-to-one mapping with a positive Jacobian there exists a matrix $G(\xi,\eta)=G^\top > 0$ so that the minimum of the functional (4.40) is attained under this mapping.

Instead of G it is convenient to introduce the matrix $\tilde{G}$ with the elements

$$\tilde{G}_{lm}=G_{lm}/\sqrt{det(G)}\,.$$

Then the functional takes the form

$$I = \frac{1}{2}\int_0^1\int_0^1 \frac{(x_\xi^2 + y_\xi^2)\tilde{G}_{22} - 2(x_\xi x_\eta + y_\xi y_\eta)\tilde{G}_{12} + (x_\eta^2 + y_\eta^2)\tilde{G}_{11}}{x_\xi y_\eta - x_\eta y_\xi}\,d\xi d\eta\,. \tag{4.42}$$

Since $det(\tilde{G})$=1, the elements of the matrix $\tilde{G}$ are not independent. They are defined by two smooth functions $f_1(\xi,\eta)$ and $f_2(\xi,\eta)$

$$\tilde{G}_{11} = f_1\ ,\ \ \tilde{G}_{12} = \tilde{G}_{21} = \sqrt{f_1 f_2 - 1}\ ,\ \ \tilde{G}_{22} = f_2\ . \tag{4.43}$$

The Euler-Lagrange equations for the functional (4.42) are

$$\begin{aligned} \alpha x_{\xi\xi} - 2\beta x_{\xi\eta} + \gamma x_{\eta\eta} &= x_\xi(-P\tilde{G}_{22} + Q\tilde{G}_{12}) + x_\eta(P\tilde{G}_{12} - Q\tilde{G}_{11})\ , \\ \alpha y_{\xi\xi} - 2\beta y_{\xi\eta} + \gamma y_{\eta\eta} &= y_\xi(-P\tilde{G}_{22} + Q\tilde{G}_{12}) + y_\eta(P\tilde{G}_{12} - Q\tilde{G}_{11})\ , \end{aligned} \tag{4.44}$$

where

$$\alpha = x_\eta^2 + y_\eta^2\ ,\ \ \beta = x_\xi x_\eta + y_\xi y_\eta\ ,\ \ \gamma = x_\xi^2 + y_\xi^2\ ,$$

$$P{=}\frac{1}{2}\left(\alpha\frac{\partial\tilde{G}_{11}}{\partial\xi} - 2\beta\frac{\partial\tilde{G}_{12}}{\partial\xi} + \gamma\frac{\partial\tilde{G}_{22}}{\partial\xi}\right)\ ,\ \ Q{=}\frac{1}{2}\left(\alpha\frac{\partial\tilde{G}_{11}}{\partial\eta} - 2\beta\frac{\partial\tilde{G}_{12}}{\partial\eta} + \gamma\frac{\partial\tilde{G}_{22}}{\partial\eta}\right)\ .$$

Consider the class of all homeomorphisms from the unit square onto the domain Ω, satisfying the same boundary conditions as $(x_b, y_b) : \partial Q \to \partial\Omega$. Recall that $(x_b, y_b) : \partial Q \to \partial\Omega$ is a homeomorphism between the boundaries. Above we have shown that for every mapping from this class there exists a matrix $G(\xi,\eta) = G^\top > 0$ so that the minimum of the functional (4.42) is attained under this mapping.

It may be conjectured that the class of all smooth two-dimensional homeomorphisms is equivalent to the class of all smooth symmetric and positive definite 2×2 matrices with a determinant equal to a unity at every point of the square.

To substantiate this conjecture, it remains to show that for any smooth symmetric and positive definite matrix $G(\xi,\eta)$ there exists a homeomorphism, under which the minimum of F is attained. The class of such homeomorphisms is defined by all possible pairs of smooth functions $f_1(\xi,\eta)$ and $f_2(\xi,\eta)$ satisfying the conditions (4.43).

We formulate the conjecture in the form of the variational principle, allowing to single out a class of homeomorphisms among all possible mappings of the square onto Ω with a defined one-to-one mapping between the boundaries.

Optimality principle. *A smooth mapping of the unit square Q onto a domain Ω is a homeomorphism if and only if it delivers the minimum of the functional (4.42) with some symmetric and positive definite matrix-function G.*

If this hypothesis holds true, then the following proposition is correct. A mapping, being not a homeomorphism, cannot minimize the functional F with any matrix $G(\xi,\eta){=}G^\top{>}0$ and thereby satisfy the Euler-Lagrange equations (4.44).

4.11 Variational Principle at Discrete Level

The grid generation problem is considered in the following formulation. In a simply connected domain Ω in plane (x, y) a grid $\{(x,y)_{ij}, i{=}1,\ldots,i^*;\ j{=}1,\ldots,j^*\}$ must be constructed with given coordinates of boundary nodes $(x,y)_{i1}$, $(x,y)_{ij^*}$, $(x,y)_{1j}$, $(x,y)_{i^*j}$.

This problem is a discrete analog of the problem on seeking the functions $x(\xi,\eta)$, $y(\xi,\eta)$, providing an one-to-one mapping of the square Q onto the domain Ω with a given correspondence between the boundaries $(x_b, y_b) : \partial Q \to \partial\Omega$.

Consider the cell with number $i{+}1/2, j{+}1/2$ and enumerate its vertices as shown in Fig. 4.4, i.e., the grid node (i,j) corresponds to vertex 1, node $(i{+}1, j)$ to vertex 2, etc. Each vertex corresponds to a triangle as follows: vertex 1 corresponds to $\triangle_{412}$, vertex 2 corresponds to $\triangle_{123}$, etc. Denote by J_k the doubled area of the corresponding triangle (k=1, 2, 3, 4); for example,

$$J_1 = (x_2 - x_1)(y_4 - y_1) - (y_2 - y_1)(x_4 - x_1) =$$

$$(x_{i+1,j} - x_{ij})(y_{i,j+1} - y_{ij}) - (y_{i+1,j} - y_{ij})(x_{i,j+1} - x_{ij}) .$$

A discrete analog of Jacobian positiveness may be written in the form of the inequalities

$$[J_k]_{i+1/2,j+1/2} > 0, \quad k = 1,2,3,4; \quad i = 1,\ldots,i^* - 1; \quad j = 1,\ldots,j^* - 1 , \qquad (4.45)$$

where $J_k{=}(x_{k+1} - x_k)(y_{k-1} - y_k) - (y_{k+1} - y_k)(x_{k-1} - x_k)$, the index k is cyclic, and when calculating J_k we set $k{-}1{=}4$ if $k{=}1$ and $k{+}1{=}1$ if $k{=}4$. When the conditions (4.45) are fulfilled, every cell is a convex quadrilateral.

The set of grids, satisfying the inequalities (4.45), we denote as W_D. This set belongs to Euclidean space $\mathbb{R}^{N_{in}}$ where the total number $N_{in} = 2(i^* - 2)(j^* - 2)$ of degrees of freedom is the doubled number of the interior grid nodes. In this space, W_D is an open bounded set. Its boundary ∂W_D is the set of grids for which at least one inequality turns into equality. Henceforth, the set W_D is assumed to be nonempty.

Thus, any admissible set of coordinates of the boundary nodes defines a class of unfolded (convex) grids satisfying the inequalities (4.45). With the purpose to characterize this class it is naturally to use a discrete analogue of the variational principle, formulated in Section 4.10.

The discrete analogue of the functional (4.40) is

$$I^h = \frac{1}{(i^* - 1)(j^* - 1)} \sum_{i=1}^{i^*-1} \sum_{j=1}^{j^*-1} \sum_{k=1}^{4} \frac{1}{4} [F_k]_{i+1/2,j+1/2} , \qquad (4.46)$$

$$F_k = \frac{(r_{k+1}{-}r_k)^2(\tilde{G}_{22})_k{-}2(r_{k+1}{-}r_k)(r_{k-1}{-}r_k)(\tilde{G}_{12})_k{+}(r_{k-1}{-}r_k)^2(\tilde{G}_{11})_k}{2[(x_{k+1} - x_k)(y_{k-1} - y_k) - (y_{k+1} - y_k)(x_{k-1} - x_k)]} , \qquad (4.47)$$

where

$$r_k = (x_k, y_k)^\top , \quad (r_{k+1} - r_k)^2 = (x_{k+1} - x_k)^2 + (y_{k+1} - y_k)^2 ,$$

$$(r_{k+1} - r_k)(r_{k-1} - r_k) = (x_{k+1} - x_k)(x_{k-1} - x_k) + (y_{k+1} - y_k)(y_{k-1} - y_k) ,$$

$$(r_{k-1} - r_k)^2 = (x_{k-1} - x_k)^2 + (y_{k-1} - y_k)^2 ,$$

$(\tilde{G}_{lm})_k$ are the elements of the matrix

$$\tilde{G}_k = \tilde{G}_k^\top > 0 , \quad det(\tilde{G}_k) = 1 ,$$

referred to the triangle with the local number k of the $(i+1/2, j+1/2)$th cell.

Consider a symmetric and positive definite matrix $(g)_k$ with elements

$$(g_{11})_k = (r_{k+1} - r_k)^2\,, \quad (g_{12})_k = (g_{21})_k = (r_{k+1} - r_k)(r_{k-1} - r_k)\,, \quad (g_{22})_k = (r_{k-1} - r_k)^2\,. \tag{4.48}$$

Substituting (4.48) in (4.47), we obtain

$$F_k = \frac{tr(\tilde{G}_k^{-1} g_k)}{2\sqrt{det(\tilde{G}_k^{-1} g_k)}} = \frac{\lambda_1 + \lambda_2}{2\sqrt{\lambda_1 \lambda_2}} \geq 1\,, \tag{4.49}$$

where λ_1 and λ_2 are the eigenvalues of the matrix $\tilde{G}_k^{-1} g_k$. The equality in (4.49) is attained if and only if $\lambda_1 = \lambda_2$. Therefore, we have $F_k \geq 1$, and $F_k = 1$ if and only if

$$(\tilde{G}_{lm})_k = (g_{lm})_k / \sqrt{det(g_k)}\,.$$

The function I^h posses the barrier property, which is established by the following lemma.

Lemma. *For any set of symmetric and positive definite matrices at every cell* $\{\tilde{G}_k(ij),\ k=1,2,3,4;\ i=1,\ldots,i^*-1;\ j=1,\ldots,j^*-1\}$ *such that* $det(\tilde{G}_k)=1$, *the function* I^h *has an infinite barrier on the boundary of the class of unfolded grids* W_D, *i.e., if in a cell the area of one of the triangles tends to zero, at the same time remaining positive, then* $I^h \to +\infty$.

Proof. Assume that $J_k \to 0$ in some cell while I^h does not tend to $+\infty$. Then the numerator in (4.47) must tend to zero as well. From this and inequality

$$\frac{(r_{k+1} - r_k)^2 (G_{22})_k - 2(r_{k+1} - r_k)(r_{k-1} - r_k)(G_{12})_k + (r_{k-1} - r_k)^2 (G_{11})_k}{\sqrt{(G_{11})_k (G_{22})_k - (G_{12})_k^2}}$$

$$\geq \sqrt{\frac{\lambda_{min}(G_k)}{\lambda_{max}(G_k)}} \left((r_{k+1} - r_k)^2 + (r_{k-1} - r_k)^2\right)$$

it follows that the lengths of two corresponding sides of the cell tend to zero, and, consequently, the areas of all triangles, including these sides, tend to zero as well. Reiterating this argumentation as many as necessary one finds that the lengths of all sides of all cells must tend to zero. In other words the grid will shrink to a point, and this contradicts the boundary conditions (the boundary grid nodes belong to the domain boundary). □

Now we prove the discrete analogue of the variational principle formulated in Section 4.10.

Theorem. *A structured grid* $\{(x,y)_{ij},\ i=1,\ldots,i^*;\ j=1,\ldots,j^*\}$, *constructed in a domain* Ω *by the given coordinates of the boundary nodes, is an element of the class of unfolded grids* W_D *if and only if it delivers the minimum of the functional* I^h *for some set of symmetric and positive definite matrices* $\{G_k(ij),\ k=1,2,3,4;\ i=1,\ldots,i^*-1;\ j=1,\ldots,j^*-1\}$.

Proof of the necessity. Let a grid, satisfying the inequalities (4.47), be given. In every cell, we introduce four matrices $G_k=\alpha_k g_k$, $\alpha_k>0$, $k=1,2,3,4$, where the elements of the matrix g_k are given by (4.48). Instead of G_k we define the matrices $\tilde{G}_k$ with the elements $(\tilde{G}_{lm})_k=(g_{lm})_k/\sqrt{det(g_k)}$, determined uniquely by the elements of g_k and $det(\tilde{G}_k)=1$. It is easily seen that in this case every summand in (4.45) is equal to 1

$$F_k = \frac{tr(\tilde{G}_k^{-1} g_k)}{2\sqrt{\tilde{G}_k^{-1} det(g_k)}} = \frac{tr(g_k^{-1} g_k)}{2\sqrt{det(g_k^{-1} g_k)}} = 1 .$$

Consequently, for every grid from W_D there exists such a set of matrices $\tilde{G}_k(ij)$ that I^h takes a minimal value equal to 1. Let us show that the grid, which corresponds to the minimum $I^h=1$ for the defined set of matrices $\tilde{G}_k(ij)$, is unique.

Really, if $I^h=1$, then each term in (4.45) takes the minimal possible value equal to $F_k=1$. Considering the sum of four terms for some cell, we can see that if the coordinates of two adjacent vertices are specified, the last two are determined uniquely from the conditions $F_k=1$, $k=1,2,3,4$. Thus, if $I^h=1$, the mesh can be constructed sequentially, beginning from the boundary. Initially, by the coordinates of two boundary vertices of a boundary cell we determine the coordinates of two remaining vertices. Next this procedure is repeated for the neighboring cell, etc. As a result the grid sought will be constructed. The constructed grid is unique.

Proof of the sufficiency. Let the set of symmetric and positive definite matrices $\{G_k(ij),\ k=1,2,3,4;\ i=1,\ldots,i^*-1;\ j=1,\ldots,j^*-1\}$ be given. The function I^h possesses an infinite barrier on the boundary of the class of unfolded meshes W_D according to the Lemma of the barrier property. From (4.48) it follows that I^h is bounded from below by 1. Since I^h is a continuous function of coordinates of internal grid nodes, there exists at least one grid from W_D, which delivers the minimum of I^h and the equations (the necessary conditions of minimum)

$$\partial I^h/\partial x_{ij} = 0 , \quad \partial I^h/\partial y_{ij} = 0 ,$$

have at least one solution from the class of unfolded grids W_D. This completes the proof. □

Remark 1. From this principle it follows that the class of nondegenerate grids W_D is equivalent to the class of all sets of symmetric and positive definite matrices $\{\tilde{G}_k(ij),\ k=1,2,3,4;\ i=1,\ldots,i^*-1;\ j=1,\ldots,j^*-1\}$, so that $det(\tilde{G}_k) = 1$.

Remark 2. The theorem can be extended to the case when the triangle areas in the inequality (4.45) may be both positive and negative. It is possible, for instance, for the triangles corresponding to the corners of the square Q, if the angle at the corner point in the domain Ω is greater than π.

Remark 3. The problem can be formulated for a block-regular grid consisting of several regular grids. This is the case when the parametric domain, consisting of several rectangles (rather than a parametric square), is mapped onto the physical domain. The principle can be also extended to the case of an arbitrary irregular grid.

4.12 Numerical Results

4.12.1. Horseshoe domain. We begin with grid generation in the horseshoe domain, shown in Fig. 4.5. Its bottom boundary is a half-circle with radius R=1 and top is a half-ellipse

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 ,$$

with the semi-axes a=2 and b=2R, R=4.5. Two other sides are the straight-line segments. This domain is of interest, because the Winslow's method [26] produces a folded mesh and besides all interior grid nodes are located near the bottom boundary. Meanwhile, a suitable mesh is constructed by drawing the straight-lines joining the opposite sides, see Fig. 4.5 *a*.

The variational barrier method [11] (functional (4.40) with the Euclidean second metric G: G_{11}=G_{22}=1, G_{12}=0) delivers the mesh presented in Fig. 4.5 *b*. Some grid cells, attached to the top boundary, are very stretched and this mesh is unfit to calculation.

The presented method allow to obtain both the mesh from Fig. 4.5 *a* and 4.5 *b*. For instance, if as an initial guess to define the mesh from Fig. 4.5 *b* and as an objective shape to prescribe the shape of corresponding grid cells from Fig. 4.5 *a*, then minimizing the energy density functional we get the mesh from Fig. 4.5 *a* in several tenths of mesh iterations.

If as an initial guess to define the mesh from Fig. 4.5 *a* and as an objective shape to prescribe the square for every cell, then, minimizing the energy density functional, we get the mesh from Fig. 4.5 *b* with maximum value of the energy density (over all cells) equal to 7.9 and average equal to 1.9. By this values one can judge how far the produced mesh from the square grid. Note that the minimum value of the energy density, equal to 1, is not attained, however the minimization problem possesses the unique solution. For more complicated domains the uniqueness of the solution can be lost.

There are possible different ways to define the cell objective shape to attain mesh orthogonality near the boundary. In Section 4.5, it was presented a simple way to define the boundary cell shape and the mesh, constructed using this method of orthogonalization, is shown in Fig. 4.5*c*. The width of the objective boundary cell is equal to 0.1. The width of the interpolation layer along j (number of cell layers) is equal to 5. For remaining cells, as an objective shape, we define the cell shapes of the mesh depicted in Fig. 4.5 *a*. The maximum value of the energy density over all cells is equal to 1.4 and average equal to 1.07. The mesh is "more orthogonal" near the top boundary than the mesh of Fig. 4.5 *a*. However, the shape of the boundary cells is insufficiently close to the objective shape.

Analogous results are obtained when using the square as the objective cell shape everywhere except the boundary cell layer and interpolation layer. This mesh is shown in Fig. 4.5 *d*. Here the width of the objective boundary cell is equal to 0.1 and interpolation is executed along five cell layers. The maximum value of the energy density over all cells is equal to 6.1 and average equal to 2.48. In toto the mesh differs substantially from the grid of Fig. 4.5 *c*, however, the shape of some boundary cells is rather far from the objective ones.

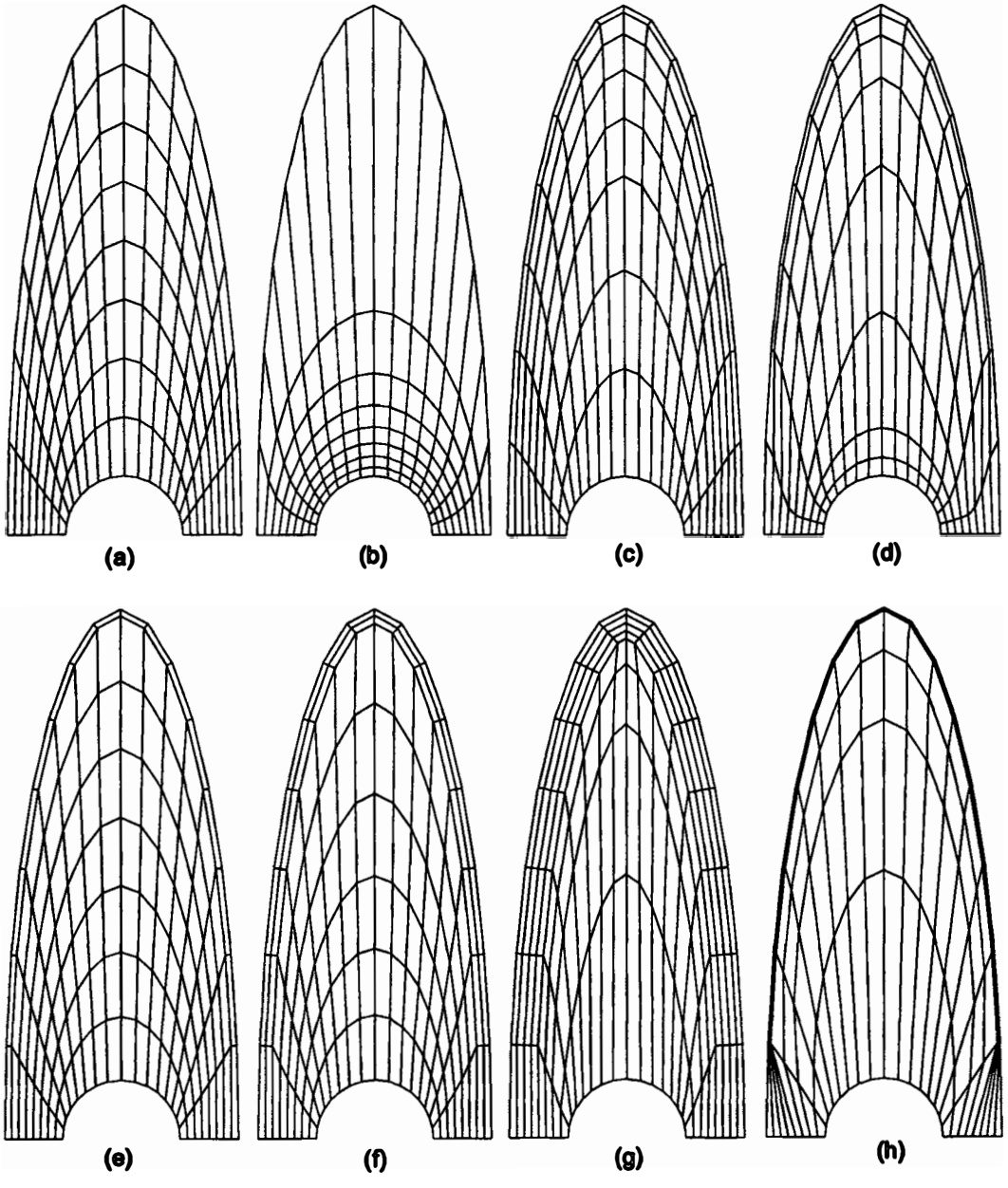


Figure 4.5: Examples of grids in the horseshoe domain.

Hence, there is a necessity to obtain such a boundary cell shape which is exactly equal to the objective shape. It can be attained as follows. Besides the objective shape of the boundary cells we define an initial mesh, and cell shape of this grid is defined to be an objective shape outside the boundary layer. Next the function of the average energy density deformation of the initial grid is minimized. The resulting mesh is considered as the objective one (except the boundary cell layer, for which the cell shapes are defined from the beginning). Next again the energy density is minimized, etc. Solving sequentially the minimization problems, in the end we obtain the mesh with cell shapes exactly equal to the objective ones. Actually, both the cell shape and size are equal to the objective ones, respectively. However, one may expect that if this procedure treats some (not a large) part of grid cells, the cells, located far from treated cells, will not be deformed. This is confirmed by the example depicted in Fig. 4.5 *e*. The mesh is produced for one layer of the boundary cells with the prescribed width of the boundary cell equal to 0.1. As initial it is used the mesh of Fig. 4.5 *a*. Both the maximum and average value of the energy density is equal to 1 (within 7 figures after the decimal point). Consequently, the shape and size of the boundary cells are equal to prescribed.

Fig. 4.5 *f*, *g* demonstrates the grids obtained using the above procedure for two and four layers of boundary cells, respectively. The width of all layers are equal to 0.1. Accuracy of the solution is the same as that of Fig. 4.5 *e*. The last case demonstrates that, if the number of boundary layers is equal to 4, the number of grid nodes is insufficient for precise approximation of the domain and there appears too stretched cells in the center of the domain and near the bottom boundary.

The mesh, presented in Fig. 4.5 *h*, has four layers of boundary cells with the layer width equal to 0.01. We see that the presented method can condense significantly the grid lines to the boundary. We also performed computations up to the case when the width of the first boundary layer is equal to 10^{-5}. This mesh is similar to presented in Fig. 4.5 *h*, and boundary cell layer is so narrow that can not be seen (that is why we do not present it here).

4.12.2. Grid in the external aerodynamics problem. When solving external aerodynamics problems there are specific requirements for the grid generation methods. This is because the countable domain on one side is bounded by the streamline contour, whose shape is described rather precisely and requirements for nodes location are stiff, and on the other side by the outer shock wave (if supersonic flow) or by a far outer boundary (if subsonic flow) where requirements for nodes location are weak. Especially stiff conditions are imposed when modeling the viscous flow near an airplane. Here the width of the boundary layer is about $10^{-3} - 10^{-5}$ of the typical linear size. Models of turbulence require that the mesh to be orthogonal near the body surface. The actual problem is construction of a grid of a simple structure, e.g., O-type mesh in the 2D case, being orthogonal near the boundary with prescribed number of boundary cell layers of a necessary width.

An example of the O-type 111×37 mesh is presented in Fig. 4.6 *a*. The internal boundary of the domain is the airplane contour (due to symmetry we demonstrate only the right-half of the domain). The boundary layer has the width about 10^{-3} of the typical line size and consists of 7 cell layers. Fragments of the mesh are presented in Fig. 4.6 *b*, *c*. The mesh

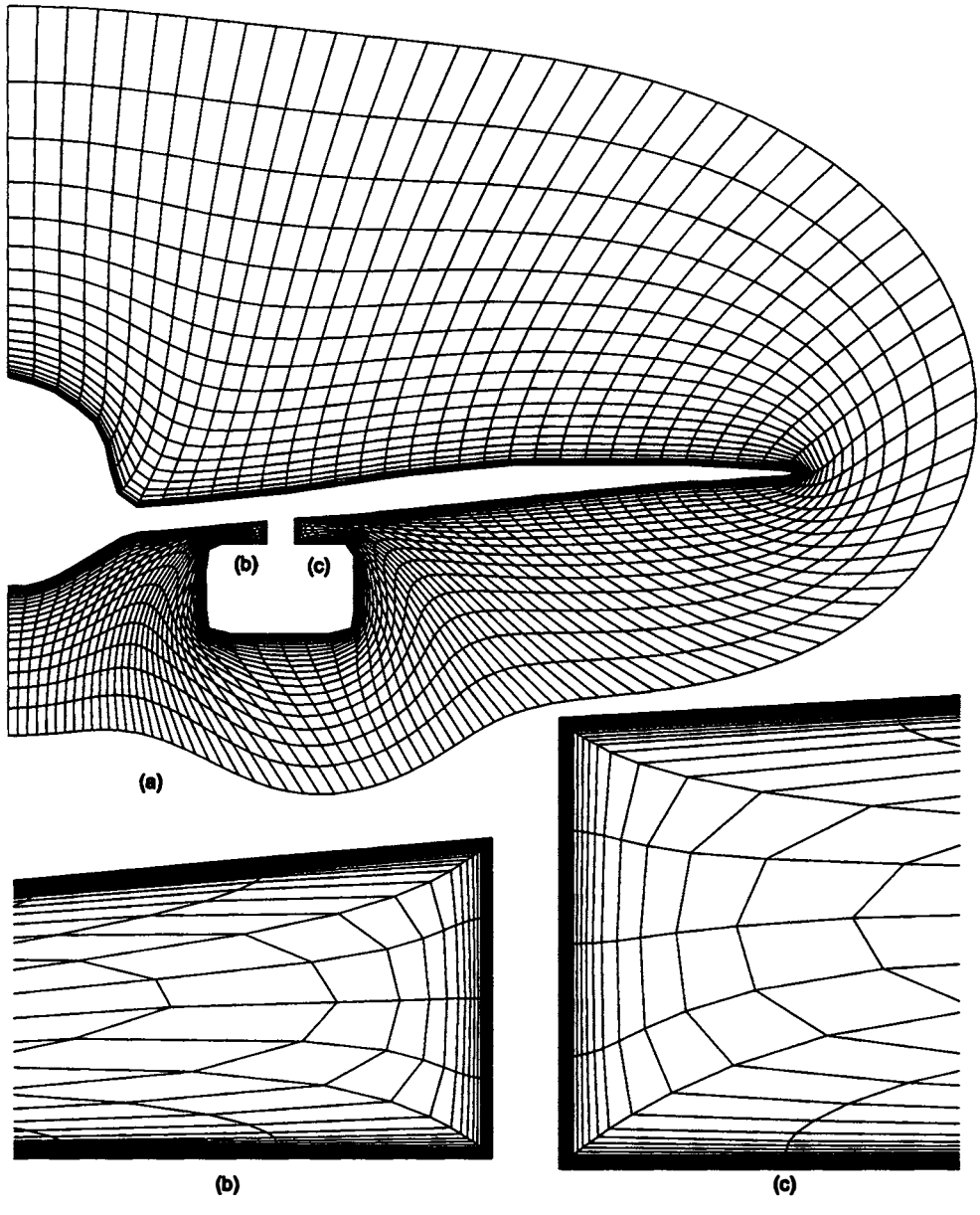


Figure 4.6: O-type 111×37 mesh around airplane contour (a) and close-up (b), (c). Due to symmetry only the right-half of the domain is shown.

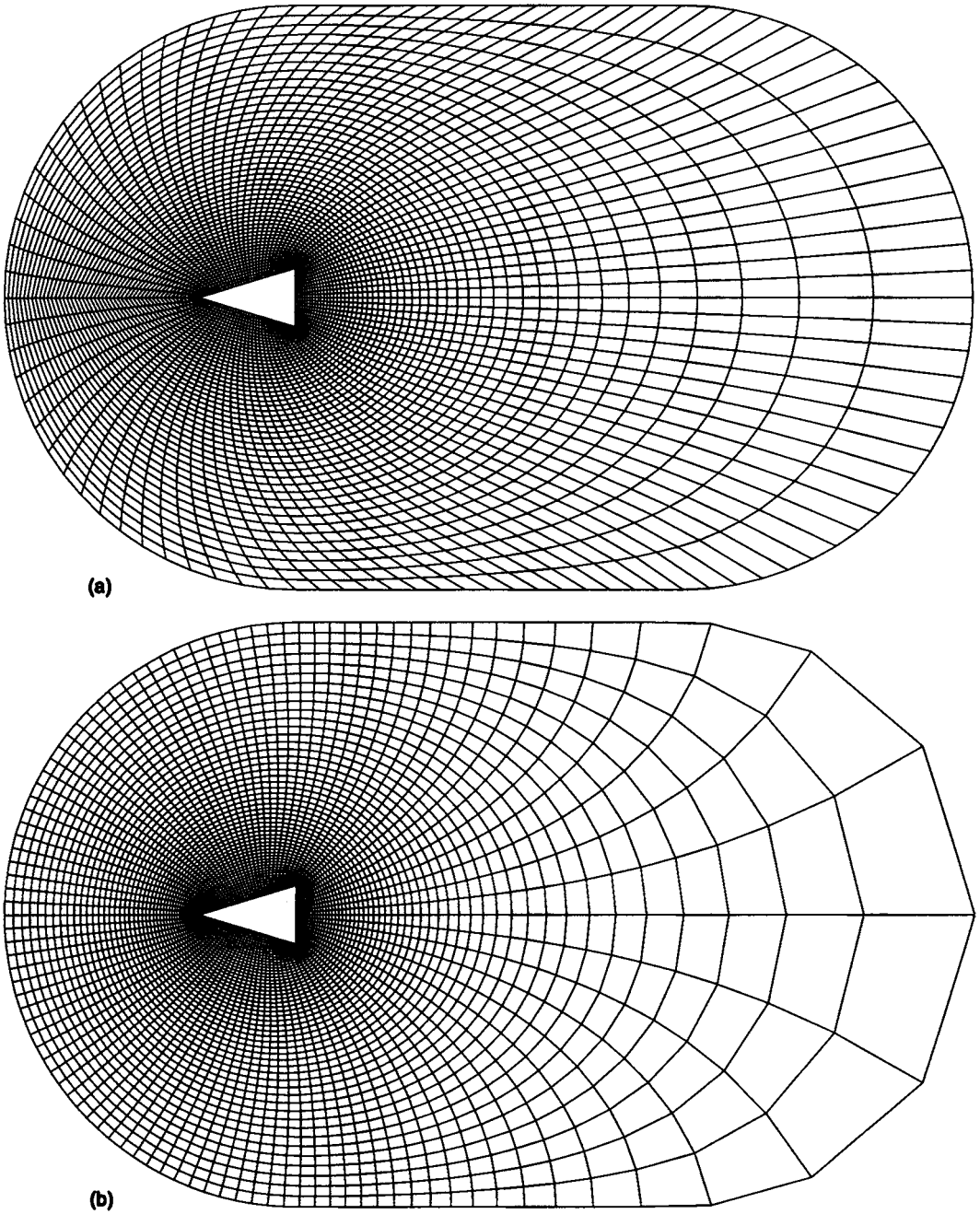


Figure 4.7: O-type 101×51 mesh with fixed boundary nodes (a) and quasi-conformal grid (b) generated with constrained minimization of boundary nodes.

is orthogonal (in the sense defined in Section 4.5) near the body.

4.12.3. Quasi-conformal mesh. We demonstrate the quasi-conformal mesh generation in the sense stated in Remark 2 of Section 4.9. Fig. 4.7 *a* presents the mesh constructed around a triangle with the variational barrier method. On the outer boundary (left and right outer boundary is a half-circle) the grid nodes are fixed. Next we employ the unconstrained minimization for the boundary nodes and obtain the mesh shown in Fig. 4.7 *b*, which is a discrete realization of the quasi-conformal mapping. Here the boundary nodes move along the axis of symmetry (boundary from the left grid end to the triangle left vertex) as well. The boundary nodes along the triangle contour are fixed, since in general, the conformal mapping near the internal boundary produces the mesh which may not be applicable for modeling. Meanwhile, in addition we can execute the mesh orthogonalization (with condensing if necessary) near the triangle contour as in the example of item 4.12.2.

4.12.4. Modeling detonation wave motion. Note that simultaneous employing the adaptation and Riemann metric G in the parametric square, i.e., using the functional (4.25), is possible only if when adapting the direction of grid nodes motion to the singularity of the solution (shock waves, slip lines, boundary layers, etc.) coincides (or nearly) with the direction into which we move the nodes using the second metric G. If these two mechanisms employ the nodes motion in different directions, the mesh distorts (remaining unfolded) and is not fit for modeling. This occurs due to the following. In [3], for the model example, it was shown that, when adapting to the discontinuity of the flow problem solution (without the second metric G), the discrete functional has no minimum and the nodes travel back and forth along the discontinuity providing grid lines condensing towards the singularity. If to apply an additional constrain by "switching on" the second metric G, this a "weak-stable" state is disrupted that causes distortion of the mesh. It does not mean that one method can substitute the other. For instance, to adapt the mesh to the boundary layer with width being smaller than the cell size, first, we should "catch" it by significant condensing the grid lines to this narrow zone, and to do it we use the functional (4.40). After several cells have been placed into the boundary layer, we "switch off" the second metric and "switch on" adaptation, i.e., employ the functional (4.25) with the Euclidean second metric G. Note that the latter turns into the functional of smoothness written on the surface of the control function f suggested in [20].

Example of the adaptive mesh, obtained when modeling the unstable detonation wave motion (for details see [4]), is presented in Fig.4.8. The pressure is used as a control function f in the functional (4.25) with the Euclidean metric G. To execute a stable mesh adaptation we employ the constrained minimization for the boundary nodes redistribution, i.e., minimize the functional (4.33).

In the second case, presented in Fig.4.9, we employ the shock-tracking procedure for the heating shock wave and apply the functional (4.40) to condense the grid lines towards the right moving boundary (being the heating shock) so as to resolve accurately the zone behind the wave. The height-to-width ratio in the boundary cells is achieved 10. Here we distribute the nodes along the moving right boundary employing the functional (4.33).

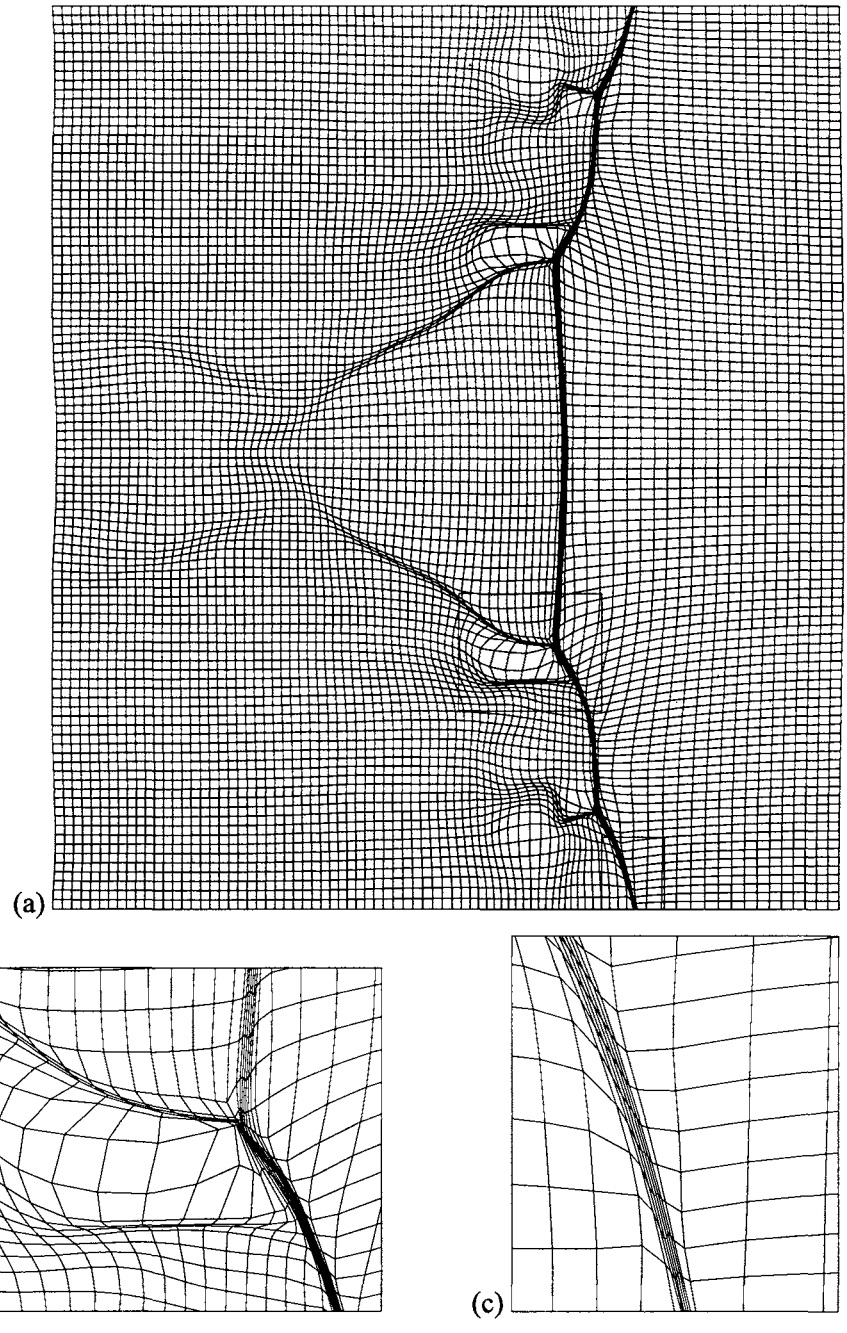
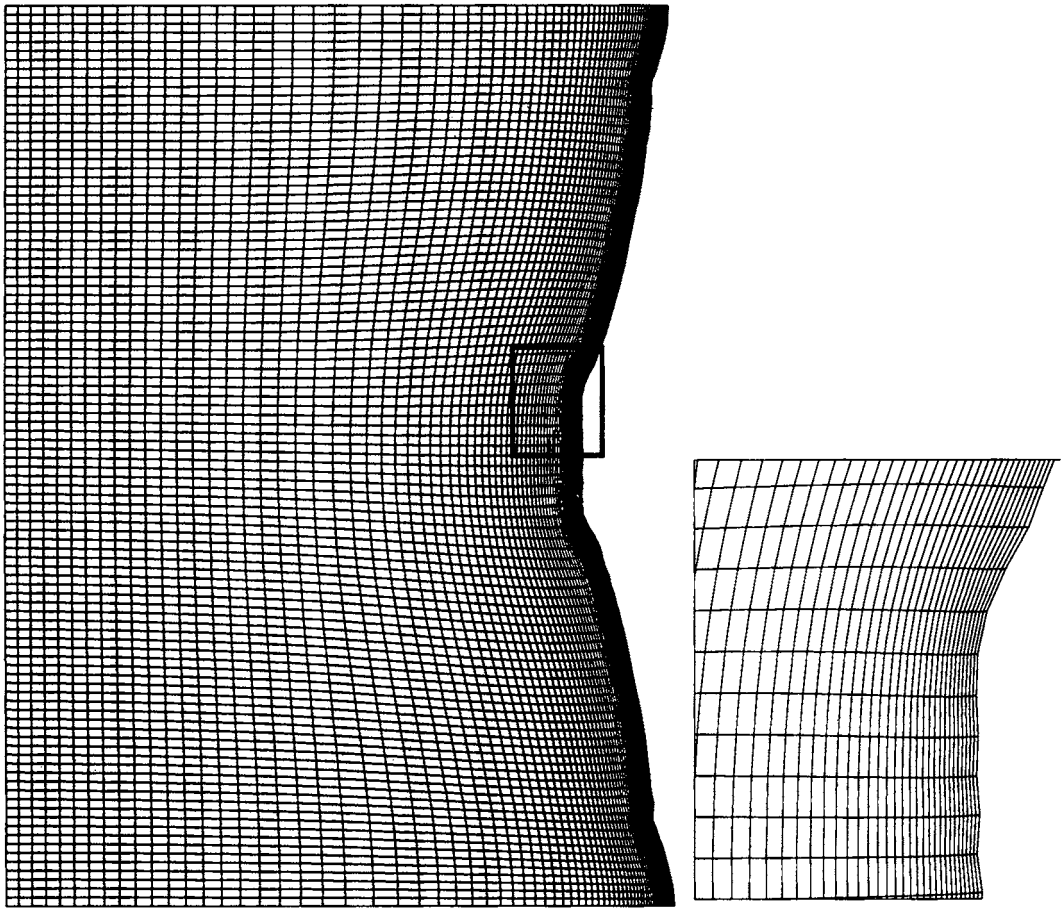


Figure 4.8: Adapted mesh (a) of 100 cells in height and close-up $(b), (c)$. Pressure is a control function f.



(a) (b)

Figure 4.9: Shock-tracking and condensing the grid lines to the shock using the metric G. Mesh (a) and close-up (b).

4.13 Energy Density in Three-Dimensional Case

Consider an extension of the energy density for the case of a linear mapping of 3D Euclidian spaces defined by different linear transformations of space (ξ, η, μ). Thus, we discuss the mapping $(X, Y, Z) \rightarrow (x, y, z)$, where $x(\xi, \eta, \mu)$, $y(\xi, \eta, \mu)$, $z(\xi, \eta, \mu)$ and $X(\xi, \eta, \mu)$, $Y(\xi, \eta, \mu)$, $Z(\xi, \eta, \mu)$ are two linear transformations with matrices c and C, respectively. The vector length squared x^2+y^2 and X^2+Y^2 is calculated using positive definite quadratic forms with the matrices $g=(g_{ij})=c^\top c$ and $G=(G_{ij})=C^\top C$, respectively. The quadratic form, corresponding to the mapping $(X, Y) \rightarrow (x, y)$, is defined by the matrix $(C^{-1})^\top c^\top c C^{-1}$. The characteristic equation is of the form

$$det[(C^{-1})^\top c^\top c C^{-1} - \lambda I] = det(C)^{-2} det(c^\top c - \lambda C^\top C) = 0 .$$

Hence, subject to the matrix C is invertible, the characteristic equation for the matrix $(C^{-1})^{\top}c^{\top}cC^{-1}$ possesses the same solutions λ_1, λ_2 as the characteristic equation of the pair of quadratic forms

$$det(g-\lambda G)=det(G)det(G^{-1}g-\lambda I)=0\,.$$

The orthogonal invariants d_1, d_2, and d_3 are written in terms of the eigenvalues $\lambda_{1,2,3}$ of the matrix $(G^{-1}g)$

$$d_1=\lambda_1+\lambda_2+\lambda_3=tr(G^{-1}g)=g_{11}G^{11}+2g_{12}G^{12}+2g_{13}G^{13}+g_{22}G^{22}+2g_{23}G^{23}+g_{33}G^{33}\,,$$

$$d_2=\lambda_1\lambda_2+\lambda_1\lambda_3+\lambda_2\lambda_3=det(G^{-1}g)tr(g^{-1}G)\,,$$

$$d_3=\lambda_1\lambda_2\lambda_3=det(G^{-1}g)=\frac{det(g)}{det(G)}\,.$$

Here G^{ij} are the elements of the matrix G^{-1}, inverse to G. The energy density of the mapping $(X,Y,Z)\rightarrow(x,y,z)$ is written in the form

$$e(x,y,z)(X,Y,Z)=\frac{1}{3}\frac{tr(G^{-1}g)}{\sqrt{det(G^{-1}g)}}$$
$$=\frac{1}{3}\frac{g_{11}G^{11}+2g_{12}G^{12}+2g_{13}G^{13}+g_{22}G^{22}+2g_{23}G^{23}+g_{33}G^{33}}{\sqrt{det(g)}/\sqrt{det(G)}}\,.$$

Here g^{ij} are the elements of the matrix g^{-1}, inverse to g. In contrast to the 2D case, the energy densities of the direct and inverse 3D mappings are not equal each other

$$e(x,y,z)(X,Y,Z)\neq e(X,Y,Z)(x,y,z)\,.$$

We write these values in terms of the eigenvalues $\lambda_{1,2,3}$ of the matrix $(G^{-1}g)$

$$e^{+}=e(x,y,z)(X,Y,Z)=\frac{1}{3}\frac{\lambda_1+\lambda_2+\lambda_3}{\sqrt{\lambda_1\lambda_2\lambda_3}}=\frac{1}{3}\frac{d_1}{\sqrt{d_3}}\,,$$

$$e^{-}=e(X,Y,Z)(x,y,z)=\frac{1}{3}\frac{\lambda_1\lambda_2+\lambda_1\lambda_3+\lambda_2\lambda_3}{\sqrt{\lambda_1\lambda_2\lambda_3}}=\frac{1}{3}\frac{d_2}{\sqrt{d_3}}\,.$$

It is easily seen that the inverse quantities $1/e^{+}$ and $1/e^{-}$ are the tetrahedron shape measures according to the definition in [21, 7].

In [18] it was suggested to use the condition number k of the matrix cC^{-1}, defined to be

$$k^2=\|cC^{-1}\|^2\|Cc^{-1}\|^2=tr(G^{-1}g)tr(g^{-1}G)=\frac{d_1d_2}{d_3}$$
$$=(\lambda_1+\lambda_2+\lambda_3)\left(\frac{1}{\lambda_1}+\frac{1}{\lambda_2}+\frac{1}{\lambda_3}\right)\,.$$

One can see that the condition number k is proportional to the geometric mean of the energy densities of the direct and inverse transformations, i.e., $k^2=9e^+e^-$. Using the inequality

$$3(\lambda_1\lambda_2+\lambda_1\lambda_3+\lambda_2\lambda_3)\le(\lambda_1+\lambda_2+\lambda_3)^2\,,$$

we can obtain the following estimates for the condition number

$$9(e^-)^{3/2}=\frac{\sqrt{3}(\lambda_1\lambda_2+\lambda_1\lambda_3+\lambda_2\lambda_3)^{3/2}}{\lambda_1\lambda_2\lambda_3}\le k^2\le\frac{(\lambda_1+\lambda_2+\lambda_3)^3}{3\lambda_1\lambda_2\lambda_3}=9(e^+)^3\,.$$

Consider the energy densities for tetrahedrons. Let a tetrahedron in space (x,y,z) be given such that one of the vertices r_0=0, the other vertices be defined by three vectors r_1, r_2, and r_3. Similarly a tetrahedron in space (X,Y,Z) is defined by R_0=0, R_1, R_2, and R_3. If the tetrahedron in (X,Y,Z) is formed by the unit orthogonal vectors, then the energy densities for the direct and inverse transformations are the following

$$e^+=\frac{1}{3}\frac{r_1^2+r_2^2+r_3^2}{V^{2/3}}\,,\quad e^-=\frac{1}{3}\frac{(r_1\times r_2)^2+(r_2\times r_3)^2+(r_3\times r_1)^2}{V^{4/3}}\,,$$

where $V=r_3\cdot(r_1\times r_2)$ is the triple tetrahedron volume. For the equilateral objective tetrahedron in space (X,Y,Z) it is easy to obtain that the energy density of the direct transformation is inverse to the mean ratio, defined in [7]

$$e^+=\frac{(r_1-r_2)^2+(r_2-r_3)^2+(r_1-r_3)^2+r_1^2+r_2^2+r_3^2}{12V^{2/3}}\,.$$

The energy density of the inverse transformation e^- is proportional to the ratio of the sum of areas squared of all faces over the volume in the corresponding power.

The energy density of a hexahedron can be defined similarly to the 2D case as a sum of 8 energy densities e^+ of the corner tetrahedrons. In this case, the procedure of minimizing the corresponding functional is analogous to that in 2D, only the expressions are more cumbersome [17]. This functional possesses a barrier property, associated with conditions of positive volumes of all corner tetrahedrons for all hexahedral grid cells.

4.14 Conclusion

The presented method allows us to obtain any mesh of a given structure. One only needs to know the shape of all grid cells. The variational principle, determining classes of one-to-one mappings, is formulated. This principle can be extended to the three-dimensional grids. However, to substantiate it in the continuous level, in a variational formulation one should prove existence of the class of one-to-one mappings of the cube onto an arbitrary 3D domain. Note, that in the 3D case it is unknown yet whether a harmonic map of an arbitrary domain onto a convex domain (unit cube) with a given one-to-one mapping between the boundaries is always a homeomorphism. There is an example (see [6]), when, subject

to some conditions imposed on the boundaries, the harmonic mapping is not a homeomorphism.

The method allows generating the mesh and its adapting to the solution of the flow problem.

It is of interest to extend this method to the 3D case. The way of its realization on the irregular tetrahedral grids is clearly seen. However, on the hexahedral mesh its direct application leads to rather cumbersome expressions for the functional and its derivatives. An additional study is required here.

Acknowledgment

This work was supported by the Russian Foundation for Basic Research, project 02–01–00236.

Bibliography

[1] Azarenok B.N. and Ivanenko S.A., Application of adaptive grids in numerical analysis of time-dependent problems in gas dynamics. *Comput. Math. Math. Phys.*, **40**(9), 2000, pp. 1330–1349.

[2] Azarenok B.N. and Ivanenko S.A. Application of moving adaptive grids for numerical solution of nonstationary problems in gas dynamics. *Intern. J. for Numer. Meth. in Fluids.*, **39**(1), 2002, pp. 1–22.

[3] Azarenok B.N. Variational barrier method of adaptive grid generation in hyperbolic problems of gas dynamics. *SIAM J. Numer. Anal.*, **40**(2), 2002, pp. 651–682.

[4] Azarenok B.N. and Tang T. Second-order Godunov-type scheme for reactive flow calculations on moving meshes. Submitted to *J. Comp. Phys.*, **206**(2005), pp. 48–80.

[5] Charakhch'yan A.A. and Ivanenko S.A. A Variational form of the Winslow grid generator. *J. Comp. Phys.*, **136**(2), 1997, pp. 385–398.

[6] Farrell F.T. and Jones L.E. Some non-homeomorphic harmonic homotopy equivalences. *Bull. London Math. Soc.*, **28**, 1996, pp. 177–182.

[7] Dompierre J., Labbe P., Guibault F. and Camarero R. Proposals of Banchmarks for 3D Unstructured Tetrahedral Mesh Optimization. *In Proceedings of the 7th International Meshing RoundTable'98*, 1998, pp. 459–478.

[8] Godunov S.K. and Prokopov G.P. On computation of conformal transformations and construction of difference meshes. *USSR Comput. Maths. Math. Phys.*, **5**(5), 1967, pp. 1031–1059.

[9] Godunov S.K., Zhukov V.T. and Feodoritova O.V. An algorithm for construction of quasi-isometric grids in curvilinear quadrangular regions. *In: Proceedings of the 16th Int. Conf. on Numerical Methods in Fluid Dynamics*, Arcachon, France, July 6-10, 1998, pp. 49–54.

[10] Handbook of Grid Generation. Eds. Thompson J.F., Soni B.K., and Weatherill N.P. CRC Press, Boca Raton, Fl., 1999.

[11] Ivanenko S.A. and Charakhch'yan A.A. Curvilinear grids of convex quadrilaterals. *USSR Comput. Maths. Math. Phys.*, **28**(2), 1988, pp. 126–133.

[12] Ivanenko S.A. Generation of non-degenerate meshes. *USSR Comput. Maths. Math. Phys.* **28**(5), pp. 141–146, 1988.

[13] Ivanenko S.A. Adaptive-Harmonic Grid Generation. Moscow, Computing Center of Russian Academy of Sciences, 1997, 182 p. (in Russian).

[14] Ivanenko S.A. Harmonic Mappings, *Chapt. 8 in Handbook of Grid Generation.* CRC Press, Boca Raton, Fl, 1999.

[15] Ivanenko S.A. Control of cell shape in the construction of a grid. *Comp. Math Math. Phys.* **40**(11), 2000, pp. 1662–1684.

[16] Ivanenko S.A. Existence of equations describing the classes of nondegenerate curvilinear coordinates in arbitrary domains. *Comp. Math Math. Phys.* **42**(1), 2002, pp. 43–48.

[17] Ivanenko S.A. Selected Chapters on Grid Generation and Applications. Moscow, Computing Center of Russian Academy of Sciences, 2004, 238 p.

[18] Knupp P.M. Matrix Norms and the Condition Number. A General Framework to Improve Mesh Quality via Node-Movement. *In: Proceedings of 8th International Meshing Roundtable*, South Lake Tahoe, CA, USA, 1999, pp. 13–22.

[19] Knupp P., Margolin L., and Shashkov M. Reference Jacobian optimization-based rezone strategies for arbitrary Lagrangian Eulerian methods, *J. Comput. Phys.*, **176**, 2002, pp. 93–128.

[20] Liseikin V.D. On generation of regular grids on n-dimensional surfaces, *USSR Comput. Maths. Math. Phys.*, **31**(11), 1991, pp. 47–57.

[21] Liu A. and Joe B. Relationship Between Tetrahedron Shape Measures, *Bit*, **34**, 1994, pp. 268–287.

[22] Shoen R. and Yay S.T. On univalent harmonic maps between surfaces. *Invent. Math.* **44**, f. 3, 1978, pp. 265–278.

[23] Spekreijse S.P. Elliptic grid generation based on Laplace equations and algebraic transformations. *J. Comp. Phys.*, **118**, pp. 28–61, 1995.

[24] Tiniko-Ruiz J.G. and Barrera-Sanchez P. Area Functionals in Plane Grid Generation. *In: Proceedings of the 6th International Conference on Numerical Grid Generation in Computational Field Simulation*, University of Greenwich, July 6-9, pp. 293–302, 1998.

[25] Thompson J.F., Warsi Z.U.A., and Mastin C.W. Numerical grid generation. North-Holland, N.Y. etc. 1985.

[26] Winslow A. Numerical solution of the quasi-linear Poisson equation in a nonuniform triangle mesh, *J. Comput. Phys.*, **1**(2), 1966, pp. 149–172.

Chapter 5

MOVING MESH CALCULATION IN UNSTEADY TWO-DIMENSIONAL PROBLEMS

Genadii P. Prokopov
Keldysh Institute of Applied Mathematics

Variational functionals for calculating two-dimensional difference grids, applied when solving unsteady problems of mathematical physics with moving boundaries, are suggested. Their coefficients are defined by the metric parameters of the grid, obtained at the preceding time step, and corrected additionally using other variational functionals taken with weight coefficients, being proportional to the time step. Discretization of the functionals and iteration procedures for the obtained system of equations are considered.

5.1 Introduction

We accentuate that major scientific results of the present paper was published in [1, 2]. Nevertheless, this publication is of interest since readers, not knowing Russian, may not be informed of it. Besides, the present work includes additional information and gives more clear accents to some peculiarities of the problem. The present paper does not pretend to a review and contains only the references required by the essence of the questions discussed.

5.1.1. Our interest to the grid generation problems arose many years ago during the work headed by S.K. Godunov (at present Academician) on developing algorithms for solving two-dimensional unsteady gas dynamics problems. One result of that work was the monograph [3]. In the introduction of [3] they noted the substantial role of applying moving and deforming grids, what was connected with contact (slip) boundaries, shock waves, etc. tracked when modeling.

Since grid generation is not a final purpose and only a means for subsequent solving rather complicated physical problems, we were eager towards the simplest mesh structure in the point of view of arranging the grid cells. When computing a problem in a domain with complex geometry, it is divided into several subdomains. Next, in every subdomain we generate a regular mesh consisting of quadrilateral cells, which (mesh) looks like a crooked

rectangular (or rather like a parallelogram) mesh. Under such an approach, cell numeration is quite simple (like numeration of matrix elements by strings and columns). Such grids we refer to as regular.

5.1.2. In the simplest realization (cf. [3], § 23) one family of the grid lines can be the system of the straight-line segments, being fixed when modeling, and the mesh movement is provided by motion of the second family of the grid lines (nodes on the domain boundaries along those straight-line segments). In every segment of the first family the node coordinates are obtained by some distribution laws. These distribution laws can be either conserved invariable (e.g. when using the simplest uniform nodes redistribution) or controlled via some parameters (including dependence on time). Joining the adjacent nodes (by numbers of the first family segments), we generate the second family of the grid lines.

In unsteady gas dynamics problems the domain boundary moves by virtue of physical laws and concrete distribution of the physical parameters near the boundary at the underlying time. Thus, in general the boundary nodes redistribution is performed within separated stage (usually before interior nodes redistribution).

5.1.3. One more simplest and effective grid generation algorithm is based on applying explicit formulas, like interpolation ones, which allow for computing the interior node coordinates using known coordinates of the boundary nodes (see details in Section 5.7).

In the case of arbitrary domains these formulas can give unsatisfactory results, because we can not cut the domain without folds and self-intersection of the grid cells. Violation of such a regular structure we refer to as grid degeneration.

5.1.4. A regular grid generation problem can be considered as a discrete realization of an invertible (univalent) mapping of the parametric domain onto the physical (countable) domain defined by its boundaries at the underlying time.

Within long time many authors were suggesting approaches and algorithms for solving this problem. Unfortunately, rather seldom their substantiation was brought to the level of a clear formulated mathematical problem assuming existence and uniqueness of the solution.

5.1.5. It may be of interest that the matter on existence of such algorithms was always beyond any doubt. Really, we know from the theory of functions of complex variables that the Riemann theorem states: "There is a conformal mapping of an arbitrary planar simply connected domain, bounded by a piecewise smooth curve, onto another analogous domain. This mapping is defined uniquely if to specify a correspondence between three arbitrary boundary points (observing the bypass direction to the boundaries)".

Let us assign four "corner" points on the boundary of the simply connected domain, i.e. we define it as a curvilinear quadrilateral. Then there is (and undoubtedly unique) a conformal mapping of a rectangular with some sides ratio onto this domain. Under this mapping four corners of the rectangular correspond to four "corners" of the domain. The sides ratio is a conformal invariant of the curvilinear quadrilateral, it is unknown beforehand and is to be determined during calculation. Besides, it must be defined a one-to-one correspondence between given points on the domain boundary and strictly given points on the contour of the parametric rectangular.

We introduce an additional dilatation of the coordinates so as to make consistent dimen-

sions of the conformal rectangular and parametric one, defined by number of intervals along two space variables (or so as to transform it into the canonical unit square). Superposition of two mappings is one of possible solutions of the underlying problem in the differential form.

Images of the straight-line segments in the parametric rectangular, joining mentioned points at the opposite sides and corresponding to the given points on the physical domain boundary, can be taken as the grid lines. Intersection points between the lines of two families are the grid nodes sought (more precisely, one of possible examples of the grid).

5.1.6. Direct realization of the above idea was described in [4]. Its main advantage is the theoretical substantiation. Unfortunately, in the point of view of practical applications there were found two substantial drawbacks. The first is complexity of the numerical algorithm and large number of numerical operations when realizing. It was rather serious with a weak power of computers. Secondly, at discrete realization, with a concretely defined number of nodes in the case of "strongly bent" domain boundaries, the generated mesh may be unfit to simulation. In general the grid is not orthogonal. This is because, when fixing the boundary nodes, the corresponding grid in the parametric rectangular is not orthogonal, and the conformal mapping preserves the angles between the grid lines. Meanwhile, generation of the orthogonal or maximal close to them (i.e. "quasi-orthogonal") grids was users' requirement. First of all it concerned the grids closed to polar, especially with nonuniform redistribution by the angle and radius.

Users' requirement could be realized in this algorithm as well if to allow node "free flow" along two boundaries of the domain (the algorithm is significantly simplified if to allow node "free flow" along all four boundaries). However, for the sake of the main problem we refused of such a formulation in grid generation.

5.1.7. In some sense the publication of [4] was stimulated by appearing of [5], where with the purpose of grid construction it was used a nonlinear system of elliptic equations, being the invertible Laplace equations.

Our attempts, preceding the work [4], to avoid complexity, connected with nonlinearity of the equations, by using the direct Laplace equations and their generalization failed. Later, the respective analytical examples were suggested (of course, it does not prevent such algorithms from applying in simple cases).

At a later time the idea that in grid construction we can use the mapping, sought as a solution of the system of elliptic equations, was widely extended. The most natural realization of the variational approach is the use of the finite element method.

5.1.8. In one Russian folk fairy tale the hero is ordered: "Go there nobody knows where and bring that nobody knows what". In grid problems we meet an analogy with that case. Uncertainty of requirements imposed upon the grid, except one, that is nondegeneracy, gives an illusion that the generation algorithms can be rather arbitrary. Indeed, the user make a choice within his restricted possibilities. The selected method must provide with an unique solution. Otherwise there is no a chance that it will succeed when realizing on a computer.

Indeed, for solving the main problem it is of interest the mesh providing the minimal

error of the solution sought in a flow problem. These meshes are referred to as adaptive (or adapted), and there is a lot of works where these grids are obtained. However, in the case of the domain with a complex boundary, the problem of regular grid generation is rather difficult. Adaptation towards the solution of a concrete problem can be partially achieved by a special boundary nodes redistribution. Thus, as noted above, we refuse of freedom in boundary nodes redistribution for the sake of the grid generation algorithms.

5.1.9. If we consider the grid generation problem in a given domain with fixed boundaries, there are several attempts (if necessary) to obtain the mesh. Looking at the mesh even at the level "I like it or don't" and having the set of grid algorithms, the user may hope that a suitable mesh can be produced. This way does not work when solving unsteady problems with moving boundaries, where the mesh should be generated at every time step and number of time steps can achieve several hundred thousand and even millions.

Deformation of the moving boundaries is hardly predictable beforehand. Even if we can foresee (i.e., using preliminary rough calculations), then the initially successfully selected mesh algorithm can lead to undesirable or even catastrophic ("break") results during modeling.

This results in necessity to change something. Even if a new solution is successful, the transition process is very complicated due to the high speed of the grid nodes, which should switch from one grid equations to others (even if the equations remain the same and we only vary their parameters).

One more difficulty, connected with a high grid node velocity, is that in general the iteration processes do not attain convergence, because it is not realistic and not necessary. This means that the residual of the grid equations is not small enough. Meanwhile, the node velocity depends (directly or indirectly) on the residual value. Hence, if in the main flow problem the time step is significantly reduced (independently by a physical reason or it is "grid's" fault), the node velocity is steeply increased and next it follows the "chain reaction" of failures causing a break of modeling.

The presented approach should solve the above problems connected with regulation of the grid node velocity.

The concrete form of the coefficients to the variational functional, suggested in [6], plays the substantial role in our study. It is of universal character, since any unfolded grid can be produced if to specify properly these parameters. Foundation of the discrete model to this functional is presented in [7] (see also chapter 4).

To describe algorithms we use the simplest variant of the regular mesh, specified by two indices, in an isolated domain which can be considered as a quadrilateral with curvilinear boundaries. Since the described algorithms are applied in local subdomains, obviously, they can be used in an arbitrary domain (or "map") consisting of these subdomains.

5.2 Variational Functionals

5.2.1. In the above sense the grid generation problem can be performed as a discrete realization of the invertible mapping of the unit square Q: $(0 \leq \xi, \eta \leq 1)$ onto a given domain Ω

in the plane of the variables (x, y).

With this purpose we use the Ivanenko's functional [6]

$$F = \int_0^1 \int_0^1 E d\xi d\eta \,, \tag{5.1}$$

where the integrand E, referred to as the energy density of the mapping, is

$$E = \frac{g_{11}G_{22} - 2g_{12}G_{12} + g_{22}G_{11}}{2g_o\sqrt{G_{11}G_{22} - G_{12}^2}}. \tag{5.2}$$

The elements of the symmetric and positive defined matrix

$$g = \begin{Vmatrix} g_{11} & g_{12} \\ g_{12} & g_{22} \end{Vmatrix}$$

are the metric parameters of the mapping

$$\begin{aligned} g_{11} &= x_\xi^2 + y_\xi^2 \,, \qquad g_{12} = x_\xi x_\eta + y_\xi y_\eta \,, \\ g_{22} &= x_\eta^2 + y_\eta^2 \,, \qquad g_o = x_\xi y_\eta - y_\xi x_\eta \,. \end{aligned} \tag{5.3}$$

Invertibility of the mapping and positive definiteness of the matrix g are interconnected by virtue of the identity

$$g_{11}g_{22} - g_{12}^2 \equiv g_o^2 \,. \tag{5.4}$$

value G_{11}, G_{12}, G_{22} are the elements of the analogous symmetric and positive defined matrix G, given at every point of the unit square Q.

The functional is minimized in the class of the functions $x(\xi, \eta)$, $y(\xi, \eta)$, being the smooth extension of the functions x_b,y_b, given on the boundary, inwards the square Q. The latter execute the smooth one-to-one mapping of the square Q boundary onto the boundary of the domain Ω, where the grid is to be generated.

One can readily prove that $E \geq 1$. The proof is presented in [6] and below in Section 5.8. Therefore, the absolute minimum of the functional is $F_{min} = 1$.

5.2.2. Let $x(\xi, \eta)$, $y(\xi, \eta)$ be an arbitrary smooth invertible mapping of the square Q onto the domain Ω and

$$G_{11} = g_{11} \,, \quad G_{22} = g_{22} \,, \quad G_{12} = g_{12} \,. \tag{5.5}$$

Then, by virtue of the identity (5.4), we obtain $E \equiv 1$, $F = 1$. Hence, this arbitrary mapping is realized as a minimization problem for the functional (5.1)-(5.2) with matrix (5.5).

It is convenient to introduce a matrix $\tilde{G}$ with normalized elements

$$\tilde{G}_{11} = G_{11}/G_o \,, \quad \tilde{G}_{12} = G_{12}/G_o \,, \quad \tilde{G}_{22} = G_{22}/G_o \,, \tag{5.6}$$

where the normalizing factor G_o is

$$G_o = \sqrt{G_{11}G_{22} - G_{12}^2} \,. \tag{5.7}$$

Consider a variational functional differing from (5.1) by the integration domain. Here, instead of the parametric square, we integrate over the countable domain Ω

$$F^* = \iint\limits_{\Omega} E dx dy = \iint\limits_{Q} E^* d\xi d\eta \,. \tag{5.8}$$

Since $dxdy = (x_\xi y_\eta - y_\xi x_\eta) d\xi d\eta = g_o d\xi d\eta$, we obtain the energy density of the mapping E^*

$$E^* = \frac{1}{2}(g_{11}\tilde{G}_{22} - 2g_{12}\tilde{G}_{12} + g_{22}\tilde{G}_{11}) \,. \tag{5.9}$$

Thus, the functional (5.8)-(5.9) differs from (5.1)-(5.2) by absence of the Jacobian of the mapping in the denominator. This decision was made by the authors in [4], where the functional (5.8)-(5.9) was suggested intentionally so as to simplify the problem of solving the Euler-Lagrange equations.

If to define the metric parameters G according to (5.5), then minimizing the functional (5.8)-(5.9) we obtain the absolute minimum of F^* equal to the domain Ω area.

Consequently, defining (5.5) and applying the functional (5.8)-(5.9) we can construct any mapping as well. Thus, the functionals F and F^* are the universal grid generators.

5.2.3. In particular, if to set

$$G_{11} = G_{22} = 1 \,, \quad G_{12} = 0 \,, \tag{5.10}$$

then the harmonic mappings and corresponding harmonic grids are realized. In the functional F the energy density of the mapping is

$$E = \frac{g_{11} + g_{22}}{2g_o} \,,$$

and in the functional F^* it is

$$E^* = \frac{g_{11} + g_{22}}{2} \,.$$

As variational Euler-Lagrange equations for the functional F^* we obtain the Laplace equations

$$x_{\xi\xi} + x_{\eta\eta} = 0 \,, \quad y_{\xi\xi} + y_{\eta\eta} = 0 \,. \tag{5.11}$$

After transformation, we obtain the inverse Laplace equations as variational Euler-Lagrange equations for the functional F

$$g_{22}x_{\xi\xi} - 2g_{12}x_{\xi\eta} + g_{11}x_{\eta\eta} = 0 \,, \quad g_{22}y_{\xi\xi} - 2g_{12}y_{\xi\eta} + g_{11}y_{\eta\eta} = 0 \,. \tag{5.12}$$

It is agreed that the equations (5.12) were used in [5] in grid generation for the first time.

We constructed an example to the equations (5.11), where in the simple domain they produce a degenerate mapping, see Section 5.8. Thus, at arbitrary definition of the matrix $G(\xi, \eta)$ the functional (5.8)-(5.9) does not guarantee invertibility of the mapping.

In turn, in [7] (see also chapter 4) it is proved in the discrete form that the functional (5.1)-(5.2) provides an invertible mapping with arbitrary positive defined matrices $G(\xi,\eta)$. It is a substantial advantage of this functional. However, the functional (5.8)-(5.9) possesses a number of advantages over the functional (5.1)-(5.2). Thus, it is worth considering the both functionals.

For the sake of clearness we will refer (5.1)-(5.2) to as the functional with the Jacobian and (5.8)-(5.9) to as the functional without the Jacobian.

5.2.4. The variational Euler-Lagrange equations for the functional without the Jacobian are

$$\begin{aligned}
&\frac{\partial}{\partial\xi}\left(\tilde{G}_{22}\frac{\partial x}{\partial\xi}-\tilde{G}_{12}\frac{\partial x}{\partial\eta}\right)+\frac{\partial}{\partial\eta}\left(\tilde{G}_{11}\frac{\partial x}{\partial\eta}-\tilde{G}_{12}\frac{\partial x}{\partial\xi}\right)=0\,, \qquad (5.13)\\
&\frac{\partial}{\partial\xi}\left(\tilde{G}_{22}\frac{\partial y}{\partial\xi}-\tilde{G}_{12}\frac{\partial y}{\partial\eta}\right)+\frac{\partial}{\partial\eta}\left(\tilde{G}_{11}\frac{\partial y}{\partial\eta}-\tilde{G}_{12}\frac{\partial y}{\partial\xi}\right)=0\,.
\end{aligned}$$

We will denote it as the system $S^*[G]$. It is a linear system with given $G{=}G(\xi,\eta)$. Analogous and more cumbersome nonlinear system can be derived for the functional with the Jacobian. As its consequence in [7], it was derived nonlinear equations generalized (5.12) to the arbitrary matrices $G(\xi,\eta)$. For the completeness, we write them as well in our notations

$$\begin{aligned}
&g_{22}x_{\xi\xi}-2g_{12}x_{\xi\eta}+g_{11}x_{\eta\eta}=A\cdot x_\xi+B\cdot x_\eta\,, \qquad (5.14)\\
&g_{22}y_{\xi\xi}-2g_{12}y_{\xi\eta}+g_{11}y_{\eta\eta}=A\cdot y_\xi+B\cdot y_\eta\,,
\end{aligned}$$

where

$$A=-P\tilde{G}_{22}+Q\tilde{G}_{12}\,,\quad P=\frac{1}{2}\left(g_{22}\frac{\partial\tilde{G}_{11}}{\partial\xi}-2g_{12}\frac{\partial\tilde{G}_{12}}{\partial\xi}+g_{11}\frac{\partial\tilde{G}_{22}}{\partial\xi}\right),$$

$$B=-P\tilde{G}_{12}+Q\tilde{G}_{11}\,,\quad Q=\frac{1}{2}\left(g_{22}\frac{\partial\tilde{G}_{11}}{\partial\eta}-2g_{12}\frac{\partial\tilde{G}_{12}}{\partial\eta}+g_{11}\frac{\partial\tilde{G}_{22}}{\partial\eta}\right).$$

We will denote it as the system $S[G]$.

Relative simplicity and linearity of the variational equations (5.13) in comparison with (5.14) is an substantial advantage of the functional without the Jacobian.

5.3 Unsteady Grid Problem

5.3.1. Now we turn to the matter of grid generation when solving unsteady problems with moving boundary.

We introduce the following notations: N is the time step number, $\tau=\tau^{(N)}$ is the time step, $\boldsymbol{r}^{(N-1)}$ is the grid, generated at the preceding time step. The vector $\boldsymbol{r}$ includes the coordinates (x,y) of all grid nodes.

For the sake of definiteness consider a gas dynamics problem. Accordingly to the splitting scheme, calculation of the next time step includes several stages.

At the 1st stage the boundary of the countable domains is shifted. Its motion is caused by the physical laws and distribution of the physical quantities in the boundary vicinity at the underlying time. Suppose that the sequences $\boldsymbol{r}_{\mathrm{b}}^{(N-1)}$ define the boundary location at the preceding time step and turn into the sequences $\boldsymbol{r}_{\mathrm{b}}^{(N)}$ at the present time step. We can assume that

$$|\boldsymbol{r}_{\mathrm{b}}^{(N)} - \boldsymbol{r}_{\mathrm{b}}^{(N-1)}| = O(\tau)\,. \tag{5.15}$$

Let us agree that (5.15) and analogous equalities are fulfilled to each coordinate of the vector $\boldsymbol{r}$.

By reasons, stated in [8], at the 2nd stage we determine the shift of the boundary nodes

$$\triangle \boldsymbol{r}_{\mathrm{b}}^{(N)} = \boldsymbol{r}_{\mathrm{b}}^{(N)} - \boldsymbol{r}_{\mathrm{b}}^{(N-1)}\,, \tag{5.16}$$

and then, by interpolation formulas, compute the shift of the internal nodes, and obtain an initial guess

$$\boldsymbol{r}^{0(N)} = \boldsymbol{r}^{(N-1)} + \triangle \boldsymbol{r}^{(N)}\,, \tag{5.17}$$

to the nodes at the Nth time step. Boundary nodes take a correct location, which does not change at the next stages of this time step.

Interpolation formulas will be presented in Section 5.7 and it will be clear that

$$|\boldsymbol{r}^{0(N-1)} - \boldsymbol{r}^{(N-1)}| = |\triangle \boldsymbol{r}^{(N)}| = O(\tau)\,. \tag{5.18}$$

5.3.2. Now consider the 3d stage, definition of the matrix G coefficients to the above functionals. We suggest to use the metric parameters of the grid from the preceding time step

$$G = G^{(N)} = G[\boldsymbol{r}^{(N-1)}]\,. \tag{5.19}$$

As noted in Section 1, this definition reproduces the grid of the preceding step, i.e.

$$\begin{aligned} S[G^{(N)}](\boldsymbol{r}^{(N-1)}) &\equiv 0 \quad \text{for the functional (5.1)-(5.2)}\,, \\ S^*[G^{(N)}](\boldsymbol{r}^{(N-1)}) &\equiv 0 \quad \text{for the functional (5.8)-(5.9)}\,. \end{aligned} \tag{5.20}$$

By virtue of (5.18), we may assume that

$$S[G^{(N)}](\boldsymbol{r}^{0(N)}) = 0(\tau)\,, \tag{5.21}$$

and similarly to the second functional.

Strictly speaking one needs to formulate requirements to be imposed upon smoothness of the domain boundary and grid, etc. We omit it, more over that these requirements should be formulated to the discrete variant, which will be described in Section 5.4.

Under well-conditionality of the system of equations we assume that the sought grids $\boldsymbol{r}^{(N)}$, determined from the equations

$$S^*[G^{(N)}](\boldsymbol{r}^{(N)}) = 0 \quad \text{or} \quad S[G^{(N)}](\boldsymbol{r}^{(N)}) = 0\,,$$

to satisfy the condition

$$|\boldsymbol{r}^{(N)} - \boldsymbol{r}^{(N-1)}| = O(\tau)\,. \tag{5.22}$$

This would solve completely the grid generation problem. Unfortunately, in practical calculations it is not so. When defining the coefficients by (5.19), the functionals exhibit "indifference" to the "gird fate", because to *any* grid the functional value is close to the absolute minimum, differing from it only by a value of $O(\tau)$ due to deflection of the boundary conditions (5.15). If there is no a feedback, deterioration of the grid quality can lead to irreversible consequences, including break. Therefore, definition (5.19) should be corrected.

5.3.3. One way of correction reflects the eagerness to make the grid closer to "quasi-orthogonal".

We suggest a substitution for the functional (5.1)-(5.2) with the Jacobian

$$\tilde{E} = \frac{(1+p_o\tau)(g_{11}G_{22}+g_{22}G_{11}) - 2g_{12}G_{12}}{2g_oG_o} = E + p_o\tau E_o\,,$$

$$E_o = \frac{g_{11}G_{22}+g_{22}G_{11}}{2g_oG_o}\,. \tag{5.23}$$

For the functional (5.8)-(5.9) without the Jacobian, the analogous substitution is

$$\tilde{E}^* = E^* + p_o\tau E_o^*\,, \quad E_o^* = \frac{g_{11}G_{22}+g_{22}G_{11}}{2G_o}\,. \tag{5.24}$$

Here $p_o > 0$ is the control parameter.

To realize (5.23) or (5.24) it is sufficient to change the normalization formulas (5.6)-(5.7)

$$\tilde{G}_{11} = \frac{G_{11}(1+p_o\tau)}{G_o}\,, \quad \tilde{G}_{22} = \frac{G_{22}(1+p_o\tau)}{G_o}\,, \quad \tilde{G}_{12} = \frac{G_{12}}{G_o}\,, \tag{5.25}$$

holding $G_o = \sqrt{G_{11}G_{22} - G_{12}^2}$ and definition (5.19).

It may be of interest to note that, when defining the correcting functionals by (5.5), the energy densities E_o and E_o^* take the form

$$E_o = \frac{g_{11}g_{22}}{g_o^2}\,, \quad E_o^* = \frac{g_{11}g_{22}}{g_o}\,. \tag{5.26}$$

The first was suggested in [9] as a functional of "orthogonality" and the second in [10] as a variant for obtaining "quasi-orthogonal" grids. Both cases produce incorrectness when realizing, which was overcame by introducing "regularizing" means. In the suggested variant as such means we use the "support" functionals (5.1)-(5.2) or (5.8)-(5.9) which they are added to with the coefficient $p_o\tau$.

In general, for correcting the support functionals we suggest to apply arbitrary "good functionals" with control coefficients $p\tau$. The term "good functionals" means well-conditionality of the corrected functional and influence upon the mesh to provide its quality suitable for solving the main problem. We pay attention to concordance of the functionals dimensionality with respect to the variables (x, y), see details in Section 5.8.

Note that, considering dimensionality of the variable t, instead of $p\tau$ it is convenient to define the weight coefficients as

$$p(\tau) = \frac{p^*\theta(1+p^*)}{1+p^*\theta}\,, \tag{5.27}$$

where $\theta = \tau^{(N)}/\tau^{(N-1)}$.

In particular, the use of the functional, producing the harmonic meshes, with the coefficient $p_h(\tau)$ is executed by applying

$$\tilde{G}_{11} = \frac{G_{11}(1+p_o(\tau))}{G_o} + p_h(\tau)\,, \quad \tilde{G}_{12} = \frac{G_{12}}{G_o}\,, \tag{5.28}$$
$$\tilde{G}_{22} = \frac{G_{22}(1+p_o(\tau))}{G_o} + p_h(\tau)\,, \quad G_o = \sqrt{G_{11}G_{22} - G_{12}^2}\,,$$

instead of (5.25).

5.4 Discretization

5.4.1. The next stage in the grid generation algorithm is transition from the differential form of the variational functional to discrete. The simplest and efficient way is to change the differential Euler-Lagrange equations (5.13) of the functional (5.8)-(5.9) and consequences (5.14) of the functional (5.1)-(5.2) by difference ones. However, opinion that this stage is reduced to mechanical (formal) substitution of the differential expressions by difference is incorrect. When approximating, some important properties of the differential model can be lost.

In the problem considered, it is very important that the *discrete model* provides the conditions (5.20), i.e. the *residual* of the difference equations (appeared after the coefficients G have been defined via (5.19), when substituting the grid of the preceding step into these equations) must be *identically equal to zero*. Violation of this requirement, in the hope that an approximation error will be rather small, can lead to uncontrolled value of the node velocity. A special procedure, well-known as a variational barrier method, guarantees that this purpose can be achieved. It was suggested in [11] for constructing the harmonic grids. We will set forth it for completeness with additional remarks.

The discrete analog of the functional (5.1)-(5.2) is written in the following form

$$F^h = \sum_n \sum_m E_{n+1/2,m+1/2}\,, \tag{5.29}$$

where summation is performed over all grid cells of the countable domain. In general, one uses half-integer indices $n+1/2, m+1/2$. The normalizing factor, number of cells, is omitted in (5.29), since it does not influence upon the result.

The difference equations to the grid node with number (n, m) are

$$\frac{\partial F^h}{\partial x_{n,m}} = 0\,, \quad \frac{\partial F^h}{\partial y_{n,m}} = 0\,. \tag{5.30}$$

Equations (5.30) can be derived if we consider the "stencil" of four cells, attached to this node. It is depicted in Fig. 5.1. For simplicity of the formulas, we introduce a short notation to the node and cell number, see Fig. 5.1. The cells are enumerated as $L=1,\ldots,4$. Then, the equations (5.30) are written as follows

$$\left(\sum_{L=1}^{4}\frac{\partial E_L}{\partial x_1}\right)_{n,m}=0\,,\qquad \left(\sum_{L=1}^{4}\frac{\partial E_L}{\partial y_1}\right)_{n,m}=0\,. \tag{5.31}$$

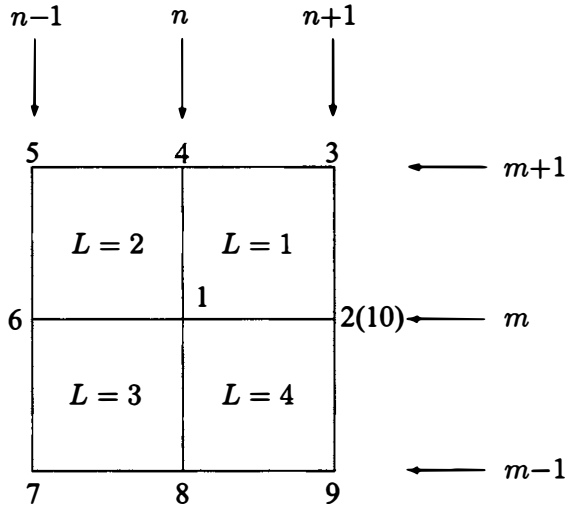


Figure 5.1: Stencil for derivation of equations in node (n,m).

In turn, every Lth cell, with vertices $(1, 2L, 2L{+}1, 2L{+}2)$, is divided into two pairs of triangles by drawing the diagonals $(1, 2L{+}1)$ and $(2L, 2L{+}2)$. The obtained triangles are enumerated as $k=1,\ldots,4$, see Fig. 5.2, with correspondence to the cell vertices. Vertices of the kth triangle are enumerated as $k-1, k, k{+}1$, see Fig. 5.3*a*. The other vertices are enumerated by analogy. Next we can write

$$E_L=\frac{1}{4}\sum_{k=1}^{4}E_{k,L}\,,\qquad E_{k,L}=\frac{E^*_{k,L}}{(g_o)_{k,L}}\,. \tag{5.32}$$

$$E^*_{k,L}=\frac{1}{2}\Big[(g_{11})_k(\tilde G_{22})_k-2(g_{12})_k(\tilde G_{12})_k+(g_{22})_k(\tilde G_{11})_k\Big]_L\,. \tag{5.33}$$

Hereinafter the index (k, L) denotes that the corresponding value relates to the kth triangle of the Lth cell.

It is important to note that in (5.29) we assume that

$$(g_o)_{k,L}>0\quad\text{for all}\quad k, L\,. \tag{5.34}$$

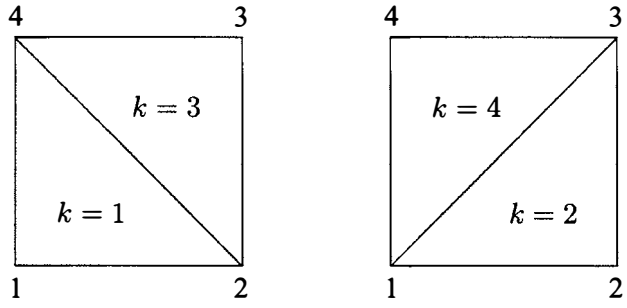


Figure 5.2: Cell partitioning into two pairs of triangles.

Further it will be clear that the value $(g_o)_{k,L}$ is the double area of the corresponding triangle. Therefore, the requirement (5.34) means that all grid cells should be convex quadrilaterals. In practical calculations this requirement is weakened, see discussion in Section 5.6. Now we assume the condition (5.34) to be satisfied.

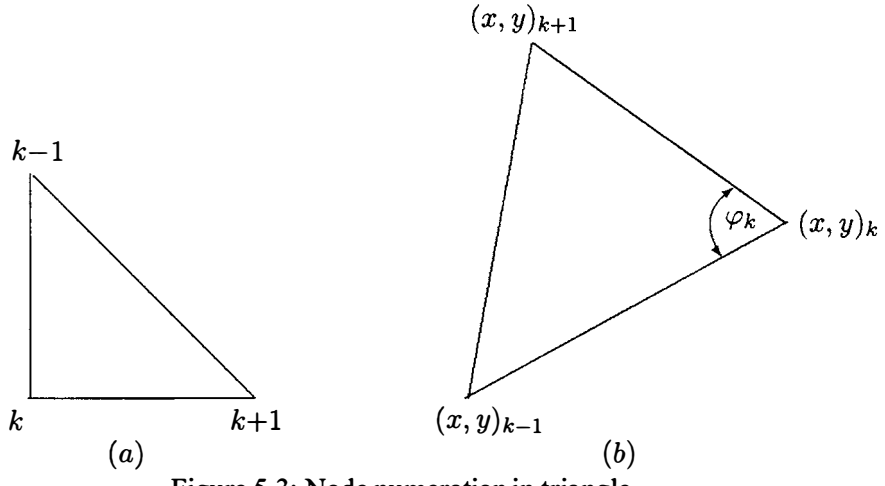


Figure 5.3: Node numeration in triangle.

5.4.2. To determine the metric parameters in the kth triangle we use the formulas, which after introducing notations

$$u_{k+1} = x_{k+1} - x_k\,, \quad \nu_{k+1} = y_{k+1} - y_k\,, \qquad (5.35)$$
$$u_{k-1} = x_{k-1} - x_k\,, \quad \nu_{k-1} = y_{k-1} - y_k\,,$$

are of the form

$$(g_{11})_k = u_{k+1}^2 + \nu_{k+1}^2\,, \quad (g_{12})_k = u_{k+1}u_{k-1} + \nu_{k+1}\nu_{k-1}\,, \qquad (5.36)$$
$$(g_{22})_k = u_{k-1}^2 + \nu_{k-1}^2\,, \quad (g_o)_k = u_{k+1}\nu_{k-1} - \nu_{k+1}u_{k-1}\,.$$

The formulas (5.36) satisfy the discrete analog of (5.4)

$$(g_{11})_k(g_{22})_k - (g_{12})_k^2 = (g_o)_k^2\,. \qquad (5.37)$$

Besides, we can write the following equalities

$$(g_{12})_k = \sqrt{(g_{11})_k (g_{22})_k}\,\cos\varphi_k\,, \quad (g_o)_k = \sqrt{(g_{11})_k (g_{22})_k}\,\sin\varphi_k\,, \tag{5.38}$$

where φ_k is the angle at the kth vertex of the triangle, see Fig. 5.3 *b*.

The quantities $(G_{11})_k$, $(G_{12})_k$, $(G_{22})_k$, $(G_o)_k$ are defined via similar formulas using the node coordinates of the preceding time step, and further they are corrected, e.g. by formulas (5.28).

The difference equations (5.31) for the functional (5.8)-(5.9) without the Jacobian can be written in node 1 of the stencil in the form

$$\sum_{L=1}^{4} c_L(x_1 - x_{2L}) + \sum_{L=1}^{4} b_L(x_1 - x_{2L+1}) = 0\,, \tag{5.39}$$
$$\sum_{L=1}^{4} c_L(y_1 - y_{2L}) + \sum_{L=1}^{4} b_L(y_1 - y_{2L+1}) = 0\,,$$

where the coefficients c_L, b_L are

$$\begin{aligned} c_L \;=\; & (\tilde{G}_{22})_{1,L} - (\tilde{G}_{12})_{1,L} + (\tilde{G}_{22})_{2,L} - (\tilde{G}_{12})_{2,L} + \\ & (\tilde{G}_{11})_{1,L-1} - (\tilde{G}_{12})_{1,L-1} + (\tilde{G}_{11})_{4,L-1} - (\tilde{G}_{12})_{4,L-1}\,, \end{aligned} \tag{5.40}$$

$$b_L = (\tilde{G}_{12})_{2,L} + (\tilde{G}_{12})_{4,L}\,. \tag{5.41}$$

Note that the above formulas do not include the quantities with index $(3, L)$, corresponding to the triangle with number k=3. It is not occasionally, since "contribution" of this triangle to the sum (5.29) does not change when varying the node (x_1, y_1). Therefore, we need not treat the triangle with number k=3.

5.4.3. Next, if $\tilde{G}_{12} \neq 0$, then, in general, the negative coefficients arise in the formulas (5.40)-(5.41). Therefore, the principle of maximum is not satisfied to the resulting system of linear equations. This is one more reason of constructing carefully the system of difference equations based on using the functional without the Jacobian.

When obtaining the coefficients $\tilde{G}$, we use $(G_o)_k$ to normalize them. By virtue of (5.38) they depend on $\sin\varphi_k$, i.e. on the angles between grid lines at the preceding time step. Therefore, the system of difference equations "responds" to small values of these angles or if they approach π, and, in turn, it should prevent the cells from degeneration. This is rather a serious reason to sophisticate the algorithm by cutting the cell into triangles. If not, in formulas (5.40)-(5.41) we have "average" values $\tilde{G}_{12}$ and area of the whole cell that "smears" the effect of degeneracy.

Consider the case when $\tilde{G}_{12} \equiv 0$. Then $b_L \equiv 0$ and the system (5.39) is reduced to more simple

$$\sum_{L=1}^{4} c_L(x_1 - x_{2L}) = 0\,, \quad \sum_{L=1}^{4} c_L(y_1 - y_{2L}) = 0\,, \tag{5.42}$$

Next from (5.40) we obtain

$$c_L = (\tilde{G}_{22})_{1,L} + (\tilde{G}_{22})_{2,L} + (\tilde{G}_{11})_{1,L-1} + (\tilde{G}_{11})_{4,L-1} \,. \tag{5.43}$$

Therefore, this system satisfies the principle of maximum. Meanwhile in the formulas (5.6), we obtain

$$\tilde{G}_{11} = \sqrt{G_{11}/G_o}\,, \quad \tilde{G}_{22} = \sqrt{G_{22}/G_o}\,.$$

Consequently, it is not necessary to cut the cells into triangles.

That is why the algorithms of constructing "quasi-orthogonal" grids are attractive in the domains with a fixed boundary. Unfortunately, in the unsteady problem definition $\tilde{G}_{12}$=0 conflicts with the ideology, set forth in Section 5.3, where it is allowed only variation of the coefficients $O(\tau)$. Consequently, it will not be discussed, though, these algorithms can be used for steady grid generation.

Now we turn for the functional (5.1)-(5.2) with the Jacobian. The corresponding system of grid equations (5.31) is nonlinear due to presence of the denominator in (5.32) and it is of more complicated form. It is described together with the iteration procedure in [6] and chapter 4. Corresponding formulas will be presented in Section 5.5.

5.5 Iteration Procedures

5.5.1. We begin with the method of solving the system of difference equations (5.39) for the functional without the Jacobian. Note that there is a good initial guess $\boldsymbol{r}^{0(N)}$ to the solution sought, defined by formula (5.17), and, by virtue of (5.18) and (5.22), it differs from the solution by a value of $O(\tau)$. Keeping it in mind and considering the structure of the equations (5.39), we suggest to use the simplest iteration procedure

$$x_1^{(i+1)} = x_1^{(i)} - \frac{w_i}{c_o} \sum_{L=1}^{4} \left[c_L (x_1^{(i)} - x_{2L}^{(i)}) + b_L (x_1^{(i)} - x_{2L+1}^{(i)}) \right] , \tag{5.44}$$

$$y_1^{(i+1)} = y_1^{(i)} - \frac{w_i}{c_o} \sum_{L=1}^{4} \left[c_L (y_1^{(i)} - y_{2L}^{(i)}) + b_L (y_1^{(i)} - y_{2L+1}^{(i)}) \right] .$$

Here the coefficients c_L, b_L are constant at iterations and

$$c_o = \sum_{L=1}^{4} (c_L + b_L) \,. \tag{5.45}$$

The iteration parameter ω=ω_i can be changed during iterations and it will be considered below. To provide the stability of the iteration procedure it should be $0 < \omega_i < 1$.

5.5.2. According to ideas stated in the end of Section 5.4, for the functional (5.1)-(5.2) with the Jacobian the formulas of the iteration procedure are

$$x_1^{(i+1)} = x_1^{(i)} - w_i \left(\frac{R_x}{R_{xx}} - \frac{R_y}{R_{yy}} \cdot \frac{R_{xy}}{R_{xx}} \right)^{(i)} \Bigg/ \left(1 - \frac{R_{xy}}{R_{xx}} \cdot \frac{R_{xy}}{R_{yy}} \right)^{(i)}, \quad (5.46)$$

$$y_1^{(i+1)} = y_1^{(i)} - w_i \left(\frac{R_y}{R_{yy}} - \frac{R_x}{R_{xx}} \cdot \frac{R_{xy}}{R_{yy}} \right)^{(i)} \Bigg/ \left(1 - \frac{R_{xy}}{R_{xx}} \cdot \frac{R_{xy}}{R_{yy}} \right)^{(i)}.$$

Here R_x, R_y, R_{xx}, R_{xy}, R_{yy} are the first and second derivatives of the discrete functional F^h, respectively, at the central point (x_1, y_1) of the stencil, described in Section 5.4. These derivatives are calculated on the grid of the *preceding iteration*. In F^h we use the matrix coefficients $\tilde{G}_{11}, \tilde{G}_{12}, \tilde{G}_{22}$ obtained on the grid of the *preceding time step* with correction (5.28). These matrix coefficients are constant within all iterations. Every quantity R_x, R_y, R_{xx}, R_{xy}, R_{yy} is the sum of 12 summands, corresponding in every of four cells with numbers L=1, ..., 4 to three triangles with numbers k=1, 2, 4. It is a cumbersome and simple exercise on differentiation and the user himself can execute it. Note that these formulas will be different for the triangles k=1, 2, 4 due to different role of the node (x_1, y_1).

In practical computations, two variants are possible. In the first, we select all the quantities from the stencil of an internal node and perform calculations up to obtaining the result of $(x_1, y_1)^{(i+1)}$. In the second variant we reserve a special array for gaining the quantities R_x, R_y, R_{xx}, R_{xy}, R_{yy}, by consequent treating all grid cells, as described in [11, 6] and chapter 4. The second variant is preferable by number of operations. However, if to calculate on a multi-processor computer an additional check is required to make a final choice.

In Section 5.4 we began to describe the algorithm in conformity with the first variant so as to present its essence and structure of the coefficients (5.40)-(5.41) in the equations (5.39).

5.5.3. As noted in [11, 6], other papers, and chapter 4, the iteration process is the variant of the quasi-Newton procedure, differing from the standard method by using only the block-diagonal in the matrix of the second derivatives. This allows us to execute the explicit iterations and eliminating solution of a complex nonlinear system of equations. This procedure was applied in harmonic mesh generation.

It may be of interest to note the following. Since for the functional (5.8)-(5.9) without the Jacobian we have

$$R^*_{xx} = 0\,, \quad R^*_{xx} = R^*_{yy} = \begin{cases} \tilde{G}_{11} - 2\tilde{G}_{12} + \tilde{G}_{22} & \text{at } k = 1\,, \\ \tilde{G}_{22} & \text{at } k = 2\,, \\ \tilde{G}_{11} & \text{at } k = 4\,, \end{cases} \quad (5.47)$$

then the formulas (5.46) take the form

$$x_1^{(i+1)} = x_1^{(i)} - w_i \frac{R^*_x}{R^*_{xx}}\,, \quad y_1^{(i+1)} = y_1^{(i)} - w_i \frac{R^*_y}{R^*_{yy}}\,. \quad (5.48)$$

Taking into account (5.47) one can readily check that the formulas (5.48) are identical to (5.44), since $R^*_{xx} = R^*_{yy} = c_o$.

Therefore, the iteration procedure (5.44) is also a variant of the quasi-Newton procedure for the functional without the Jacobian. In the discrete form, this functional is a positive-defined quadric form of the grid node coordinates. Note that the sum c_o of the coefficients in (5.45) is positive, since according to (5.47) every of twelve triangles gives a positive "contribution" into it. Actually,

$$G_{11} - 2G_{12} + G_{22} > 0\,,$$

since

$$|G_{12}| < \sqrt{G_{11}G_{22}} \le (G_{11} + G_{22})/2\,.$$

5.5.4. Now we turn to definition of the iteration parameter ω_i. Specific is that one should care for providing nondegeneracy of the mesh, i.e. the grid lines must not overlap. For the functional (5.1)-(5.2) with the Jacobian in the denominator this requirement is more stiff: we must hold convexity of all grid cells. As noted in Section 5.4, we assume that the requirement (5.34) to be satisfied on the initial mesh.

Consequently, after every iteration the mesh $(x, y)^{(i+1)}_{n,m}$ is checked for convexity of the cells. In every cell, we compute four values $(g_o)_k$ using the formulas (5.35)-(5.36) by the vertex coordinates. If $(g_o)_k{>}0$ for all $k{=}1, \ldots, 4$, then the cell is convex.

Violation of this condition at least in one cell leads the whole grid to be updated again with a smaller parameter ω_i. If to reduce ω_i by half, the necessary result is obtained if to take a half-sum of the grids

$$\tilde{x}^{(i+1)}_{n,m} = (x^{(i+1)}_{n,m} + x^{(i)}_{n,m})/2\,, \quad \tilde{y}^{(i+1)}_{n,m} = (y^{(i+1)}_{n,m} + y^{(i)}_{n,m})/2\,. \tag{5.49}$$

The updated mesh $(\tilde{x}, \tilde{y})^{(i+1)}_{n,m}$ is checked again and, if necessary, the parameter ω_i is reduced by half again, etc. (To avoid recycling of this procedure, one should stipulate the number of such "cuttings" of ω_i).

The stiff procedure of updating the whole grid, when appearing at least one "bad" cell, serves to hold mesh smoothness. Practical calculations have shown that improving the grid by treating only "bad" cells leads to undesirable consequences.

When performing the next iteration it is useful to begin with ω_i, taken from the preceding iteration or this value multiplied by a coefficient (> 1). It is caused by desire "not to delay the procedure" which should improve at convergence of iterations with decreasing the residual of the equations.

The matter of the initial guess ω_o in the first iteration is not simple, since it can depend on the countable domain shape at this time. One of possible solutions is using the last ω_i from the preceding time step or this value multiplied by a coefficient (> 1) (then these values should be kept as "regional" control parameters), and this matter is reduced to definition of ω at the first time step.

5.5.5. Now consider the convergence rate of the iteration procedures. Despite encouraging words that these procedures are the modified variants of the quasi-Newton procedure,

in practical computations the convergence rate is extremely slow. Two circumstances rescue the situation. Firstly, at the next step the system of equations is constructed so that if the correcting parameters equal to zero, the grid of the preceding time step satisfies them with the residual equal to 0 (in the discrete form). Thus, in accordance with (5.18) and (5.22), iterations should extinguish the residual of $O(\tau)$. This is a distinction of the suggested algorithms from another which produces "underiterated" equations which can not be driven to convergence. Secondly, along with extinguishing the residual of $O(\tau)$, the iteration procedure should provide grid smoothness with the deformed boundary and direct change of the grid in the proper way (desirably in necessary) not allowing for degrading mesh quality, suitable for solving the main flow problem. This is achieved by using corresponding correcting functionals. If above two purposes are achieved, the matter, whether the residual $O(\tau)$ will be extinguished completely, is not of importance. The next step begins from the "clean sheet", i.e. constructing a new system of difference equations. Hence, we can hope that the required number of iterations (even slowly converging) will not be too large. Naturally, the domain shape plays the substantial role. Besides, the value of the control parameters $p(\tau)$ in the correcting functionals, is of importance.

Is there an alternative, allowing for improving the matter of convergence ? In the monograph [12] (page 263) we read: "In the general theory of the iteration methods, it is considered the methods of two kinds: 1) methods which use a-priory information about operators, and 2) methods which do not use it (variational methods). The first methods do not use closeness of the initial guess to the solution. In the variational methods the iteration parameters are selected under condition of minimum of some functionals connected with the original equation. In this case the iteration parameters possess the property to take into account the quality of the initial guess".

Thus, recommendations incline to the variational methods. In point of view of developing automatical computer codes on grid generation, this approach can be perspective. However, by virtue of the above specific of the problem, it may happen that the iteration parameter is unfit and leads to a "bad" grid. Besides, numerical experiments have shown that employing more sophisticated procedures (e.g. as the standard Newton procedure, see chapter 4), taking into account additional numerical operations, does not provide significant gainings in running time.

5.6 On Mesh Degeneracy and Nonconvex Cells

5.6.1. Hitherto, when discussing the grid generation algorithms, we assumed that all mesh cells were convex, as stipulated by the condition (5.34) in Section 5.4. Unfortunately, the practical calculations require that this restriction to be weakened.

In Fig. 5.4 we present illustrations, when in the countable domain it is required to admit presence of quadrilateral cells degenerating (or nearly) into triangles, see the cells attached to the point O (or O_1,O_2) in Fig. 5.4*a*-*c*. Moreover, in the case, presented in Fig. 5.4*d*, one cell in the initial formulation is nonconvex. Degeneracy (or nearly degeneracy) of the quadrilateral cells into triangles may occur when computing an unsteady problem and the

boundary is becoming more complicated with time, while it is approximated by a restricted number of grid nodes, stipulated in initial data.

To execute the algorithm, when degenerating the cell into a triangle, we stipulate a correction of the algorithm.

As noted, the value $(g_o)_k$, given by the formulas (5.35)-(5.36), is the double area of the kth triangle. Then in this cell, we have

$$(g_o)_C = (g_o)_1 + (g_o)_3 = (g_o)_2 + (g_o)_4 \,. \tag{5.50}$$

Correction of the algorithm is of the following. Let

$$|(g_o)_k| \le \varepsilon^* (g_o)_C \,, \quad \text{where} \quad \varepsilon^* \ll 1 \,. \tag{5.51}$$

Then the kth triangle is *excluded* from calculation.

Actually it means that in the discrete variant of the formulas (5.1) and (5.8) we exclude an insignificant part of the integration domain, corresponding to degenerated triangles. (This integration domain is covered by the other pair of triangles). Note that "break-free" modifications (e.g. change of the degenerated Jacobian (equal to 0) in the denominator by something other), generally suggested by authors, which works when treating steady grids, lead to uncontrolled node velocity values when employing unsteady meshes.

The above technique (5.51) exhausts the measures connected with degenerating the quadrilateral cells into triangles.

5.6.2. The case of appearing nonconvex cells is more complicated. They can be revealed if in a "bad" triangle we get

$$(g_o)_k < -\varepsilon^* (g_o)_C \,. \tag{5.52}$$

Hypothetically there are the following alternative variants to resolve this case.

a) Not to pay attention (nothing should be changed in formulas). For instance, in the case, depicted in Fig. 5.4*d*, the only "bad" (nonconvex) cell is the "corner" one (the point O is one of its vertices) in the countable domain. Corresponding "bad" triangle is of the number k=3 and it is not treated as noted in Section 5.4. Here we need to take care that the computer code admits this "bad" cell, meanwhile not allowing for appearing another nonconvex cell.

b) The "bad" triangle is excluded from treating. This variant is admissible. For example, when constructing the harmonic meshes in [13], it is considered the algorithm where nonconvex cells are cut only with one "good" diagonal (i.e. passing inside the triangle), and even in the convex cells only one diagonal is drawn (realizing the minimal value of the functional). Validity of this algorithm has been confirmed by practical calculations.

c) In formula (5.2), the value g_o is replaced by $|g_o| = \sqrt{g_{11}g_{22} - g_{12}^2}$. Obviously, that positive definiteness of the matrices G_k will be hold. For the functional with the Jacobian, according to the above mentioned theorem from [7], existence of an unfolded grid for this set of the matrices G_k is guaranteed. The problem is how to overcome the variational barrier, if some grid node(s) jumps over it, and what a node velocity will be developed.

d) To take measures on preventing from appearing nonconvex cells. The following technique can be used. In the case of (5.52), the kth node is projected onto the segment

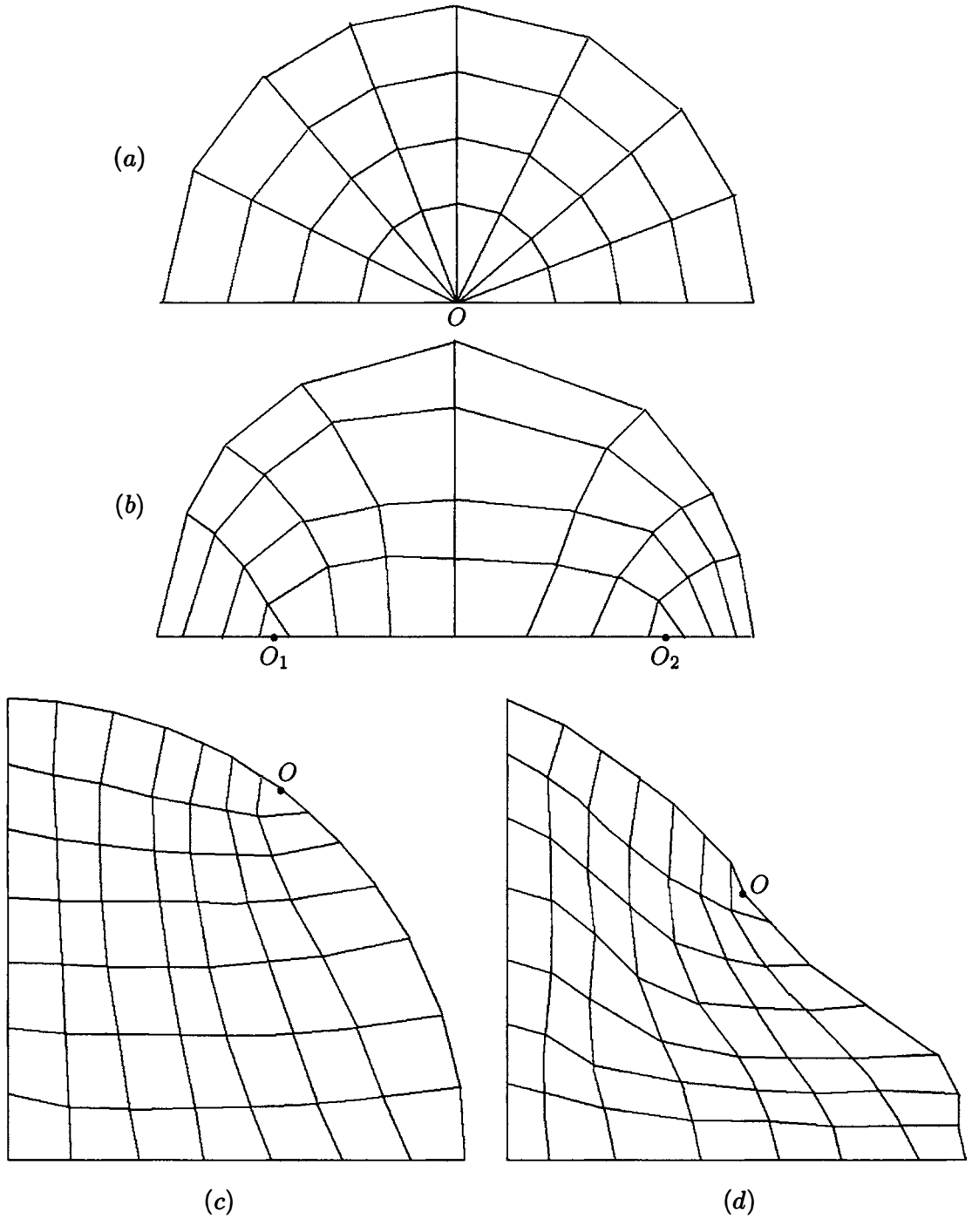


Figure 5.4: Examples of grid with triangle (a), (b), (c) and nonconvex (d) cells.

attached to the $(k-1)$th and $(k+1)$th nodes, and obtained point $(\tilde{x}_k, \tilde{y}_k)$ is entered into the list instead of (x_k, y_k). Then the nonconvex cell turns into the triangle. We give the corresponding formulas. Denote

$$\triangle x = x_{k+1} - x_{k-1} \,, \quad \triangle y = y_{k+1} - y_{k-1} \,.$$

Using the equation of the line, passing through the points $(x, y)_{k-1}$ and $(x, y)_{k+1}$, we can write

$$\tilde{x}_k = x_{k-1} + \tilde{t}\triangle x \,, \quad \tilde{y}_k = y_{k-1} + \tilde{t}\triangle y \,. \tag{5.53}$$

According to the equation of the perpendicular, passing through the points $(\tilde{x}, \tilde{y})_k$ and $(x, y)_k$, we have

$$(\tilde{x}_k - x_k)\triangle x + (\tilde{y}_k - y_k)\triangle y = 0 \,,$$

and readily obtain that

$$\tilde{t} = \frac{(x_k - x_{k-1})\triangle x + (y_k - y_{k-1})\triangle y}{\triangle x^2 + \triangle y^2} \,. \tag{5.54}$$

Next we calculate $(\tilde{x}, \tilde{y})_k$ via the formulas (5.53).

One should remember that this volitional transformation of the nonconvex cell into the triangle can lead to uncontrolled increase of the node velocity.

Nevertheless, practical calculations have shown that admission of appearing nonconvex cells can lead to failure, since their shape may be very "bad". On such a mesh the approximation of even the first derivatives, needed for consequent solving the main problem, deteriorates.

Thus, from the above variants in the case of appearing nonconvex cells, probably, the latter is preferable. That is one should *get rid of nonconvex cells in the initial mesh* (e.g. using the above technique), and further *not allow for their appearing*. Corresponding technique, based on selecting the iteration parameter ω_i, was described in Section 5.5.

Note that the recommended solution originates from the interest of calculating gas dynamics problems on the quadrilateral grids. It can be another one if, for instance, we calculate on the triangle grids or in the cases *b* and *c*, when nonconvex cells are admitted.

5.7 Problem of Nodes Velocity Control

5.7.1. As noted in [14] (page 15) "... calculation of fitted shock waves, whose motion induces the movement of the two-dimensional or three-dimensional grid nodes, has not been thoroughly studied yet, neither theoretically nor experimentally. A feed-back of the nodes movement upon the shock wave traveling is not studied yet, and it is not clear what effects this feed-back gives rise to." [1].

[1] Cited by the Russian issue: Godunov S.K. Reminiscenses about difference schemes. Report at the International Symposium "Godunov's method in gas dynamics", Michigan Univ., May 1997, Nauchnaya kniga, Novosibirsk, 1997.

In support of the above words we can say that a negative experience gained, first of all, relates to an inadmissible node velocity, appearing when modeling due to imperfection of the grid generation algorithms. Indeed, the grid node velocity influences substantially upon computing the main flow problem. It can be seen even by the example of the elementary Riemann problem, being the key item in the method of calculating gas dynamics problems (see [3]).

When the boundary moves, the grid generation algorithm should continuously provide *adequate* change of the nodes location inside the domain. The requirement, that the velocity of the "interior" nodes should be within the limits of changing the velocity of the "boundary" nodes, seems natural. (This is analogously to the principle of maximum to the elliptic equations). However, in spite of its naturalness, one can readily suggest examples when this requirement is not fulfilled, for instance, when applying the interpolation formulas in grid construction.

5.7.2. Now we will give the answer to this question, as promised in Section 5.3. The results of comparison to several variants of generating two-dimensional meshes using interpolation formulas are presented in [15].

One of the algorithms, proved to be suitable in the real-world calculations, is a transfinite interpolation which can be recommended for using. In the simplest two-dimensional variant the interpolation formula for the function $f(\xi,\eta)$ in the unit square $0 \le \xi,\eta \le 1$, with given boundary values, is

$$f(\xi,\eta) = (1-a_o)f(0,\eta) + a_1 f(1,\eta) + (1-b_o)f(\xi,0) + b_1 f(\xi,1) - (1-a_o)(1-b_o)f(0,0) - a_1 b_1 f(1,1) - (1-a_o)b_1 f(0,1) - a_1(1-b_o)f(1,0)\,. \quad (5.55)$$

Here $a_o{=}a_o(\xi)$, $a_1{=}a_1(\xi)$, $b_o{=}b_o(\eta)$, $b_1{=}b_1(\eta)$ are the monotone increasing functions with respect to their arguments in the segment $[0,1]$, and

$$a_o(0) = a_1(0) = b_o(0) = b_1(0) = 0\,, \quad a_o(1) = a_1(1) = b_o(1) = b_1(1) = 1\,.$$

One can readily check that to the arbitrary functions a_o, a_1, b_o, b_1, satisfying the above conditions, the interpolant $f(\xi,\eta)$ equals the boundary values at the boundary points.

Considering the sequences of grid node coordinates, defined at the boundary points, as values of some continuous functions at the points $\xi_n{=}n/n^*$, $\eta_m{=}m/m^*$, and using the formula (5.55) independently for each spatial coordinate x and y, we obtain the interpolation formulas for calculating the interior node coordinates.

To define the control functions $a_o(\xi)$,$a_1(\xi)$,$b_o(\eta)$,$b_1(\eta)$ one successfully uses so-called laws of boundary nodes redistribution, described in the monograph [3] (pp. 180-182)

$$S_n^{\text{bot}},\ S_n^{\text{top}},\ (n=0,\ldots,n^*)\,;\ \ S_m^{\text{lft}},\ S_m^{\text{rght}},\ (m=0,\ldots,m^*)\,.$$

In [15] in the simple example, when constructing a mesh in the convex quadrilateral with *direct-line* boundaries and specially imposed nonuniform boundary nodes redistribution, unexpected results have been obtained. We present those results with an insignificant modification, concerning the domain shape. The underlying domain is very simple, a unit

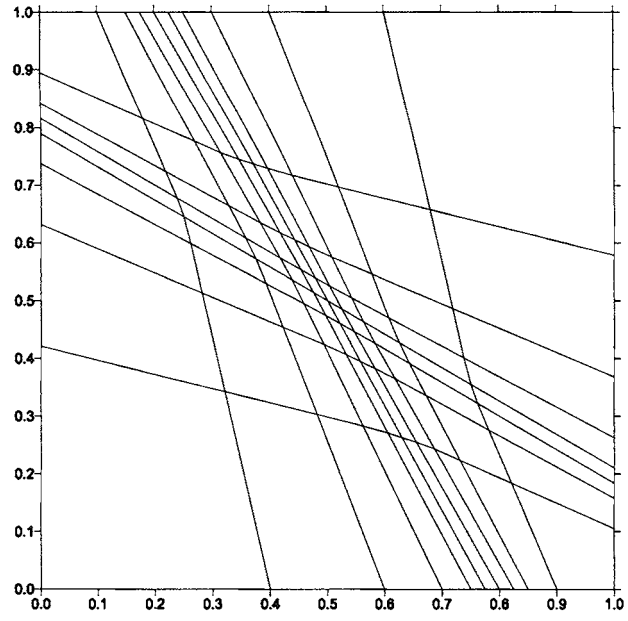


Figure 5.5: Grid generation in the unit square with transfinite interpolation (5.55) when coefficients are defined via (5.56).

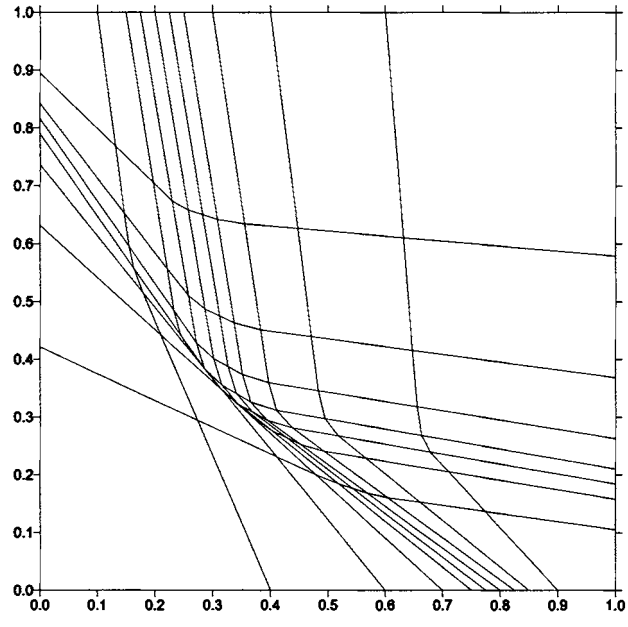


Figure 5.6: Grid generation with transfinite interpolation (5.55) when coefficients are defined via (5.57).

square $0 \le x, y \le 1$. The boundary nodes redistribution is defined as recommended in [15]. For the sake of clarity the number of grid nodes 11×9 (n^*=10, m^*=8) is not large. The distribution laws are the following

$$\tilde{S}_n^{\rm bot} = \{0, 16, 24, 28, 30, 31, 32, 33, 34, 36, 40\},\ \tilde{S}_m^{\rm lft} = \{0, 16, 24, 28, 30, 31, 32, 34, 38\},$$

and normalized values are determined as

$$S_n^{\rm bot} = \tilde{S}_n^{\rm bot}/\tilde{S}_{n^*}^{\rm bot}\,,\quad S_m^{\rm lft} = \tilde{S}_m^{\rm lft}/\tilde{S}_{m^*}^{\rm lft}\,.$$

Two other distribution laws are

$$S_n^{\rm top} = 1 - S_{n^*-n}^{\rm bot}\,,\quad S_m^{\rm rght} = 1 - S_{m^*-m}^{\rm lft}\,.$$

For example, definition of the control functions in the form of

$$a_n^o = a_n^1 = (S_n^{\rm bot} + S_n^{\rm top})/2\,,\quad b_m^o = b_m^1 = (S_m^{\rm lft} + S_m^{\rm rght})/2 \tag{5.56}$$

provides unfolded mesh generation in the square, see Fig. 5.5. When using the formulas (5.56), unfolded mesh generation is hold with an *arbitrary* boundary nodes redistribution. Recall that in [15] a more general case (domain is a quadrilateral with direct-line boundaries) was considered. If we define

$$a_n^o = S_n^{\rm bot}\,,\ a_n^1 = S_n^{\rm top}\,,\ b_m^o = S_m^{\rm lft}\,,\ b_m^1 = S_m^{\rm rght}\,, \tag{5.57}$$

i.e. as it seems in a more natural way than that of in (5.56), the above boundary nodes redistribution causes a folded mesh in the domain, see Fig. 5.6. The formulas

$$a_n^o = a_n^1 = n/n^*\,,\ b_m^o = b_m^1 = m/m^*\,, \tag{5.58}$$

can lead to more unsatisfactory result, see Fig. 5.7. Enlarging the number of nodes on the boundary (holding already located points), we do not change the position of the interior nodes.

As noted in [20], the transfinite interpolation is widely used when solving various problems of mathematical physics with the finite element method. Thus, in our opinion, the results, presented in Figs. 5.5-5.7, is of interest as a methodical example of applying the formulas (5.56)-(5.58) in the discrete realization of (5.55).

5.7.3. Note, that in the case of arbitrary domains, even the formulas (5.56) can give unsatisfactory results (that is why we need to invent sophisticated grid generation algorithms). However, "viability" of the interpolation formulas, applying (5.56), is higher than that of using (5.57) and (5.58).

Concerning the grid node velocity, calculated by interpolation formulas using (5.55), we can note the following. In (5.55) presence of the both positive and negative weight coefficients at the corner points allows us to suggest examples, when the interior node velocity is out of the interval, within which the boundary node velocity is changed. It

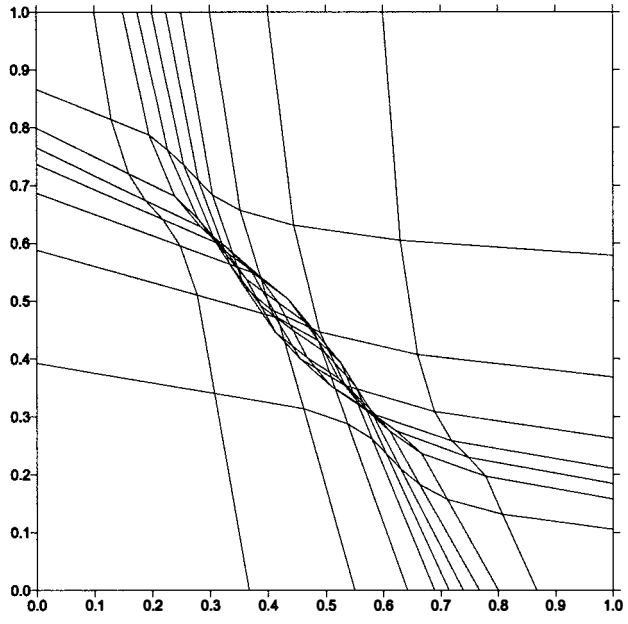


Figure 5.7: Grid generation with transfinite interpolation (5.55) when coefficients are defined via (5.58).

can occur to each of the coordinates x and y, and moreover to the total node velocity. In Section 5.4, we noted the analogous case for the system of linear equations (5.39).

This impedes automatization of the control for the node velocity in the practical calculations. For example, when generating, one can calculate W_{bnd}, the maximal node velocity on the domain boundary, and W_{int}, the maximal node velocity inside the domain. The question is what a deviation of W_{int} from W_{bnd} can be considered as admissible.

Note that the present algorithms can be also used to grid reconstruction with a fixed boundary (so as to have the grid, more suitable for modeling the flow problem). When reconstructing the grid, we have $W_{\text{bnd}}=0$ and $W_{\text{int}}\neq 0$.

5.8 On Correcting Functionals

The suggested grid technology uses the universal variational functionals (5.1)-(5.2) or (5.8)-(5.9). Their coefficients are defined by the metric parameters of the grid already produced. For the sake of brevity, they are referred to as the support functionals. However, as noted in Section 5.3, they should be corrected by another functionals with rather small weight coefficients. The choice of these functionals by the user is directed towards achieving necessary purposes.

5.8.1. As noted the meshes, generated by the orthogonal mapping, are of particular interest to the practical calculations. They should satisfy the condition

$$g_{12} = x_\xi x_\eta + y_\xi y_\eta = 0 . \tag{5.59}$$

Taking into account the Euler-Lagrange equations (5.13) or (5.14), we have the case when *two* functions $x(\xi,\eta)$ and $y(\xi,\eta)$ sought should satisfy *three* equations with corresponding boundary conditions. The matter on existence of this solution is not clear. Consider possibilities of the algorithms presented.

In [4], suggesting the functional (5.8)-(5.9), the authors accompanied it by the following identity

$$\frac{1}{2}(\tilde{G}_{22}g_{11} - 2\tilde{G}_{12}g_{12} + \tilde{G}_{11}g_{22}) \equiv x_\xi y_\eta - x_\eta y_\xi + \frac{1}{2}(A^2 + B^2) , \tag{5.60}$$

where

$$\begin{aligned} A &= \sqrt{\tilde{G}_{22}}\left(x_\xi \sin\frac{\omega}{2} + y_\xi \cos\frac{\omega}{2}\right) + \sqrt{\tilde{G}_{11}}\left(x_\eta \sin\frac{\omega}{2} - y_\eta \cos\frac{\omega}{2}\right) , \\ B &= \sqrt{\tilde{G}_{22}}\left(-x_\xi \cos\frac{\omega}{2} + y_\xi \sin\frac{\omega}{2}\right) + \sqrt{\tilde{G}_{11}}\left(x_\eta \cos\frac{\omega}{2} + y_\eta \sin\frac{\omega}{2}\right) . \end{aligned} \tag{5.61}$$

The value of ω ($0 \le \omega \le \pi$) is defined as

$$\cos\omega = \tilde{G}_{12}/\sqrt{\tilde{G}_{11}\tilde{G}_{22}} .$$

From (5.60) it follows that the inequality $E \ge 1$ is fulfilled for the energy density E of the invertible mapping, given by (5.2).

5.8.2. Consider the case, when minimizing the functional (5.1)-(5.2) or (5.8)-(5.9) we attain its *absolute* minimum. Then the functions $x(\xi,\eta), y(\xi,\eta)$ sought, realizing this minimum, should satisfy the equations

$$A = 0 , \quad B = 0 ,$$

where A, B are defined by the formulas (5.61). By an immediate check one can see that the following consequence is correct

$$x_\xi = \tilde{G}_{11}y_\eta - \tilde{G}_{12}y_\xi , \quad x_\eta = \tilde{G}_{12}y_\eta - \tilde{G}_{22}y_\xi . \tag{5.62}$$

In the theory of the quasi-conformal mappings they are known as Beltrami equations.

Suppose that the sought orthogonal mapping exists and is invertible. If the *absolute* minimum of the functional is attainable, this mapping should satisfy the equations (5.62). Substitute the equations (5.62) into (5.59) and after a simple derivation we get

$$g_{12} = (\tilde{G}_{11}y_\eta - \tilde{G}_{12}y_\xi)(\tilde{G}_{12}y_\eta - \tilde{G}_{22}y_\xi) + y_\xi y_\eta = \tilde{G}_{12}(\tilde{G}_{11}y_\eta^2 - 2\tilde{G}_{12}y_\xi y_\eta + \tilde{G}_{22}y_\xi^2) .$$

Since the matrix $\tilde{G}$ is positive defined, then

$$\tilde{G}_{11}y_\eta^2 - 2\tilde{G}_{12}y_\xi y_\eta + \tilde{G}_{22}y_\xi^2 > 0\,,$$

except the trivial case of $y_\xi{=}y_\eta{=}0$. Therefore, to satisfy the condition $g_{12}{=}0$ it is necessary that $\tilde{G}_{12}{=}0$.

If $\tilde{G}_{12}{=}0$, then the equations (5.62) take the form

$$x_\xi = \tilde{G}_{11}y_\eta\,,\quad x_\eta = -\tilde{G}_{22}y_\xi\,. \tag{5.63}$$

Since in this case $\tilde{G}_{11}\tilde{G}_{22} = 1$, the condition of orthogonality (5.59) is satisfied.

Thus, if the absolute minimum of the functional (5.1)-(5.2) or (5.8)-(5.9) is attained, the functional provides the orthogonal grid if and only if $\tilde{G}_{12} = 0$.

5.8.3. For the functional (5.8)-(5.9) without the Jacobian we suggested the following example, cf. [1]. Consider the mapping, defined by

$$x(\xi,\eta) = \frac{1}{2}(\xi^2 - \eta^2) - \frac{2}{3}\xi\,,\quad y(\xi,\eta) = \xi\eta + \frac{1}{2}\xi - \frac{1}{3}\eta\,. \tag{5.64}$$

in the unit square Q : $\{0{\le}\xi,\eta{\le}1\}$. Obviously, the mapping (5.64) satisfies the Laplace equations (5.11) and, consequently, minimizes the functional (5.8)-(5.9) when

$$\tilde{G}_{11} = \tilde{G}_{22} = 1\,,\quad \tilde{G}_{12} = 0\,.$$

The Jacobian of the mapping (5.64) is

$$J(\xi,\eta) = x_\xi y_\eta - x_\eta y_\xi = \left(\xi - \frac{2}{3}\right)\left(\xi - \frac{1}{3}\right) + \eta\left(\eta + \frac{1}{2}\right)\,.$$

Since $J(\xi,0){=}(\xi{-}2/3)(\xi{-}1/3){<}0$ in the domain $\eta{=}0$, $1/3{<}\xi{<}2/3$, the mapping has the fold in the vicinity of the image of the square bottom boundary $\eta{=}0$, see Fig. 5.8. This example is of interest, because the shape of the domain Ω, enclosed within the image of the square Q, is rather simple.

The maximal deepness of fold penetration into the square Q is at $\xi{=}1/2$ and achieves the coordinate $\eta{=}\eta_o$, defined by the condition $\eta_o(\eta_o{+}1/2){=}1/36$. From it we get $\eta_o{=}(\sqrt{13}{-}3)/12{=}0.0505$. Therefore, the mapping (5.64) is invertible in the rectangle Q_o : $\{0{\le}\xi{\le}1,\ \eta_o{\le}\eta{\le}1\}$ and the fold occupies a zone $0{\le}\eta{\le}\eta_o$. In (5.64), η can be substituted for $(\eta{-}\eta_o)/(1{-}\eta_o)$. Then we get the unfolded mapping in the entire square Q, and folded one in the rectangle $0{\le}\xi{\le}1$, $-\eta_o/(1{-}\eta_o){\le}\eta{\le}1$.

5.8.4. The above example attests, firstly, when defining an arbitrary positive defined matrix $G(\xi,\eta)$, the functional (5.8)-(5.9) without the Jacobian does not guarantee invertibility of the mapping, as already noted in Section 5.2.

Secondly, definition $\tilde{G}_{12}{=}0$ is insufficient so as to provide orthogonality of the mesh. A serious reason, caused by the mathematical formulation of the problem, is that the *absolute minimum* of the functional *may be not attainable*. If it is attainable, then with $\tilde{G}_{11}{=}\tilde{G}_{22}{=}1$

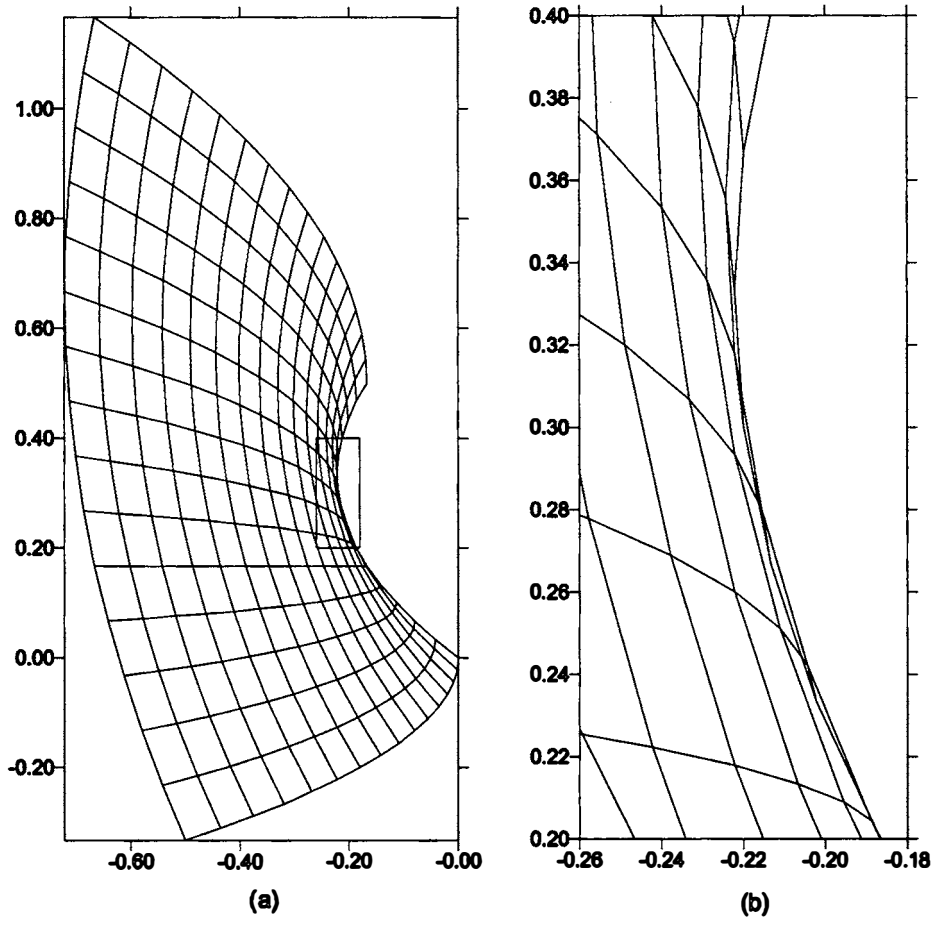


Figure 5.8: Folded 16×16 mesh is obtained by using the mapping (5.64).

the equations (5.63) would be the equations of the conformal mapping. Meanwhile, they are not satisfied to the mapping (5.64). Why? According to the Riemann theorem (mentioned in Introduction) under the conformal mapping one can define a correspondence only between three points at the boundary contour, and not to the all boundary points. Consequently, when minimizing the functional, the set of acceptable functions is very narrow and does not contain the conformal mapping. Meanwhile, it is the conformal mapping that would provide the *absolute* minimum of the functional.

Thus, a prudent stipulation that we can *only hope* that the orthogonal mapping exists, has every reason. Actually, when realizing a discrete mapping, we can only expect to produce a mesh being close (in some sense) to orthogonal. We refer it *conditionally* to as a quasi-orthogonal mapping.

Exceptions can be only in the favorable cases of lucky correspondence between the

boundary nodes.

We can conclude that, when minimizing the functional, there is a principal difference between the cases when the absolute minimum of the functional is attainable or not. In particular, note that in [7] for the discrete variant of the functional (5.1)-(5.2) it is proved that the unique solution exists only if the metric parameters provide its absolute minimum. The matter of uniqueness of the solution, when the absolute minimum is not attainable, is still open, and moreover, we call doubt on it.

5.8.5. Tendency to extract a special class of quasi-conformal mappings, which one can substantiate existence and uniqueness to, was realized for the functional (5.8)-(5.9) in [16]. The corresponding numerical algorithm was published in the monograph [3] (pp. 237-241). Extension of this trend was executed in number of works, where the quasi-isometric mappings are studied.

The mapping $x(\xi,\eta)$,$y(\xi,\eta)$ is referred to as *quasi-isometric*, if the ratio of the distance between any two (rather close) points to the distance between their images is limited from the top and bottom:

$$0 < \sigma_1 \le \frac{\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}}{\sqrt{(\xi_2-\xi_1)^2+(\eta_2-\eta_1)^2}} \le \sigma_2 \,.$$

Quasi-isometric mesh generation is outside the present study. For detail information we refer the reader to [17] and chapter 2.

5.8.6. One more interesting example of the variational functional, used in grid generation, is considered in the works by Academician A.F. Sidorov and his followers, cf. [9, 18], chapter 10. Its two-dimensional variant in the discrete formulation is

$$F_p = \sum_n \sum_m \left[(r_{n+1,m} - r_{n-1,m})^2 \left(\frac{1}{r^2_{n+1,m}} + \frac{1}{r^2_{n-1,m}} \right) + \right. \tag{5.65}$$
$$\left. (r_{n,m+1} - r_{n,m-1})^2 \left(\frac{1}{r^2_{n,m+1}} + \frac{1}{r^2_{n,m-1}} \right) \right] ;$$

here

$$r_{n+1,m} = \sqrt{(x_{n+1,m} - x_{n,m})^2 + (y_{n+1,m} - y_{n,m})^2}$$

is the distance between the central node of the stencil $(x,y)_{n,m}$ and neighbouring node $(x,y)_{n+1,m}$. The other values are defined similarly. In contrast to (5.29), summation is executed by grid nodes, and not by cells.

Minimization of the functional is directed towards obtaining uniform meshes. That is why it is referred to as a criterion of uniformity.

5.8.7. The author believes that instead of (5.65) one can use another formula as the

criterion of *uniformity*

$$F_R = \sum_n \sum_m \left[(r^2_{n+1,m} - r^2_{n-1,m}) \left(\frac{1}{r^2_{n-1,m}} - \frac{1}{r^2_{n+1,m}} \right) + \right. \tag{5.66}$$
$$\left. (r^2_{n,m+1} - r^2_{n,m-1}) \left(\frac{1}{r^2_{n,m-1}} - \frac{1}{r^2_{n,m+1}} \right) \right] .$$

Note that to any $a, b > 0$ the equality

$$\frac{(a-b)^2(1/a^2 + 1/b^2)}{(a^2 - b^2)(1/a^2 - 1/b^2)} = \frac{a^2 + b^2}{(a+b)^2} ,$$

and inequality

$$(a+b)^2/2 \le a^2 + b^2 < (a+b)^2 \le 2(a^2 + b^2)$$

are correct. Applying these estimates to every summand in the sums (5.65) and (5.66), we can see that the criteria (5.65) and (5.66) are *energetic* equivalent

$$F_R/2 \le F_p < F_R \le 2F_p .$$

Therefore, they are equivalent in the point of view of the variational approach. The form (5.66), not including the square root, is preferable in computing the first and second derivatives when iterating, as described in Section 5.5.

Note that in contrast to that of in Section 5.4, to derive the difference equations for the functional of uniformity in the (n, m)th node, the nine-point stencil (see Fig. 5.1) should be added by four nodes more with numbers $(n+2, m)$, $(n, m+2)$, $(n-2, m)$, and $(n, m-2)$. It requires a special consideration, when deriving the equations in the neighboring nodes to the boundary, so as to exclude non-existent summands in the functional.

It is quite natural, since the formula (5.65) is the discrete model of the variational functional suggested in [9]. This functional includes both the first and second derivatives of the functions sought. A consequence is that the Euler-Lagrange equations to this functional are the fourth-order PDEs, and not second-order ones as the equations (5.13) or (5.14).

5.8.8. We could continue the list of the functionals suggested by various authors. In particular, it is of interest the variational functionals, mentioned in Introduction, oriented to obtaining adaptive meshes. One such a functional was suggested in [6] (see also chapter 4) as generalization of (5.1)-(5.2) and is applied with the purpose to adapt the mesh towards a control function, obtained when modeling the unsteady flow problem. This function can be vector-valued one [19], i.e. include several quantities (density, pressure, components of velocity, etc.). Once the control function has been selected, they compute its derivatives in the grid nodes and use a bit more complicated formulas to calculate the first and second derivatives of the discrete functional needed in the iteration procedure, described in Section 5.5.

Every functional can be used as correcting one to the support functionals (5.1)-(5.2) or (5.8)-(5.9). As described in Section 5.3, it can be done, for instance, defining the energy of the mapping as

$$\tilde{E} = E + p_k(\tau)E_k\,,$$

where E_k is the energy of the correcting functional.

Dimensions of E and E_k should be matched. Dimensionless formulas (5.65) or (5.66) can be used for correcting the support functional (5.1)-(5.2). In the functional (5.8)-(5.9) we should enter an additional co-factor of Jacobian dimensionality, e.g. the sum of areas of four cells, attached to the underlying mesh node.

With the purpose to accelerate the convergence rate of the iteration procedures, when constructing the meshes in the domains with fixed boundary (for example, when preparing the initial data at initial time of the unsteady problem), the correcting functionals can be used independently, as implied by the authors. The functionals can be used either individually or in combination with weight coefficients. In the latter case the author recall one proverb: "If to run simultaneously after two hares, no one will be caught". Discussion on these questions is beyond the scope of the present work.

5.9 Conclusion

1. It is suggested to use (5.1)-(5.2) or (5.8)-(5.9) as the support functionals when constructing the meshes in unsteady problems. Their coefficients are the metric parameters of the mesh, obtained at the preceding time step of unsteady simulation.

2. If discretization of these functionals is performed carefully, it is obtained the system of equations, substituting into which the grid of the preceding step, we get the residual identically equal to zero. Hence, with the boundary conditions, corresponding to the next time step, the residuals are of $O(\tau)$ (τ is the time step). Therefore, one can expect that the node velocity will be within the reasonable interval relative to the velocity of the boundary. By virtue of set forth in Section 5.7, a more clear statement can hardly be declared.

3. The study of the support functionals, when there is no a feed-back, may lead the mesh to spoil irreversibly during unsteady modeling, and it becomes unsuitable to the main flow problem. To prevent it, one should add another functionals with weight coefficients $p(\tau)$ to the support functionals.

In the simplest variant it can be realized via the formula (5.28) so as to correct the coefficients $\tilde{G}_{11}, \tilde{G}_{22}, \tilde{G}_{12}$. One can conditionally say that the use of the coefficient p_o connects up the functional of "orthogonality", and the coefficient p_h is harmonic. The other cases have been discussed in Section 5.8.

4. In the case of the functional (5.8)-(5.9) without the Jacobian the system of difference equations (5.39) with coefficients (5.40)-(5.41) is realized. From this formulas one can see that correction (5.28) is reduced to increasing the coefficients c_L in the system of equations (5.39). The matter, concerning the choice of the correcting functionals and weight coefficients, should be the subject of a special experimental study.

5. Now we discuss which of two functionals (with the Jacobian or without) one should prefer. As noted, the functional with the Jacobian guarantees the unfolded mesh generation with any positive-defined symmetric matrices of the coefficients G, and the functional without the Jacobian does not. However, one should note that with small coefficients in the correcting functionals we calculate in the vicinity of the support functionals. Both of them provides nearly identical results, reproducing the grid from the preceding time step (if the boundary is taken at this time step as well), which is implied to be unfolded. Therefore, with rather small parameters p^* one can hope that the functional without the Jacobian will provide an unfolded mesh as well.

Presence of the Jacobian for the mesh G_o from the preceding time step in the denominator of the coefficients (5.6)-(5.7) of the functional (5.8)-(5.9) is of importance. Besides, when calculating, we verify that mesh is unfolded.

The above argumentation is added by advantages of the functional without the Jacobian. First of all, it is a linearity of the equations (5.39), and, therefore, the coefficients (5.40) are constant during iterations. The matrix of these coefficients (8 numbers in every internal grid node) can be determined only once at every time step, and then it is used at all iterations. The iteration is reduced to simple formulas (5.44) and control for mesh convexity. Thus, if necessary a large number of iterations is not burdensome. The computational formulas for the functional with the Jacobian are substantially more complicated, and, therefore, running time significantly increases.

Probably, in practical computations one need to have the codes, realizing the both functionals. The use of the functional with the Jacobian should be executed if we have exhausted all possibilities of the functional without the Jacobian.

6. Note that in practical computations at the first stage one calculates the mesh by interpolation formulas. This stage is executed till it provides with a suitable mesh.

It is important that transition from the interpolation stage to using the functionals is "painless" (without appearing an uncontrolled node velocity) due to the ideology of one functional, set forth above. The same concerns change of the control parameters in the functionals.

However, practical calculations have shown that this is realized only if such a reconstruction begins "in advance" so as to have sufficient "time reserve" not causing the break in modeling the flow problem.

7. In view of "survivability" of the grid generation algorithm and suitability of the produced meshes to the flow problem it is of importance whether to admit or not nonconvexity of the cells. This matter was discussed in Section 5.6. It is preferable to dispose of the nonconvex cells in initial data (if possible) and not to admit their appearance (stipulating this requirement in the code on mesh quality control). Realization of this requirement was considered in Section 5.5 by automatical selection of the iteration parameter ω_i.

8. In spite of above, the author does not pretend to developing completely automatic grid generation algorithms, allowing for modeling unsteady problems without user's control. Assume that we could experimentally develop a technology of switching on the correcting functionals, described above. It should be realized in the form of concrete rec-

ommendations, allowing the user for working with the code. (It would be ideal to define automatically the control parameters at every time step based on a concrete situation).

There still remains the cases when the modeling can not be completed due to necessity to change the topological map of the problem (to change "cutting" into countable subdomains), to change the number of grid nodes so as to reflect adequately appearing singularities (e.g. distortion of the boundary shape), etc. Automation of solving these questions was and is an actual problem, have not quite solved in practical applications.

Acknowledgment

This work was supported by the Russian Foundation for Basic Research, project 02–01–00236.

Bibliography

[1] Prokopov G.P. Universal variational functionals for 2D grid generation. *Prepint of Keldysh Institute for Applied Mathematics of Russian Academy of Science* , **1**(2001), 36 p. (in Russian).

[2] Prokopov G.P. Variational methods of 2D grids calculating for nonstationary poblems. *Prepint of Keldysh Institute for Applied Mathematics of Russian Academy of Science* , **4**(2003), 32 p. (in Russian).

[3] Godunov S.K. (Ed.), Zabrodin A.V., Ivanov M.Ya., Kraiko A.N., and Prokopov G.P. Numerical Solution of Multi-Dimensional Problems in Gas Dynamics, *Nauka Press* (Moscow, 1976); (French translation: Résolution Numérique des Problémes Multidimensionnels de la Dynamique des Gaz, *Mir*, Moscou, 1979.)

[4] Godunov S.K. and Prokopov G.P. On computation of conformal transformations and construction of difference meshes. *USSR Comput. Maths. Math. Phys.*, **7**(5), 1967, pp. 1031–1059.

[5] Winslow A. Numerical solution of the quasi-linear Poisson equation in a nonuniform triangle mesh, *J. Comput. Phys.*, **1**(2), 1966, pp. 149–172.

[6] Ivanenko S.A. Control of cell shape in the construction of a grid. *Comp. Math Math. Phys.* **40**(11), 2000, pp. 1662–1684.

[7] Ivanenko S.A. Existence of equations describing the classes of nondegenerate curvilinear coordinates in arbitrary domains. *Comp. Math Math. Phys.* **42**(1), 2002, pp. 43–48.

[8] Antonova R.N., Prokopov G.P., and Sofronova O.I. Adaptive difference grid generator and initial approximation problem for the grid construction in complex domain. *Voprosy Atomnoj Nauki i Tehniki (VANT). Ser. Mathematical modelling of physical processes.* Issue 1-2, pp. 84–90. 1996. (in Russian).

[9] Sidorov A.F. and Shabashova T.I. On a method of constructing optimal grids in multidimensional domains. *Chisl. Metody Mechaniki Sploshnoy Sredy*. Novosibirsk, **12**(5), pp. 106–124. 1981 (in Russian).

[10] Prokopov G.P. Methodology of variational approach to quasi-orthogonal grid construction. *VANT. Ser. Mathematical modelling of physical processes*. Issue 1, 1998, pp. 37-46. (in Russian).

[11] Ivanenko S.A. and Charakhch'yan A.A. Curvilinear grids of convex quadrilaterals. *USSR Comput. Maths. Math. Phys.* **28**(2), pp. 126–133. 1988.

[12] Samarskii A.A. and Nikolaev E.S. Methods of solving difference eqations. *Nauka Press* (Moscow, 1978).

[13] Uskov V.M. Generation of grids consisting of non degenerate quadrangles using the Delaunay criterion. *VANT. Ser. Mathematical modelling of physical processes*. Issue 2, 1994, pp. 12-18. (in Russian).

[14] Godunov S.K. Reminiscenses about difference schemes, *J. Comput. Phys.*, **153**, pp. 6–25. 1999.

[15] Antonova R.N. and Prokopov G.P. Comparing 2D difference mesh generation variants with interpolation formulas. *VANT. Ser. Mathematical modelling of physical processes*. Issue 1, pp. 78–84. 1994. (in Russian).

[16] Belinskii P.P., Godunov S.K., Ivanov Yu. B., and Yanenko I.K. The use of a class of quasiconformal mappings to construct difference grids in domains with curvilinear boundaries. *USSR Comput. Maths. Math. Phys.* **15**(6), pp. 131–139. 1975.

[17] Godunov S.K., Zhukov V.T. and Feodoritova O.V. An algorithm for construction of quasi-isometric grids in curvilinear quadrangular regions. *In: Proceedings of the 16th Int. Conf. on Numerical Methods in Fluid Dynamics,* Arcachon, France, July 6-10, pp. 49–54. 1998.

[18] Sidorov A.F., Khairullina O.B., and Ushakova O.V. Variational methods of Construction of optimal grids. Chapt. 36 in: Handbook of Grid Generation, J.F. Thompson et al. eds., CRC Press, Fl, (1999).

[19] Azarenok B.N. and Ivanenko S.A. Application of moving adaptive grids for numerical solution of nonstationary problems in gas dynamics. *Intern. J. for Numer. Meth. in Fluids*. **39**(1), pp. 1–22. 2002.

[20] Thompson J.F., Warsi Z.U.A., and Mastin C.W. Boundary-fitted coordinate systems for numerical solution of partial differential equations. A review. *J. Comp. Phys.* **47**(2), pp. 1–108. 1982.

Chapter 6

GENERATION OF CURVILINEAR GRIDS IN MULTIPLY CONNECTED DOMAINS OF COMPLEX TOPOLOGY

Nataliya A. Artyomova, Aleksey F. Khairullin, and Olga B. Khairullina
Institute of Mathematics and Mechanics
Ural Branch of the Russian Academy of Sciences

A survey of the authors works on the automatic generation of optimal multiblock curvilinear grids of large size (about hundreds of millions of nodes) in two-dimensional domains of an arbitrary connectivity and configuration using minimum input data is presented. Basic features of the proposed approach are: a special way of formalization of the criterion of closeness of grids to a uniform ones, smoothness of grids at block boundaries, automatization of generation of arbitrary topology grids in multiply connected domains, and the use of combined iterations, which include both discrete and variational interpretations, resulting in an effective computational process. An algorithm of the automatic generation of three-dimensional block-structured grids is described for a wide class of axisymmetric three-dimensional simply and multiply connected domains.

6.1 Introduction

The construction of grids of high quality (sufficiently smooth, close to uniform and orthogonal ones [1, 2], with a large number of nodes) is an important step in solving problems of mathematical physics. This construction should be maximally automated and use minimum input information. The calculating time of a grid must be much less than the total calculating time of the problem.

A vast survey of the literature on constructing grids is given in [3, 4]. The methods of construction of grids are rather different: with the help of conformal and quasiconformal mappings, solutions of the Euler equations of elliptic type obtained from variational problems, and so on. Some algorithms take into account only geometry of the domain, other

algorithms consider, besides geometry, singularities of the solutions of a concrete problem in this domain. The constructed grids may have a regular and irregular structure. In the case of regular (structured) grids, most of the works are devoted to constructing curvilinear grids in two- and three-dimensional domains topologically equivalent to a rectangle and parallelepiped, respectively. Smaller part of works are devoted to designing block-regular (composed or block-structured) grids. Very few works describe grid generation in multiply connected domains of complex topology (with elements of basis types of the topologies H, O, C [3] of grids or non-one-sheeted mapping).

In this paper, we consider questions related to constructing optimal block-structured curvilinear grids in two- and three-dimensional simply and multiply connected domains of complex geometries in the class of coordinate grids. In this case, parts of the boundary Γ of a given domain D are considered to be coordinate lines or coordinate surfaces in the plane of curvilinear coordinates (p, q) (the plane of node numbers). A two-dimensional domain is represented as a rectangle or a union of rectangles and a three-dimensional domain is represented as a parallelepiped or a union of parallelepipeds (blocks) in which a regular uniform orthogonal grid is introduced.

6.2 Optimal Two-Dimensional Grids

6.2.1 Optimality Criteria

We consider a grid in a domain D to be optimal if the functions

$$x=x(p, q),\ y=y(p, q)$$

mapping the grid from the plain (p, q) in the physical plane (x, y) under a prescribed arrangement of nodes on the boundary Γ minimize the functional [1, 2]

$$\Phi = \iint_D \left[\left(\ln \sqrt{x_p^2 + y_p^2} \right)_p^2 + \left(\ln \sqrt{x_q^2 + y_q^2} \right)_q^2 + A \frac{\left(x_p^2 + y_p^2\right)\left(x_q^2 + y_q^2\right)}{\left(x_q y_p - x_p y_q\right)^2} \right] dp\, dq. \quad (6.1)$$

The functional Φ formalizes the optimality criteria, i.e., criteria of closeness of grids to uniform ones with respect to the distance between nodes and to orthogonal ones at the intersection points of coordinate lines. We can control the quality of a grid by changing the weight $A > 0$. The subscripts in (6.1) denote the differentiation of the corresponding function with respect to variables p, q.

Let $r_{i\pm1,j}$ denote the Euclidean distance between the grid nodes with coordinates $(x_{i,j}, y_{i,j})$ and $(x_{i\pm1,j}, y_{i\pm1,j})$, $r_{i,j\pm1}$ denote the distance between nodes $(x_{i,j}, y_{i,j})$ and $(x_{i,j\pm1}, y_{i,j\pm1})$, and $\alpha_{i,j}^{(l)}$ $(l=1, 2, 3, 4)$ denote the angles between the rays connecting the node (i, j) and the nodes $(i+1, j)$, $(i, j+1)$, $(i-1, j)$, and $(i, j-1)$. Then the functional (6.1) is a continuous analog of the discrete functional [1]

$$F = F_R + AF_Q, \quad (6.2)$$

$$F_R = \sum_{i,j} \left[(r_{i+1,j} - r_{i-1,j})^2 \left(\frac{1}{r_{i+1,j}^2} + \frac{1}{r_{i-1,j}^2} \right) + (r_{i,j+1} - r_{i,j-1})^2 \left(\frac{1}{r_{i,j+1}^2} + \frac{1}{r_{i,j-1}^2} \right) \right],$$

$$F_Q = \sum_{i,j} \sum_{l=1}^{4} \sin^{-2} \alpha_{i,j}^{(l)}.$$

The functional F_R enables us to minimize in the average the relative difference of the distances between adjacent nodes, and the functional F_Q enables us to construct nondegenerate grids ($\alpha_{i,j}^{(l)} \neq 0, \pi$) which are close to orthogonal ones.

A distinctive feature of the variational method used in the algorithm is a special way of formalization of the criterion of closeness of grids to uniform ones with respect to the areas of adjacent cells in combination with the criterion of closeness of grids to orthogonal ones. Formalization of the uniformity criterion of the arrangement of grid nodes turned out to be crucial for constructing economical and stable procedures of calculation of grids. The functional corresponding to this formalization yields a system of the Euler–Ostrogradsky equations, hyperbolic in a broad sense. It has allowed us to consider new wider types of boundary conditions and to develop effective algorithms and programs of grid generation for rather complicated domains. Economical and efficient procedures of calculation of grids are connected with the application of iterative processes, which use both a special nonstationary modification of the Euler–Ostrogradsky equations and direct geometrical methods of minimization of discrete functionals formalizing the optimality criteria.

A formulation of the problem of constructing such grids and corresponding iterative calculating methods for domains of complex geometry are described in [1, 2]. In detail, an economical successive algorithm of designing optimal grids in multiply connected domains is described in [5, 6].

The grids constructed can contain elements of basic grids of the type O, C, H [3], for which the mappings of grids from the physical plane (x, y) into the plane (p, q) and inversely are not one-valued. Namely, they are multivalued along the splits in the plane (x, y) (grids of the H type, marked lines) and along slits (grids of the O and C type, bold lines) in the plane of curvilinear coordinates (Fig. 6.1). The image of the prescribed domain in the plane (p, q) can be non-one-sheeted (Fig. 6.1*c*). The use of the subdomain overlapping method [2] enabled one to construct grids which are smooth on the boundaries of adjacent blocks.

6.2.2 Sequential and Parallel Algorithms

To reduce the computation time for multiblock optimal grids of large size (dozens or hundreds of millions of nodes), a parallel algorithm [7]–[10] is elaborated on the basis of the sequential algorithm for a multiprocessor computational systems (MCS) [11]. The use of distributed memory of the system essentially reduced both the memory necessary for storage of the matrices of coordinates of grid points, densely packing them independently of dividing the domain into blocks, and the calculation time of a grid.

Memorizing the coordinates of grid points. A multiblock grid is formed as a result of joining regular grids in the blocks covering this domain. In the sequential algorithm, the structure of the matrix of coordinates of grid points is determined, using a special numeration [2, 5], by the geometry of the topological image (Fig. 6.1 c) of a grid in the plane (p, q). This image is inscribed into a rectangle of the size $M \times N$. The matrix is filled by the "flag" method. If a point does not belong to the domain, then "a flag", i.e., the number 10^{18}, is transferred into the corresponding element of the matrix. In this case, the memory is used uneconomically. The matrix can be filled more densely if we divide the domain into blocks in another fashion. To memorize the ambiguity of the coordinates of a split, we introduce two columns in the matrix if this line is vertical (marked lines in Figs. 6.1 a and 6.1c) or two rows if the line is horizontal. To each point of a slit in the plane (x, y), at least two elements of the matrix correspond, since the slit in the plane (p, q) has two images (bold lines in Figs. 6.1a and 6.1c). If more than one slit emerge from a point, then from two up to four elements of the matrix can correspond to the endpoints of the slits.

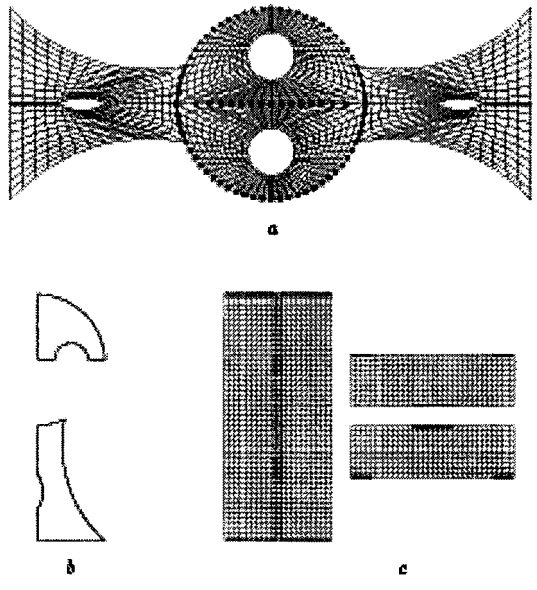


Figure 6.1: a an optimal grid; b boundaries of given fragments of the domain; c the topological image of the grid.

In the parallel algorithm, a block structure is suggested for memorizing the coordinates of grid points. The coordinates of the block grid points form one-dimensional arrays. The points are numbered from the bottom on vertical lines and from the left on horizontal lines and written one after the other. As a result, the array of the coordinates of grid points is densely packed independently of the way of dividing the domain into blocks. The matrix requires no extension to memorize the ambiguity of the coordinates of splits.

Each block k in the plane (p, q) can be described by coordinates p_{1k}, q_{1k}, p_{2k}, and q_{2k}

of its vertices, which are used to arrange connections between blocks. Knowing the number K of a block and the number of the element in the common array, which is the beginning of the one-dimensional array for this block, enables us to restore the structure of the image of the constructed grid. On the other hand, for solving problems on large grids we may not recover the structure of the image but solve problems block by block on several processors, using the distributed memory of MCS and information on the neighboring blocks. As each block has at most 8 adjacent blocks (one at each side and one at each angle), relations between the problems being solved on different processors are easily organized.

Constructing grids with smooth coordinate lines. If we construct grids in blocks independently of coordinates of grid points of the adjacent blocks, then the grid lines on the common boundaries may not be smooth. When solving problems, such grids are poor if the unknown quantities have large gradients in a neighborhood of lines of block junctions [12]. In this case, we use the block overlapping method for calculation of smooth grid lines.

In the sequential algorithm, each block whose boundary has a common part with the boundary of another block (inner boundary) is extended beyond this boundary by the addition of one coordinate strip. Respectively, a horizontal or vertical line of the adjacent block are taken as the boundary of the extended block. After that, the grid is calculated at each iteration in all blocks successively. Since blocks are overlapping, points of the grid on the inner boundaries of the domain are calculated in correspondence with the given optimality criteria. When calculating grids in multiply connected domains of complex topology, the movement of nodes on slits and splits is organized in a similar way, but in these cases the analysis of geometrical possibilities of block junctions is much more complicated [5, 6].

To organize the block overlapping in the parallel algorithm, we consider indications c_{lk} $(l = 1, 2, 3, 4)$ of sides of each block k. They characterize the type of the block boundary and can take the value $c_{lk} = 0$ for the given boundary of the domain, $c_{lk} = 1$ for an internal boundary, $c_{lk} = 31$ for an internal boundary belonging to a split, and $c_{lk} > 40$ for slits. Each block is considered as a given simply connected domain with a given arrangement of nodes on the boundary.

In the parallel algorithm, each block is extended, by the addition one coordinate strip, beyond all sides independently of the type of the boundary. Hence, every block will consist of two interconnected parts: internal and external. The external part is referred to as "a border" (the shaded strip in Fig. 6.2).

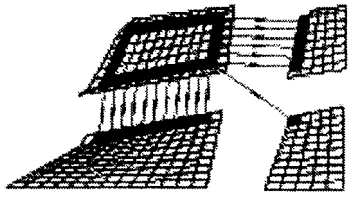


Figure 6.2: The structure of a block and the arrangement of the block overlapping.

If c_{ik}=1 or c_{ik}=31 on some side of the block, then the block border overlaps the adja-

cent block and the coordinates of points of the external line of the block border are defined to equal the values of the coordinates of points of this line in contiguous block. For calculation of the grid in the block k with a slit $c_{ik}=m$ ($m>40$), we seek the block l one of whose sides is a slit with the same number $c_{jl}=m$. The coordinates of points of an internal line of the border of the block k, corresponding to this slit, and its external line as well, are defined to equal the values of the coordinates of points of the boundary and near-boundary lines of the block l. If the border is adjoined to the boundary of the domain, then the coordinates of points of its external line are labeled by the corresponding "flag".

Distributing blocks into groups. To provide smooth grids, the blocks whose borders are not completely labeled cannot be simultaneously calculated on different processors in an arbitrary order.

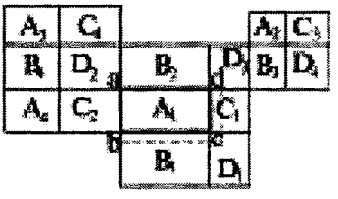


Figure 6.3: Principal scheme of parallelizaton.

Let us consider, for example, the block A_1 (Fig. 6.3), which is a part of the image of some domain. Its sides bc and cd has the indications $c_2 = c_3 = 1$. When using several processors, the grid in the adjacent blocks B_1 and C_1 must be calculated in different time in order to transfer information about the boundary, obtained when calculating the grid in one of the blocks, to block A_1. The coordinates of a point laying on the intersection of bc and cd must be calculated also in correspondence with the given optimality criteria. For this purpose, the calculation of the grid in the block D_1 should be separated in time from calculation of the grid in A_1, B_1, and C_1. The other sides of this block and the remaining blocks should be similarly analyzed. As a result of such analysis, we obtain blocks A_i (i=1...,I_a), B_i (i=1...,I_b), C_i (i=1...,I_c), and D_i (i=1...,I_d), which are combined into four corresponding groups. Distributing the blocks into groups allows us to generate the grid in blocks of one group simultaneously on different processors and by turns in blocks of different groups.

The use of the border enabled us to arrange the overlapping more universally and to calculate the grid in blocks with several slits. In the case when the grid is generated by the sequential algorithm, the block being calculated cannot have more than one slit [5, 6]. Otherwise, since a block may have up to four slits in the general case, this problem was solved by duplicating the block into several (depending on the number of slits) blocks, and a particular slit is assigned to each of them.

For calculation of an optimal curvilinear grid in a given domain by the iterative method, we should specify the initial allocation of grid nodes and the coordinates of block vertices. Using these data, the indications of block sides are determined, the external line of borders

is filled, and the blocks are divided into groups. All these operations are performed by the program.

Loading of processors. The calculation time of an optimal grid on a multiprocessor computer consists mainly of the calculation time on processors for the grid in blocks and the time of exchange of information between these processors. A uniform loading of the processors used and minimum amount of information to be transmitted are basic criteria of the calculation time minimization.

We suppose that the calculation time of coordinates of each node of a block is approximately the same. The calculation time of the grid in a block k is determined by its weight L_k — the number of nodes of the block. We introduce a parameter B_k, which initially equals L_k. If blocks k and l have the same slit and are in one group, then, to reduce the amount of information to be transferred between processors, they are sent to one processor, the parameter B_k is defined to equal the sum of weights of these blocks ($B_k = L_k + L_l$), and the parameter B_l is defined to equal zero. In the group under consideration, the block k can generally have common slits with several "neighbors" (for example, the block k has common slits with blocks l_1 and l_2 and the block l_1, besides the block k, has a common slit with the block l_3, and all of them are in one group). Then

$$B_k = L_k + \sum_j L_{l_j}, \qquad B_{l_j} = 0 \text{ for } j = 1, 2, \ldots .$$

Blocks of a group are numbered in decreasing order of parameter B_k, and information on them is written in an array M_1. The number of blocks with $B_k \neq 0$ determines the maximal number of processors P_1 that are necessary for calculation of this group of blocks. There arise two situations, depending on the relation between a prescribed number P and the required number P_1 of processors.

If $P \geq P_1$, then blocks with $B_k \neq 0$ are distributed among processors, and blocks with $B_l = 0$ are sent to processors on which blocks with the same slits must be calculated. In this case, processors work unevenly and some of them either stand idle or work for a short space of time.

If $P < P_1$, it is possible to load processors more uniformly. For each group, we form a matrix M_2 with the number of columns corresponding to the given number of processors and the number of rows equal to the number of blocks in the group. Elements of a column m_i of the matrix M_2 are the numbers of blocks calculated on the ith processor. The average value S of parameters B_k of all the blocks of the group is determined. If $B_k \geq S$ for some blocks, then their numbers are immediately transferred from M_1 to the array M_2, and the sum of quantities B_k for every ith processor is accumulated in S_i. The remaining blocks are distributed in accordance with the decrease in the weight B_k. If the condition $S_i + B_k < S$ is satisfied, then the block number k with the maximal value B_k is transferred from the array M_1 to M_2, and S_i changes accordingly. If, after such distribution of blocks among P processors, the array M_1 contains the block numbers for which $B_k \neq 0$, then the procedure above is iterated and the value S is increased by 1% in comparison with the value S on the previous iteration. Each iteration increases the unevenness in the loading of

processors. After distributing all blocks with $B_k \neq 0$ among processors, the block numbers with $B_l = 0$ are added to the array M_2 in columns which contain the block numbers with the same slits.

To reduce the exchange of information, we unite blocks which are on the same processor and have a common side with indication $c_{ik} = 1$. It takes few seconds of the counting time to completely analyze blocks and distribute them among processors.

After distributing blocks among processors, we send to each processor the coordinates of grid points of the initial approximation of blocks which are assigned to this processor. Secondary information about these blocks (size, the number in line, overlappings to be transferred) is also sent to the processor.

The iteration is completed if the grid has been calculated in blocks of the four groups. The last processor, as the least loaded, gathers the values of the functional in blocks from the remaining processors, summarize them, decides whether one should calculate the next iteration or finish the grid generation, and then informs the remaining processors about its decision. After ending the calculation, each processor writes its part of the grid in the file in which the coordinates of grid points of all blocks are stored in a packed form.

Results of grid generation. For completeness of the investigation, the grids were calculated in different domains with the number of nodes about 600000 and 900000. We measured the calculation time, the delay of calculations start (the work of MCS on routing), the time of reading input data and writing calculation results, the time of transferring overlappings and waiting for reception of them. Coefficients K_y of acceleration of computations (the ratio of the calculation time on one processor to the time on N processors) and coefficients $K = K_y/P$ of the effectiveness of utilization of processors depending on the number of processors are calculated.

The dependence, on the number P of processors, of the grid calculation time with taking (total) and without taking (pure) the time of reading the initial approximation into account and as well as the dependence of the time of recording the results and start delay of calculations were studied. These investigations have shown that using two or three processors sharply decreases the time expenditure. Beginning with six or eight processors, the computation time varies little. This is explained by the fact that, for specified divisions of domains into blocks, processors are loaded non-uniformly when increasing the number P. The total computation time determines the computation time on the first processor on which the block with the largest weight B_k is located. A more uniform load can be obtained if we split this block into two or more parts.

Overhead expenses (the time expenditure on start delay, reading data, and writing the computation results) for different numbers P of the given processors are approximately the same, though their shares in the total time expenditure increases as P increases, but inessentially. So, the overhead expense share on 10 processors increases about twice in comparison with two processors. It consists of the start delay $\approx 2.5\%$, reading data $\approx 12\%$, and writing results $\approx 15\%$.

The greatest coefficient of acceleration $K_{1y} = 5.2$ (without account of the overhead expenses, which corresponds to the pure computing time) in the case of a grid of 600000

nodes is obtained on $P = 10$ processors. Actually, with taking the overhead expenses into account, $K_y = 4.2$. The effectiveness coefficient of the load of processors is $K_1 = 0.51$ ($K = 0.42$), that is, when calculating this grid, processors were practically half loaded. For a grid of 900000 nodes, we obtain $K_y = 7$ on 14 processors.

The investigations show that the time spent on sending and receiving overlappings essentially depend on the uniformity of the loading of processors. The time spent on opening buffers and passage to a new group is inessential.

For comparison, we give an average time of one iteration for a grid with 600000 (900000) nodes:

- using the sequential algorithm, 19 min. 11 sec. (21 min. 28 sec.) on PC Pentium 100 MHz (150 MHz);
- using the parallel algorithm, 6 min. 7 sec. (10 min. 6 sec.) on one processor of MCS, 49 sec. (2 min. 57 sec.) on four processors, and 36 sec. (1 min. 49 sec.) on eight processors.

The time on PC is much greater than that on MCS, which is explained by another way of organization of computations. Depending on the weight, the initial approximation, and the splitting into blocks, $5 \div 15$ iterations are required to complete the grid generation.

The parallel algorithm enables us to construct optimal block-regular two-dimensional grids of large size (about hundreds of millions of nodes).

6.2.3 Automatic Generation of the Initial Approximation of a Grid

To construct an optimal grid by an iterative method, it is necessary to specify or calculate some initial approximation of it without self-intersecting cells. In [2, 5, 6] an initial approximation was constructed semi-automatically, i.e., to a great extent by hand. This requires the user to possess some practical skills.

In [13] an algorithm was elaborated for the automatic dividing of domains of complex configuration into blocks—simply connected curvilinear quadrangles. In this case, the domain boundary in the physical plane and its topological image in the curvilinear coordinates are given. The criterion of the choice of a topological image is often either the conditions of the problem being solved in the domain (for example, heterogeneity of physical processes) or the required directions of coordinate lines of the curvilinear grid (for example, the coincidence of one of the families of grid lines with the streamlines estimated). Specifying the location of nodes on the boundary, we can take into account peculiarities of the problem.

In [14] a geometrical method was elaborated for automatic generation of the initial approximation of a regular grid in simply connected domains of an arbitrary geometry using minimum input information, namely, the description of the domain boundary and the location of nodes on it.

The problems of automatic organization of the block overlapping when constructing a multiblock grid and automatic determination of lines that may be mapped ambiguously in domains of complex topology are solved in [2, 5, 6].

Using these methods, we managed to automate the optimal grid generation in two-dimensional domains of any connectivity and configuration. As a result, a universal package of programs MOPS-2A [15] was composed.

Let us consider these algorithms in detail, taking the grid constructed in the domain in Fig. 6.4 as an example.

Automatic dividing of a domain into blocks.

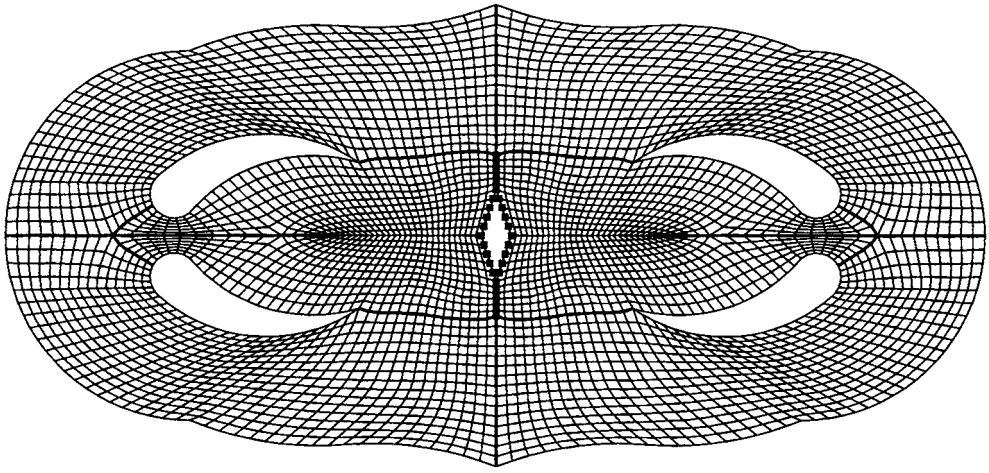


Figure 6.4: Optimal grid.

The boundary Γ of a given domain D in the physical plane (x, y) and in the curvilinear coordinates (p, q) is represented in the form of one (Fig. 6.5a) or several (Fig. 6.1b) lines, which can be both closed and non-closed. The program package MOPS-2A applies the operations *"Symmetry"*, *"Turn"*, and *"Shift"*, which essentially reduces the information processing in the cases when the domain or some of its parts are symmetric, respectively, about a line or a point, or coordinates of points of one of subdomains can be obtained by shifting another subdomain along some vector. Then the boundary is described only for

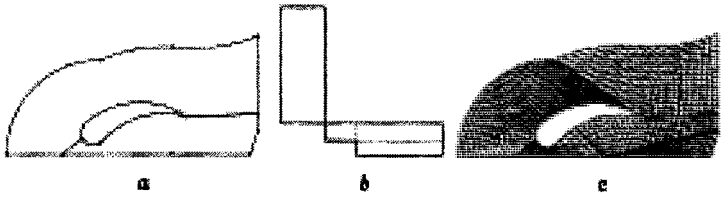


Figure 6.5: a the boundary of a fragment of the domain; b its topological image; c the grid initial approximation.

an initial fragment of the domain (Figs. 6.1 *b* and 6.5*a*) and the grid in the whole domain (Figs. 6.1*a* and 6.4) is automatically constructed by the program using minimum information. For example, for the operation *"Symmetry"* this is the coordinates of two points through which the symmetry line of this fragment passes.

The user describes lines in input data by sets of nodal points A_k (13 points in Fig. 6.5*a*), each of which is determined by four numbers x_k, y_k, p_k, and q_k — the coordinates in the physical plane and the curvilinear coordinates. We may go around the boundary starting from any point and moving in any direction. The program connects nodal points in the physical plane by segments of straight lines or arcs of circles of prescribed radii in a prescribed direction (clockwise or counter clockwise). The images of points A_k in the curvilinear coordinates are connected by segments of the straight lines $p = p_k$ or $q = q_k$. As a result, the topological image of the given domain D in the curvilinear coordinates is represented as a quadrangle, polygon, or a union of them (Ω) with sides parallel to the coordinate axes $p = 0$ and $q = 0$ (Fig. 6.5*b*).

To automatically divide the domain into blocks, information on the boundary is analyzed and the geometry of the image in the plane of curvilinear coordinates and the direction in which we should go around its boundary are determined. From nodal points of the boundary, the supporting points of its image are selected (the vertices of the polygon Ω) in which, when moving along the boundary, the passage from a grid coordinate line of one family to a coordinate line of another family is realized. From supporting points, the concavity points of the image of the boundary are selected and subdivided into 5 types (Fig. 6.6).

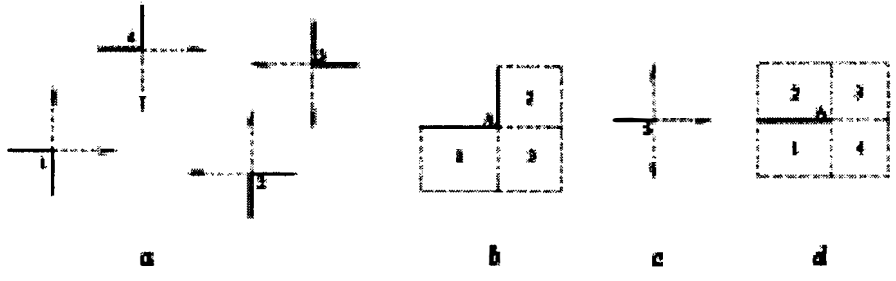


Figure 6.6: *a*, *c* types of concavity points; *b*, *d* examples of selected blocks.

The fifth type of points is the endpoints of the given non-closed curves or turning points on splits. The segment marked by the bold line can be both horizontal and vertical, and the considered point can be on either side of the segment.

Through the found points, depending on their type, the lines dividing the domain into blocks are drawn in the directions pointed by arrows. Near each point of the type 1–4, three blocks (Fig. 6.6*b*) can be selected and two of them are adjoined to the domain boundaries. Two sides of the third block are dividing lines and one of the vertices, as well as in the case of the first two blocks, is a concavity point of the boundary. Four blocks (Fig. 6.6 *d*) can be similarly chosen near points of the type 5.

The blocks are selected at the instance when the coordinates of points of three sides of the block are already known. The lengths of the block sides are determined so that the opposite sides of the block have the same length in the plane of curvilinear coordinates. Viewing all supporting points of the boundary by turns allows us to completely divide the domain into blocks and then to form a data array for the blocks obtained (coordinates of their vertices).

In the case of multiply connected domains, it is necessary to know where the grid must be constructed: inside the closed boundary curve or outside. This information is written in the input data. If the grid must be calculated outside this curve, then the direction of moving along the boundary curve is taken to be opposite.

The boundary curve of the image Ω of the given domain D of complex configuration and complex topology consists mostly of several closed curves (2 curves in Fig. 6.5 *b*). They are automatically selected and considered in the sequel as independent boundary curves.

In the plane of curvilinear coordinates (Fig. 6.5 *b*) and in the physical plane (bold lines in Fig. 6.5*c*), the dividing lines are segments of straight lines and points on them are placed uniformly by the linear interpolation method. Whether or not a dividing line intersects the domain boundary in the plane (x, y) is checked by the program. If there are intersections, additional dividing lines are drawn.

When dividing the domain into blocks is completed, the coordinates of grid points on the block boundaries become known. In every block it is necessary to construct a regular grid of the initial approximation without self-intersections.

Constructing a regular grid in a block. The method of automatic construction of a structured grid without self-intersections in the block consists in analyzing the boundary curve with the purpose to find its convexity and concavity points, to arrange grid points in the domain, and to control and improve the quality of the grid.

As in dividing the domain into blocks, a key moment of this method is to know convexity and concavity points of the boundary, but these points must here be determined in the plane of physical coordinates (x, y). To find these points, the following equation of an oriented straight line is used in the work [16]:

$$f_k(x, y) = (y_k - y_{k-1})x + (x_{k-1} - x_k)y + (x_k y_{k-1} - x_{k-1} y_k) = 0.$$

If the condition $f_k(x_{k+1}, y_{k+1}) < 0$ is satisfied, then the vertex A_{k+1} (Fig. 6.7*a*)

lies on the left of the straight line $A_{k-1}A_k$, the inner angle $\angle A_{k-1}A_kA_{k+1}$ is greater than π, the point A_k is a concavity point of the boundary. If $f_k(x_{k+1}, y_{k+1}) > 0$, then the point A_k is a convexity point. However, when using this method, we are to go around the boundary in a certain direction, namely, clockwise. This is an essential shortcoming in constructing a multiblock grid.

The convexity and concavity points can be found using the condition that these points belong to the closed domain, which does not depend in which direction we go around the boundary. If we connect points A_{k-1} and A_{k+1} by the segment of the straight line (Fig. 6.7*b*), we obtain a new domain whose boundary does not contain point A_k. For each point $A_k(x_k, y_k)$, the number n of the intersection points of the straight line $x = x_k$ and

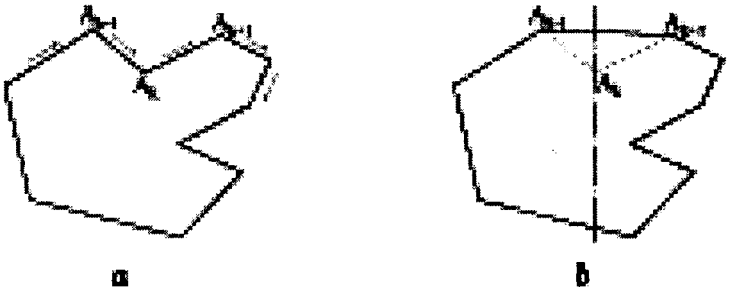


Figure 6.7: Determining the convexity and concavity points.

the domain boundary with $y > y_k$ is determined. All parts A_iA_{i+1} (i=1, 2, ...) of the boundary are checked whether they intersect or not. If n is odd, the point A_k lies inside the bounded domain and it is a concavity point; otherwise, it is a convexity point. The time of determination of points by this method increases as the number of the specified points on the boundary increases.

In the algorithm under consideration, points of the boundary are analyzed by a method, which combines the speed of the first method and universality of the second. Using the oriented straight line equation, we determine the indication of each specified point, i.e., we check whether the boundary at this point is convex or concave. This indication is searched for any point of the boundary by the second method. If the indications of a point obtained by the two methods coincide, then the boundary is described clockwise and the indications of points are defined correctly. Otherwise, the boundary is described counter clockwise, the indications of all of its points are taken to be opposite.

We use the priority principle in arranging grid points inside the block. For this purpose, a particular quantitative indication t is assigned to each of the grid points of the block. The concavity points of the boundary have the highest priority ($t = 31$), the priority of the points of its convexity is lower ($t = 21$), and the remaining points of the boundary have $t = 1$. The grid points inside the block with coordinates x, y unknown for the present have $t = 0$.

Grid lines are sequentially drawn first from the concavity points of the boundary and then from the convexity points. As a result, the domain is divided mostly into convex polygons, in which the grid is generated by the linear interpolation method.

Suppose that it is required to draw a grid line from the point T_1 (Fig. 6.8a).

In the plane (p, q), we search its nearest neighboring point T_2 with the indication $t = 0$. The coordinates of this point in the plane (x, y) are not defined yet. The position of the point T_2 coincides with one of eight positions, which are outlined in Fig. 6.8 a, and depends on the position of the point T_1 on the boundary or in the domain. If such a point exists, then from T_1 in the direction of the found point T_2 we search a point T_3 with an indication $t \geq 1$ and the known coordinates x, y. An intersection of the segment T_1T_3 and the boundary curve is checked. If T_1T_3 does not intersect the boundary curve, then the grid points are placed on the segment by the linear interpolation method and the indication $t = 1$ is assigned to them.

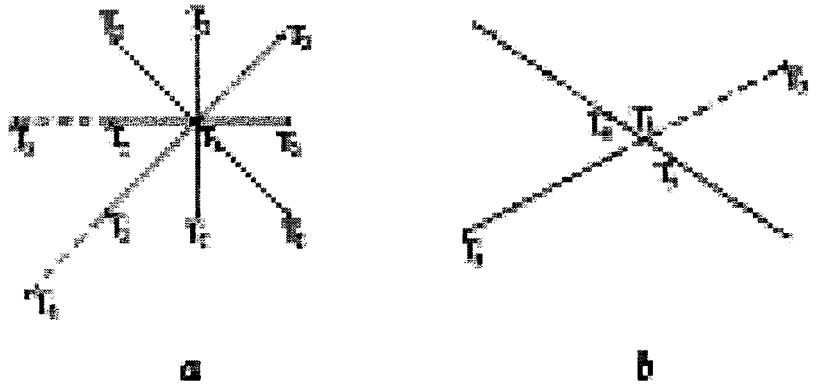


Figure 6.8: a scheme of choosing the calculation direction of grid points; b scheme of choosing the calculation direction of a temporary grid node.

If there is an intersection, then no grid points are placed in this direction, since otherwise they will be beyond the computation domain. We attempt to place the grid points in another direction, choosing the following point T_2 with unknown x, y.

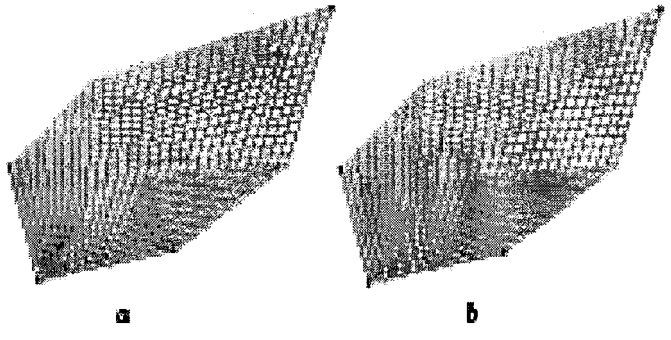


Figure 6.9: Grid for different ways of determining the intersection points of diagonals.

When placing the points in the diagonal direction, the intersection point of diagonals may not belong to the grid as the straight line T_1T_3 in the plane of curvilinear coordinates passes between points T_4 and T_5 (Fig. 6.8b) with indications $t \neq 0$. This results in an incorrect computation of the step of arrangement of points on the diagonal, and the grid can have self-intersecting cells (Fig. 6.9a). Introducing a temporary node T_3' with fractional coordinates allows us to calculate a grid without self-intersections (Fig. 6.9 b).

After calculation of a grid line, whose image is T_1T_3, the following boundary point is selected. Such a point is also chosen in the case when all points T_2 surrounding T_1 have the indication $t = 1$ or the segments T_1T_3 intersect the block boundary. Processing of the arrays of the concavity and convexity points of the boundary is continued till the complete

determination of the point coordinates in the computation domain.

The grid quality control and improvement. When constructing the initial approximation of the grid is completed, an automatic control of its quality is carried out.

A number of criteria of the quality of a grid are suggested in the work [17] and are broadened in [14]. A criterion based on the analysis of the areas of cells is used for the quality control of the grid initial approximation.

Each cell is determined by the coordinates of four its vertices (Fig. 6.10).

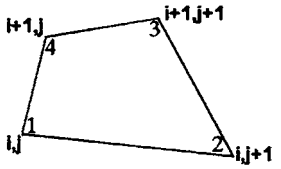


Figure 6.10: Numeration of the cell vertices.

Assigning numbers k=1, 2, 3, 4 to vertices, we calculate the areas [17]

$$2\sigma_k = (x_k - x_{k-1})(y_{k+1} - y_k) - (y_k - y_{k-1})(x_{k+1} - x_k)$$

of four triangles obtained when the diagonals of the quadrangle are drawn. In this case, $(x_0, y_0) = (x_4, y_4)$ and $(x_5, y_5) = (x_1, y_1)$.

For each cell, the signs of σ_k are checked, which determine the type of the cell. The following five types of cells are possible:

- convex cell (Fig. 6.10) if all areas $\sigma_k > 0$;
- nonconvex cell (Fig. 6.11a) or triangular cell if three areas $\sigma_k > 0$ and one area $\sigma_k \leq 0$;
- self-intersecting cell (Fig. 6.11b) if two areas $\sigma_k > 0$ and two areas $\sigma_k < 0$;
- nonconvex inverted cell (Fig. 6.11c) or triangular cell if three areas $\sigma_k < 0$ and one area $\sigma_k \geq 0$;
- convex inverted cell (Fig. 6.11d) if all areas $\sigma_k < 0$.

When constructing grids in practice, convex and nonconvex cells are admissible for solving problems. We will call them nondegenerate. The grids containing self-intersecting and inverted cells are unsuitable for calculations. We will call such cells degenerate [14, 15].

When constructing the grid initial approximation, sometimes there arise a situation when the grid generation in a domain of complex configuration cannot be completed or the constructed grid is of poor quality, i.e., the grid contains self-intersecting or inverted cells or, because going out of the domain is strictly prohibited, coordinates of some nodes are not defined in the plane (x, y) (we will call them "empty" cells) (Fig. 6.12a).

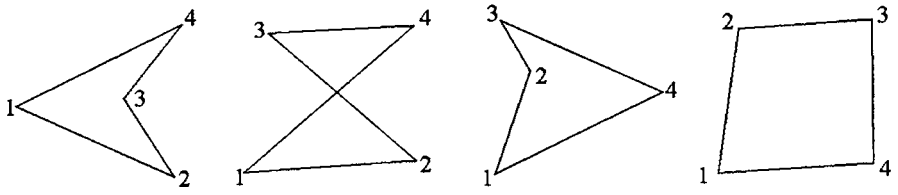


Figure 6.11: Types of cells.

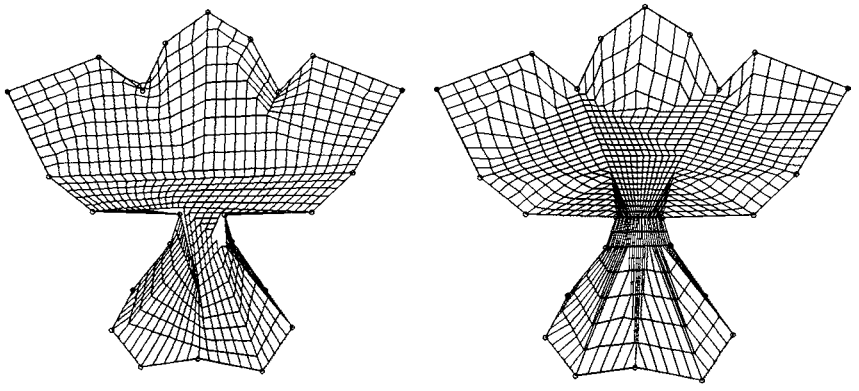


Figure 6.12: The grid initial approximation before and after improvement.

When checking the quality of a grid, information on such cells is written in the indication t of the corresponding nodes. Several methods were elaborated for improvement of these cells.

It was found that the quality of the grid initial approximation in domains of complex configurations essentially depends on the order of choosing the boundary points which are used at the first stage of calculation. To construct a qualitative initial approximation, it is required to examine several different versions of sequences of points. The following four versions are stipulated in the program: nodal points of the boundary are scanned clockwise, counter clockwise, on vertical and then horizontal lines of the boundary, and, conversely, first on horizontal and then vertical lines. From these four versions, the grid with the best quality parameters is taken.

In the case when the constructed grid is of low quality, some subdomain is chosen that contains degenerate and empty cells but has a boundary consisting only of nondegenerate cells. Indications t of all nodes lying inside this boundary are set to be zero, the convexity and concavity points are selected on the subdomain boundary, the repeated calculation of the coordinates of internal points is implemented by the algorithm above. This process is iterated under increase in the size of the subdomain till the quality of the grid becomes

satisfactory or the specified number of attempts of improving the grid is exhausted.

The presence of empty cells can result in degenerate cells, which are concentrated mainly near nodal points of the boundary concavity. A horizontal or vertical line is drawn through such boundary points, which splits the block into two blocks. The grid is anew constructed in each of the new blocks. Passing from the block to the block, the grid initial approximation is constructed in the whole domain.

The designed procedure of dividing a domain of complex geometry into blocks and constructing the initial approximation of a structured grid in blocks in combination with overlapping of blocks [2, 5, 6] enabled us to solve the problem of automatic generation of both the grid initial approximation and an optimal multiblock curvilinear grid in two-dimensional domains of any connectivity and configuration. Minimum input information is used in this construction; namely, specification of the boundary, arrangement of nodes on it in the physical plane, and the image of the boundary in the plane of curvilinear coordinates.

The time of self-dividing the domain into blocks and constructing a multiblock grid of the initial approximation with the number of nodes about one million on a computer of the type Pentium-100 does not exceeded 1 min.

6.3 Block-Structured Grids in Axisymmetric Three-Dimensional Multiply Connected Domains

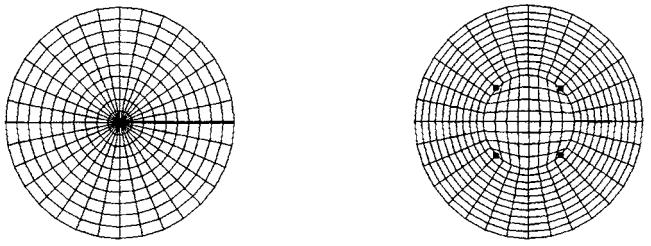
In the work [18] algorithms are designed for constructing three-dimensional curvilinear structured grids in simply connected domains, obtained by rotation of a directrix about an axis through an angle ϕ ($\phi_0 \le \phi \le \pi$). The grids are constructed without singularities (there are no degeneration on the axis and self-intersections in the domain) [19].

In this work, three-dimensional block-structured grids are constructed in axisymmetric domains of any connectivity in the case when the rotation angle ϕ of the directrix around the axis is equal to 360°. Such grids are required when solving problems of mathematical physics in three-dimensional axisymmetric domain with nonsymmetric boundary conditions (for example, gasdynamic flow around axisymmetric bodies at nonzero angle of attack).

A curvilinear grid in a domain is proposed to be constructed by rotation of a two-dimensional radial section of the domain about the symmetry axis. This section can be simply or multiply connected with one-valued or multivalued mapping into the plane of curvilinear coordinates. A two-dimensional structured or block-structured optimal grid should be generated in the section by the use of algorithm MOPS-2A.

For automatic generation of a three-dimensional grid it is necessary to set the number K_s of sectors into which the domain of rotation should be divided.

If a three-dimensional grid is constructed using only rotation of a section, then on the symmetry axis we obtain an ambiguity of the mapping like O and C grids (Fig. 6.13*a*). To eliminate this, we must change the grid in blocks or parts of blocks adjacent to the symmetry axis (Fig. 6.13*b*).



a *b*

Figure 6.13: Grid in the cross section of the central part of a three-dimensional domain with a singularity and after elimination of it.

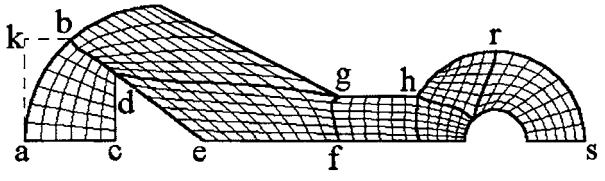


Figure 6.14: Radial section of a three-dimensional domain.

The lateral faces of blocks can be represented as conical (bd) and cylindrical (gh) surfaces or their combination (rs). Each of the bases can be a circle (cd), spherical (ab) or conic surface (ed), or more complicated surface whose directrix of rotation is a broken line (combination of the lateral faces of truncated cones) (gf) (Fig. 6.14).

Let us first consider a rather simple structure of the block—a truncated cone with circle bases. To reconstruct the grid, we "cut out" a part of the block adjacent to the symmetry axis. In the remaining part of the block, if such a part exists, the grid is constructed by rotation of the segment of the original section of the domain that belongs to this part of the block.

When constructing the grid in the central part of the block, we renumber the boundary points of its bases so that this boundary in the plane of curvilinear coordinates is represented as a curvilinear square (Fig. 6.13 *b*). To maintain the symmetry in the grid reconstructed, the number K_s of given sectors for generation of a three-dimensional grid, determining the number of the boundary points of the block base in which the grid is reconstructed, must be multiple of 8. Otherwise, in handling input data the nearest $K_c > K_s$ multiple of 8 will be automatically chosen. A grid close to a uniform one is constructed in the base of the central block by linear interpolation. Points are placed first on the vertical and horizontal symmetry axes of the circle and then in the left top quadrant. For maintaining the symmetry, the grid in the top of this quadrant is symmetrically mapped with respect to the diagonal to its bottom. Then the grid obtained in this quadrant is symmetrically mapped to other quadrants. As a

result, we get a symmetric grid of size $L = (K_c/4 + 1) \times (K_c/4 + 1)$ in the base of the central block.

Having performed analogous constructions in cross sections of the block for specified $z = z_k$, we obtain in the central part a grid close to a uniform one and without singularities.

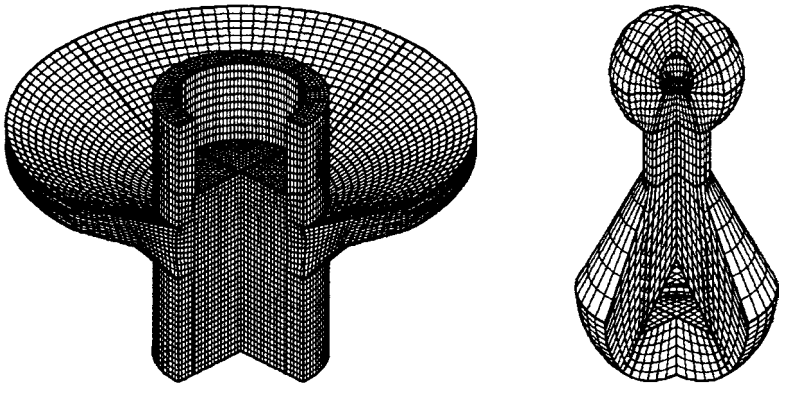
Let now the block base is the lateral face of a right circular cone (de in Fig. 6.14). We will generate a grid on such surface as follows. We represent the base dc of this cone in the plane curvilinear coordinates (pq) as a square in which a uniform orthogonal grid will be generated. In the plane of physical coordinates, we construct a circle of radius dc and project the grid points of the generator de to the axes $x = 0$ and $y = 0$. In the general case, we can obtain a nonuniform grid on the axes. Then, on lines $q = \text{const}$ in the left top quadrant, we arrange points so that the ratios of the distances between grid points are the same as on the axes. To ensure the symmetry, the grid in the top of this quadrant is symmetrically mapped with respect to the diagonal to its bottom. The grid obtained in this quadrant is symmetrically mapped to other quadrants. As a result, we get a symmetric grid in the cone base, which is projected to the lateral face of the cone.

The composed surface of the cone base (for example, ab in Fig. 6.14) can be represented as a combination of the lateral faces of truncated cones $(K = \sum K_i)$ with heights $h_i = z_i - z_{i-1}$ $(i = 1, 2, ...)$.

To construct a grid on such surface, we project the grid points of the section line (ab), corresponding to this surface, to the ordinate axis (ak). In the plane (xy), we construct a circle of radius ak and a grid in it by the method described above. To project the obtained grid to the base face, it is necessary to take into account that this surface is a combination of the lateral faces of truncated cones and to rightly choose the required cone.

Having performed analogous constructions for the section lines inside the block for specified $z = z_k$, we obtain in the central part of the domain a grid close to uniform one and without singularities.

The coordinates of the grid points are stored in a block form. When solving problems



a *b*

Figure 6.15: Three-dimensional grids in domains with simply connected radial sections.

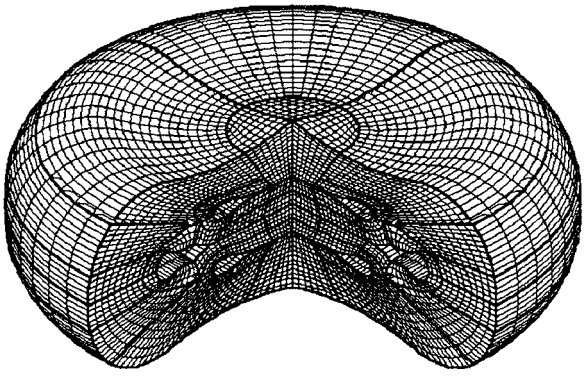


Figure 6.16: A three-dimensional grid in a multiply connected domain.

on these grids, connections between blocks are automatically organized.

A three-dimensional grid in a simply connected domain is presented in Fig. 6.15 *a* and in multiply connected domains of simple and complex topologies in Figs. 6.15 *b* and 6.16. The radial section in Fig. 6.14 was used for designing the grid presented in Fig. 6.15 *b*. To make the grids more viewable, we cut a quarter out of each domain. The bold lines in Figs. 6.14–6.16 mark the block boundaries.

6.4 Using Optimal Grids

A non-orthogonal optimal two-dimensional curvilinear grids were used to solve complicated gas dynamics problems. Using these grids, a method of calculating parameters of stationary vortical flows of compressible and incompressible fluid in axisymmetric and plane parallel simply connected channels of complex configuration is elaborated [2, 12, 20]. The method is realized by the program package SOKOL. It enables us to reliably diagnose zones of the subsonic part of channels in which vortices are formed, to determine how the gas-dynamic parameters and peculiarities of the channel design influence the formation of the pressure and velocity field and the formation of stationary large scale vortices in the flow. As a result, using various constructive techniques, we can exclude, even at the stage of designing, the vortices giving rise to acoustic oscillations of pressure in these channels.

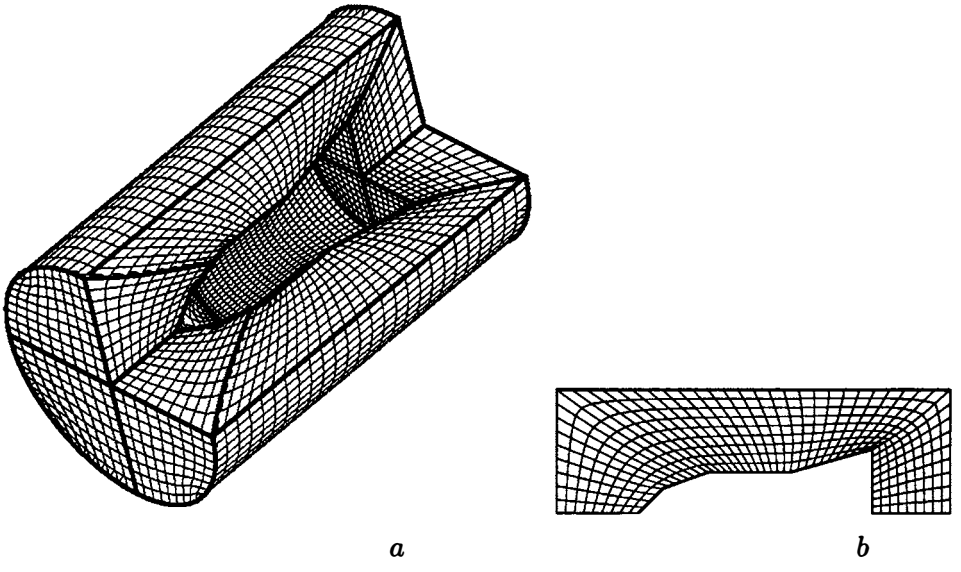
The methods designed and the program SPECTR for calculation of parameters of acoustic oscillations propagating in a moving media in these channels enabled us to calculate enough effectively acoustic parameters of combustion chambers [20]–[22]. The effect was revealed that frequencies perceptibly decrease when vortical zones are present in the flow.

It is precisely the use of optimal curvilinear grids that enabled us to free ourselves of the essential restrictions on the class of the considered geometries of channels, and the high approximative quality ensured a success in constructing rapid iterative processes for calculating flows and acoustic oscillations in channels of complex configuration. The grid generation time is 5–10% of the computation time of the problem. The more complicated

structure of the gas flow the less percentage.

The grids constructed by this technique were successfully used for solving problems of the gas flow around axisymmetric spatial bodies, in particular, for calculating counter and overtaking diffraction on a sphere [23]. A three-dimensional grid was designed as a union of two-dimensional optimal grids. These grids are constructed on the evolvent of the lateral face of circular cones which are one of the families of coordinate surfaces in the given domain.

The three-dimensional grids described in Section 6.3 were used for solving problems of stationary flows of an ideal gas around axisymmetric bodies at a nonzero angle of attack. As an example, the results of calculation of the flows around a body composed of cones and a cylinder are given. A three-dimensional grid in the domain was constructed by rotation of the section presented in



a *b*

Figure 6.17: *a* a three-dimensional grid in the computation domain; *b* the original two-dimensional grid in its radial section.

Fig. 6.17*b*. To make the grid viewable, a quarter of the domain in Fig. 6.17 *a* was cut out. Below, the results of calculations are presented for the cases when the flow was both subsonic with the Mach number $M_0 = 0.5$ and supersonic with the Mach number $M_0 = 2$. The velocity vector of the flow lies in the plane xz at the angle of attack $\alpha = 15°$.

Figures 6.18 and 6.19 show isobars in sections xoz and xoy obtained for the subsonic flow around a body. The sections xoz and xoy are symmetry planes of the streamlined body. Since the velocity vector of the flow lies in the plane xz, the pattern of the gas flow in the section xoz is symmetric.

Figure 6.20 shows isolines of the Mach numbers in the section xoz obtained for a supersonic flow around a body.

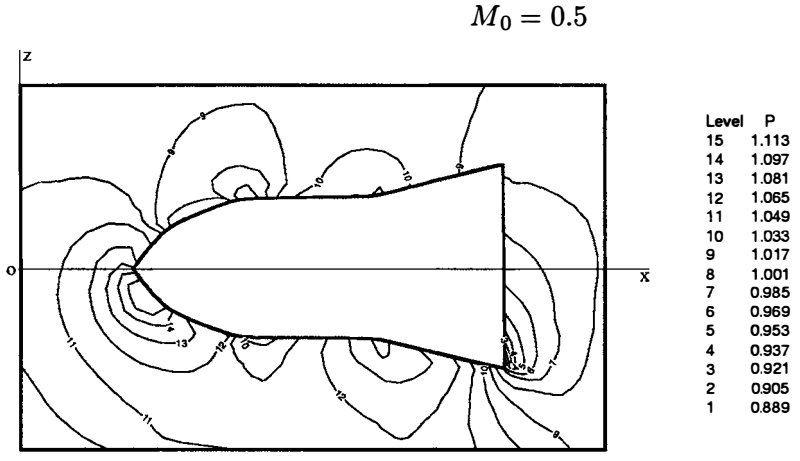


Figure 6.18: Isobars in section xoz for the subsonic flow.

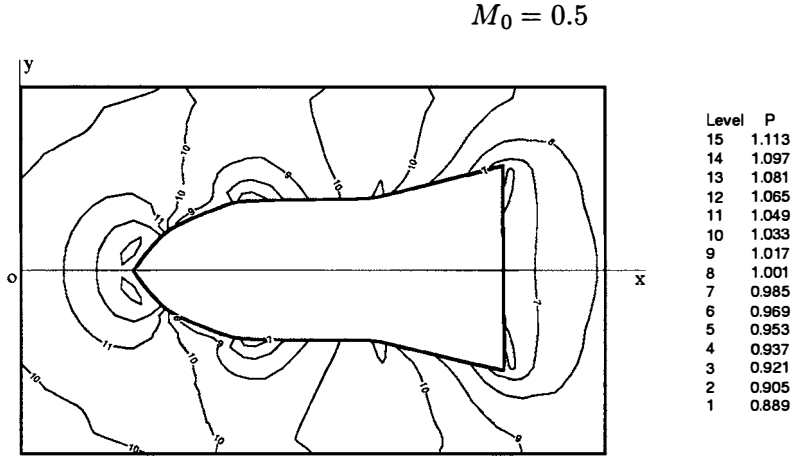


Figure 6.19: Isobars in section xoy for the subsonic flow.

The numbers at lines denote the numbers of isolines. The corresponding values of pressure P and the Mach numbers are given in the tables.

Calculations of gasdynamic flows were carried out by programs elaborated in the Keldysh Institute of Applied Mathematics of Russian Academy of Sciences.

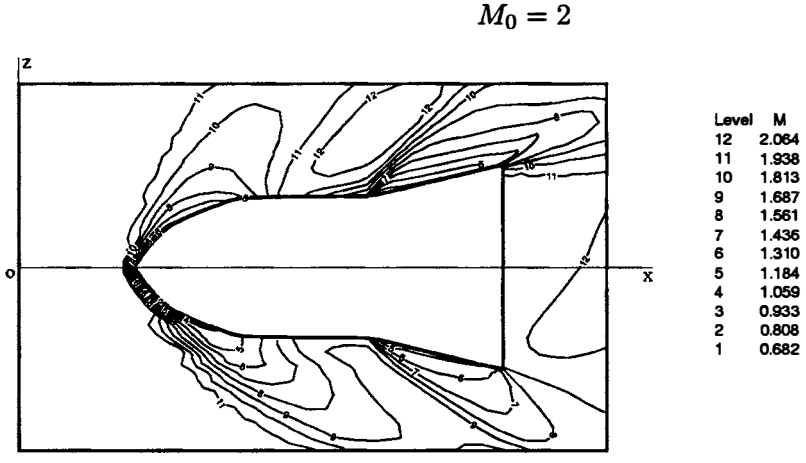


Figure 6.20: Isolines of the Mach numbers in section xoz for the supersonic flow.

6.5 Conclusion

The numerous calculations show that the hyperbolicity of the system of equations solved does not give rise to discontinuities. No additional singularities appear when passing to a new system of coordinates, though the general theorems on smoothness, existence, and uniqueness of the solution of the system of Euler-Ostrogradsky equations are not derived [2]. However, the experience in using optimal grids based on the approach proposed shows that the grids constructed have high computational and approximative advantages. Their application allows us to solve complicated gas dynamics problems with high degree of accuracy.

Automatization of constructing both two-dimensional and three-dimensional grids in specified domains makes it possible to use MOPS-2A for solving nonstationary problems.

Acknowledgment

This work was supported by the Russian Foundation for Basic Research, project nos. 02–01–00236 and 02–01–00047, and the Program of Fundamental Research of the Presidium of the Russian Academy of Sciences.

Bibliography

[1] Serezhnikova, T.I., Sidorov, A.F., and Ushakova, O.V., On one method of construction of optimal curvilinear grids and its applications, *Soviet J. Numer. Anal. Math. Modelling*, 1989, vol. 4, no. 2, pp. 137–155.

[2] Khairullina, O.B., Sidorov, A.F., and Ushakova O.V., Variational methods of construction of optimal grids, in *Handbook of Grid Generation*, Ed. by J.F. Thompson, B.K. Soni, and N.P. Weatherill, CRC Press, Boca Raton etc., 1999, pp. 36-1– 36-25.

[3] Thompson, J.F., Warsi, Z.U.A., and Mastin, C.W., *Numerical Grid Generation: Foundations and Applications*, North–Holland, New York, 1985.

[4] Handbook of Grid Generation, Ed. by J.F. Thompson, B.K. Soni, N.P. Weatherill, CRC Press, Boca Raton etc., 1999.

[5] Khairullina, O.B., Method of constructing block regular optimal grids in two-dimensional multiply connected domains of complicated geometries, *Russ. J. Numer. Anal. and Math. Modelling*, 1996, vol. 11, no. 4, pp. 343–358.

[6] Khairullina, O.B., A method of constructing block regular optimal grids in two-dimensional multiply connected domains of complex configuration (MOPS-2a), in *Collection of Sci. Works*, UrO RAN, Ekaterinburg, 1998, 56 p.

[7] Artyomova, N.A., Khairullin, A.F., and Khairullina, O.B., Constructing optimal grids in multiply connected domains of complex topology on multiprocessor computers, in *Algorithms and Program Tools of Parallel Computations* , IMM UrO RAN, Ekaterinburg, 1998, issue 2, pp. 22–38.

[8] Sidorov, A.F., Khairullina, O.B., and Khairullin, A.F., Parallel algorithms of generation of optimal multiblock structured two-dimensional and three-dimensional grids of large size, *Numerical Grid Generation in Computational Field Simulation: 6th International Conference, University Greenwich*, July 6-9, 1998, ISGG, Greenwich, 1998, pp. 759–768.

[9] Khairullin, A.F., Khairullina, O.B., and Artyomova, N.A., Computation of optimal grids of large size in multiply connected domains by using the distributed memory

of MVS-100, in *Algorithms and Program Tools of Parallel Computations* , IMM UrO RAN, Ekaterinburg, 1999, issue 3, pp. 239–251.

[10] Artyomova, N.A., Khairullin, A.F., and Khairullina, O.B., Parallel algorithm of computation of optimal grids, *Vychisl. Tekhnol.*, 2001, vol. 6, no. 2, pp. 3–13.

[11] Zabrodin, A.V., Levin, V.K., and Korneev, V.V., The massively parallel computer system MVC-100, in *Third International Conference on Parallel Computing Technologies*, PaCT-95, St. Petersburg, 1995, pp. 341–355.

[12] Khairullina, O.B., Modelling subsonic vortex gas flows in channels of complex geometries, *Russ. J. Numer. Anal. and Math. Modelling*, 1998, vol. 13, no. 3, pp. 191–218.

[13] Khairullin, A.F. and Khairullina, O.B., Dividing the domain into blocks in constructing a multiblock curvilinear grid, in *Algorithms and Program Tools of Parallel Computations*, IMM UrO RAN, Ekaterinburg, 2001, issue 5, pp. 261–274.

[14] Khairullin, A.F. and Khairullina, O.B., Automatic generation of the initial approximation of a curvilinear grid, in *Algorithms and Program Tools of Parallel Computations* , IMM UrO RAN, Ekaterinburg, 2000, issue 4, pp. 273–286.

[15] Khairullin, A.F. and Khairullina, O.B., Automatic generation of the initial approximation of a block-structured curvilinear grid, in *Vychisl. Tekhnol.*, 2002, vol. 7, no. 5, pp. 96–107.

[16] Gasilova, I.A., Algorithm of automatic generation of the initial approximation of a curvilinear grid for the star type domains, *Voprosy Atomnoi Nauki i Tekhniki. Ser.: Mat. Model. Fiz. Protsessov*, 1994, issue 3, pp. 33–40.

[17] Prokopov, G.P., On the comparison between algorithms and programs for constructing regular two-dimensional difference grids, *Voprosy Atomnoi Nauki i Tekhniki. Ser.: Mat. Model. Fiz. Protsessov*, 1989, issue 3, pp. 98–108.

[18] Bronina, T.N., Gasilova, I.A., and Ushakova, O.V., Algorithms of constructing three-dimensional structured grids, *Zh. Vychisl. Mat. Mat. Fiz.*, 2003, vol. 43, no. 6, pp. 875–883.

[19] Ushakova, O.V., Conditions of nondegeneracy of three-dimensional cells. A formula of a volume of cells, *SIAM J. Sci. Comput.*, 2001, vol. 23, no. 4, pp. 1273–1289.

[20] Kokovihina, O.V., Sidorov, A.F., and Khairullina, O.B., Mathematical modelling of gas-dynamic and acoustic effects in combustion chambers, in Theory of Combustion of Powder and Explosives, Ed. by A.M. Lipanov, *Nova Sci. Publ.*, New York, 1996, pp. 191–202.

[21] Kokovikhina, O.V., Solving method for the problem of propagating acoustic oscillations in channels of complex form, *Mat. Model.*, 1998, vol. 10, no. 11, pp. 47–62.

[22] Kokovikhina, O.V., Amplitudes of parameters of acoustic oscillations in axisymmetric channels, *Mat. Model.*, 2001, vol. 13, no. 11, pp. 88–96.

[23] Martyushov, S.N., Program package MODAMS for calculating ideal gas flows around bodies, *Vychisl. Tekhnol.*, 1992, vol. 1, no. 2, part 2, pp. 232–239.

Chapter 7

ANISOTROPIC GRID ADAPTATION APPLIED TO AERODYNAMIC PROBLEMS

Alexander M. Sorokin and Natalia A. Vladimirova
Central Aerohydrodynamic Institute (TsAGI)

Two anisotropic grid adaptation methods are considered. The first one is the algorithm of constrained Delaunay triangulation with specified stretching of triangular cells adjacent to the boundary applied in multiply connected regions. The grid generation procedure incrementally inserts the nodes to make the distribution of circum-circles around triangular elements smooth and meet the requirement imposed on the cells stretching adjacent to specified curves. The generated grids were used in Navier-Stokes numerical simulation of high Reynolds number turbulent viscous flows around isolated and multi-element airfoils.

The second method is anisotropic edge refinement adaptation technique of three-dimensional unstructured grids. The adaptation procedure identifies three directions of adaptation at each grid node, performs the refinement of grid edges aligned to these directions and reconnects edges. The interpolation based error indicator at an edge is determined as a weighted sum of the first and second order derivatives of monitor function along an edge direction. The edge refinement utilizes this error indicator. The developed adaptation algorithm was applied to convection-diffusion problems. The solutions to these problems simulate three-dimensional curved viscous wakes with large gradients of flow parameters. The explored dependencies of numerical errors via the number of grid nodes showed good correlations with theoretical predictions. The industrial application to the Euler transonic flow over a wing with winglets is considered.

7.1 Introduction

The accuracy of the numerical solution with large gradients calculated on the uniform grid can be significantly improved with the optimal placement of grid nodes. The tessellation of the given set of nodes is not unique and nodes connectivity signi-ficantly influences

the accuracy of the numerical scheme. If the problem parameters vary differently in the different directions, the optimal grid should comprise the anisotropic grid elements.

Two different anisotropic grid adaptation techniques are presented in this paper. The first one constructs the constrained Delaunay triangulations with specified cell stretching in the regions with large gradients of flow parameters. This algorithm does not rely on flexible push button technology, but it produces high quality grids that meet all the requirements imposed by employed numerical Navier-Stokes schemes. These requirements are:

- the Delaunay grid property
- the grid steps small enough to resolve the flow parameters in viscous dominated flow regions
- the control of maximal angle in triangular cells
- the smooth transition between isotropic and anisotropic parts of grid
- reasonable grid quality

The other method utilizes the edge refinement technique and can be the base for automatic grid generation technology. The grid is generated through iterations consisting in the following steps:

- computation of flow parameters on the current wing
- calculation of edge-based error indicators
- refinement and derefinement grid adaptation according to the values of error indicator

7.2 Generation of Constrained Delaunay Triangulations with Specified Cells Stretching near the Boundaries

The Navier-Stokes numerical simulation of high Reynolds number turbulent viscous flows needs the regular tessellations with extremely high cell aspect ratio in viscous dominated flow regions. Moreover, most of the numerical schemes prefer the Delaunay triangulation because the Delaunay property improves and for some problems even ensures the stability of numerical schemes. The triangular tessellation of the given region is considered to be Delaunay if interior of the circum-circle of each triangle contains no points of the triangulation. Everywhere in this paper the boundary constrained Deluanay triangulations are considered, i.e. points that are occluded by boundary edges are not taken into account in the in-circle test. The aspect ratio of the cell is the ratio of its circum-radius to in-radius.The Delaunay property in highly stretched grids imposes tough requirements on the grid generation technique. Although the Delaunay criteria can be easily met with the edge reconnection, this procedure generally seriously corrupts the quality of the highly stretched grids. The angles

in triangular elements may be arbitrary increased and the high connectivity at isolated nodes may appear.

Generally the viscous dominated flow regions close to the boundary are the crucial regions in calculations on Navier-Stokes flows. The straightforward approach to this problem is to use the triangulated rectangular grid near boundaries and isotropic grid in the rest part of the flow field [1]. Generally this tactics introduces sudden and considerable discrepancies in the cells shape and size on the boundary of the isotropic and anisotropic parts of the computational region. Two alternate strategies of isotropic grid generation can be outlined. One of them incrementally inserts nodes in the centers of circum-circles of mesh triangles to improve the grid quality. The grid quality is defined with the size and shape of grid cells [2]-[5]. The other technique starting from the region boundaries incrementally cuts off the triangles from the computational regions, until the region is exhausted [6]. One of the possible approaches to generation of anisotropic grids is to apply the isotropic grid generation technique in locally transformed space with specified metric of Euclidean space [7]. Unfortunately, this approach does not control the grid quality.

The developed approach specifies the cells stretching near boundaries of the region and adjacent to the cuts inside it. The initial triangulation of the region based on the boundary nodes is incrementally enriched with the nodes to meet the requirement on the stretching of adjacent to the boundary triangular cells and to produce the smooth distribution of radii of circum-circles of mesh triangles. The developed technique does not employ the tactics of node insertion in the centers of circum-circles with maximal normalized radius [7]. In the developed algorithm the gradient vectors of circum-circle radii are calculated numerically at the grid nodes and new nodes are injected in the gradient direction from the nodes with maximal module of the gradient vector. The new node is injected from the old one at the step that assured the smooth distribution of circum-circle radii. The same algorithm was used in the isotropic and anisotropic parts of the mesh. The notable feature of this algorithm is that it may produce the highly stretched grid structures close to the triangulated orthogonal rectangular grids in viscous dominated flow regions. The algorithm controls the maximal angle value of in the grid triangles.

7.3 The Algorithm of Anisotropic Grid Generation

The algorithm of the anisotropic grid generation is based on the following steps:

- The computational region boundaries and the cuts defined by the streamlines starting from the sharp ends of the bodies are discretized.

- The aspect ratio of the cells close to the bodies and the cuts is specified; actually the length of the edges entering the body and the cuts are prescribed.

- The initial tessellation of the region is produced. The tessellation contains only the nodes on the boundaries and the cuts.

- The gradient of circum-circles radii (P_x, P_y) is calculated integrating over the Voronoi control volume. The gradient of circum-circles radii at node $i = (x_i, y_i)$ is obtained by integrating over the appropriate Voronoi control volume

$$P_x = \left(\sum_{n=1}^{N} C_{in} R_{in} \Delta x_{in} \right) \left[\sum_{n=1}^{N} C_{in} (\Delta x_{in}^2 + \Delta y_{in}^2) \right]^{-1}$$

$$P_y = \left(\sum_{n=1}^{N} C_{in} R_{in} \Delta y_{in} \right) \left[\sum_{n=1}^{N} C_{in} (\Delta x_{in}^2 + \Delta y_{in}^2) \right]^{-1},$$

the node i shares the edges $(i, n), n = 1, N$. The value of R_{in} is equal to half of the sum of circum-circles radii around triangles sharing the edge connecting the nodes i and n, $\Delta x_{in} = x_n - x_i, \Delta y_{in} = y_n - y_i$. $C_{in} = L_w / L_e$, where the value of L_w is equal to the length of the side of Voronoi cell, corresponding to the edge (i, n), and the value of L_e being equal to the length of the edge (i, n).

- Some number of nodes (the parameter of the algorithm), where the module of the gradient of circum-circle radius is the largest is chosen. The new nodes are inserted near the chosen nodes. The location of the new nodes is defined at the step along the gradient direction from the old ones. This tactic means the mesh enrichment in the regions where it is mostly rarefied. The step is equal to the normalized length of the edge entering the node and aligned to the direction opposite the gradient vector. The normalizing value defines the ratio of grid stretching. If the node with large gradient of circum-circle radii locates on the boundary or on the cut, then the step value is chosen to produce the specified cell stretching on this boundary or the cut.

- Before insertion of the new node the check is made whether the presence of this node would cause the illegal grid structures such as grid overlapping, when the new node locates outside the region or lies too close to the old nodes. If it is so, the insertion of new node is forbidden.

- The edge reconnection is fulfilled to meet the Delaunay property.

- If all nodes that are candidates for insertion are rejected, the grid generation is considered to be finished. If not so, the gradient of the circum-circle radii is calculated at the nodes, where it was changed by the node insertion and edge reconnection, and the next iteration step is made.

7.4 Computational Results

The developed grid generation technique was used in Navier-Stokes numerical si-mulation of high Reynolds number turbulent viscous flows around isolated and multi-element airfoils.

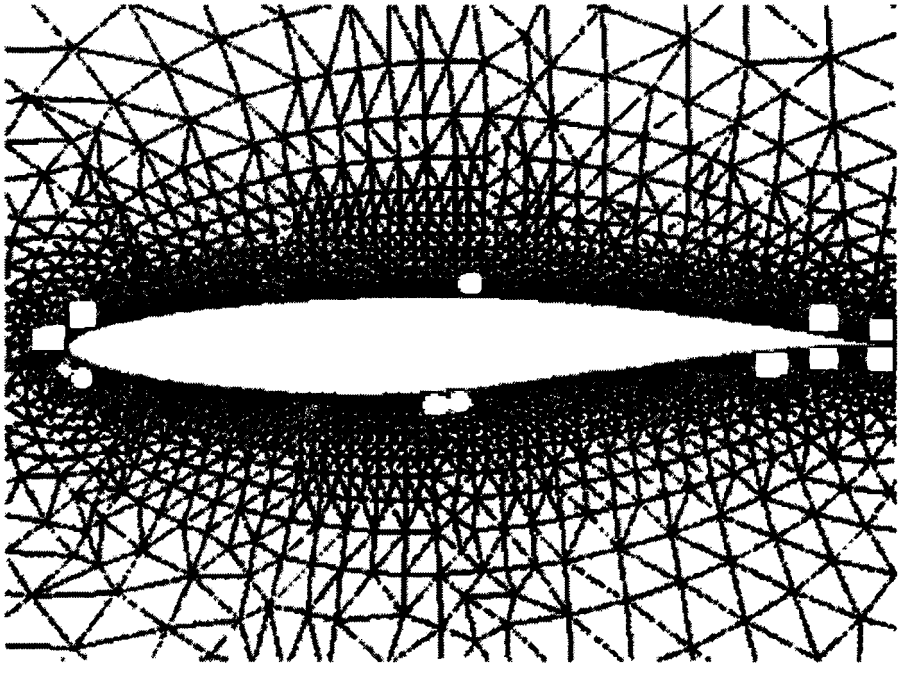


Figure 7.1: Grid around RAE-2822 airfoil, 5138 nodes

7.4.1 RAE 2822 Airfoil

The Navier-Stokes grid generated around RAE-2822 airfoil is shown in Figure 7.1. The cell aspect ratio, being equal to 1000, was specified only at the airfoil surface. The grid comprised 5138 nodes. The maximal angle in triangular cells did not exceed 160°. The Navier-Stokes calculation with incorporated Spalart-Allmaras turbulence model [8] was performed with FLUENT package on this grid at $M = 0.725$, $\alpha = 2.92°$, $Re = 6.5 \cdot 10^6$. Figure 7.2 displays good correlation of pressure distribution on the airfoil surface between numerically predicted values and experimentally determined data [9].

7.4.2 Two-Element Airfoil GA(W)-1

The important feature of the developed algorithm is the ability to create highly stretched Delaunay grid not only adjacent to the boundaries of the computational domain but to curves inside the region. The illustration of this feature is the case of Navier-Stokes grid around airfoil GA(W)-1. The cell aspect ratio equal to 1000 was specified both on airfoil surface and on the wakes. The flow at $M = 0.21$, $\alpha = 0$, $Re = 2.2 \cdot 10^6$ was considered. At first the grid with specified density only near airfoil boundaries was generated (see Figure 7.3) and the Navier-Stokes code was run on it at the considered flow regime. After that the cuts were generated as a stream lines in the calculated flow field going from the sharp corners of the body. Then the grid with grid density near body and defined cuts was produced. This

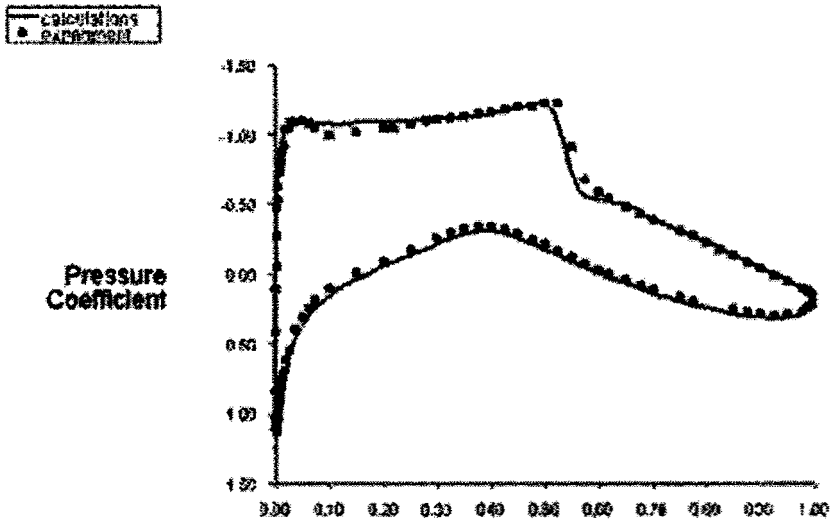


Figure 7.2: Comparison of pressure distribution on the RAE-2822 airfoil surface between numerically predicted values and experimentally determined data ($M = 0.725$, $\alpha = 2.92°$, $Re = 6.5 \cdot 10^6$)

grid is depicted in Figure 7.4. It consists of 15463 nodes. The cells adjacent to the body have the aspect ratio equal to 1000. The maximal angle in triangular elements does not exceed 160°. Figure 7.5 presents the fragment of this grid near flap. Figure 7.6 shows the Mach number distribution around this airfoil obtained with FLUENT package on this grid at $M = 0.21$, $\alpha = 18°$, $Re = 2.2 \cdot 10^6$. The algorithm produces grids of poor quality when fronts of dissimilar scales are encountered. That is why the grid density on the closely placed curves should be fitted to each other.

7.4.3 Four-Element Airfoil

This example demonstrates the capabilities of the algorithm for complex multiply connected geometries. The flow parameters were equal $M = 0.2$, $\alpha = 8.09°$, $Re = 2.85 \cdot 10^6$. Besides gaps between close surfaces, close locations of multiple boundaries and wakes, some elements of this airfoil have blunt trailing edges. The wake from such blunt trailing edge is modeled with two streamlines going from its sharp ends. At first the grid with prescribed density only near airfoil boundaries was generated and Navier-Stokes calculation at the considered flow parameters were performed. After that, the streamlines going from the body sharp ends were constructed according to computational flow fields. Then the gird with cell aspect ratio equal to 1000 near body surface and wakes was generated. The grid comprised 41813 nodes. The maximal angle in grid triangles does not exceed 175°. The general view of the grid is shown in Figure 7.7. The grid fragments are depicted in Figure 7.8, Figure 7.9 and Figure 7.10.

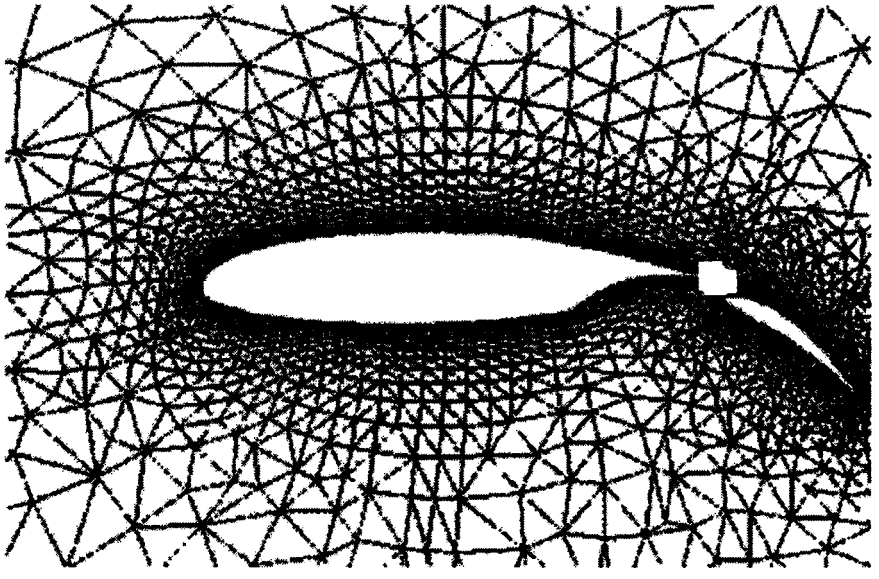


Figure 7.3: Grid around GA(W)-1 airfoil without adaptation to the wake, 3091 nodes

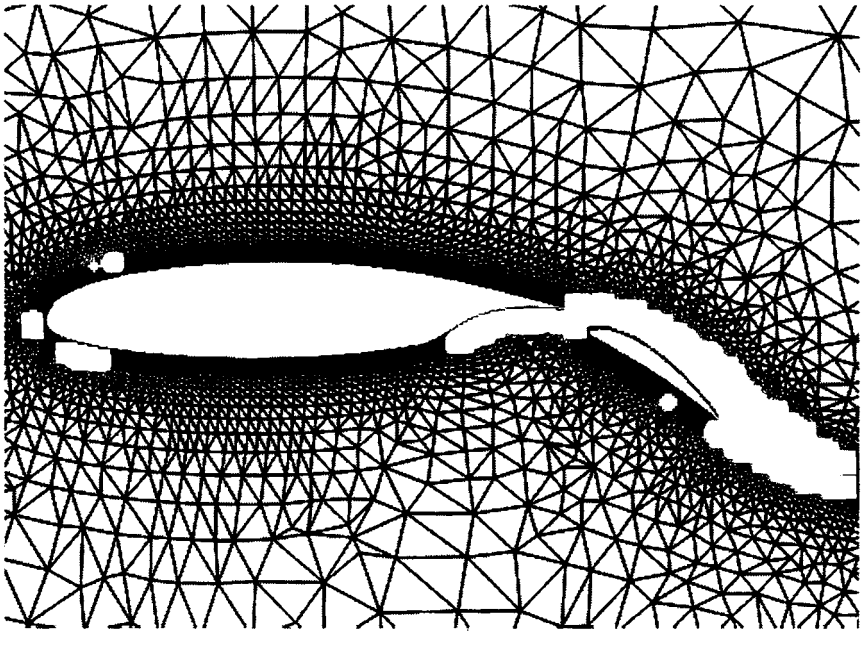


Figure 7.4: Grid around GA(W)-1 airfoil with adaptation to the wakes, 15463 nodes

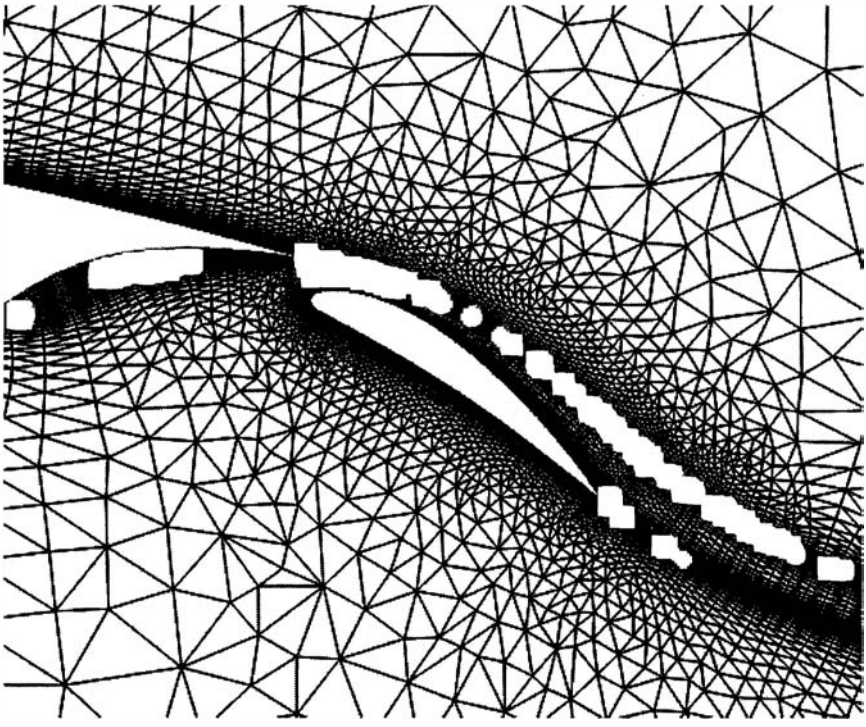


Figure 7.5: Fragment of the grid around GA(W)-1 airfoil near the flap

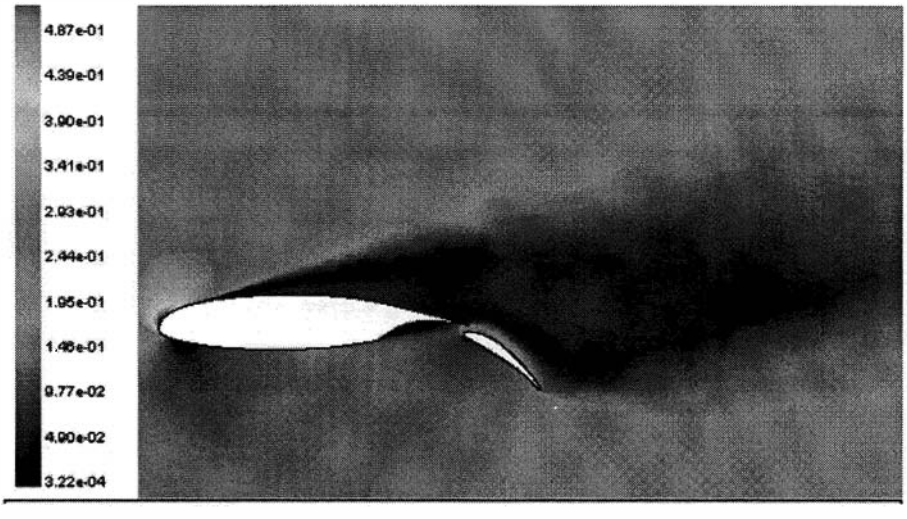


Figure 7.6: Mach number distribution around GA(W)-1 airfoil, $M = 0.21$, $\alpha = 18°$, $Re = 2.2 \cdot 10^6$

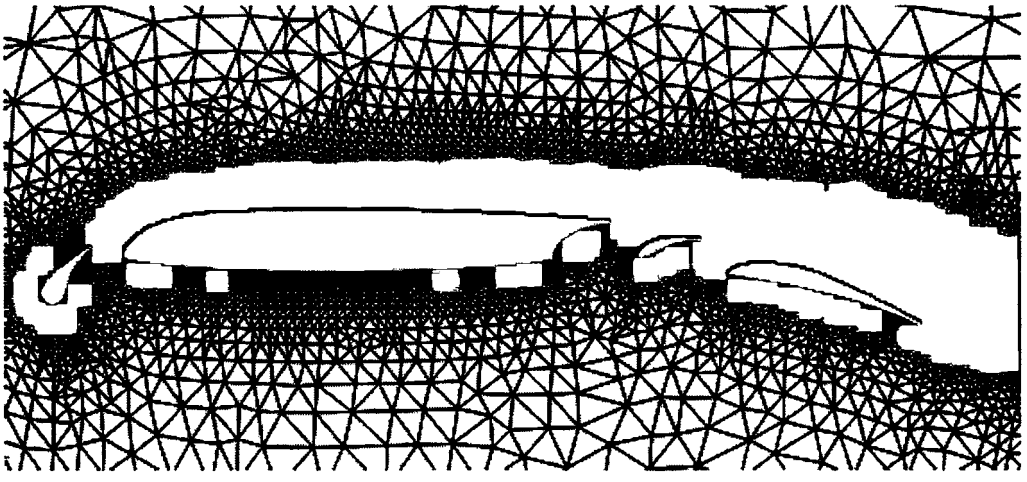


Figure 7.7: Grid around four-element airfoil, 41813 nodes

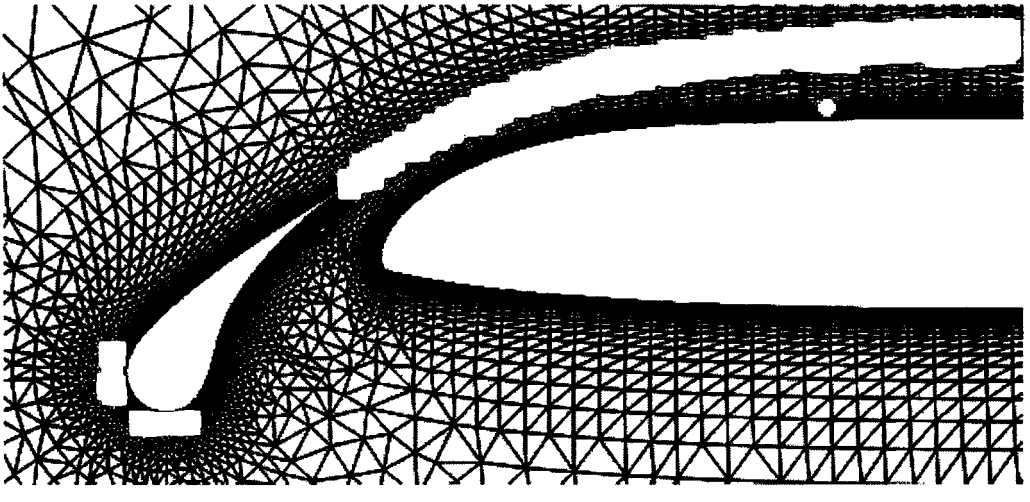


Figure 7.8: Grid fragment near slat

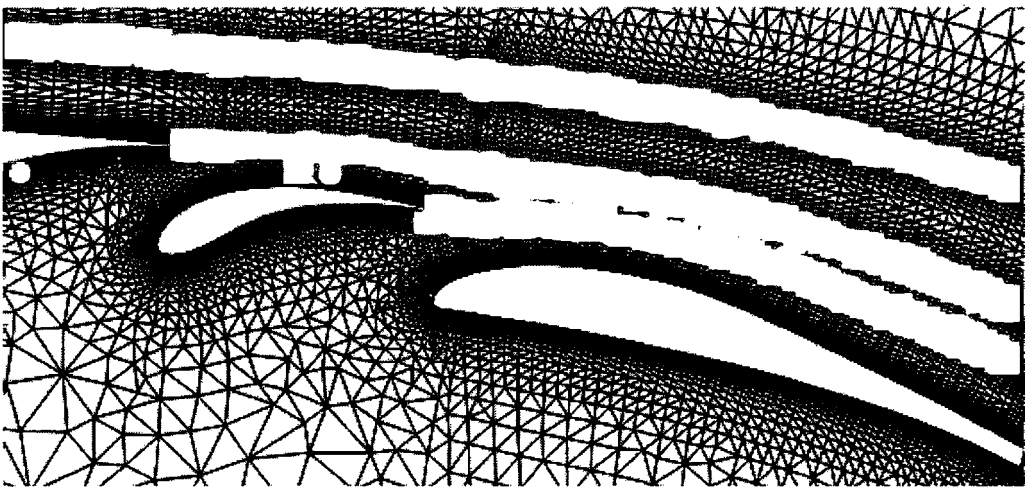


Figure 7.9: Grid fragment near flaps

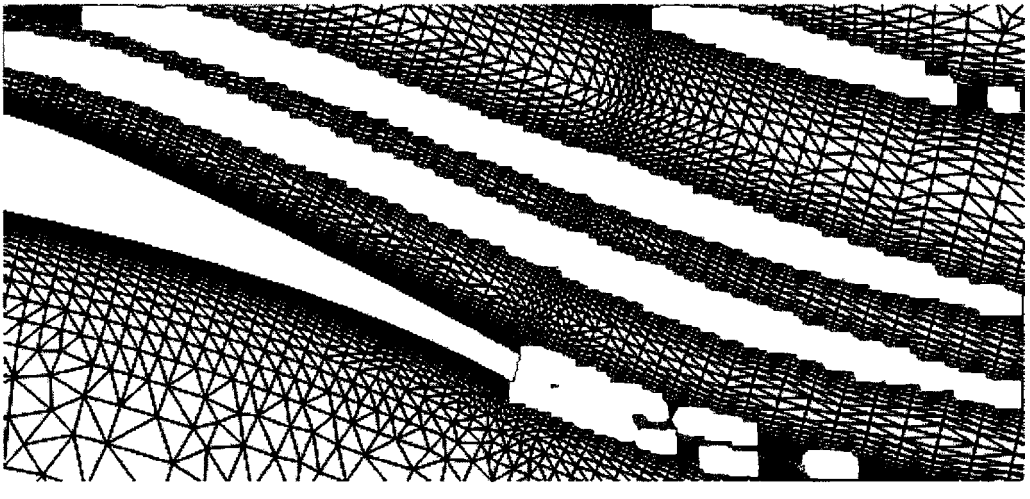


Figure 7.10: Grid fragment near flap trailing edge

7.5 Anisotropic Adaptation Applied to Unstructured Grids

The general adaptation procedure estimates local numerical errors with error indicator and achieves the uniform distribution of error indicator in the computational domain with the help of various grid transformations. The error indicator can be attributed to nodes, edges or tetrahedrons. The most efficient among them is the edge based error indicator because it gives directional information about adaptive properties of unstructured grid. The error of the linear interpolation of one-dimensional quadratic function can be estimated with the following expression [10], [14], [15]

$$Kh^2 \left| \frac{d^2u}{dx^2} \right|,$$

where K is a constant and h is a grid step. If the optimal grid is considered to be the grid with uniform error distribution, then the formula

$$Kh^2 \left| \frac{d^2u}{dx^2} \right| = C$$

determines the grid step in the optimal grid. The symbol C in the upper formula is a positive constant.

In three-dimensional case, the Hessian of monitor function u is used instead of second derivative in presented formula, [10]-[13]. In multidimensional problems the monitor function is a function the grid is adapted to. The Hessian is decomposed in the following manner

$$H = R\Lambda R^T$$

where Λ is a diagonal matrix composed of Hessian eigenvalues and matrix R is a matrix of its eigenvectors. As the error is always positive, the matrix H is replaced generally with the following expression

$$\widetilde{H} = R|\Lambda| R^T = SS^T,$$

where $S = R\sqrt{|\Lambda|}$. The matrix R defines the rotation transformation and S can be interpreted as a stretching transformation of the element in the direction of the principle axes of $\widetilde{H}$. The matrix $\widetilde{H}$ specifies the metric that defines the length of the edge connecting the node i with the node j using the expression

$$e_{ij} = \sqrt{(\mathbf{r}_i - \mathbf{r}_j)^T \widetilde{H} (\mathbf{r}_i - \mathbf{r}_j)},$$

where $\mathbf{r}_i$ is a radius-vector of a node i. The adaptation procedure transforms the given initial grid to the isotropic grid in the metric defined with the matrix $\widetilde{H}$. This can be done applying nodes movement, node insertion/extraction and edges/faces reconnection. In this work, only two last transformations are employed.

The transformations producing isotropic grid in metric $\widetilde{H}$ are the crucial issue for adaptive grid generation technique. The uniform distribution of error indicator with the same

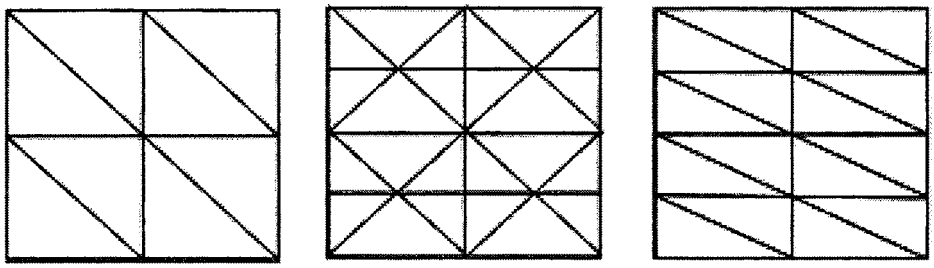


Figure 7.11: The refinement of two-dimensional triangulated rectangular grid

error level can be achieved on the grids with different structure, and comprising different number of nodes. Let us consider the two-dimensional simple problem on the initial triangulated rectangular grid uniform in both directions (Fi-gure 7.11 *a*). The rectangular grid has a grid step h in both directions. Let us consider on this grid the following monitor function u constant along x direction

$$u = \frac{|y|}{h}$$

Let the error indicator at an edge (i, j) is determined as $e_{ij} = |u_i - u_j|$. Then the edge refinement according to criteria $e_{ij} > 0.001$ refines both vertical and diagonal edges (Figure 7.11 *b*). Adaptation of this type is not efficient, because it refines the space not only in the vertical direction, but also in the horizontal direction, despite the constant behavior of monitor function in this direction. In this work, all edges are sorted according to directions of adaptation. One of these directions is the direction of maximal change of monitor function (the vertical direction in the considered example). Another direction of adaptation is the direction of minimal values of error indicator (the horizontal direction in the example). The diagonal edges are not aligned to directions of adaptation in the example. If the edge refinement is applied only to edges aligned to directions of adaptation where the error indicator exceeds the specified threshold, the efficient adaptive grid is obtained. The grid is shown in Figure 7.11 *c*. This adaptation technique does not refine the horizontal direction.

The important problem is the stretching direction of the grid elements. The tetrahedrons should be stretched in the directions of minimal values of error indicator. Otherwise, most of the edges will be subjected to refinement to distribute the error indicator uniformly on the edges, and the produced grid structure will be not efficient. The edge alignment to the direction of minimal values of error indicator is achieved in this work with the edge reconnection according to proper criteria.

The other important problem is the control of the grid elements shape. Namely, the value of the maximal angles in triangles and tetrahedrons should be limited, because unlimited angles corrupt the accuracy of numerical schemes [14]. In this work, the angles in triangular cells are limited by the procedure of recovery of the edges aligned to direction of

maximal changes of error indicator. Besides local reconnection is not made if it results in large angles.

7.6 The Basics of Adaptation Algorithm

The adaptation algorithm iteratively calculates the error indicators at each grid edge using the numerical solution obtained at the previous iteration level, sorts the edges according to directions of adaptation, refines the edges aligned to directions of adaptation, where the error indicator exceeds the given threshold, extract the nodes, where the error indicator at the edges entering a node is less than the given threshold, reconnects the edges to minimize the value of error indicator. The second derivatives of monitor function along the edge direction are calculated with the help of gradients of the monitor function at the nodes. The gradients are calculated according to conventional Green formula. The numerical solution obtained on the grid of the previous iteration level is used to construct adaptive grid on the next level. The iterations are repeated until the needed accuracy is obtained.

7.7 Edge Refinement along Directions of Adaptation

The three-dimensional adaptive algorithm employs the following procedures:
Sorting. The behavior of monitor function at the node is considered to be isotropic, if absolute values of the first derivatives of monitor function along an edge direction are close to each other at all edges entering this node and the same is true for the absolute values of the second derivatives. Three directions of adaptation are attributed to each node with anisotropic behavior of monitor function: the direction of minimal values of error indicator, the direction of maximal changes of monitor function and the direction orthogonal to the previous two. All edges are identified as aligned to one of these directions or nonaligned. All faces in tetrahedral grid are identified as aligned to a pair of the described above three directions or nonaligned.
Edge refinement. Only aligned edges are subjected to refinement. The error indicator at an edge is based on the estimate of the error of linear interpolation of quadratic function and presented with weighted sum of first and second order derivatives along the edge [14], [15]

$$Error(\mathbf{e}) = w_1 \left|\frac{du}{d\mathbf{e}}\right| + w_2 \left|\frac{d^2u}{d\mathbf{e}^2}\right| = w_1 \left|\mathbf{e}\nabla u\right| + w_2 \left|\mathbf{e}^T H \mathbf{e}\right|^{1/2},$$

where $\mathbf{e}$ is an edge unit vector and u is a function the grid is adapted to. The term with the first derivative in the present expression regularizes the generation of grids with little number of nodes. The derivatives are scaled by their average value over all edges in the grid. The derivatives along the edge are calculated using the derivatives provided by the solver or the values of monitor function at the nodes. If the error indicator at the aligned edge exceeds the threshold, this edge is refined. The nonaligned edges are closed to refinement. All edges entering a node with isotropic behavior of monitor function are opened to

refinement.

Three-dimensional reconnection. Faces/edges are swapped to align edges along the direction of minimal values of error indicator. Generally, three-dimensional reconnection swaps three tetrahedrons into 2 and vise versa [16]. Faces aligned to gradient of monitor function are closed to swapping.

Derefinement. The node with isotropic behavior of monitor function is removed if the value of error indicator at all edges entering this node is less than a threshold. At the node with anisotropic behavior only edges aligned to directions of adaptation are taken into account. It may be impossible to implement tessellation of polyhedron appeared as a result of removing all tetrahedrons sharing the removed node. In this case, additional node is inserted.

Restoring the gradient aligned edges. The node insertion and derefinement may result in three-dimensional grids some nodes of which do not have the gradient aligned edges entering them (the other types of aligned edges exist because of edge swapping). If the behavior of monitor function is anisotropic and there are no edges aligned to direction of maximal change of monitor function entering this node, these edges are restored with the help of nodes insertion and reconnection. There must be faces that pass through the aligned edges. That is why after recovery of gradient aligned edges the gradient aligned faces are restored with reconnection or node insertion.

7.8 Sorting Edges along Directions of Adaptation

The following directions of adaptation are defined:

The direction of maximal change of monitor function. An edge is considered to be aligned to this direction if absolute value of the scalar product of the monitor function gradient at this edge with edge unit vector takes on the maximal value among all the edges entering the end points of this edge at the angle that does not exceeds the given threshold (70° in practice).

The direction of minimal values of error indicator. An edge is considered to be aligned to this direction if the error indicator takes on the minimal value among all the edges entering the ends of this edge at the angle that does not exceeds the given threshold (70° in practice).

The third direction of alignment. This direction is defined as orthogonal to the previous two. For any edge t the set of edges $\{l_i | i = 1, \ldots n\}$ entering the ends of the edge and aligned to the first two directions of adaptation is considered. The edge is considered to be aligned to the third direction of adaptation if the functional

$$F(t) = \sum_{i=1}^{n} \cos^2(l_i, t)$$

takes on the minimal value among all the edges entering its ends at some angle to this edge.

7.9 Multilevel Three-Dimensional Reconnection

The developed adaptation algorithm employs edge reconnection to achieve minimum value of error indicator in c norm. Namely, it means that an edge is swapped to the tetrahedral face if the error indicator at this edge is greater than the maximum of the error indicators on the face sides.

The possible reconnection in three-dimensional case consists in changing two neighboring tetrahedrons into three ones and vice versa. Besides, 4 to 4 or 2 to 2 reconnections on the boundary are possible. In two-dimensional case, any triangulation of a prescribed set of nodes can be obtained from the given one with two-dimensional edge swapping. It is not true in three-dimensional case. Three-dimensional reconnection may stick the local optima. This situation often takes place when large number of nodes is inserted in chaotic manner and the followed three-dimensional swapping fails to provide proper alignment. To overcome this difficulty the multilevel swapping is applied. This means that nodes are inserted by portions followed by three-dimensional swapping aligning edges to the direction of minimal values of error indicator. As a result, each new portion of nodes is inserted in aligned grid structure. This tactics provides good alignment for complex grid structures.

7.10 Application of Anisotropic Adaptation to Three-Dimensional Convection-Diffusion Problem

The developed adaptation algorithm was applied to three-dimensional convection-diffusion problem in a cube with solution similar to curvilinear viscous wake. The following convection-diffusion equation is considered

$$\partial_x u + \partial_z \left[f(x,y) u \right] = \mu \, \partial_z^2 u, \qquad f(x,y) = A \sin\left(Bx + C\right),$$

where the symbol μ is a diffusion coefficient and function $f(x,y)$ defines the shape of a surface of diffusive discontinuity, the values A, B and C are the functions of (x,y) in a unit cube [1,0,0] $\times$ [0,1,0] $\times$ [0,0,1]. The considered problem has the following solution in a unit cube

$$u(x,y,z) = erf(\xi), \qquad \xi = \frac{z - z_s(y) - f(x - x_s, y)}{\sqrt{4\mu(x - x_s)}},$$

where erf is an error function, $erf(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$ and point $(x_s, y, z_s(y))$ defines the beginning of the discontinuity in each cube section $y = const$.

The boundary conditions on the cube faces were prescribed according to the exact solution considered in a half plane $x \geq x_s$ with boundary conditions

$$u(x_s, z) = sign(z - z_s(y)), \qquad u(x, \pm\infty) = \pm 1, \qquad x \geq x_s$$

At $x = x_s$ the initial profile of the solution has discontinuity. For $x_s < 0$ the initial profile is smooth. In calculations the following value was chosen $x_s = -0.1$ (smooth profile),

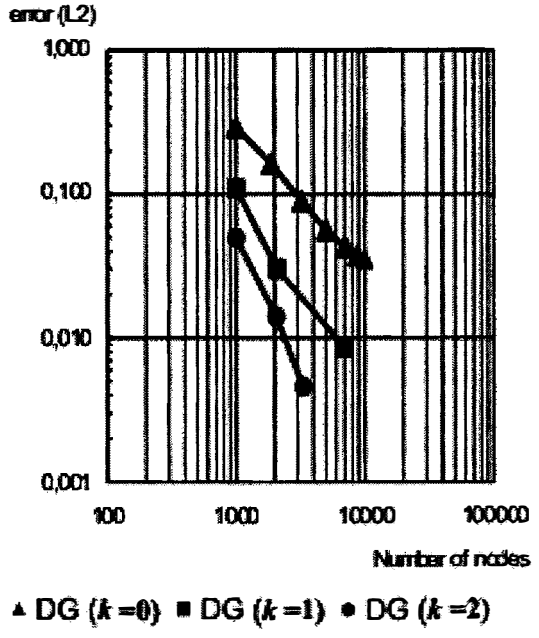


Figure 7.12: Convergence of the error in $\mathbf{L_2}$ norm via number of nodes for different degrees of polynomials in finite element basis $k = 0, 1, 2$

$z_s(0) = 0.6$, function $z_s(y)$ is linear, and linear functions A, B and C define the shape of discontinuity in each cross section $y = const$.

The discontinuous Galerkin finite element scheme was used to solve three-dimen-sional convection-diffusion problem [18], [19], [20]. The code developed by Galeev D.M. [21] was employed. Three sequences of adaptive grids for different orders of accuracy of finite element schemes corresponding to different degrees of polynomials in finite element basis $k = 0, 1, 2$ were generated and the convergence rates of errors via number of grid nodes were explored. In all these sequences the diffusion coefficient was equal $\mu = 5 \cdot 10^{-3}$. Figure 7.12 presents the convergence of error in $\mathbf{L_2}$ norm via number nodes in these sequences. Top fragments in Figure 7.14 show the distribution of solution on the cube faces (on the left) and the grid structure on these faces obtained by removing the grid tetrahedrons close to the boundary (on the right).

The grid adaptation procedure was applied to convection-diffusion problem with different values of solution gradients corresponding to different diffusion coefficient chosen in the range $5 \cdot 10^{-5} \leq \mu \leq 5 \cdot 10^{-3}$. Figure 7.13 depicts the dependencies of the error in $\mathbf{L_2}$ norm via number of nodes for different values of diffusion coefficient. The initial uniform grid $5 \times 5 \times 5$ was used in these calculations. Figure 7.14 (top, middle and bottom fragments) shows the distribution of solution on the cube faces (on the left) and the grid structure on these faces obtained by removing the grid tetrahedrons close to the boundary (on the right) for $\mu = 5 \cdot 10^{-5}$, $5 \cdot 10^{-4}$ and $5 \cdot 10^{-3}$ accordingly.

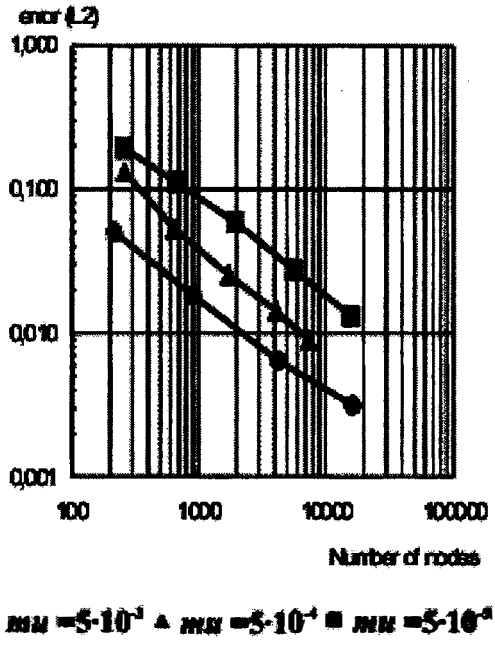


Figure 7.13: Convergence of the error in $\mathbf{L_2}$ norm via number of nodes for different values of diffusion coefficient

7.11 Anisotropic Refinement Coupled with Euler Solver

An ONERA M6 wing with winglets configuration has been used to demonstrate the transonic shock capturing capability of developed adaptive gridding scheme. The finite volume scheme was used to calculate the Euler flow [22]. The flow conditions were at $M = 0.84$ and $\alpha = 3.06°$. The initial grid was isotropic in computational field and almost uniform on the body. The grid density was produced only at the wing leading edge. The initial grid and the final adaptive grid are shown in Figure 7.15 *a* and Figure 7.15 *b*. The fragments of these grids near winglet are presented in Figure 7.16 *a* and Figure 7.16 *b*.

The Mach number was employed as a monitor function. The initial grid contained 6513 nodes on the body surface and 37317 total nodes. An inviscid Euler flow computations on this grid indicates the presence of λ - shock wave on the upper surface of the wing and strong shocks on the winglet. The Mach number contours of numerical solutions calculated on the initial and final adapted grid is presented in Figure 7.17 *a* and Figure 7.17 *b*. The close-up view of these pictures near winglets is presented in Figure 7.18 *a* and Figure 7.18 *b*. As expected, the shock wave was smeared on the coarse initial grid and became sharp on the adapted grid.

The body Mach number distributions in $z = const$ cross wing section and $y = const$ cross winglet section are depicted in Figure 7.19 and Figure 7.20. Two levels of adaptation were performed. The first level increased the number of nodes on 18216 nodes and the

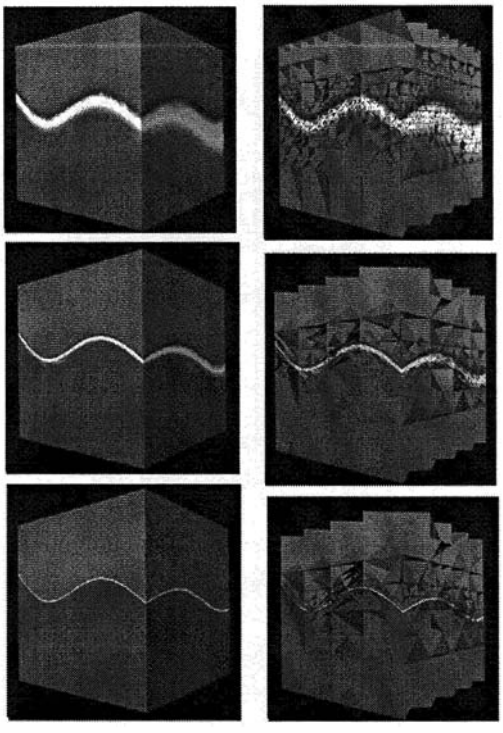
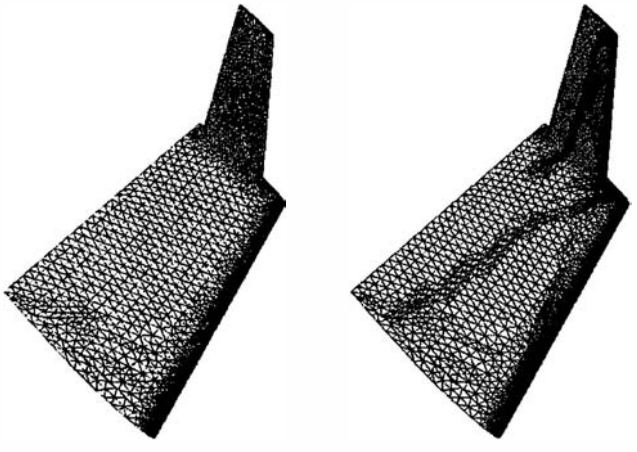
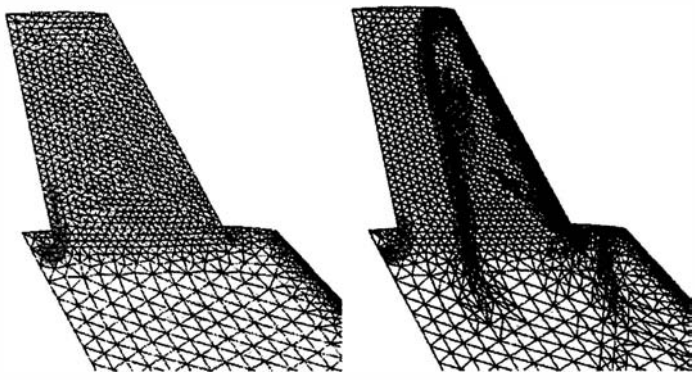


Figure 7.14: Top - $\mu = 5 \cdot 10^{-3}$, middle - $\mu = 5 \cdot 10^{-4}$, bottom - $\mu = 5 \cdot 10^{-5}$



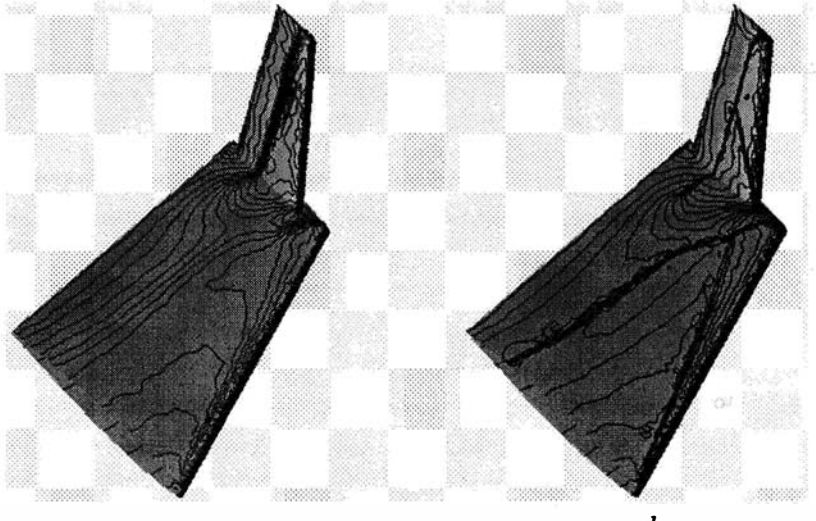
a *b*

Figure 7.15: Initial grid (37315 nodes) and final adaptive grid (87326 nodes)



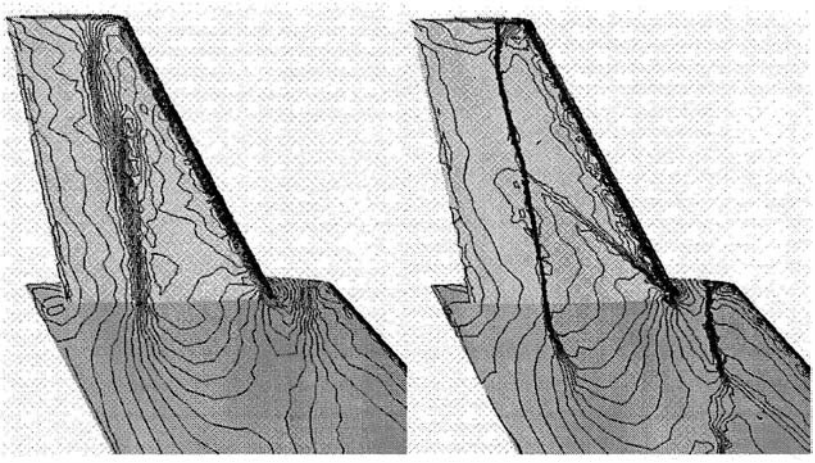
a *b*

Figure 7.16: Initial grid and final adaptive grid (fragments near the winglet)



a *b*

Figure 7.17: Mach number contours for initial and final adaptive grids



a *b*

Figure 7.18: Mach number contours for initial and final adaptive grids (fragments near the winglet)

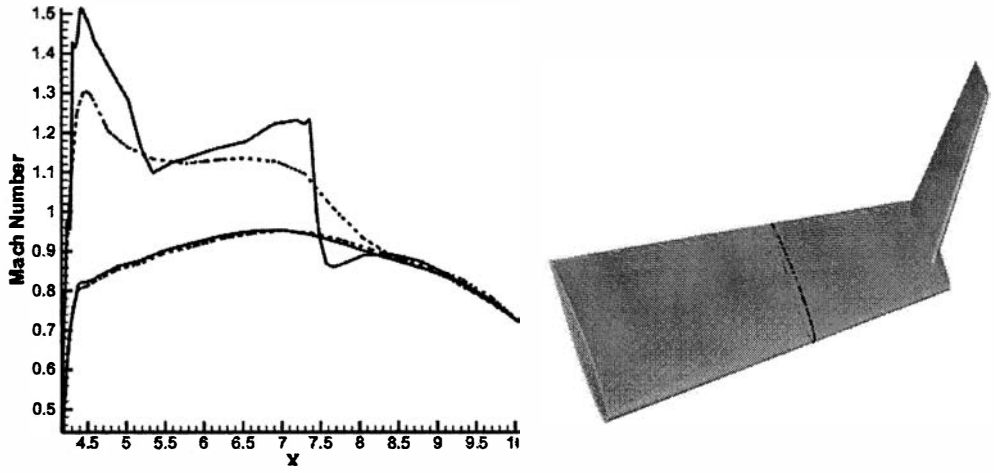


Figure 7.19: Mach number distribution in the wing section (full line - adaptive mesh, dashed line - initial mesh)

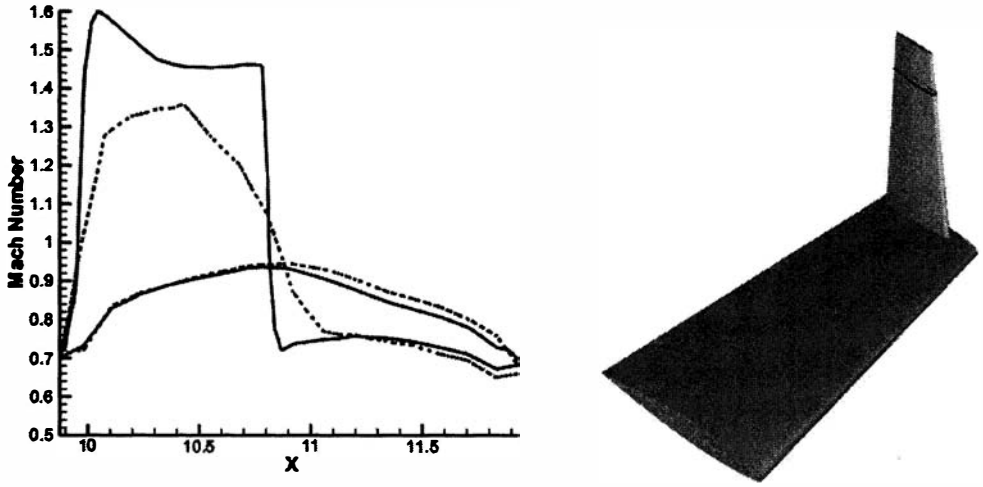


Figure 7.20: Mach number distribution in the winglet section (full line - adaptive mesh, dashed line - initial mesh)

second level on 31797 nodes. The final grid comprised 12881 nodes on the body surface and 87326 total nodes. As a result, the grid is efficiently refined at shock locations and near the leading edges of the wing and winglet. The adaptation was performed at the suction peak where Mach number gradients are large. The presented results show that an adaptation algorithm effectively refines the grid in the regions with large gradients of flow parameters. Figure 7.19 and Figure 7.20 illustrate how the grid adaptation significantly improves the accuracy of numerical solution.

7.12 Conclusion

The algorithm of constrained Delaunay triangulations with specified stretching of boundary triangles is developed. The algorithm produces grids of good quality with extreme stretching in viscous dominated flow regions. The developed technique was successfully applied to produce grids for Navier-Stokes calculations of turbulent viscous high Reynolds number flows around multi-element airfoils. The grids resolved both the boundary layers and multiple wakes.

Three-dimensional algorithm generating anisotropic adaptive grids was deve-loped and coupled with discontinuous Galerkin high order scheme for convection-diffusion problem. The order properties of developed adaptive technique were investigated for different levels of anisotropy. The adaptive algorithm did not display significant change of convergence rate with increase of adaptation anisotropy.

The adaptive algorithm was applied to transonic Euler flow computations around wing

with winglets. The adaptive meshing efficiently refines grid at the shock locations and at the leading edges of the wing and winglets. The adaptive procedure significantly improves the Mach number distribution in the flow field.

Bibliography

[1] Lorner, R., Matching semi-structured and unstructured grids for Navier-Stokes calculations, *AIAA Paper* 93-3348, 1993

[2] George, P.L., Hecht, F., Saltel, E. Automatic mesh generator with specified boundary, *Computer Methods in Applied Mechanics and Engineering* **33**, 1991, 975–995

[3] Weatherill, N.P., Hassan, O., Marcum, D.L., Calculation of steady compressible flow fields with the finite element method, *AIAA Paper* 93-0341, 1993

[4] Anderson, W.K., A grid generation and flow solution method for the Euler equations on unstructured grids, *J. Comp. Phys.* **110** (1), 1994, 23–38

[5] Barth, T.J., Steiner triangulation for isotropic and stretched elements, *AIAA Paper* 95-0213, 1995

[6] Lohner, R., Parikh, P., Generation of three-dimensional unstructured grids by the advancing-front method, *International Journal for Numerical Methods in Fluids* **8** (10), 1988, 1135–1150

[7] Mavriplis, D.J., Adaptive mesh generation for viscous flows using Delaunay triangulation, *J. Comp. Phys.* **90** (2), 1990, 271–291

[8] Spalart, P.R., Allmaras, S.R., A one equation turbulence model for aerodynamic flows, *Recherche Aerospatiale* **1**, 1994, 5–21

[9] Cook, P.H., McDonald, M.A., Firmin, M.C.P., Airfoil RAE 2822: Pressure distributions, boundary layer and wake measurements, *AGARD Advisory Report* 138, 1979

[10] Peraire, J., Vahdati, M., Morgan, K., Zienkeiwicz, O.C., Adaptive Remeshing for Compressible Flow Computations, *J. Comp. Phys.* **72** , 1987, 449–466

[11] Habashi, W.G., Fortin, M., Dompierre, J., Vallet, M.G., Ait-Ali-Yahia, D., Bourgault, Y., Robichaud, M.P., Tam, A., Bolvin, S., Anisotropic Mesh Optimization for Structured and Unstructured Meshes, *VKI Lecture Series* 1997-02, 1997

[12] George, P.L., Hecht, F., *Nonisotropic Grids. Handbook of Grid Generation* . Edited by Thompson, Soni and Weatherill. CRC Press, 1999, 20-1–20-30

[13] Dompierre, J., Labbe, P., Garon, A., Camarero, R., Unstructured Tetrahedral Mesh Adaptation for Two-Dimensional Space-Time Finite Elements, *AIAA Paper* 2000-0810, 2000

[14] Babushka, I., Aziz, A., On the angle condition in the finite element method, *SIAM J. Numer. Anal.* **13**, 1976, 214–226

[15] Vasilevskii, Yu. V., Lipnikov, K. N., An adaptive algorithm for quasioptimal mesh generation, *Comput. Math. and Math. Phys.* **39** (9), 1999, 1468–1486

[16] Barth, T., Aspects of unstructured grids and finite-volume solvers for the Euler and Navier-Stokes equations, *AGARD Report* 787, 1992

[17] Hughes, T.J.R., Mallet, M., A new finite element formulation for CFD III: the generalized streamline operator for multidimensional advective-diffusive systems, *Computational Methods in Applied Mechanics and Engineering* **58**, 1986, 305–328

[18] Bey, K., A Runge-Kutta discontinuous finite element method for high speed flows, *AIAA Technical Report* 91-1575, 1991

[19] Cockburn, B., Lin, S.Y., Shu, C.W., TVD Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws III: One dimensional systems, *J. Comp. Phys.* **84**, 1989, 90–113

[20] Cockburn, B. and Shu, C.W., The Runge-Kutta discontinuous Galerkin method for conservation laws V: Multidimensional system, *ICASE NASA Langley R C Technical Report* 201737, 1997

[21] Galeev, D.M., Analysis of the properties of a high-order accurate scheme approximating conservation laws, Materialy XIII Shkoly-seminara "Aerodinamika letatel'nykh apparatov" (Proc. Of XIII Workshop "Aerodynamics of flight vehicles"), Moscow: TsAGI, 2002, 35–36 (in Russian)

[22] Sakovich, V.S., Numerical solution of Euler and Navier-Stokes equations on unstructured grids, *Proc. 6th Internat. Conf. on Numer. Grid Generation in Comput. Field Simulations*, July 6-9, Greenwich, UK, 1998, 931–941

Part II

THREE-DIMENSIONAL ALGORITHMS

Chapter 8

NUMERICAL SIMULATION OF 3D MULTI-COMPONENT VORTEX FLOWS BY MAH-3 CODE

Nina N. Anuchina, Vladimir I. Volkov, Viatcheslav A. Gordeychuk, Nikolay S. Es'kov, Olga S. Ilyutina and Oleg M. Kozyrev
Russian Federal Nuclear Center - Zababakhin Institute of Technical Physics

Brief description of the MAH-3 code for simulation of complicated multi - component hydrodynamic flows in movable Eulerian coordinates is given. The interfaces between substances are described by Lagrange coordinate surfaces of a hexahedral mesh or by mixed cells and a triangular mesh of markers in the case of strong deformations. The MAH-3 code provides a simultaneous fulfillment of the difference analogs of the conservation laws for mass, pulse, total energy and preserves 2D symmetry (planar and cylindrical) as well as 1D symmetry (planar, cylindrical and spherical). A few numerical examples illustrate the efficiency of the MAH-3 code.

8.1 Introduction

Dynamics of multi-component media with strong interface deformations is very important field of applied researches in many scientific areas, such as, high density matter and energy physics (inertial confinement fusion, explosive processes), astrophysics (birth and evolution of stars, supernew stars), physics of Earth's atmosphere and hydrosphere.

Significant difficulties arise in solving a lot of practical problems since the most of considered flows are instable. Strong distorted interfaces appear because of KH (Kelvin-Helmholtz) [12, 15], RT (Rayleigh-Taylor) [27, 33] and RM (Richtmyer-Meshkov) [19, 28] instabilities or their combination. Beside hydrodynamic instability, vortex and stream flows as well as large transfer of matters lead to strong deformations of interfaces too. Limited possibilities of analytical treatment for these phenomena make necessary to use numerical simulation in researches.

Mathematical modelling of hydrodynamic flows with strong interface deformations and a loss of an initial topological structure is very difficult problem.

Difference methods using Lagrangian variables and regular grids [31, 32, 36] are simple for implementation and allow one to describe interfaces and small details of flows well. But they become unfit under strong distorted interfaces and large transfer of substances. In these methods, a neighborhood of Lagrangian particles is not changed in calculations. This leads to the strong distorted grid, i.e. to a loss of approximation and accuracy. In the case, the calculations often become impossible.

To solve various applied problems with multi-component flows the movable Eulerian coordinates are used [9, 37]. In such approach, a movable regular difference grid for each computational domain is generated where the interfaces are coordinate lines. The grid generation depends on the domain boundaries. Considerable and sometimes unsolvable difficulties arise for the strong distorted boundaries.

In modern computational gas dynamics, there are two main directions to develop numerical methods for calculations of multi-component flows in which the interfaces may be strong distorted up to destruction.

First direction. This direction was proposed by S.Ulam [34]. He considered a medium as a set of particles whose dynamics obeys the Lagrange equations. A neighborhood of particles is not fixed by a grid. This idea, for example, was implemented in [20, 25], where Dirichlet cells for calculating gradients of gas dynamical fields were used. Methods based on irregular difference grids [6, 8, 26] for gradient calculations as well as SPH (Smoothed Particles Hydrodynamics) methods [18] using gas dynamical fields as a sum of regularized delta-functions are intensively developed.

An implementation of these methods meets with essential obstacles such as the support of sufficient uniform density of Lagrangian particles, interpolation of fields on irregular disposed points as well as lack of conservation in some methods and complexity of numerical algorithms.

Second direction. This direction is based on the movable and unmovable regular Eulerian grid with various descriptions of the interfaces not being coordinate lines of the grid. In this case, there are cells with a few substances (mixed sells). The interfaces are localized in the mixed cells. There exist some approaches to develop numerical methods in the direction.

First approach is related with the "particles in cells", method proposed by F.Harlow [10]. The principal feature of the method is the presence of two discrete grids, the unmovable rectangular Eulerian grid for calculating gradients of gas dynamical variables and the Lagrangian particles grid for finding fluxes at the boundaries of Eulerian cells. A substance of each particle is known. Therefore, particles movement allows to obtain the interface movement. This method and its modifications [1] were found very effective for numerical simulation of vortex multi-component flows with strong distorted and destroyable interfaces.

Second approach is based on the method of concentrations and the concept of continuous fluxes [4]. Under this approach the interfaces are not distinguished and locally

calculated from the concentration field at each time step. That is required for determining mass and energy fluxes, and momentum in the mixed cell. The most complete review of numerical methods calculating the interfaces with concentrations is given in [29]. In these methods, additional assumptions about the interface form are often induced. In the methods VOF (Volume Of Fluid) [14] and SLIC (Simple Line Interface Calculation) [23], the interface within the cell is approximated by a straight line in parallel to coordinate lines. Various modifications of the concentration method are widely used in 2D codes (for example [38]).

Third approach necessarily contains an explicit description of the interface moving through the Eulerian grid. This information permits to reduce the numerical diffusion at the interfaces. First such approach for calculating 2D flows was proposed in [22]. There the interface is presented as a broken line which movement with respect to a difference grid is determined in the process of calculations. Algorithmic complexities arise in the method implementation. The method application is difficult for strong interface deformations and practically impossible in turbulent flows.

Two methods of interface description are often used in publications. The first method is a Lagrangian one known as "front tracking" [5]. Interface movement is characterized by the movement of points at this surface. Velocities of the points are obtained from the solution of the Riemann problem.

The second method, "level set" [24], uses an Eulerian approach to describe the interfaces. Instead of considering the interface as a set of points, this method uses a level set function of Eulerian variables, treating the interface as a specific level of this function.

There exist some other methods of describing the interfaces, for example [7], in which Lagrangian interface approximation is based on information about intersection of the interfaces with Eulerian coordinate lines.

In many important applied problems, it is necessary to consider distinct gas dynamical flows. Shock-wave, laminar, vortex and shear flows without strong interface deformation as well as vortex ones with strong interface distortions and a topological change of physical domains are often required to model effectively (simultaneously or at differ time intervals) and with sufficient accuracy. Therefore, in our view, the most perspective approach to effective mathematical modelling is a development of adaptive, i.e. adjusting to flows, computational algorithms. The algorithm adaptation may be achieved by decomposition of the computational field into a few simple domains, by description of strong and weak breaks, by grid generation taking into account integral and local details, by nonuniform numerical methods and so on. In the adaptive algorithms the grid generation must take into account the domain geometry and character of processes there. Algorithms of the grid generation and program tools for the geometrical domain description are widely used in mathematical modelling of various mathematical physics problems [16]. The adaptive methods using a priori knowledge about the flow features allow to obtain more meaningful information.

In this paper, the brief description of the MAH-3 code is given. The code is intended for simulation of complicated multi-component hydrodynamic flows in movable Eulerian coordinates [13]. The interfaces between substances are described by coordinate grid sur-

faces or by mixed cells [4] and a triangular mesh of markers [35] in the case of strong deformations. A few numerical examples illustrate the efficiency of the MAH-3 code.

8.2 Adaptive Difference Method of Calculation of Multi-component Vortex Flows in MAH-3 Code

The difference method implemented in the MAH-3 code allows to calculate complicated non-stationary hydrodynamic flows from a wide range, in particular, multi-component vortex flows with strong interface deformations, and with changing of topological structure of the physical fields. The adaptive computational method using a priori information on the irradiating flows, can apply algorithms of MAH-3 code in various combinations and at different stages of modelling for effective solution with the accuracy needed [2, 3, 35].

8.2.1 Main Characteristics of MAH-3 Difference Method

- The calculated system depending on a priori information on the problem is presented by a set of computational domains. Each domain is a topological parallelepiped. The boundaries may be contact or external borders of the problem. The external borders involves free boundaries, rigid walls, borders with given pressure and Eulerian borders with the material outflow or inflow.

- In each domain, one uses an arbitrary hexahedral grid of its own. In particular, the grid may be Lagrangian [31], stationary Eulerian [10], movable Eulerian [13], or their combination over the coordinate directions.

- The difference method is based on splitting into Lagrangian and Eulerian stages [13].

- The Lagrangian stage may be calculated by explicit or implicit schemes. This allows to model the flow with Mach number from a sufficiently wide range. The implicit difference equations are solved by the Jacobi-Newton iteration with an iterative parameter. The parameter is determined from the condition of a maximal rate of the iteration convergence.

- The difference scheme of the Eulerian stage is an explicit one in which locally adaptive algorithms for the movable Eulerian mesh generation are used. The convective fluxes are approximated by the upstream scheme with first order of accuracy or the Lax-Wendroff scheme [17] with second order of accuracy.

- The interfaces may be coordinate surfaces of the grid. In order to calculate strong deformations of the interfaces the markers [35] are used. The marker coordinates are determined at each time step. The markers indicate the interface position at any time moment. While moving, the interfaces can intersect the Eulerian grid in an arbitrary manner. As a result the mixed cells [4] including several different matters arise. In the mixture calculation the hydrodynamic equilibrium of components and continuity of

the velocity at the interface are mainly applied [4]. In computations of the convective fluxes across the mixed cell edges the interface position determined by the markers is used. In the case of strong interface deformations the matter concentrations are used too.

- At each stage the difference equations approximate the conservation laws written for a curvilinear hexahedron, i.e. the scheme is divergent with respect to mass, pulse and total energy.

The MAH-3 code, unlike many others, provides a simultaneous fulfillment of the difference analogs of the conservation laws for mass, pulse, total energy and preserves 2D symmetry (planar and cylindrical) as well as 1D symmetry (planar, cylindrical and spherical) of the difference solution.

8.2.2 Brief Description of Calculation Algorithms

Equation Systems at Lagrangian and Eulerian Stages

Let t, $\vec{r}$, $\vec{u}$, $\vec{v}$, $\vec{w} = \vec{u} - \vec{v}$ be time, radius-vector in immovable Cartesian coordinate system of (x, y, z), medium velocity field, observer's velocity field, medium velocity relative to the observer, respectively. The substantial time derivate will be denote by $\frac{\mathrm{d}}{\mathrm{d}t}$. In the observer's coordinate system, the time derivative will be denote by $\frac{\delta}{\delta t}$.

The integral conservation laws of mass, pulse and energy for a certain volume D moving in accordance with observer's velocity field may be written as follows

$$\frac{\delta \vec{r}}{\delta t} = \vec{v}, \qquad \frac{\delta}{\delta t}\int_{\mathrm{D}} \rho \mathrm{d}\vec{r} + \int_{\partial \mathrm{D}} \rho \vec{w} \cdot \vec{n}\,\mathrm{d}s = 0,$$

$$\frac{\delta}{\delta t}\int_{\mathrm{D}} \rho \vec{u}\,\mathrm{d}\vec{r} + \int_{\partial \mathrm{D}} p\vec{n}\,\mathrm{d}s + \int_{\partial \mathrm{D}} \rho \vec{u}\,\vec{w} \cdot \vec{n}\,\mathrm{d}s = 0,$$

$$\frac{\delta}{\delta t}\int_{\mathrm{D}} \rho E \mathrm{d}\vec{r} + \int_{\partial \mathrm{D}} p\vec{u} \cdot \vec{n}\,\mathrm{d}s + \int_{\partial \mathrm{D}} \rho E \vec{w} \cdot \vec{n}\,\mathrm{d}s = 0,$$

where ρ, p, $E = e + 0.5\vec{u}^2$, e, $\vec{n}$ denote density, pressure, total specific energy, intrinsic specific energy, unit vector of the external normal to the boundary ∂D of volume D, respectively. The system of equations is closed by the state equation of the form $p = f(\rho, e)$.

Splitting into the Lagrangian and Eulerian stages corresponds to the presentation of the observer's velocity as the difference $\vec{v} = \vec{u} - \vec{w}$ between the medium velocity and the medium velocity with respect to the observer.

The system of equations for Lagrangian stage:

$$\frac{\mathrm{d}\vec{r}}{\mathrm{d}t} = \vec{u}, \qquad \frac{\mathrm{d}}{\mathrm{d}t}\int_{\mathrm{D}} \rho \mathrm{d}\vec{r} = 0,$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\mathrm{D}} \rho \vec{u}\,\mathrm{d}\vec{r} + \int_{\partial \mathrm{D}} p\vec{n}\,\mathrm{d}s = 0,$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\mathrm{D}} \rho E \mathrm{d}\vec{r} + \int_{\partial \mathrm{D}} p\vec{u} \cdot \vec{n}\,\mathrm{d}s = 0.$$

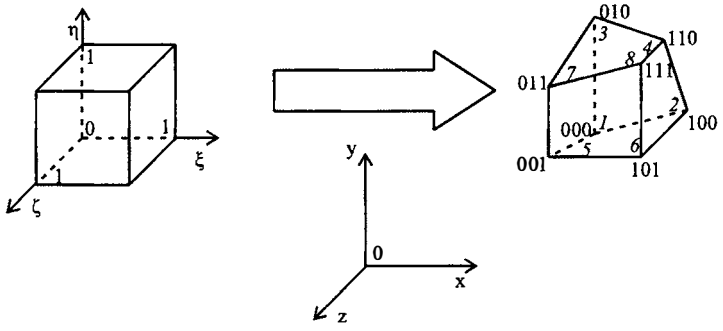


Figure 8.1: Unit cube in a parametric space of (ξ, η, ζ) and ruled cell in a physical space of (x, y, z)

The system of equations for Eulerian stage:

$$\frac{\delta \vec{r}}{\delta t} = -\vec{w}, \qquad \frac{\delta}{\delta t}\int_{\mathrm{D}} \rho \mathrm{d}\vec{r} + \int_{\partial \mathrm{D}} \rho \vec{w} \cdot \vec{n} \mathrm{d}s = 0,$$
$$\frac{\delta}{\delta t}\int_{\mathrm{D}} \rho \vec{u} \mathrm{d}\vec{r} + \int_{\partial \mathrm{D}} \rho \vec{u}\, \vec{w} \cdot \vec{n} \mathrm{d}s = 0,$$
$$\frac{\delta}{\delta t}\int_{\mathrm{D}} \rho E \mathrm{d}\vec{r} + \int_{\partial \mathrm{D}} \rho E \vec{w} \cdot \vec{n} \mathrm{d}s = 0.$$

Spatial Approximation of Gas Dynamical Parameters

The cell of a computational domain is obtained by a nonsingular trilinear mapping of a unit cube to the 3D physical space, where the cube vertices correspond to the cell ones. So, the cell has six faces of every which is a part of ruled surface, twelve edges being segments of straight lines and eight vertices coinciding with grid nodes (Fig. 8.1).

The radius-vector $\vec{r}$ and velocity vector $\vec{u}$ are related to the cell vertices, while density ρ, pressure p, and specific intrinsic energy e are specified in the geometrical center of a cell. The additive values as volume V which is the function of radius-vectors of vertices, mass Mc, internal energy $Ei = Mc \cdot e$, kinetic energy Ek, total energy $Et = Ek + Ei$ are also assigned to the cell.

The cell mass is distributed between its eight vertices $Mc = \sum_{l=1}^{8} Mt_l$, where Mt_l is the vertex mass. Consequently, the cell kinetic energy is defined as $Ek = \frac{1}{2}\sum_{l=1}^{8} Mt_l \cdot \vec{u}_l^2$.

The mass of a grid node Mn is obtained by addition of masses of appropriate eight vertices. The kinetic momentum $\vec{I} = Mn \cdot \vec{u}$ is related to the grid nodes (Fig. 8.2).

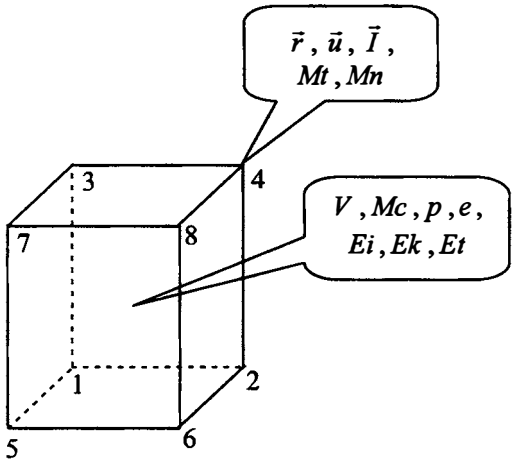


Figure 8.2: Approximation of the main and derived values

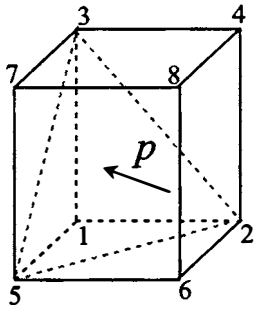


Figure 8.3: The increment of vortex (1) pulse caused by pressure is connected with oriented area of triangle (235)

Difference Scheme of Lagrangian Stage

The Lagrangian stage can be calculated by explicit and implicit schemes. In the explicit scheme the increments of pulse and kinetic energy of the vertices caused by pressure, and the increment of intrinsic energy for each cell are defined (Fig. 8.3).

The new volume of a cell is defined by the radius-vectors of vertices calculated with new velocities of the grid nodes.

For shock waves calculations, the artificial viscosity as a combination of linear and square viscosities is introduced. The effect of artificial viscosity forces is determined by the conservative explicit formulas. At this stage, volume forces are taken into account too.

In the implicit scheme, pulse and total energy equations differ from those in the explicit scheme just by the fact that pressure is related to advanced time moment. The equations

are interpreted as a stage of the corrector of the common scheme predictor-corrector. The pressure is determined at the predictor stage involving node trajectory equations , motion equation, energy equation in non-divergent form, and state equation. The implicit nonlinear difference equations are solved by the Jacobi-Newton iteration. The values obtained by explicit scheme are posed as the initial approximation. In the linear approximation the iterations converge for any time step.

Difference Grids

In each computational domain, an arbitrary hexahedral grid of its own is used. The grid may be Lagrangian, immovable Eulerian, movable Eulerian or any combination of them over coordinate directions. For example, one can generate the immovable Eulerian, movable Eulerian, and Lagrangian grids along three different coordinate directions. In the case of an arbitrary movable Eulerian coordinate system the splitting over parametric directions is applied to create a new grid.

In the algorithms of locally-adaptive grid generation, in order to find the new radius-vector of the current node its nearest neighbors in this parametric direction are invoked only. If needed, one can globally remap the grid in one or several parametric directions. Some analysis of the quality of the produced grid and its correction (e.g. a "smoothing" algorithm for "very sharp" angles between the edges) are provided.

The choice of the grid modification needed is determined by the specificity of the calculated problem.

Difference Scheme of Eulerian Stage for Calculating of Single-Component Medium

The difference scheme of the Eulerian stage is an explicit scheme with the locally-adaptive algorithms for generating a movable Eulerian grid. The convective flows are approximated by a one-sided upstream scheme with the first order of accuracy or by the Lax-Wendroff scheme [17] with the second order of accuracy.

At the Eulerian stage, the convective flows of mass, pulse and total energy, associated with the transition of gas dynamic fields from the Lagrangian to the Eulerian grid [13] are calculated. The flows are calculated along each parametric direction, and are connected with the change in the cell volume caused by the change in the position of any face of the cell in the Lagrangian and Eulerian grids (Fig. 8.4). In the scheme of the first order, the convective flow approximation is based on the principle of a donor-cell [13].

Let the flow come from the cell A to the cell B. One calculates the volume increment δV, which is equal to the volume of a ruled hexahedron $(2^L 24^L 46^L 68^L 8)$.

By volume δV we find an increment of the mass and total energy calculated with the help of corresponding specific values of the cell A. The change in the mass of the cells leads to a corresponding pulse exchange between the four triads of the nodes $(1 \to 2^L \to 2\widehat{\ }), (3 \to 4^L \to 4\widehat{\ }), (5 \to 6^L \to 6\widehat{\ }), (7 \to 8^L \to 8\widehat{\ })$.

In the calculation of the flows by the scheme of the second order of accuracy the specific values from both cells are used. The stability of the Lax-Wendroff scheme is ensured by

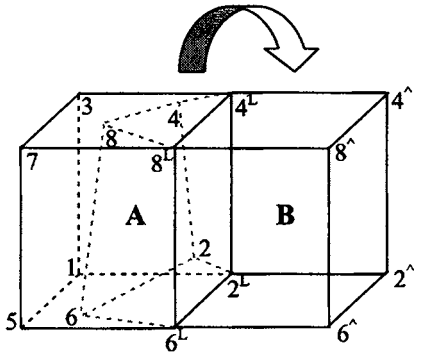


Figure 8.4: Calculation of flows between the cells A and B

the application of stabilizing terms.

Calculation of Contact Surfaces

In the MAH-3 code, the interfaces are described as the coordinate surfaces of the grid in the computational field, or with the help of triangular mesh of markers and mixed cells, or by the mixed cells only.

In detailed study of the interface evolution the following calculation technology is used:

- the interface with small distortions is described as a Lagrangian coordinate surface of the grid;
- the interface with significant distortions is described by means of transition to mixed cells and Lagrangian triangular mesh of markers, which is not connected with the movable Eulerian grid of a computational field;
- in the description of turbulent flows when the interfaces are destroyed and there appears a zone of mixed matters, the method of concentrations is used.

Algorithms for describing the interface as Lagrange coordinate surface of the grid

Such description of the interface leads to the following peculiarities in the calculation algorithm:

- with the generation of a new Eulerian grid after the Lagrangian stage the deviation of the nodes at the interface is performed only in the tangent plane;
- at the Eulerian stage convective flows through the interface are absent.

Algorithms for describing the interface as Lagrange triangular mesh of markers

In MAH-3 code, strong deformations of the matter interfaces are calculated by a triangular mesh of markers. The markers define the interface position at any time. The interfaces may intersect the Eulerian grid in an arbitrary manner. In this case, the mixed cells which contain several different matters are formed. The mixture calculation is based on the conditions of hydrodynamic equilibrium of the components and continuity of the velocity vector at the interface. The convective fluxes in the vicinity of mixed cells are calculated according to the direction of the medium flow and the composition of the matters in the cells. The matter composition in the cells is determined by the markers. For strongly deformed interfaces the calculation algorithms using the markers may be divided into two groups:

- geometrical algorithms for generating the triangular mesh of markers and maintaining its optimum in the course of interface evolution;
- algorithms for processing the mixed cells:
  - [1] calculation of pressure in the mixed cells at the Lagrangian stage;
  - [2] calculation of convective fluxes at the face of mixed cells with account for the motion of the markers at the Eulerian stage.

Geometrical algorithms for describing the interface as a Lagrangian triangular mesh of markers

The interface is a 2D surface in a 3D space. A triangular mesh of markers for describing the surface is used. Each node of the mesh is a marker whose motion is defined by the velocity field of the medium.

The marker is characterized by an ordinal number m, the indices (i_m, j_m, k_m) of its cell, the Cartesian coordinates $\vec{r}_m(x, y, z)$ and the vector $\vec{\sigma}_m(\xi, \eta, \zeta)$, where (ξ, η, ζ) are the local Lagrangian coordinates of the marker with respect to its cell.

The relation between $\vec{r}_m(x, y, z)$ and $\vec{\sigma}_m(\xi, \eta, \zeta)$ is given by a trilinear mapping of a unit cube into a 3D physical cell containing the considered marker.

At the Lagrangian stage, the local Lagrangian coordinates of the marker do not change. After the Lagrangian stage the coordinates $\vec{\sigma}_m(\xi, \eta, \zeta)$ are transformed into the Eulerian coordinates $\vec{r}_m(x, y, z)$. At the Eulerian stage, after generation of a new grid we determine the position of the markers with the relation to the cells of a new grid, and find new local Lagrangian coordinates, which in a general case do not coincide with the old ones.

The sides of the triangles determining the relation between the markers at the interface give the grid topology.

The requirements imposed on the triangular mesh of the markers are as follows:

- the mesh should cover all the interface by nonintersecting triangles;
- the mesh should be symmetrical if the problem has the corresponding symmetry (plane, cylindrical and spherical);
- the mesh should be close to the uniform one.

Creation of initial triangular mesh of markers

It is assumed that initially the interface coincides with the coordinate surface of the grid and is presented by a regular grid composed of spatial quadrangles. When the description of the interface by coordinate surface is substituted for the description by a triangular mesh of the markers, the markers are placed into the nodes of a regular quadrangular grid and geometrical centers of spatial quadrangles.

Addition of the marker

In order to add one new marker the corresponding side of the triangle is split into two equal intervals, and an additional new point is then connected with the opposite vertices of the triangles related to the side.

Such simple algorithm of the marker addition does not provide the conservation of symmetry of the numerical solution with modelling the appropriate symmetrical flow. So, the initial algorithm has been completed by special sorting of the mesh edges, that allows to hold the symmetry of the numerical solution.

Remapping of the marker triangular mesh

In calculations the triangular mesh can become nonuniform. In this case the marker mesh is remapped. The remapping is a local algorithm which displaces the markers in the tangent plane to the interface in order to equalize distances between the neighboring markers.

Processing of mixed cells

Topology of the marker triangular mesh given at the interface defines a direction of the normal. It makes the interface oriented and separates the left part of the space from the right with respect to the normal. Each interface divides the concrete matters. The direction of the normal allows to know from which side of the interface one or other matter is situated.

The matter composition of each mixed cell is defined. Each matter i of the mixed cell is characterized by the volume concentration σ_i, mass M_i, and specific intrinsic energy e_i.

Calculation of pressure in mixed cells at the Lagrangian stage

For a mixed cell, the total pressure and intrinsic energy of the matters of mixture are defined by a single-velocity model of a multi-component medium and the conditions of joint deformation of the components [4].

Let N matters be in a cell. For each matter i, we know the mass M_i, the specific intrinsic energy e_i, the fraction of the volume σ_i, occupied by the matter in the cell volume V, and the equation of state of the matter $p_i = f_i(\rho_i, e_i)$ is given, where $\rho_i = M_i/(\sigma_i \cdot V)$ is the density of the matter i in the cell.

In order to calculate all the values in the mixed cell at the Lagrangian stage, one should determine the pressure. In the pressure calculation, we make use one of the following assumptions:

- the compression of all the mixture components should be equal;
- all the matters are of the same pressure.

In the case of the explicit difference scheme under the first assumption, and from additivity of the total intrinsic energy of the components it follows that:

$$p = \sum_i \sigma_i \cdot p_i.$$

Under the second assumption the pressure p and σ_i are defined by the following system of equations:

$$p_1 = p_2 = ... = p_N,$$

$$\sum_i \sigma_i = 1,$$

$$p_i = f_i(M_i/(\sigma_i \cdot V), e_i),$$

which is solved by an iterative method.

If at the Lagrangian stage the implicit difference scheme is applied, the relationships above for determining pressure in the mixed cells are introduced into the iterations.

Calculation of convective flows at the faces of mixed cells with account for the marker displacement

At the next time, the matter composition in the cells is determined by the analysis of the marker positions relative to a new Eulerian grid.

Each marker entering the cell of the new grid supplements its composition by the matters, which are separated by the marker surface. If all the markers come out of the cell, then, may be, only one matter will stay in the cell. This matter will be the same as in the nearest neighboring non-mixed cell, if such exists. For analysis we take only those cells, which have a common face with considered cell. If the cell does not contain markers, and the markers have not entered the cell, the matter composition is not changed.

The convective flows are calculated only between the cells having a common face. So, it is necessary to define the flows through six faces of the cell.

For each cell A, we find the volume variation δV_a (Fig. 8.4) caused by the motion of the face a resulting from the grid modification. In the case shown in Fig. 8.4, the flow comes from the cell A into the cell B. In calculations, it is convenient to determine only the outgoing flows for each cell.

At the beginning we define the matters participating in the fluxes and the change of their volumes. For the cells A and B, both the old and new matter compositions are known. The flows from the cell A to the cell B are calculated for only those matters, which are contained in the cell A, and will be contained in the cell B.

For each matter i, flowing out from the cell A through the face a, it is necessary to determine the volume variation δV_{ia}, so that $\sum_i \delta V_{ia} = \delta V_a$. At first for the matter i, the preliminary values $\delta \tilde{V}_{ia}$ are found:

$$\delta \tilde{V}_{ia} = 0.5 \left(\frac{\sigma_{iA}}{\sum_{\acute{i}} \sigma_{\acute{i}A}} + \frac{\sigma_{iB}}{\sum_{\acute{i}} \sigma_{\acute{i}B}} \right).$$

Here σ_{iA} and σ_{iB} are the volume concentrations of the matter i in the cells A and B obtained at the Lagrange stage. The summation is made for all the matters flowing out from the cell A to the cell B through the face a.

After calculating the $\delta\tilde{V}_{ia}$ for all the matters flowing out through all the faces of the mixed cell A, the situation is possible when the calculated volume of some matter flowing out from the mixed cell A exceeds the volume of the matter in the cell A, or is insufficient to remove any matter from the cell, although this component should not be present at the next moment of time. So, the $\delta\tilde{V}_{ia}$ are corrected to obtain the final values of δV_{ia}. The correction is made by a redistribution of the volumes of flowing out components.

After this correction the convective flows of the mass, total energy and pulse are calculated separately for each component.

The use of markers in describing the interface allows to localize its position with the accuracy up to one cell of the difference grid. This gives possibility to trace in detail the evolution of the interface up to its destruction.

Calculation of convective flows at the faces of mixed cells by concentrations

In the description of turbulent flows, when the interfaces are destroyed and mixture of matters is formed, the method of concentrations is applied. In this case, the algorithms for determination of δV_{ia} are essentially simplified and are reduced to the following.

Initially only those matters contained both in the cell A and in the cell B at the current time flow out from the cell A to the cell B. When volume of some matter calculated for all the faces exceeds its volume in the mixed cell A, then the preliminary obtained volumes of the matter are corrected at the expense of the volumes of other matters flowing out from the cell. If the latter fails then the rest components of the mixed cell are used.

8.3 Illustration of Capabilities of Calculational Algorithms by Test Problems

To confirm practically that the calculational algorithms preserve the plane, cylindrical and spherical symmetry of the flows and to evaluate the accuracy of them the calculations of several test problems were carried out. Part of these problems have analytical solutions.

8.3.1 Scattering of Gas Cloud

The numerical simulation of scattering of a gas cloud into the vacuum was performed. At the initial time, the gas is a rest perfect one with the density and pressure distributions given in a special way:

$$\rho = \left(1 - \left(\frac{x^2}{a_x^2} + \frac{y^2}{a_y^2} + \frac{z^2}{a_z^2}\right)\right)^{\frac{1}{\gamma-1}}, \qquad p = \left(1 - \left(\frac{x^2}{a_x^2} + \frac{y^2}{a_y^2} + \frac{z^2}{a_z^2}\right)\right)^{\frac{\gamma}{\gamma-1}}.$$

The following shapes of the gas cloud have been considered: a circular cylinder, an elliptic cylinder, a ball, a biaxial ellipsoid and triaxial ellipsoid. Such given shapes of the gas cloud, except for the last one, correspond to 1D cylindrical, 2D plane, 1D spherical and 2D cylindrical symmetry of the flow.

The ellipsoid semi-axes will be denoted a_x, a_y, a_z along the directions x, y, z, respectively. For an perfect gas with adiabatic index $\gamma = 1.4$, the following initial geometries have been considered.

Table 8.1: Initial geometries

a_x	a_y	a_z	Type of symmetry
1	1	∞	1D cylindrical
1	2	∞	2D plane
1	1	1	1D spherical
1	2	2	2D cylindrical
1	2	3	-

Using a separation of variables for all cases listed above the problem is reduced to the system of ordinary differential equations [21], which can be numerically integrated with high degree of accuracy. The solution of this system is said to be the "exact" solution.

The picture of the flow is as follows. The cloud scattering begins from a rest state. In the course of scattering, the ratio between the semi-axes changes. The smaller semi-axis becomes the greater one, and vice versa. In time the scattering becomes the stationary flow when velocity along the axial directions is constant.

In calculations, the Lagrange variables and the movable Eulerian grid were used. The free surface served as coordinate surface. Two intervals were taken along the coordinate directions where the medium was immovable due to the symmetry.

Figs. 8.5-8.10 demonstrate the calculated profiles of pressure and the semi-axes ratio depending on time as compared to the "exact" solution.

In all calculations, the corresponding symmetry of the flow is preserved. The numerical solutions practically coincide with the "exact" ones.

8.3.2 Comparison of the Marker and Concentration Methods

In the calculations using Eulerian methods, an interface may be not a coordinate surface. While moving, the interface can intersect the Eulerian grid in an arbitrary manner. As a result, the mixed cells containing a few different matters are formed. The accuracy of these methods depends on the algorithms calculating the convection flows through the mixed cell faces.

Interesting model problems for the comparison of different numerical methods of such type are proposed in [30] for two spatial variables.

The problems are generalized for the case of three spatial variables. These problems have been calculated by MAH-3 code in order to compare the algorithms describing the

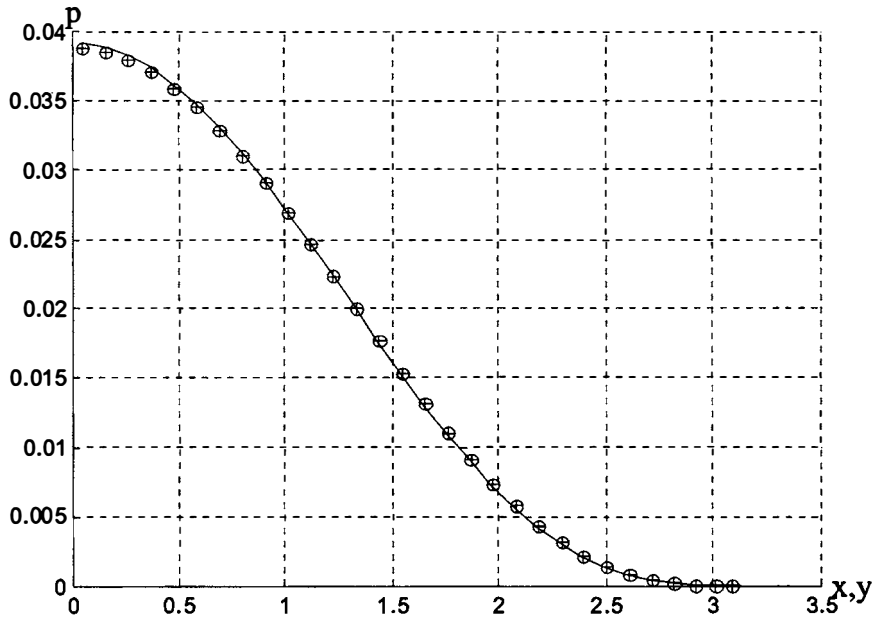


Figure 8.5: Scattering of circular cylinder. Pressure profiles at $t = 1$ over directions x, y: $-$ - exact solution, $\circ$ - numerical solution along direction x, $+$ - numerical solution along direction y

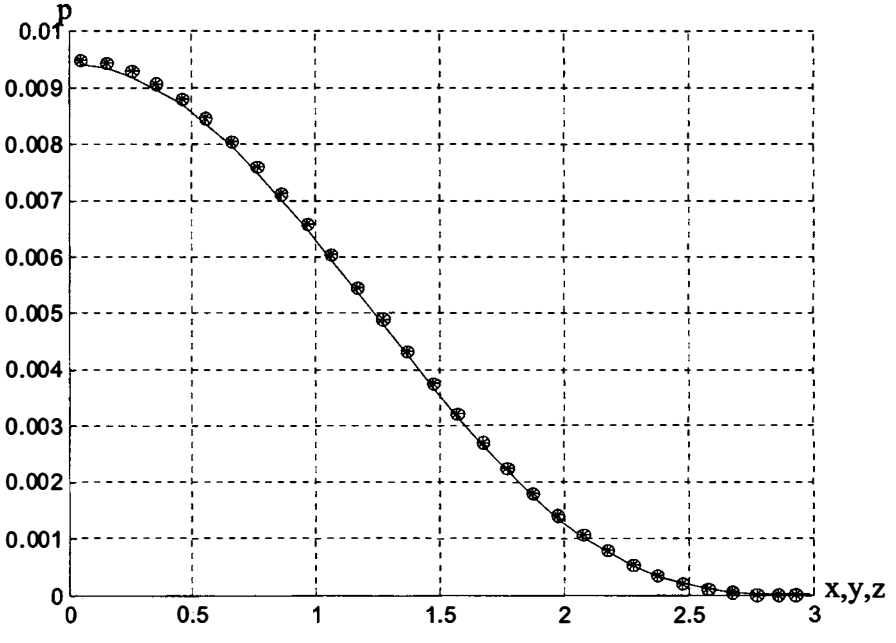


Figure 8.6: Scattering of a ball. Pressure profiles at $t = 1$ over directions x, y, z: $-$ - exact solution, $\circ$ - numerical solution along direction x, $+$ - numerical solution along direction y, $\times$ - numerical solution along direction z

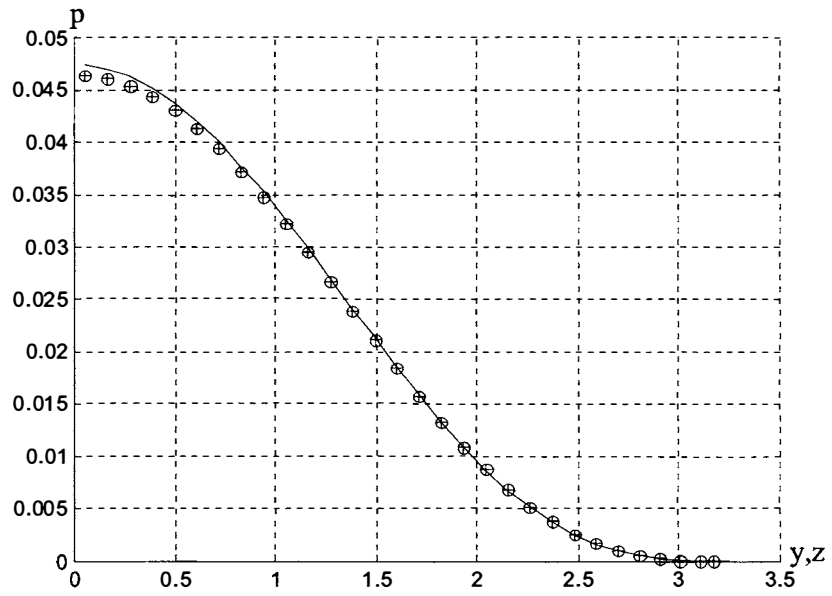


Figure 8.7: Scattering biaxial ellipsoid. Pressure profiles at $t = 1$ over directions y, z: $-$ - exact solution, $\circ$ - numerical solution along direction y, $+$ - numerical solution along direction z

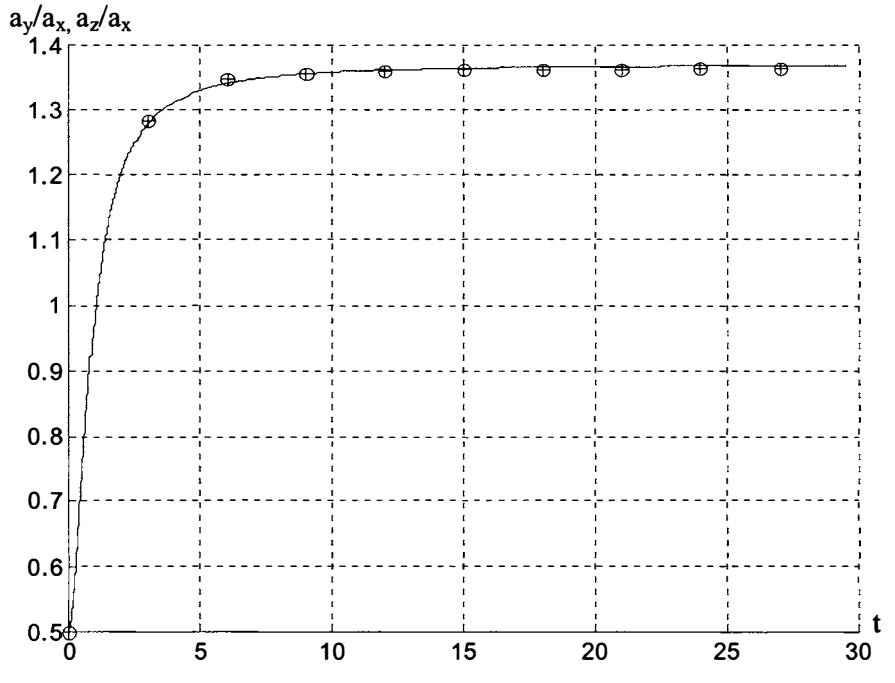


Figure 8.8: Scattering of biaxial ellipsoid. Semi-axes ratio versus time: $-$ - exact solution, $\circ$ - numerical solution for a_y/a_x, $+$ - numerical solution for a_z/a_x

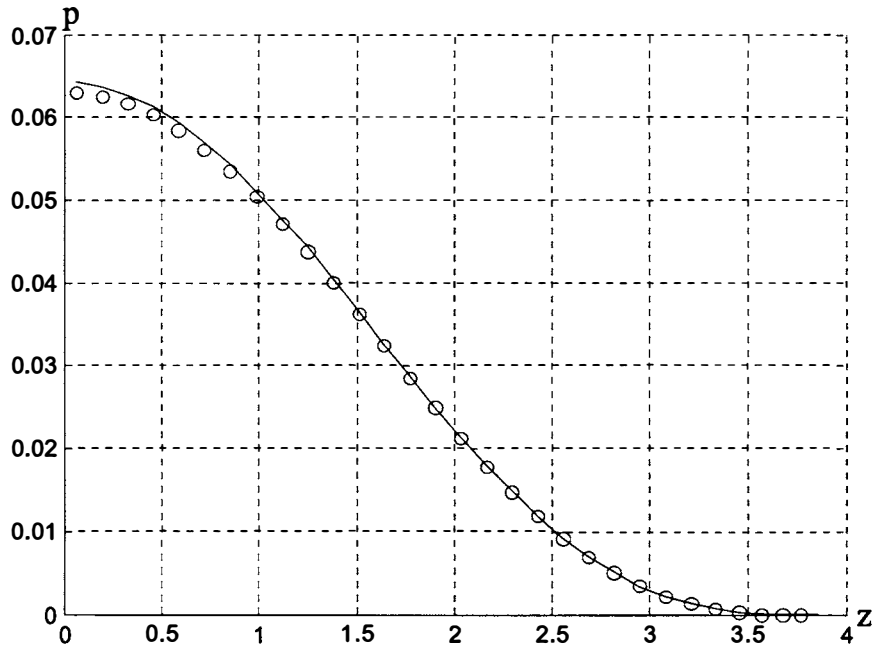


Figure 8.9: Scattering triaxial ellipsoid. Pressure profiles at $t = 1$ over direction z: – - exact solution, o - numerical solution

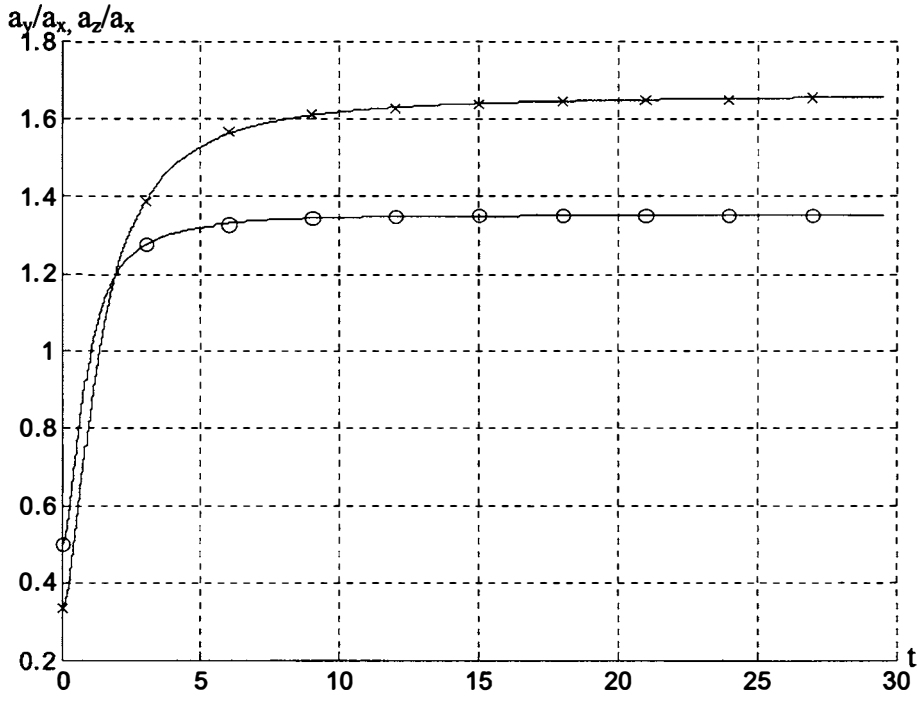


Figure 8.10: Scattering of triaxial ellipsoid. Semi-axes ratio versus time: – - exact solution, o - numerical solution for a_y/a_x, × - numerical solution for a_z/a_x

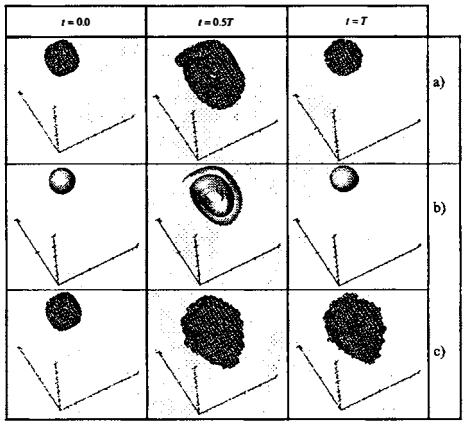


Figure 8.11: Results at $T = 8$: a) mixed cells (M); b) marker surface (M); c) mixed cells (C)

strong distorted interfaces with the markers (M), and with the concentrations (C) only.

In the cube $0 \leq x, y, z \leq 1$ with the rigid walls, there is an incompressible homogeneous liquid with the parameters $p = 0$, $\rho = 1$. The sphere with the radius 0.15 and the center at the point $(0.75, 0.75, 0.75)$ is assumed to be a interface separating two matters. The matter number 2 is inside the sphere, and the matter number 1 is outside the sphere.

The liquid moves with the velocity $\overrightarrow{v}(u, v, w)$:

$$u = (\sin(\pi x))^2 \sin(2\pi y) \sin(2\pi z) \cos(\pi t/T),$$
$$v = (\sin(\pi y))^2 \sin(2\pi x) \sin(2\pi z) \cos(\pi t/T),$$
$$w = -2 (\sin(\pi z))^2 \sin(2\pi x) \sin(2\pi y) \cos(\pi t/T),$$

i.e. $div\,\overrightarrow{v} = 0$.

In the comparative calculations performing by MAH-3 code with $T = 8$, an immovable Eulerian grid in which the number of cells equal to 32^3 was used.

The results obtained are shown in Fig. 8.11. The figures demonstrate the mixed cells arising in the course of calculations, and the marker surfaces.

The comparative calculations show that the calculation diffusion with describing the interface by the marker surfaces is essentially smaller. In the method of concentrations, a rather difficult problem is to reconstruct the interface position by the concentration field.

8.4 Numerical Simulation of the Rayleigh-Taylor Instability in the Spherically Convergent Systems

The studies of RT instabilities are of great practical importance, because they are an obstacle to achieve high degrees of the target compression, and thereby decrease the intensity of

thermonuclear reactions in the inertial confinement fusion (ICF).

In a typical ICF problem, a target consists of several concentric spherical shells having different functional purposes. An external layer is an ablator, the next layer is a shell, which accumulates a kinetic energy at motion to the symmetry center, and an inner layer is the gas. The conditions for the nuclear fusion are provided and exist at compression and expansion of a target. During compression, the target is twice unstable, i.e. at an external ablative surface during an acceleration phase, and at an inner gas-shell interface during a deceleration phase.

In [11], a spherically symmetric self-similar solution of the gas dynamic equations was developed. For the solution, theoretical analysis of a linear stage of the RT instability arising at a shell-gas interface during deceleration phase was done.

In this section, we represent the results of 2D and 3D numerical simulation of a linear stage and beginning of a nonlinear stage of RT instability in the setting as proposed in [11].

In the spherical coordinate system (r, ϕ, ψ), two layers of non-viscous, non-heat conducting gases of different densities are considered: ρ_1, ρ_2 $(\rho_1 < \rho_2)$ (Fig. 8.12). The gases are described by the equation of state $p = (\gamma - 1)\,\varepsilon\rho$, where $\gamma = 5/3$.

The boundary conditions: $r = 0$ is the symmetry center; $r = r_s$ is the free boundary. The moment of the maximal system compression is $t = 0$.

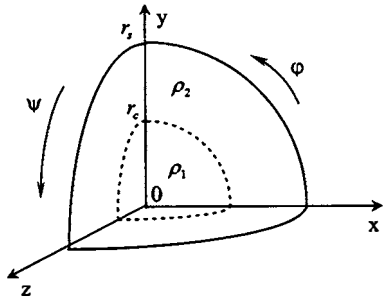


Figure 8.12: Setting of the problem

Using the MAH-3 code, a numerical simulation of evolution of a small single-mode perturbation of interface in 2D and 3D setting was carried out.

At the moment $t/\tau = -1$ (τ is typical time of deceleration), the following perturbation is introduced in the position of the interface:

$$r = r_c + a_0 Y_l^m(\phi, \psi), \qquad 0 \le \phi \le \pi/2, \qquad 0 \le \psi \le \pi/2,$$

$Y_l^m(\phi, \psi) = P_l^m(\cos\phi)\cos m\psi$, $(l = 0, 1, 2, ...; \quad m = 0, 1, 2, ..., l)$ is the tesseral spherical harmonic of the degree l and the order m (Fig. 8.13).

$P_l^m(x)\,(-1 \le x \le 1, l = 0, 1, 2, ...; \quad m = 0, 1, 2, ..., l)$ is the adjoint Legendre function of the first kind, of the degree l, and the order m.

$P_l(x)$ is the Legendre polynomial of the degree l.

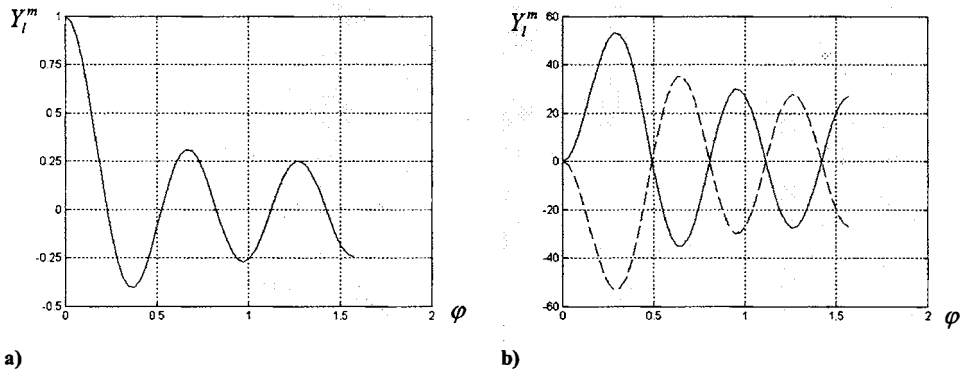


Figure 8.13: Tesseral spherical harmonic of the degree l and order m $Y_l^m(\phi, \psi)$: a) $l = 10, m = 0$; b) $l = 10, m = 2$ and $- - \psi = 0$, $- - - \psi = \pi/2$

For the known degree l of the tesseral spherical harmonic and initial amplitude a_0, the growth of perturbation amplitude on the linear stage can be found numerically with high degree of accuracy. It is said to be the "exact solution".

An initial amplitude a_0 of the perturbation was given so as a maximal value of penetration of a light matter into a heavy matter was the same for both 2D and 3D cases.

For 2D setting: $l = 10, m = 0, a_0 = 0.017$. The difference grid consists of $40 \times 50 \times 2$ intervals along the r, ϕ and ψ directions, respectively.

For 3D setting: $l = 10$, $m = 2$, $a_0 = 0.00032$. The difference grid consists of $40 \times 50 \times 20$ intervals.

Figures 8.14 and 8.15 illustrate the shape of the interface for 2D and 3D settings at different times.

Figure 8.16 illustrates the growth of a perturbation amplitude, as compared to the "exact solution". The results of 2D and 3D simulation agree well with the linear theory. After a maximum compression, one can observe a deviation from the linear theory. Within the studied time interval, the curves of the bubble growth (2D, 3D) and the curve of the jet growth (3D) coincide. They showed that for the perturbations with the same initial amplitude, the growth of perturbation amplitude is determined by the degree l of the adjoint Legendre function of the first kind, and is independent on the order m. Thus, the results of numerical simulation of development of the RT instability arising at a shell-gas interface during deceleration phase agree with the linear stage of analytical solution well.

8.5 Conclusion

- Adaptive calculation method and the program means (MAH-3) have been developed for effective mathematical simulation of the wide class of 3D gas dynamical multicomponent flows, in particular, vortex flows with strong deformations of the inter-

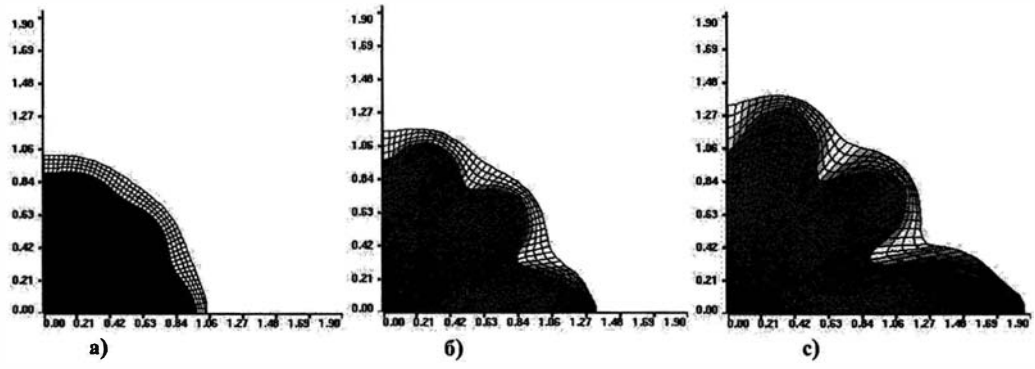


Figure 8.14: Interface shape at the following moments: a) $t/\tau = 0.03$, b) $t/\tau = 0.545$, c) $t/\tau = 0.89$

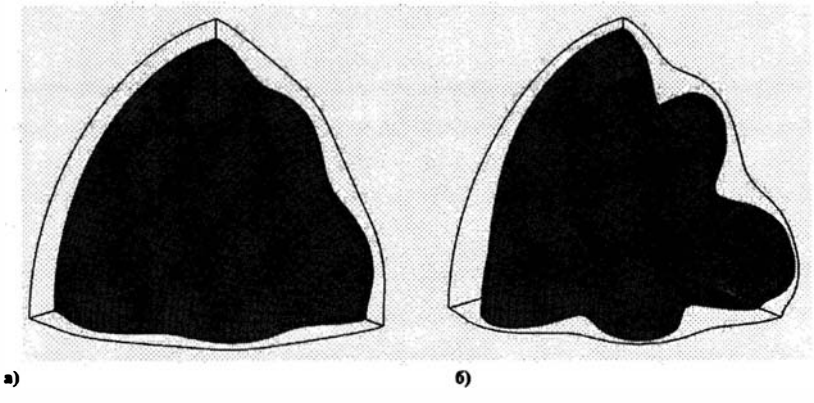


Figure 8.15: Interface shape at the following moments: a) $t/\tau = 0.03$, b) $t/\tau = 0.53$

faces, their damage, loss of topological structure of the physical regions, and mixing of substances. The method using a priori information about the studied flows allows to apply different calculation algorithms, implemented in the code in various computational regions and at different stages of the calculations. This should permit one to choose optimal way for obtaining the results of the required accuracy.

- The calculation algorithms of the MAH-3 code, unlike the other ones, provide for simultaneous fulfillment of the difference analogs of mass, pulse, total energy conservation laws, and conservation of 2D (planar and cylindrical) as well 1D (planar, cylindrical and spherical) symmetries of the difference solution. That is especially important for the solution of problems, where the influence of the asymmetrical external conditions on the parameters of the symmetrical initial system (ICF problem, for example) is studied.

- The 2D and 3D calculations, which simulate a Rayleigh-Taylor instability arising during deceleration at the gas-shell interface, were performed. At linear stage of perturbation growth, the good agreement of numerical and analytical data was obtained.

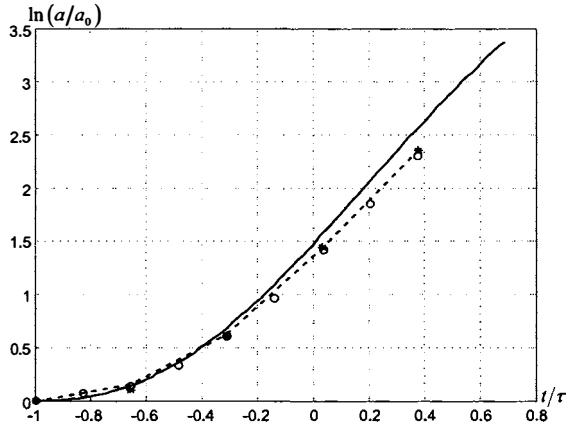


Figure 8.16: The growth of perturbation amplitude as compared to the "exact solution": — - "exact solution", - - - bubble amplitude (2D), o - bubble amplitude (3D), * - jet amplitude (3D)

Acknowledgment

The work was supported by ISTC Projects 177 and 1495.

Bibliography

[1] N.N. Anuchina, Numerical solution of nonstationary gas dynamics problems by particles in cells method, in: *Theoretical foundations and numerical algorithmic design of mathematical physics problems*, edited by K.I. Babenko, NAUKA, Moscow, USSR, 1979, pp.235–253 (in Russian).

[2] N.N. Anuchina, V.I. Volkov, N.S. Es'kov, Numerical modeling of multi-dimensional flows with large deformations, Report at Russian/U.S. Weapons Laboratories introductory technical exchange in computational and computer science, Livermore, 1992.

[3] N.N. Anuchina, V.I. Volkov, V.A. Gordeychuk et al., 3D Numerical simulation of Rayleigh-Taylor instability using MAH-3 code, *Laser and Particle Beams*, **18**, 2000, pp.175–181.

[4] S.M. Bakhrakh, Yu. P. Glagoleva, M.S. Samigulin et al., Calculating gas dynamics flows by the concentrations method, *Reports of the USSR Academy of Science*, **257**, 3, 1981, pp.366-369 (in Russian).

[5] I.-L.Chern, J. Glimm, O. McBryan et al., Front tracking for gas dynamics, *J. Comput. Phys.*, **62**, 1986, pp.83-110.

[6] W.P. Crowley, FLAG: A free-Lagrange method for numerically simulating hydrodynamics flows in two dimensions, in: *Proc. 2ICNMFD*, Springer-Verlag, Berlin, 1970, pp. 37-43.

[7] Yu. A. Dement'ev, A.I. Kirillov, About an approximation of lagrangian surfaces by spatial eulerian mesh, *VANT*, iss. **1**, 1987, pp.54-60 (in Russian).

[8] V.F. D'yachenko, The method of free point, in: *Theoretical foundations and numerical algorithmic design of mathematical physics problems*, edited by K.I. Babenko, NAUKA, Moscow, USSR, 1979, pp.275-281 (in Russian).

[9] S.K. Godunov, A.V.Zabrodin, M.Ja. Ivanov et al., *Numerical solution of multidimensional gas dynamics problems*, NAUKA, Moscow, USSR, 1976 (in Russian).

[10] F.H. Harlow, Numerical method of particles in cells for hydrodynamics problems, in: *Fundamental methods in hydrodynamics*, edited by B. Alder, S. Fernbach, and M. Rotenberg, Mir, Moscow, USSR, 1967, pp.316-342 (in Russian).

[11] F. Hattori, H. Takabe, and K. Mima Rayleigh-Taylor instability in a spherically stagnating system. *Phys. Fluids*, **29**, 5, 1986, pp.1719-1724.

[12] H.L.F. Helmholtz, Uber discontinuierlich Flussigkeitsbewegungen, Monatsberichte Konigl. Akad. Wissenschaften, Berlin, 1868, pp.215-228 (in German).

[13] C.Hirt, A.Amsden, J.Cook, An arbitrary Lagrangian-Eulerian computing method for all flow speeds, *J. Comput. Phys.*, **14**, 3, 1974, pp.227-253.

[14] C.W.Hirt and B.D.Nichols, Volume of fluid (VOF) method for the dynamics of free boundaries, *J. Comput. Phys.*, **33**, 1981, pp.201-225.

[15] Lord Kelvin (W. Thomson), *Mathematical and physical papers*, **4**, Cambrige University Press, 1910.

[16] O.B. Khairullina, A.F. Sidorov, and O.V. Ushakova, Variational methods of construction optimal grids, in: Handbook of Grid Generation, edited by J.F. Thompson, B.K. Soni, and N.P. Weatherill, CRC Press, Boca Raton, FL, 1999, pp.36-1–36-25.

[17] P.D. Lax, B. Wendroff, Systems of conservation laws, *Comm. Pure Appl. Math.*, **13**, 1960, pp.217-237.

[18] L.D. Libersky, A.C. Petschek, T.C. Carney, J.R. Hipp, F.A. Allahdadi, High Strain Lagrangian Hydrodynamics. A three-dimensional SPH code for dynamic material response, *J. Comput. Phys.*, **109**, 1993, pp.67-75.

[19] E.E. Meshkov, Instability of an interface of two gases accelerated by a shock wave, News of the Academy of sciences of the USSR, *Ser.: The mechanics of fluid and gas*, **5**, 1969, pp.151-157 (in Russian).

[20] N.V. Mikhailova, V.F. Tishkin, N.N. Tjurina et al., Numerical simulation of two-dimensional gas dynamic flow on lagrangian structure mesh of variable structure, Preprint 156, Institute of Applied Mathematics, Moscow, 1984 (in Russian).

[21] I.V. Nemchinov, Scattering of a gas triaxial ellipsoid in regular regime, *Applied Mathematics and Mechanics*, **1**, 1965, pp.134-142 (in Russian).

[22] W.F. Noh, CEL - combined eulerian-lagrangian method for calculating nonstationary two-dimensional problems, in: Hydrodynamic instability, edited by G.Birkhoff, R. Bellman, and C.C. Lin, MIR, Moscow, USSR, 1964, pp.128-184 (in Russian).

[23] W.F. Noh and P.R. Woodward, SLIC (Simple Line Interface Calculation), in: *Lecture Notes in Physics*, edited by A.I. van de Vooren and P.J. Zandbergen, 59, 1976, pp.330-340.

[24] S. Osher, R. Fedkiw, Level set methods: an overview and some recent results, *J. Comput. Phys.*, **169**, 2001, pp.463-502.

[25] 18. I.F. Podlivaev, MEDUZA method for calculating two-dimensional gas dynamic problems, in: *Theoretical foundations and numerical algorithmic design of mathematical physics problems*, edited by K.I. Babenko, NAUKA, Moscow, USSR, 1979, pp.254-274 (in Russian).

[26] V.V. Rasskazova, I.D. Sofronov, Some questions of numerical solution of three-dimensional problems of medium mechanics, Voprosy Atomnoy Nauki i Tekhniki: Ser. Matem. *Modelirovanie Fizicheskikh Processov*, **1**, 1987, pp.76-87 (in Russian).

[27] Lord Rayleigh (J.W. Strutt), Scientific papers, 1, Cambridge University Press, 1899.

[28] R.D. Richtmyer, Taylor instability in a Shock acceleration of compressible fluids, *Comm. Pure and Appl. Math.*, **13**, 2, 1960, pp.297-319.

[29] W.J.Rider, D.B.Kothe. Reconstructing volume tracking, *J. Comput. Phys.* **141**, 1998, pp.112-152.

[30] W.J. Rider, D.B. Kothe, Stretching and tearing interface tracking method, presented at the 12th AIAA CFD Conference, San Diego, June 20, 1995.

[31] W.D. Schulz, Two-dimensional difference gas dynamic equations in Lagrange variables, in: *Fundamental methods in hydrodynamics*, edited by B.Alder, S. Fernbach, and M. Rotenberg, MIR, Moscow, USSR, 1967, pp.9-54 (in Russian).

[32] I.D. Sofronov, N.A. Dmitriev, L.V. Dmitrieva, E.V. Malinovskaja, Computational method for nonstationary two-dimensional gas dynamic problems in Lagrange coordinates, in: *Theoretical foundations and numerical algorithms of mathematical physics problems*, edited by K.I. Babenko, NAUKA, Moscow, USSR, 1979, pp.175-200 (in Russian).

[33] G. Taylor, The instability of liquid surfaces when accelerated in a direction perpendicular to their planes, I, *Proc. Roy. Soc. London Ser.A*, **201**, 1950, pp.192-196.

[34] S. Ulam, Stability under calculations by the many solids method, in: Hydrodynamic instability, edited by G.Birkhoff, R. Bellman, and C.C. Lin, MIR, Moscow, USSR, 1964, pp.289-303 (in Russian).

[35] V.I. Volkov, V.A. Gordeychuk, N.S., Es'kov, O.M. Kozyrev, Numerical simulation by the MAH-3 code of the interfaces using an unstructured mesh of markers, *Laser and Particle Beams*, **18**, 2000, pp.197-205.

[36] M.L. Wilkins, Calculation of elastic-plastic flow, in: *Fundamental methods in hydrodynamics*, edited by B. Alder, S. Fernbach, and M. Rotenberg, Mir, Moscow, USSR, 1967, pp.212-263(in Russian).

[37] N.N. Yanenko, V.D. Frolov, V.E. Neuvazhaev, Splitting method for calculation of the heat conducting gas motion in curvilinear coordinates, News of the USSR Academy of Science, Siberian Department, *Ser.: Technical Science*, **8**, 2, 1967 (in Russian).

[38] Ju. V. Yanilkin, A.A. Shanin, N.P. Kovalev et. al., EGAK code for calculating two-dimensional multi-component medium flows, *Voprosy Atomnoy Nauki i Tekhniki: Ser. Matem. Modelirovanie Fizicheskikh Processov*, **4**, 1993, pp.69-75 (in Russian).

Chapter 9

NONDEGENERACY CONDITIONS FOR DIFFERENT TYPES OF GRIDS

Olga V. Ushakova
Institute of Mathematics and Mechanics
Ural Branch of the Russian Academy of Sciences

The work is devoted to the question of nondegeneracy of computational grids. This concept is strictly formulated along with three fundamental theorems guaranteeing nondegeneracy of grids. Conditions of nondegeneracy are obtained for different types of three-dimensional grids. A review and comparative analysis of different grid nondegeneracy conditions are given. New formulas of a volume are obtained for different types of cells.

9.1 Introduction

Although grid generation methods have begun to intensively develop more than forty years ago and the concept of a grid has appeared in the computational practice essentially earlier, nevertheless the full list of requirements to a grid has not been still formulated and has been filled up till now. Some of the requirements are established by a concrete solution algorithm developed on the considered grid and some of them, as in the case of adaptation, by the properties of a solution of the physical problem. Among the requirements, there are some common (main) and some more particular. Among common requirements, the nondegeneracy criteria are the most important, since on degenerate grids, numerical algorithms of solving physical problems do not work, because after substituting differential problems by systems of algebraic equations we can get badly stipulated systems and (or) can not describe the physical field phenomena with the necessary accuracy. So, nondegeneracy of a grid is the first main aim of grid generation.

Among other common requirements influencing on the accuracy (we mean the case of structured grids, see definition 1 in section 9.2), the requirements usually considered are: smoothness of grid lines, closeness of a grid to a uniform one, closeness of a grid to an orthogonal one and adaptation of a grid to the solution of a physical problem. The great

amount of requirements (often contradictory) complicated a separation of the main requirements and their formalization. Therefore, despite of importance, not in all investigations on grid generation the problem of nondegeneracy is strictly formulated and discussion of this question is often omitted as well as obtaining the reliable conditions and the ways of verification of such a property. As a rule, a nondegeneracy of a grid (cell) is understood as a one to one correspondence of a mapping used for generating a grid (cell). However, the nondegeneracy of a grid is frequently judged only visually — by checking the lack of folded or self-intersecting cells and the arrangement of cells without overlapping and gaps and so on. Such a way of estimating the grid nondegeneracy can be difficult especially in a three-dimensional case. Therefore, often for estimation of grid nondegeneracy, the condition of a positivity of cell volumes or the Jacobian of the mapping (computed, for example, in grid nodes) is used. From the other hand, it is also well known, see, for example, in the literature on grid generation [1],[2] that positivity of the Jacobian does not guarantee the one-to-one correspondence of the mapping globally, it guarantees the one-to-one correspondence of the mapping locally. A detailed discussion of a question of a necessity to find conditions that check allows one to draw a conclusion on nondegeneracy of a grid (a cell) is contained in the monograph by Ivanenko [3]. "It is surprising," is pointed out in [3], "but in accessible mathematical literature we did not managed to find publications devoted to this questions. Existing criteria are either sufficiently common and their use requires additional mathematical investigations in each concrete case or very much restricted class of mappings is considered...". For the purpose to find effective criteria, investigations [4],[5] have been undertaken on the initiative of Ivanenko. These investigations gave the foundation and theoretical background of many grid generation algorithms. At that time, local conditions guaranteeing nondegeneracy for the majority of types of cells have been obtained for a two-dimensional case. For a three-dimensional case, such conditions have not been obtained. Conditions guaranteeing nondegeneracy of hexahedral cells, for the first time, have been published in [6], and their detailed proof in [7]. Then, on the basis of [7], criteria of nondegeneracy have been obtained in [8]. Approximately at the same time, in [9], nondegeneracy conditions for pyramidal and prismatic finite elements, and later in [10], a numerical algorithm for checking the nondegeneracy of hexahedral ruled cells have been published. The purpose of these investigations was to find the conditions of the local invertibility of the mappings or conditions guaranteeing the Jacobian nonvanishing for the mappings used for generating cells. In [11], conditions of nondegeneracy have been obtained for polynomial functions used for constructing finite elements. In the same work, it was shown that obtained conditions were the conditions of global invertibility of mappings. Theorems [4],[5] gave the opportunity to use these conditions for developing the grid generation algorithms and for estimating the nondegeneracy of grids composed of above cells.

In section 9.2, the concept of a grid (cell) nondegeneracy is strictly formulated and formulations of three main theorems guaranteeing a one-to-one correspondence of a mapping globally and proved in [4, 5] for different cases and especially for the demands of grid generation practice are given. Section 9.3 is devoted to the problem of nondegeneracy of three-dimensional structured grids composed of ruled hexahedral cells. Such cells are most

widely used in three dimensions (see [12]). Earlier in [6, 7] for testing such grids on nondegeneracy, necessary conditions and sufficient conditions (not coinciding with necessary conditions) for grid cells were obtained. The question about the check of the cells satisfying necessary conditions but not satisfying sufficient conditions remained open. In [8], a special numerical algorithm of testing the Jacobian on its positivity for the mapping used for generation of cells was suggested to use. In this paper, this algorithm is described in detail. Moreover, new necessary conditions that permit to restrict essentially necessary conditions [7] are found. For the completeness of consideration, conditions [7] are also given in this paper. Conditions [7], new necessary conditions and numerical algorithm for testing the Jacobian of a mapping on its positivity give practical criteria for testing structured grids on nondegeneracy. Criteria also allow us to check a one-to-one correspondence or invertibility of trilinear mapping from a unit cube into a region defined by eight points that are vertices of a cell. Analogous criteria for the two-dimensional case can be found in [13, 14, 15]. The criteria are formulated for hexahedral cells, so they can be used for the finite elements too. In section 9.3, a question about admissibility of single degenerate hexahedral ruled cells in some configurations of the domains is discussed.

In section 9.3, examples of hexahedral cells and three-dimensional structured grids [16], nondegeneracy of which was verified by suggested conditions are given.

In section 9.4, conditions of nondegeneracy are obtained for another types of cells. The consideration of the material is compared with the results [9, 10, 11].

In section 9.5, new formulas of the volume are obtained for ruled cells: a pyramid with ruled quadrangular face and for a prism with triangular base and ruled faces. The formula of a volume [7] is given also for a hexahedron with ruled faces. The obtained formulas reveals the connection of ruled cells, in general case having a complicated form and nonplanar faces, with planar face polyhedrons with the vertices of a ruled cells. This gives the opportunities for the essential simplification (see chapter 12) of numerical algorithms developed on grids consisting of ruled cells.

9.2 Formulation of Nondegeneracy Conditions

While numerical solving the problems of mathematical physics, two fundamental types of grids differing in the way of organization of grid nodes are usually used. These are structured and unstructured grids.

For structured grids, the coordinates of nodes are organized into a special structure that is a matrix in a two-dimensional case or a three-dimensional array in a three-dimensional case. In structured grids, every grid node has the same number of neighboring nodes and the type of a shape for the grid cells is the same for all grid points.

Unstructured grids can have their own organization of grid node connectivity for each grid node. For unstructured grids, the number of neighbors can be different. Unstructured grids are composed of cells of arbitrary shape and require the description of connectivity between nodes.

We shall investigate a question of nondegeneracy on the example of structured grids. We

shall formulate this concept for a structured grid in the whole and for a cell in particular. We shall consider such the most widely used way for generating structured grids as a mapping approach.

According to this approach, the generation of a structured grid in the domain G of the solution of the physical problem, named as a physical domain and in the general case having a complex geometrical form, is performed by means of a continuous mapping $\mathbf{x}$ of some additional domain P of a simple form, called as computational domain or a space of parameters, onto a domain G. The mapping $\mathbf{x}$ is also given on the boundary of a domain. It maps the boundary ∂P of the domain P into the boundary ∂G of the domain G, i.e. the grid nodes are also arranged on the boundary of the domain G. This way of generating structured grids gives an opportunity to carry out all computations on a fixed grid in the computational domain of a fixed form. In generation of a structured grid, we consider the case when G is a simply connected domain. In this case, a computational domain is, as a rule, a rectangle or a square for a two-dimensional grid, and a rectangular parallelepiped or a cube for a three-dimensional grid. Rarely, the computational domains in the form of a circle, triangle, tetrahedron, sphere and others are used. We shall consider the computational domains of a rectangular form. For example, if a computational domain is given in the form of a rectangular parallelepiped $P = \{\xi = (\xi_1, \xi_2, \xi_3) : 0 \le \xi_l \le I_l,\ l = 1, 2, 3\}$, where I_l are positive integers determining the number of grid nodes for each of variables (coordinate directions), then the values of the mapping $\mathbf{x} : P \longrightarrow G$

$$\mathbf{x} = \mathbf{x}(\xi_1, \xi_2, \xi_3) = \{x_1(\xi_1, \xi_2, \xi_3), x_2(\xi_1, \xi_2, \xi_3), x_3(\xi_1, \xi_2, \xi_3)\} \qquad (*)$$

for $\xi_l = i_l,\ l = 1, 2, 3,\ i_l = 0, 1, ..., I_l$ give the coordinates of nodes of a three-dimensional grid $\mathbf{x}_{i_1,i_2,i_3} = \mathbf{x}(i_1, i_2, i_3)$. The images of cells of the computational domain given in the form of unit cubes

$$P_{i_1 i_2 i_3} = \{\xi = (\xi_1, \xi_2, \xi_3) : i_l \le \xi_l \le i_{l+1},\ \ l = 1, 2, 3, \},\ i_l = 0, ..., I_l - 1,$$

define the grid cells, and lines and surfaces corresponding to fixed values of indices i_l, define coordinate lines and surfaces. So, for a given mapping, a grid arises as an object consisting of nodes and cells. For this mapping, a physical domain G is represented as a hexahedron having 8 vertices, 12 edges and 6 faces.

Constructing a grid in the way described above is carried out by searching an appropriate mapping of the domain P into the domain G. This mapping, depending on the used approach, can be given in an explicit form, for example, by means of algebraic formulas, functions, or can be obtained by solving differential equations, or by minimization of special functionals, which depend on functions defining the required mapping, and by other methods. The grid itself and its cells, their properties and geometrical characteristics,— all this is determined by the considered mapping. Properties and geometrical characteristics of grids and cells are formalized and controlled via special expressions depending on the required functions and their derivatives. These expressions, as a rule, determine the form of minimized functionals in variational problems and, frequently, the form of differential equations for computing grids.

In the numerical realization of the most of the approaches, the mapping is searched only at nodes of the computational domain. At other points, the mapping can be defined in either way, where cells are determined as images of cells of the computational domain under the mapping used for constructing cells. For these purposes, when generating structured grids, a bilinear mapping of the unit square is used most frequently in the two-dimensional case, and a trilinear mapping of unit cube is used in the three-dimensional case. When generating unstructured grids, construction of cells also can be carried out by means of special mappings of unit triangles, tetrahedrons, prisms, and etc., called as a reference element or domain. For successful numerical solution of a physical problem, it is important that the grid thus constructed and its cells would be nondegenerate.

We consider the following definition of nondegeneracy of a grid (cell).

Definition 1. *A grid (cell) is called nondegenerate, if a mapping* $(*)$ *used for its generation is one to one and "onto".*

It is worth noting that the mapping $(*)$ from definition 1 will be a homeomorphism (one-to-one continuous mapping whose inverse is also continuous), since the domain of the mapping, in the considered case, is a compact set (bounded and closed set).

The nondegenerate grid does not contain folded or self-intersecting cells, self-intersecting coordinate lines and surfaces, cells or nodes sticking together, and in most of cases, cells with different orientation of edges and faces.

Usually, the mapping is demanded to be smooth that is the functions given this mapping be continuous and have continuous derivatives. Such mappings belong to $\mathbb{C}^1$ class. In this case, coordinate lines and surfaces are smooth.

If the mapping is one-to-one, smooth and "onto" and the inverse mapping is smooth (such a mapping is called diffeomorphism) and if the Jacobian of the mapping does not vanish then the mapping defines curvilinear coordinate system.

As a rule, the aim of grid generation is to find such a mapping. In addition, it is required also that the Jacobian of a mapping has to be positive (the positivity guarantees the preservation of orientation, i.e. the coordinate system is right-hand as the Cartesian coordinate system). However, the conditions guaranteeing listed requirements are included in the algorithm very rare. Quite often, the grid generation algorithm produces degenerate grids for complicated configurations of domains. Examples of such algorithms are algebraic methods (see., for example [1]).

What conditions should be included in the construction of grid generation algorithm to guarantee the nondegeneracy of grids?

Special theorems have been obtained in [2, 4, 5] that give sufficient conditions of the one-to-one correspondence of the mappings, namely conditions of homeomorphism of mappings under different assumptions about their smoothness. These conditions ensure the nondegeneracy of grids and can be applied to justify the properties of grid generation algorithms for the cases of structured and unstructured grids in the domains of complex forms. The conditions use the information about properties of the mapping on the boundaries of the domains and information about local properties of the mapping.

Let us formulate two fundamental theorems from [5]: the first one is for a smooth case, the second one is for a continuous case.

Theorem 1 (Smooth case). *Let $\mathbb{R}^n$ be the n-dimensional Euclidean space. Let P and G be bounded connected domains in $\mathbb{R}^n$ that closures $\overline{P}$ and $\overline{G}$ are diffeomorphic. Let $\mathbf{x} : \overline{P} \longrightarrow \overline{G}$ be the mapping of $\mathbb{C}^1$ class. Let $\mathbf{x}$ map homeomorphically the boundary ∂P into the boundary ∂G, and the Jacobian of the mapping (the determinant of Jacobian matrix of the mapping) satisfies the conditions*

$$J = \det \mathbf{x}'(\xi) \neq 0 \quad (\xi \in \overline{P}).$$

Then $\mathbf{x}$ is a homeomorphism from $\overline{P}$ onto $\overline{G}$.

This theorem can be applied for verification of the nondegeneracy of a grid or a cell given by a smooth transformation. For example, all curvilinear coordinate systems homeomorphic on the boundary are nondegenerate.

In the most of cases, the grid is given by a discrete set of nodes, and grid elements or cells are constructed by different mappings. So, for practice the situation when, the mapping $\mathbf{x} = \mathbf{x}(\xi)$ is only continuous, is also important.

The following theorem is valid.

Theorem 2 (Continuous case). *Let P and G be bounded connected domains and $\mathbf{x} : P \longrightarrow G$ be a continuous mapping. Let $\mathbf{x}$ map homeomorphically the boundary ∂P into the boundary ∂G and be a local homeomorphism (i.e. homeomorphism in some neighborhood of each point) from $\overline{P}$ into $\overline{G}$. Then $\mathbf{x}$ is a homeomorphism from $\overline{P}$ onto $\overline{G}$.*

The two previous theorems allowed authors of [5] to prove the theorem.

Theorem 3. *Let P and G be bounded connected homeomorphic domains. Let the domain P be decomposed into non-intersecting convex subdomains P_i $(i = 1, ..., m)$ such, that*

$$\overline{P} = \bigcup_{i=1}^{m} \overline{P}_i.$$

Let $\mathbf{x} : \overline{P} \longrightarrow \overline{G}$ be continuous mapping smooth on $\overline{P}_i$. Let us denote the restriction of the mapping $\mathbf{x}$ corresponding to $\overline{P}_i$ by $\mathbf{x}_i$ $(i = 1, ...m)$.

Let $\mathbf{x}$ map homeomorphically the boundary ∂P into the boundary ∂G, and for any $i = 1, ..., m$ its constriction $\mathbf{x}_i$ be a homeomorphism from ∂P_i onto $\mathbf{x}(\partial P_i)$ and, besides

$$\det \mathbf{x}'(\xi) > 0 \quad (\xi \in \overline{P_i}, i = 1, ..., m.)$$

Then $\mathbf{x}$ is a homeomorphism from $\overline{P}$ onto $\overline{G}$.

Theorem 3 can be applied to obtain conditions of nondegeneracy of grids given by a discrete set of nodes. When constructing structured grids, the computational domain P is divided into simple cells or elements. On these elements, different mappings into

corresponding cells of the physical domain are defined. On the basis of such mappings, the global continuous mapping is constructed.

Then, in order to verify the conditions of theorem 3, one should check whether the mappings from the elements of the computational domain onto grid cells of the physical domain are homeomorphic. The homeomorphism of a single element ensures nondegeneracy of its image or corresponding cell. In common, homeomorphism for single elements guarantees nondegeneracy of all cells and the grid as a whole. The homeomorphism of a mapping for a single element is verified in fact by means of theorem 1, since a single cell is often constructed using smooth mappings.

For example, in the considered three-dimensional case, a computational domain P is divided into elementary cells — the unit cubes $P_{i_1 i_2 i_3}$. A three-dimensional grid is given as a set of nodes $\mathbf{x}_{i_1 i_2 i_3}$. To construct the global mapping is possible using trilinear mappings of unit cubes $P_{i_1 i_2 i_3}$ (see further (9.1))

$$\mathbf{x} = \mathbf{x}(\xi), \quad \xi \in P_{i_1 i_2 i_3},$$

for given values of coordinates of grid nodes $\mathbf{x}_{i_1 i_2 i_3}$ as images of vertices of a cube. Constructed in such a way the global mapping will be continuous. To verify the conditions of theorem 3, one have to check the positivity of the Jacobians of trilinear mappings for all unit cubes.

In a similar manner, one can check (in terms of definition 1) the nondegeneracy of unstructured grids. In [5], a finite element mesh is considered as an example. "If we consider the mesh that elements are simplex then conditions of nondegeneracy of a mesh are equivalent to conditions of positivity of algebraic volumes of all mesh cells. In the case when we have more complicated cells (rectangular, hexahedral or polynomial cells), it is necessary to have the conditions of a homeomorphism from an elementary cell that is a reference element onto each mesh cell."

First, we shall give such conditions for hexahedral ruled cells.

9.3 Nondegeneracy of Hexahedral Ruled Cells

For simplicity we shall consider the cell with the corners $\mathbf{x}_{i_1 i_2 i_3} = (x^1_{i_1 i_2 i_3}, x^2_{i_1 i_2 i_3}, x^3_{i_1 i_2 i_3})$, $i_1, i_2, i_3 = 0, 1$. The mapping

$$\begin{aligned} \mathbf{x} = \mathbf{x}_{000}(1-\xi_1)(1-\xi_2)(1-\xi_3) + \mathbf{x}_{001}(1-\xi_1)(1-\xi_2)\xi_3 \\ +\mathbf{x}_{010}(1-\xi_1)\xi_2(1-\xi_3) + \mathbf{x}_{011}(1-\xi_1)\xi_2\xi_3 + \mathbf{x}_{100}\xi_1(1-\xi_2)(1-\xi_3) \\ +\mathbf{x}_{101}\xi_1(1-\xi_2)\xi_3 + \mathbf{x}_{110}\xi_1\xi_2(1-\xi_3) + \mathbf{x}_{111}\xi_1\xi_2\xi_3 \end{aligned} \tag{9.1}$$

of the unit cube P_{000} transforms the corners of a cube into the corners of a cell, the edges of a cube into edges of a cell, and faces of a cube into faces of a cell that are ruled surfaces of the second order or planes (Fig. 9.1), and defines a so-called ruled cell.

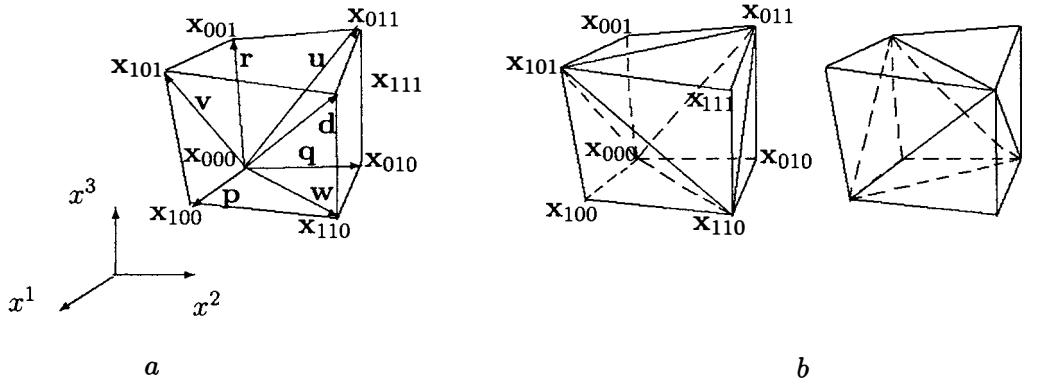


Figure 9.1: Three-dimensional cells: a hexahedron; b dodecahedrons.

The mapping is a trilinear mapping (linear in each of the variables). This is immediately seen from the representation

$$\begin{aligned}\mathbf{x}(\xi) \;=\; & \mathbf{a}_{000} + \mathbf{a}_{100}\xi_1 + \mathbf{a}_{010}\xi_2 + \mathbf{a}_{001}\xi_3 \\ & + \mathbf{a}_{110}\xi_1\xi_2 + \mathbf{a}_{101}\xi_1\xi_3 + \mathbf{a}_{011}\xi_2\xi_3 + \mathbf{a}_{111}\xi_1\xi_2\xi_3,\end{aligned} \tag{9.2}$$

to that (9.1) is reduced. The vectors $\mathbf{a}_{i_1 i_2 i_3}$ are found from the relations:

$$\begin{aligned}
\mathbf{a}_{000} &= \mathbf{x}_{000}, \\
\mathbf{a}_{001} &= \mathbf{x}_{001} - \mathbf{x}_{000}, \quad \mathbf{a}_{011} = \mathbf{x}_{011} - \mathbf{x}_{010} - \mathbf{x}_{001} + \mathbf{x}_{000}, \\
\mathbf{a}_{010} &= \mathbf{x}_{010} - \mathbf{x}_{000}, \quad \mathbf{a}_{101} = \mathbf{x}_{101} - \mathbf{x}_{100} - \mathbf{x}_{001} + \mathbf{x}_{000}, \\
\mathbf{a}_{100} &= \mathbf{x}_{100} - \mathbf{x}_{000}, \quad \mathbf{a}_{110} = \mathbf{x}_{110} - \mathbf{x}_{100} - \mathbf{x}_{010} + \mathbf{x}_{000}, \\
\mathbf{a}_{111} &= \mathbf{x}_{111} - \mathbf{x}_{110} - \mathbf{x}_{101} - \mathbf{x}_{011} + \mathbf{x}_{100} + \mathbf{x}_{010} + \mathbf{x}_{001} - \mathbf{x}_{000}.
\end{aligned} \tag{9.3}$$

In the two-dimensional case, the ruled cell is given by a bilinear mapping of the unit square and is a quadrilateral.

9.3.1 The Jacobian of the Trilinear Map

The Jacobian of mapping (9.2)

$$J(\xi_1, \xi_2, \xi_3) = \frac{\partial(x^1, x^2, x^3)}{\partial(\xi_1, \xi_2, \xi_3)} = \det\left(\frac{\partial x^i}{\partial \xi_j}\right)_{i=1,2,3, j=1,2,3}$$

is equal to the triple scalar product

$$J = \left[\frac{\partial \mathbf{x}}{\partial \xi_1}, \frac{\partial \mathbf{x}}{\partial \xi_2}, \frac{\partial \mathbf{x}}{\partial \xi_3}\right]. \tag{9.4}$$

Partial derivatives can be written in the form

$$\begin{aligned}\frac{\partial \mathbf{x}}{\partial \xi_1} &= \sum_{i_2,i_3=0}^{1} \mathbf{p}_{0 i_2 i_3} \Xi_{i_2} \Xi i_3, \\ \frac{\partial \mathbf{x}}{\partial \xi_2} &= \sum_{i_1,i_3=0}^{1} \mathbf{q}_{i_1 0 i_3} \Xi_{i_1} \Xi_{i_3}, \\ \frac{\partial \mathbf{x}}{\partial \xi_3} &= \sum_{i_1,i_2=0}^{1} \mathbf{r}_{i_1 i_2 0} \Xi_{i_1} \Xi_{i_2},\end{aligned} \tag{9.5}$$

where vectors $\mathbf{p}$, $\mathbf{q}$, $\mathbf{r}$ (see Figure 9.1 *a*) are defined in the relations

$$\begin{aligned}\mathbf{p}_{i_1 i_2 i_3} &= \mathbf{x}_{\bar{i}_1 i_2 i_3} - \mathbf{x}_{i_1 i_2 i_3}, & \mathbf{q}_{i_1 i_2 i_3} &= \mathbf{x}_{i_1 \bar{i}_2 i_3} - \mathbf{x}_{i_1 i_2 i_3}, \\ \mathbf{r}_{i_1 i_2 i_3} &= \mathbf{x}_{i_1 i_2 \bar{i}_3} - \mathbf{x}_{i_1 i_2 i_3}, & \mathbf{u}_{i_1 i_2 i_3} &= \mathbf{x}_{i_1 \bar{i}_2 \bar{i}_3} - \mathbf{x}_{i_1 i_2 i_3}, \\ \mathbf{v}_{i_1 i_2 i_3} &= \mathbf{x}_{\bar{i}_1 i_2 \bar{i}_3} - \mathbf{x}_{i_1 i_2 i_3}, & \mathbf{w}_{i_1 i_2 i_3} &= \mathbf{x}_{\bar{i}_1 \bar{i}_2 i_3} - \mathbf{x}_{i_1 i_2 i_3}, \\ \mathbf{d}_{i_1 i_2 i_3} &= \mathbf{x}_{\bar{i}_1 \bar{i}_2 \bar{i}_3} - \mathbf{x}_{i_1 i_2 i_3}.\end{aligned} \tag{9.6}$$

Here and hereafter, $\bar{0} = 1$, $\bar{1} = 0$ and $\Xi_{i_l} = i_l + (-1)^{i_l}(1 - \xi_l)$, $l = 1, 2, 3$, $i_l = 0, 1$ ($\Xi_{i_l} = 1 - \xi_l$, if $i_l = 0$, $\Xi_{i_l} = \xi_l$, if $i_l = 1$). The vectors $\mathbf{p}$, $\mathbf{q}$, $\mathbf{r}$ are directed along the edges, the vectors $\mathbf{u}$, $\mathbf{v}$, $\mathbf{w}$ play the role of "diagonals" of the faces, and vectors $\mathbf{d}$ are "inner diagonals" of the cell.

Considering ξ_l, $l = 1, 2, 3$, equal to 0 or 1 ($i_1, i_2, i_3 = 0, 1$), we compute the Jacobians at the corners of the cube P:

$$J_{i_1 i_2 i_3} = J(i_1, i_2, i_3) = \delta_{i_1 i_2 i_3} [\mathbf{p}, \mathbf{q}, \mathbf{r}]_{i_1 i_2 i_3} = 6 V^{pqr}_{i_1 i_2 i_3}.$$

Here and hereafter, $\delta_{i_1 i_2 i_3} = (-1)^{i_1 + i_2 + i_3}$, and lower indices related to brackets refer to each element inside brackets. The value $J_{i_1 i_2 i_3}$ is six times the volume of a tetrahedron with the corner $\mathbf{x}_{i_1 i_2 i_3}$ and edges $\mathbf{p}_{i_1 i_2 i_3}$, $\mathbf{q}_{i_1 i_2 i_3}$, $\mathbf{r}_{i_1 i_2 i_3}$ (the notation $V^{pqr}_{i_1 i_2 i_3}$). The notations of such type will be used to designate the volumes of tetrahedrons with the corner $\mathbf{x}_{i_1 i_2 i_3}$ and the edges corresponding to superscripts.

The properties of the Jacobian are studied in [3, 6, 7, 8, 13, 14, 15]. In two dimensions, a bilinear map of a unit square is considered. Its Jacobian is a linear function. If the Jacobian is positive at the corners of the square, then due to linearity the Jacobian will be positive everywhere in the square. The converse is also true. In two dimensions, if the Jacobian is positive, then the cell is convex, and the condition of nondegeneracy of cells is equivalent to the condition of convexity of cells.

A three-dimensional case is much more complicated. Since the faces of the cell can be nonplanar, the cell can be nonconvex. In [15] there is an example rejected the statement from [13] that the Jacobian in three dimensions is positive everywhere if and only if the Jacobian is positive at the corners of a cube. It is also demonstrated that for the positivity of the Jacobian in the interior of the cube the positivity of the Jacobian on the edges of

the cube and even on the faces is not sufficient. So, in [15] it is shown that simple tests in general situations do not guarantee nondegeneracy of cells. The Jacobian in [15] is written in terms of polynomials, the coefficients of which are the values of the Jacobian on the edges. In [6, 7, 8], another formulas of the Jacobian are obtained. These formulas can be easily reduced to the forms from [15]. By means of formulas from [6, 7, 8] the positivity of the Jacobian is investigated. Now the main results from [6, 7, 8] necessary for further consideration will be presented.

Substituting (9.5) into (9.4) we have

$$J = \left[\sum_{i_2,i_3=0}^{1} \mathbf{p}_{0i_2i_3}\Xi_{i_2}\Xi_{i_3}, \sum_{i_1,i_3=0}^{1} \mathbf{q}_{i_10i_3}\Xi_{i_1}\Xi_{i_3}, \sum_{i_1,i_2=0}^{1} \mathbf{r}_{i_1i_20}\Xi_{i_1}\Xi_{i_2} \right].$$

Decomposing the triple scalar product of sums of vectors into a sum of triple scalar products and taking common factors of vectors out of the sign of the triple scalar product we get the following lemmas from [7].

Lemma 1. *The Jacobian of trilinear map* (9.2) *can be written in the form*

$$\begin{aligned} J \;=\; & \sum_{i_1,i_2,i_3=0}^{1} \alpha_{i_1i_2i_3}\Xi_{i_1}{}^2\Xi_{i_2}{}^2\Xi_{i_3}{}^2 \\ & + \sum_{k=1}^{3} \sum_{\substack{i_l,i_m=0,\\(klm)=(123)}}^{1} \left(\sum_{i_k=0}^{1} \beta_{i_li_m}^{ki_k} \right) \xi_k(1-\xi_k)\Xi_{i_l}{}^2\Xi_{i_m}{}^2 \\ & + \sum_{k=1}^{3} \sum_{i_k=0}^{1} \left(\sum_{\substack{i_l,i_m=0,\\(klm)=(123)}}^{1} \gamma_{i_li_m}^{ki_k} \right) \Xi_{i_k}{}^2\xi_l(1-\xi_l)\xi_m(1-\xi_m) \\ & + \sum_{i_1,i_2,i_3=0}^{1} \kappa_{i_1i_2i_3}\xi_1(1-\xi_1)\xi_2(1-\xi_2)\xi_3(1-\xi_3), \end{aligned} \tag{9.7}$$

where

$$\begin{aligned} \alpha_{i_1i_2i_3} &= [\mathbf{p}_{0i_2i_3}, \mathbf{q}_{i_10i_3}, \mathbf{r}_{i_1i_20}], \quad \kappa_{i_1i_2i_3} = [\mathbf{p}_{0i_2i_3}, \mathbf{q}_{i_10\bar{i}_3}, \mathbf{r}_{\bar{i}_1\bar{i}_20}], \\ \beta_{i_2i_3}^{1i_1} = \beta_{i_1i_2i_3}^{1} &= [\mathbf{p}_{0i_2i_3}, \mathbf{q}_{i_10i_3}, \mathbf{r}_{\bar{i}_1i_20}], \quad \gamma_{i_2i_3}^{1i_1} = \gamma_{i_1i_2i_3}^{1} = [\mathbf{p}_{0i_2i_3}, \mathbf{q}_{i_10\bar{i}_3}, \mathbf{r}_{i_1\bar{i}_20}], \\ \beta_{i_3i_1}^{2i_2} = \beta_{i_1i_2i_3}^{2} &= [\mathbf{p}_{0i_2i_3}, \mathbf{q}_{i_10i_3}, \mathbf{r}_{i_1\bar{i}_20}], \quad \gamma_{i_3i_1}^{2i_2} = \gamma_{i_1i_2i_3}^{2} = [\mathbf{p}_{0i_2i_3}, \mathbf{q}_{i_10\bar{i}_3}, \mathbf{r}_{\bar{i}_1i_20}], \\ \beta_{i_1i_2}^{3i_3} = \beta_{i_1i_2i_3}^{3} &= [\mathbf{p}_{0i_2i_3}, \mathbf{q}_{i_10\bar{i}_3}, \mathbf{r}_{i_1i_20}], \quad \gamma_{i_1i_2}^{3i_3} = \gamma_{i_1i_2i_3}^{3} = [\mathbf{p}_{0i_2i_3}, \mathbf{q}_{i_10i_3}, \mathbf{r}_{\bar{i}_1\bar{i}_20}], \end{aligned} \tag{9.8}$$

and indices k, l, m *form the permutation of the cycle* (123)*; i.e.,* k, l, m *are equal to the values* $1, 2, 3, 2, 3, 1, 3, 1, 2$*, respectively.* (*The last one is denoted by* $(klm) = (123)$.)

Let us introduce the notations $\alpha_{i_2i_3}^{1i_1} = \alpha_{i_3i_1}^{2i_2} = \alpha_{i_1i_2}^{3i_3} = \alpha_{i_1i_2i_3}$.

Lemma 2. *The following relations are valid for coefficients* (9.8):

$$\alpha_{i_1i_2i_3} = [\mathbf{p}_{0i_2i_3}, \mathbf{q}_{i_10i_3}, \mathbf{r}_{i_1i_20}] = \delta_{i_1i_2i_3}[\mathbf{p}, \mathbf{q}, \mathbf{r}]_{i_1i_2i_3} = 6V^{pqr}_{i_1i_2i_3} = J_{i_1i_2i_3} \tag{9.9}$$

$$\begin{aligned} \beta^1_{i_1i_2i_3} &= [\mathbf{p}_{0i_2i_3}, \mathbf{q}_{i_10i_3}, \mathbf{r}_{\bar{i}_1i_21}] = \delta_{i_1i_2i_3}[\mathbf{p}, \mathbf{q}, \mathbf{v}]_{i_1i_2i_3} = 6V^{pqv}_{i_1i_2i_3}, \\ \beta^2_{i_1i_2i_3} &= [\mathbf{p}_{0i_2i_3}, \mathbf{q}_{i_10i_3}, \mathbf{r}_{i_1\bar{i}_20}] = \delta_{i_1i_2i_3}[\mathbf{p}, \mathbf{q}, \mathbf{u}]_{i_1i_2i_3} = 6V^{pqu}_{i_1i_2i_3}, \\ \beta^3_{i_1i_2i_3} &= [\mathbf{p}_{0i_2i_3}, \mathbf{q}_{i_10\bar{i}_3}, \mathbf{r}_{i_1i_20}] = \delta_{i_1i_2i_3}[\mathbf{p}, \mathbf{u}, \mathbf{r}]_{i_1i_2i_3} = 6V^{pur}_{i_1i_2i_3}, \end{aligned} \tag{9.10}$$

$$\begin{aligned} \sum_{\substack{i_l,i_m=0 \\ (klm)=(123)}}^{1} \gamma^{ki_k}_{i_li_m} &= \sum_{\substack{i_l,i_m=0 \\ (klm)=(123)}}^{1} \left(\bar{\gamma}^{ki_k}_{i_li_m} + \mu^{ki_k}_{i_li_m}\right) = \sum_{\substack{i_l,i_m=0 \\ (klm)=(123)}}^{1} \bar{\gamma}^{ki_k}_{i_li_m} \\ &= \sum_{\substack{i_l,i_m=0 \\ (klm)=(123)}}^{1} \left(-\alpha^{ki_k}_{i_li_m} + \beta^{li_l}_{i_mi_k} + \beta^{mi_m}_{i_ki_l}\right), \end{aligned} \tag{9.11}$$

$$\begin{aligned} \mu^{1i_1}_{i_2i_3} &= \mu^1_{i_1i_2i_3} = \delta_{i_1i_2i_3}[\mathbf{q}, \mathbf{u}, \mathbf{r}]_{i_1i_2i_3}, \\ \bar{\gamma}^{1i_1}_{i_2i_3} &= \bar{\gamma}^1_{i_1i_2i_3} = \delta_{i_1i_2i_3}[\mathbf{d}, \mathbf{q}, \mathbf{r}]_{i_1i_2i_3} = 6V^{dqr}_{i_1i_2i_3}, \\ \mu^{2i_2}_{i_3i_1} &= \mu^2_{i_1i_2i_3} = \delta_{i_1i_2i_3}[\mathbf{v}, \mathbf{p}, \mathbf{r}]_{i_1i_2i_3}, \\ \bar{\gamma}^{2i_2}_{i_3i_1} &= \bar{\gamma}^2_{i_1i_2i_3} = \delta_{i_1i_2i_3}[\mathbf{p}, \mathbf{d}, \mathbf{r}]_{i_1i_2i_3} = 6V^{pdr}_{i_1i_2i_3}, \\ \mu^{3i_3}_{i_1i_2} &= \mu^3_{i_1i_2i_3} = \delta_{i_1i_2i_3}[\mathbf{q}, \mathbf{p}, \mathbf{w}]_{i_1i_2i_3}, \\ \bar{\gamma}^{3i_3}_{i_1i_2} &= \bar{\gamma}^3_{i_1i_2i_3} = \delta_{i_1i_2i_3}[\mathbf{p}, \mathbf{q}, \mathbf{d}]_{i_1i_2i_3} = 6V^{pqd}_{i_1i_2i_3}, \end{aligned} \tag{9.12}$$

$$\sum_{i_1,i_2,i_3=0}^{1} \kappa_{i_1i_2i_3} = \sum_{i_1,i_2,i_3=0}^{1} \sum_{k=1}^{3} \beta^k_{i_1i_2i_3} - 2 \sum_{i_1,i_2,i_3=0}^{1} \alpha_{i_1i_2i_3} \tag{9.13}$$

$$= 2\bar{\kappa}_{000} + 2\bar{\kappa}_{111}, \quad \bar{\kappa}_{lll} = \delta_{lll}[\mathbf{u}, \mathbf{v}, \mathbf{w}]_{lll} = 6V^{uvw}_{lll}, \quad l = 0, 1, \tag{9.14}$$

$$\sum_{i_1,i_2,i_3=0}^{1} \sum_{k=1}^{3} \gamma^k_{i_1i_2i_3} = \sum_{i_1,i_2,i_3=0}^{1} \alpha_{i_1i_2i_3} + 4(\bar{\kappa}_{000} + \bar{\kappa}_{111}), \tag{9.15}$$

$$\sum_{i_1,i_2,i_3=0}^{1} \sum_{k=1}^{3} \beta^k_{i_1i_2i_3} = 2 \sum_{i_1,i_2,i_3=0}^{1} \alpha_{i_1i_2i_3} + 2(\bar{\kappa}_{000} + \bar{\kappa}_{111}). \tag{9.16}$$

Lemmas 1 and 2 give the following result: the Jacobian of the mapping can be represented by the sum of polynomials of the sixth degree (second degree in each of variables). The coefficients of the polynomial are expressed in terms of the volumes of tetrahedrons of four types (9.9), (9.10), (9.12) ($\bar{\gamma}^k_{i_1i_2i_3}$), (9.14). Tetrahedrons (9.9) are formed by three edges (with the common corner), tetrahedrons (9.10) by two edges and "diagonal" of one of adjacent faces, tetrahedrons (9.12) ($\bar{\gamma}^k_{i_1i_2i_3}$) by two edges and "inner diagonal" of

the cell, tetrahedrons (9.14) by "diagonals" of faces. The total number of tetrahedrons is $8+24+24+2=58$.

Finally, substituting (9.11), (9.13) into (9.7), we have in [7] the Jacobian (9.7) in the form of polynomial of the degree not higher than fourth.

Theorem 4. *The Jacobian of trilinear map* (9.2) *is a polynomial of the degree not higher than fourth* (*second in each of variables*). *It can be written in the form*

$$J = \sum_{i_1,i_2,i_3=0}^{1} \alpha_{i_1 i_2 i_3} \Xi_{i_1} \Xi_{i_2} \Xi_{i_3} (\Xi_{i_1} + \Xi_{i_2} + \Xi_{i_3} - 2) + \sum_{k=1}^{3} \sum_{\substack{i_l,i_m=0 \\ (klm)=(123)}}^{1} \left(\sum_{i_k=0}^{1} \beta_{i_l i_m}^{k i_k} \right) \xi_k (1-\xi_k) \Xi_{i_l} \Xi_{i_m}. \tag{9.17}$$

Since $\Xi_{i_1} + \Xi_{i_2} + \Xi_{i_3} - 2 = 1 - \Xi_{\bar{i}_1} - \Xi_{\bar{i}_2} - \Xi_{\bar{i}_3}$, the formula (9.17) can be reduced to the following one:

$$J = \sum_{i_1,i_2,i_3=0}^{1} \alpha_{i_1 i_2 i_3} \Xi_{i_1} \Xi_{i_2} \Xi_{i_3} + \sum_{k=1}^{3} \sum_{\substack{i_l,i_m=0 \\ (klm)=(123)}}^{1} \left(\sum_{i_k=0}^{1} \left(\beta_{i_l i_m}^{k i_k} - \alpha_{i_l i_m}^{k i_k} \right) \right) \xi_k (1-\xi_k) \Xi_{i_l} \Xi_{i_m}. \tag{9.18}$$

Formula (9.18) requires computations of eight volumes of tetrahedrons (9.9) and 12 triple scalar products

$$\begin{aligned} \phi_{i_2 i_3}^1 &= \delta_{i_1 i_2 i_3} \left[\mathbf{p}_{i_1 i_2 i_3}, \mathbf{p}_{\bar{i}_1 \bar{i}_2 i_3}, \mathbf{p}_{i_1 i_2 \bar{i}_3} \right] = \sum_{i_1=0}^{1} \left(\beta_{i_1 i_2 i_3}^1 - \alpha_{i_1 i_2 i_3} \right), \\ \phi_{i_1 i_3}^2 &= \delta_{i_1 i_2 i_3} \left[\mathbf{q}_{\bar{i}_1 \bar{i}_2 i_3}, \mathbf{q}_{i_1 i_2 i_3}, \mathbf{q}_{i_1 i_2 \bar{i}_3} \right] = \sum_{i_2=0}^{1} \left(\beta_{i_1 i_2 i_3}^2 - \alpha_{i_1 i_2 i_3} \right), \\ \phi_{i_1 i_2}^3 &= \delta_{i_1 i_2 i_3} \left[\mathbf{r}_{\bar{i}_1 i_2 \bar{i}_3}, \mathbf{r}_{i_1 \bar{i}_2 i_3}, \mathbf{r}_{i_1 i_2 i_3} \right] = \sum_{i_3=0}^{1} \left(\beta_{i_1 i_2 i_3}^3 - \alpha_{i_1 i_2 i_3} \right). \end{aligned} \tag{9.19}$$

The Jacobian J on lines $\xi_l = \text{const}$, $\xi_m = \text{const}$, $l \neq m$, $l, m = 1, 2, 3$ is a quadratic trinomial. In particular, on the edges the Jacobian is of the form

$$\begin{aligned} J(\xi_1, \xi_2, \xi_3)\Big|_{\substack{\xi_l = i_l \\ \xi_m = i_m}} &= \alpha_{i_l i_m}^{k0} (1-\xi_k)^2 + \alpha_{i_l i_m}^{k1} \xi_k{}^2 + \\ \left(\beta_{i_l i_m}^{k0} + \beta_{i_l i_m}^{k1} \right) \xi_k (1-\xi_k), \ (klm) &= (123), \quad i_l, i_m = 0, 1. \end{aligned} \tag{9.20}$$

If a hexahedral cell is a parallelepiped, then

$$\alpha_{i_1 i_2 i_3} = \alpha_{000} = \beta_{i_1 i_2 i_3}^k, \quad k = 1, 2, 3, \quad i_1, i_2, i_3 = 0, 1,$$

and the formula of the Jacobian (9.18) is essentially simplified:

$$J = \alpha_{000}(1 - \xi_1 - \xi_2 - \xi_3) + \alpha_{100}\xi_1 + \alpha_{010}\xi_2 + \alpha_{001}\xi_3 = \alpha_{000} = \text{const}.$$

9.3.2 Positivity of the Jacobian of a Trilinear Map

The condition $J > 0$ implies in [7] the inequalities

$$
\begin{aligned}
&J\ (i_1, i_2, i_3) = \alpha_{i_1 i_2 i_3} > 0,\ i_1, i_2, i_3 = 0, 1; \qquad (9.21)\\
&J\ (\xi_1, \xi_2, \xi_3)\Big|_{\substack{\xi_k = \frac{1}{2} \\ \xi_l, \xi_m = i_l, i_m}} = \frac{1}{4} \sum_{i_k=0}^{1} \left(\alpha_{i_l i_m}^{k i_k} + \beta_{i_l i_m}^{k i_k} \right) > 0,\ i_l, i_m = 0, 1;\\
&J\ (\xi_1, \xi_2, \xi_3)\Big|_{\substack{\xi_k = i_k \\ \xi_l, \xi_m = \frac{1}{2}}} = \frac{1}{8} \sum_{i_l, i_m=0}^{1} \left(\beta_{i_m i_k}^{l i_l} + \beta_{i_k i_l}^{m i_m} \right) > 0,\ i_k = 0, 1;\\
&J\left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right) = \frac{1}{16} \left(\sum_{i_1, i_2, i_3=0}^{1} \sum_{k=1}^{3} \beta_{i_1 i_2 i_3}^{k} - \sum_{i_1, i_2, i_3=0}^{1} \alpha_{i_1 i_2 i_3} \right) > 0,
\end{aligned}
$$

which compose the necessary conditions 1 of nondegeneracy of a cell.

Theorem 5 (Necessary contitions 1). *In order that the Jacobian be positive in the unit cube including its boundary it is necessary that conditions* (9.21) *be satisfied.*

The polynomials corresponding to the coefficients from (9.7) in the interior of the cube P are positive. Hence, if coefficients $\alpha_{i_1 i_2 i_3}$ are positive, and the rest of the coefficients are greater than or equal to zero, then the Jacobian is positive in the interior. It is easy to see that J is positive on the boundary. Since the coefficients $\gamma_{i_1 i_2 i_3}^{k}$, $\sum_{i_1, i_2, i_3=0}^{1} \kappa_{i_1 i_2 i_3}$ can be expressed in terms of $\alpha_{i_1 i_2 i_3}$, $\beta_{i_1 i_2 i_3}^{k}$, (see. (9.13)–(9.15)), the conditions of the positivity of the Jacobian (sufficient conditions 1) have the form (see [7])

$$
\begin{aligned}
\alpha_{i_1 i_2 i_3} &> 0, \quad i_1, i_2, i_3 = 0, 1; \qquad (9.22)\\
\sum_{i_k=0}^{1} \beta_{i_l i_m}^{k i_k} &\geq \mathrm{B}_{i_l i_m}^{k}, \quad i_l, i_m = 0, 1;\\
\sum_{i_l, i_m=0}^{1} \gamma_{i_l i_m}^{k i_k} &= \sum_{i_l, i_m=0}^{1} \left(\beta_{i_m i_k}^{l i_l} + \beta_{i_k i_l}^{m i_m} - \alpha_{i_l i_m}^{k i_k} \right) \geq \Gamma_{i_k}^{k}, \quad i_k = 0, 1;\\
2\bar{\kappa}_{000} + 2\bar{\kappa}_{111} &= \sum_{i_1, i_2, i_3=0}^{1} \sum_{k=1}^{3} \beta_{i_1 i_2 i_3}^{k} - 2 \sum_{i_1, i_2, i_3=0}^{1} \alpha_{i_1 i_2 i_3} \geq \mathrm{K},
\end{aligned}
$$

where $\alpha_{i_1 i_2 i_3}$, $\beta_{i_1 i_2 i_3}^{k}$ are calculated according to formulas (9.9), (9.10), and

$$
\mathrm{B}_{i_l i_m}^{k} = \Gamma_{i_k}^{k} = \mathrm{K} = 0. \qquad (9.23)
$$

Theorem 6 (Sufficient conditions 1). *In order that the Jacobian of the trilinear mapping be positive in the unit cube including its boundary it is sufficient that conditions* (9.22), (9.23) *be satisfied.*

The following corollary is obvious.

Corollary 1. *A hexahedral cell is nondegenerate if volumes of* 58 *tetrahedrons (9.9), (9.10), (9.12), (9.14), corresponding to coefficients* $\alpha_{i_1i_2i_3}$, $\beta^k_{i_1i_2i_3}$, $\bar{\gamma}^k_{i_1i_2i_3}$, $\bar{\kappa}_{000}$, $\bar{\kappa}_{111}$ *are positive.*

It is clear that necessary conditions assume a wider range of values for $\alpha_{i_1i_2i_3}$, $\beta^k_{i_1i_2i_3}$ than sufficient conditions 1. Both conditions include 27 inequalities(for 8 corners, 12 edges, 6 faces, and the interior part of a cell).

The expression for J is also positive if (9.19) and (9.9) are positive. However, these conditions restrict (in comparison with (9.22)) the set of values of $\alpha_{i_1i_2i_3}$, $\beta^k_{i_1i_2i_3}$ for which J is positive. Therefore, sufficient conditions 1 are more general conditions.

The conditions of corollary 1 is also more restrictive than sufficient conditions 1.

The Jacobian on edges is either a linear function or a quadratic trinomial, so it is easy to find necessary and sufficient conditions of its positivity on the edges.

Theorem 7. *In order that the Jacobian of the trilinear mapping be positive on edges of the cube* P *it is necessary and sufficient that conditions*

$$\alpha_{i_1i_2i_3} > 0,\ i_1, i_2, i_3 = 0, 1;$$

$$\sum_{i_k=0}^{1} \beta^{ki_k}_{i_li_m} > -2\sqrt{\alpha^{k0}_{i_li_m}\,\alpha^{k1}_{i_li_m}},\ (klm) = (123),\ i_l, i_m = 0, 1 \tag{9.24}$$

be satisfied.

The Jacobian J on the faces $\xi_l = 0, 1,\ l = 1, 2, 3$ and general formula (9.18) can be written (see [6],[8]) as a quadratic trinomial in one variable with fixed other variables (one or two, respectively). An attempt to find necessary and sufficient conditions of positivity of the Jacobian using its representation in the form of a quadratic trinomial fails since even in the case of faces the discriminant of a quadratic trinomial is a fourth-degree polynomial in one variable; in the general case, the discriminant will be the polynomial in two variables. Because of above reasons an analysis of the discriminant on the property of having fixed sign fails. However, it is possible to find sufficient conditions 2 more general than sufficient conditions 1 and necessary conditions 2 more restrictive than necessary conditions 1.

Theorem 8 (Sufficient conditions 2). *In order that the Jacobian of the trilinear mapping be positive in the whole cube* P *including its boundary it is sufficient that conditions* (9.22)

in strick inequality form where

$$
\begin{aligned}
\mathrm{B}^k_{i_l i_m} &= -2\min(\alpha^{k0}_{i_l i_m}, \alpha^{k1}_{i_l i_m}), \quad (klm) = (123), \quad i_l, i_m = 0, 1; \\
\Gamma^k_{i_k} &= -2 \min_{j_l, j_m = 0,1} \left(2\alpha^{k i_k}_{j_l j_m} + \sum_{i_l=0}^{1} \beta^{l i_l}_{j_m i_k} + \sum_{i_m=0}^{1} \beta^{m i_m}_{i_k j_l} \right), \quad i_k = 0, 1; \qquad (9.25) \\
\mathrm{K} &= -2 \min_{j_1, j_2, j_3 = 0,1} \left(3\alpha_{j_1 j_2 j_3} + 2 \left(\sum_{i_2, i_3 = 0}^{1} \gamma^1_{j_1 i_2 i_3} + \sum_{i_1, i_3 = 0}^{1} \gamma^2_{i_1 j_2 i_3} \right.\right. \\
&\quad \left.\left. + \sum_{i_1, i_2 = 0}^{1} \gamma^3_{i_1 i_2 j_3} \right) + \sum_{i_1=0}^{1} \beta^1_{i_1 j_2 j_3} + \sum_{i_2=0}^{1} \beta^2_{j_1 i_2 j_3} + \sum_{i_3=0}^{1} \beta^3_{j_1 j_2 i_3} \right)
\end{aligned}
$$

be satisfied.

It is obvious that cells satisfying sufficient conditions 1 satisfy sufficient condition 2 consisting of (9.22) and (9.25) . Sufficient conditions 2 are more general than sufficient conditions 1 but demand more computations. It is possible to show that when sufficient conditions 2 are satisfied, necessary conditions 1 also hold (but not vice versa).

Since on lines $\xi_l = \text{const}$, $\xi_m = \text{const}$, $l \neq m$, $l, m = 1, 2, 3$, the Jacobian is either a linear function or a quadratic trinomial, we can get necessary and sufficient conditions of Jacobian positivity on these lines, and, thus, restrict necessary conditions 1. To get necessary conditions 2, we shall write down necessary and sufficient conditions of Jacobian's positivity on the edges and midlines passing through the midpoints of faces and midpoint $(0.5, 0.5, 0.5)$ of the cube P. So, necessary conditions 2 will be composed of conditions (9.24) and the following conditions

$$
\begin{aligned}
\phi^2_{i_1 0} + \phi^2_{i_1 1} &> -2(\alpha_0 + \alpha_1 + 2\sqrt{\alpha_0 \alpha_1}), \ \alpha_j = J(i_1, j, 0.5), \\
\phi^3_{i_1 0} + \phi^3_{i_1 1} &> -2(\alpha_0 + \alpha_1 + 2\sqrt{\alpha_0 \alpha_1}), \ \alpha_j = J(i_1, 0.5, j), j = 0, 1, \\
\\
\phi^1_{i_2 0} + \phi^1_{i_2 1} &> -2(\alpha_0 + \alpha_1 + 2\sqrt{\alpha_0 \alpha_1}), \ \alpha_j = J(j, i_2, 0.5,), \qquad (9.26) \\
\phi^3_{0 i_2} + \phi^3_{1 i_2} &> -2(\alpha_0 + \alpha_1 + 2\sqrt{\alpha_0 \alpha_1}), \ \alpha_j = J(0.5, i_2, j), j = 0, 1, \\
\\
\phi^1_{0 i_3} + \phi^2_{1 i_3} &> -2(\alpha_0 + \alpha_1 + 2\sqrt{\alpha_0 \alpha_1}), \ \alpha_j = J(j, 0.5, i_3), \\
\phi^2_{0 i_3} + \phi^3_{1 i_3} &> -2(\alpha_0 + \alpha_1 + 2\sqrt{\alpha_0 \alpha_1}), \ \alpha_j = J(0.5, j, i_3), j = 0, 1,
\end{aligned}
$$

on midlines for the planes $\xi_1 = i_1$, $\xi_2 = i_2$, $\xi_3 = i_3$, $i_k = 0, 1$, respectively, and

$$
\sum_{i_1, i_2 = 0}^{1} \phi^k_{i_1 i_2} > -4(\alpha_0 + \alpha_1 + 2\sqrt{\alpha_0 \alpha_1}), \qquad (9.27)
$$

$$
\alpha_{i_k} = J(\xi_1, \xi_2, \xi_3) \Big|_{\substack{\xi_k = i_k \\ \xi_l, \xi_m = \frac{1}{2}}}, \ i_k = 0, 1, \ (klm) = (123),
$$

on midlines passing through the midpoint $(0.5, 0.5, 0.5)$. Here values of J are given in (9.21), and ϕ^k_{lm} are given in (9.19).

Theorem 9 (Necessary conditions 2). *In order that the Jacobian be positive in the unit cube including its boundary it is necessary that conditions* (9.21), (9.26), (9.27) *be satisfied.*

To see how general the obtained conditions are, a numerical experiment was carried out. The corners of hexahedron were selected randomly. Of 10^7 hexahedrons randomly generated by the computer only 36 251 were found to have positive Jacobians at the corners of a cell. From them only 14 622 cases satisfied necessary conditions 1 for edges, 14 010 cases satisfied necessary conditions 1 for faces, and 14 004 cases satisfied necessary conditions 1 for the whole cell. Necessary conditions 2 allowed us to exclude some more degenerate cells. The number of cases when necessary conditions 2 were satisfied, was equal to 11 533.

The Jacobian was positive in 11 481 cases. Sufficient conditions 1 were satisfied in 33.93% of the cases (from the number 11481), sufficient conditions 2 were satisfied in 75.08% of the cases. In 11733 cases the Jacobian was positive on the edges, 97.85% of them had positive Jacobian everywhere in the whole cell. The success rate of necessary conditions 2 was 99.54%.

Numerical results showed the following.

1. Necessary conditions allowed us to exclude a great number of cells which were degenerate.

2. In less than one third of cases when the Jacobian was positive at the corners of a cell the Jacobian was positive everywhere in the cell. Therefore, it would be unreliable to draw a conclusion about the invertibility of the Jacobian on the basis of positive Jacobians at the corners of a cell.

3. Necessary and sufficient conditions of positivity of the Jacobian on the edges (provided that necessary conditions 1 were satisfied) in a large percentage of cases gave positive Jacobian everywhere in the cell. The success rate of necessary conditions 2 was higher. However, both of these conditions also did not guarantee the invertibility of the trilinear map.

4. Sufficient conditions 2 permitted to recognize the nondegeneracy of cells in most of cases.

5. In other cases, nondegeneracy was established by a special numerical algorithm.

In Figures 9.2, 9.3, 9.4, 9.5, 9.6 and tables 9.1, 9.2, 9.3 there are hexahedrons (a edges, b faces) nondegeneracy and degeneracy of which were checked by means of obtained criteria. In Figures 9.2 and 9.3 there are nondegenerate cells that nondegeneracy was established by sufficient conditions.

9.3.3 A Special Algorithm of Testing the Jacobian on its Positivity

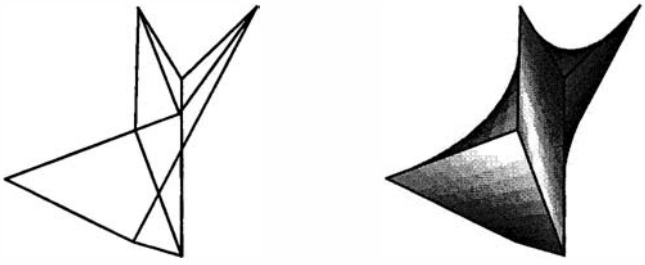
Initially in [7], to estimate the success rate of nondegeneracy conditions, computation of the Jacobian on the uniform grid with the number of nodes $10 \times 10 \times 10$ was carried out. The

Table 9.1: Nondegenerate cells. Sufficient conditions

Figure	9.2			9.3		
Values i_1, i_2, i_3	$x^1_{i_1 i_2 i_3}$	$x^2_{i_1 i_2 i_3}$	$x^3_{i_1 i_2 i_3}$	$x^1_{i_1 i_2 i_3}$	$x^2_{i_1 i_2 i_3}$	$x^3_{i_1 i_2 i_3}$
0,0,0	0.45738	0.36896	0.13358	0.88478	0.34019	0.58412
0,0,1	0.24993	0.43718	0.35387	0.43024	0.15050	0.96750
0,1,0	0.70201	0.91127	0.21267	0.51549	0.57111	0.35779
0,1,1	0.12916	0.72680	0.47524	0.33026	0.23357	0.05489
1,0,0	0.44215	0.32725	0.13254	0.72676	0.77997	0.37987
1,0,1	0.06207	0.30100	0.70946	0.91511	0.70163	0.81098
1,1,0	0.81539	0.17019	0.28804	0.08101	0.74778	0.42077
1,1,1	0.19301	0.11998	0.84236	0.05098	0.32246	0.58977

a *b*

Figure 9.2: Hexahedral cell satisfying sufficient conditions 1.



a *b*

Figure 9.3: Hexahedral cell satisfying sufficient conditions 2.

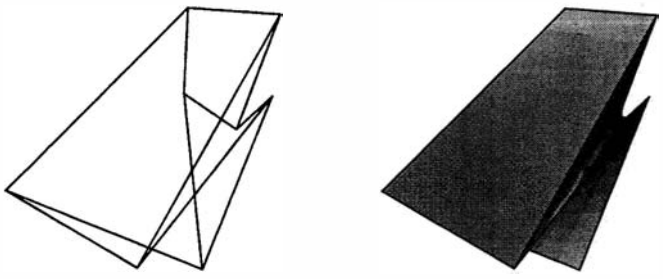
Table 9.2: Nondegenerate cell. Special algorithm

Figure	9.4		
i_1, i_2, i_3	$x^1_{i_1 i_2 i_3}$	$x^2_{i_1 i_2 i_3}$	$x^3_{i_1 i_2 i_3}$
0,0,0	0.07989	0.75985	0.95767
0,0,1	0.15525	0.16477	0.62461
0,1,0	0.87075	0.97539	0.23544
0,1,1	0.33835	0.70495	0.76928
1,0,0	0.06386	0.80644	0.68870
1,0,1	0.26285	0.50556	0.08884
1,1,0	0.69229	0.98026	0.20396
1,1,1	0.69885	0.48870	0.36548

number of cases with positive Jacobians on such a grid was 11582 provided that necessary conditions 1 were satisfied.

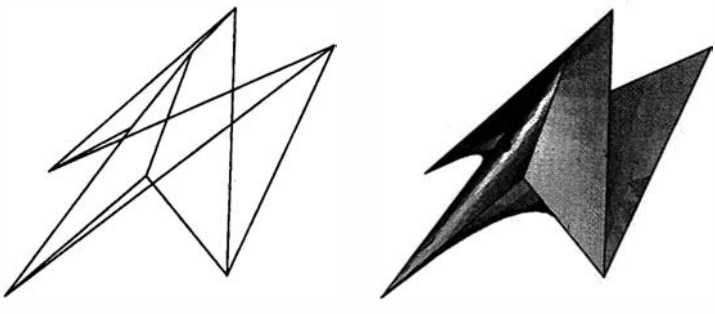
To recognize nondegenerate cells for the cases when necessary conditions 2 were satisfied but sufficient conditions 2 were not satisfied, a special algorithm of testing the Jacobian on its positivity was developed. It is an algorithm of a search for a minimal value of the Jacobian in the unit cube. It consists of two stages. First stage is a preliminary search. At this stage we find minimal values of the Jacobian on straight lines $\xi_1 = 0.1\ i_1$ and $\xi_2 = 0.1\ i_2,\ i_k = 0, 1, ..., 10,\ k = 1, 2$. Since on these lines the Jacobian is a guadratic trinomial, we can do this exactly. As a result of such a search, we have an initial approximation J^0_{min} for a minimal value of J for the next stage. At the next stage (improvement of the minimal value) a search for a minimal value is performed along coordinate directions $\xi_1,\ \xi_2,\ \xi_3$, sequentially. On each line, we also find a minimal value of the Jacobian precisely as for a quadratic trinomial. We utilize such a search on each iteration till the condition $|J^n_{min} - J^{n-1}_{min}| < \varepsilon$ will be satisfied. Here J^n_{min} is a minimal value of the Jacobian on nth iteration and ε is a small value. If, at one of the stages a negative value of the Jacobian appears, the process also stops. In this case, the cell is considered degenerate. Such a search for a minimal value of the Jacobian in a cube allowed us to exclude additionally more than 100 degenerate cases. This algorithm discovered itself as a reliable criteria. The number of grid nodes for the preliminary search was chosen by the experiment. The increase of the number of nodes up to 100×100 did not influence on the result. The small value ε was equal to 10^{-7}.

In Fig. 9.4 and table 9.2, there is an example of a cell that nondegeneracy was established by special numerical algorithm. In Figures 9.5 and 9.6 and table 9.3 there are examples of degenerate hexahedral cells satisfying necessary conditions 2 (with positive Jacobian on the edges and midlines) but with the Jacobian reaching its negative minimal value on the face (see Fig. 9.5) and with the Jacobian reaching its negative minimal value inside the cell (see Fig. 9.6). The neighborhood where the Jacobian reaches its minimal value (see table 9.4) is shown by filled circles (Figures 9.4, 9.5, 9.6).



a *b*

Figure 9.4: Nondegenerate hexahedral cell. Special algorithm.



a *b*

Figure 9.5: Degenerate hexahedral cell. Special algorithm.

a *b*

Figure 9.6: Degenerate hexahedral cell. Special algorithm.

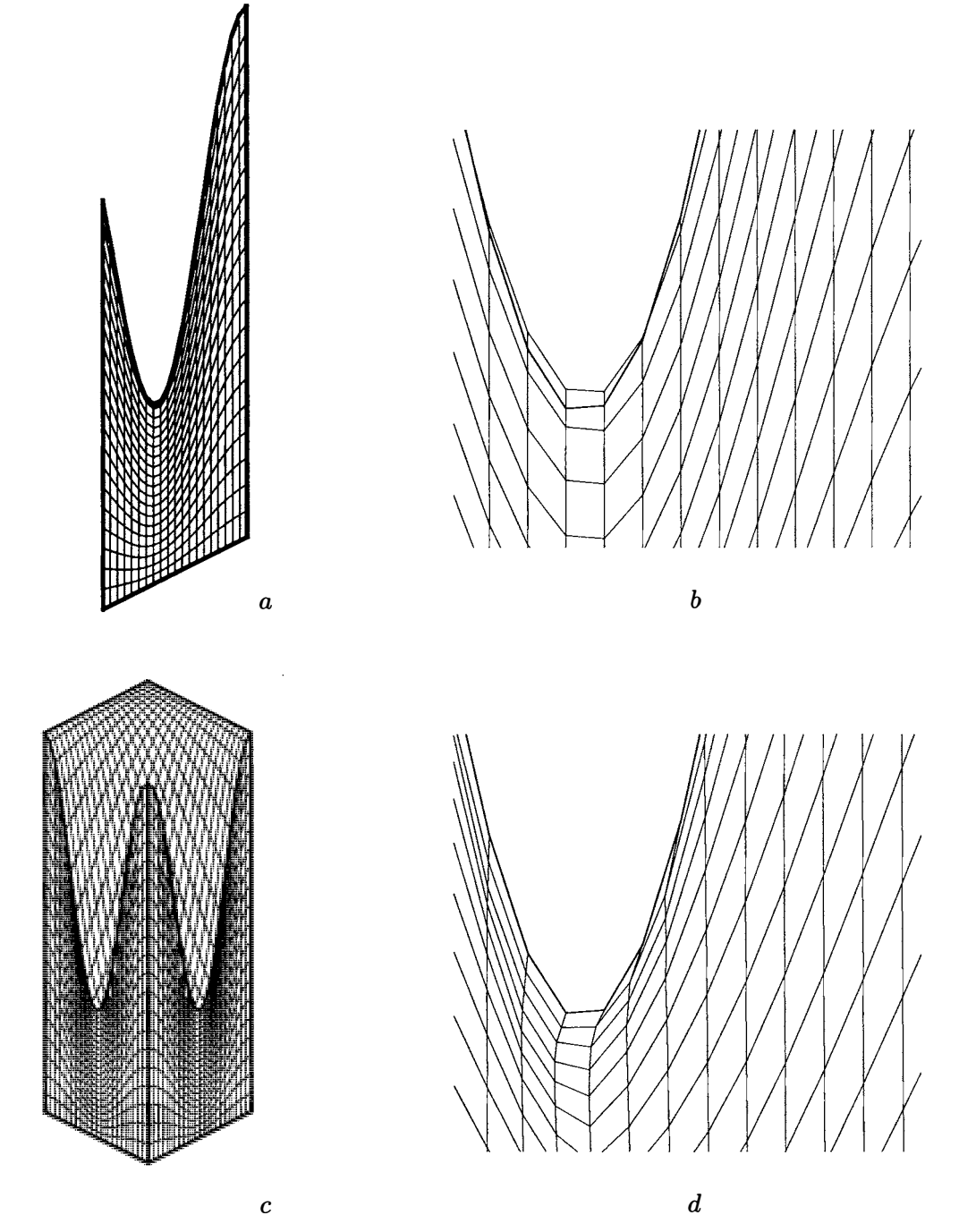


Figure 9.7: Three-dimensional grids:

a, b degenerate grid, c, d nondegenerate grid; a coordinate surface $i_2 = 17$, b fragment with degenerate cells, c grid on the boundary, d fragment of a nondegenerate grid.

Table 9.3: Degenerate cells. Special algorithm

Figure	9.5			9.6		
Values i_1, i_2, i_3	$x^1_{i_1 i_2 i_3}$	$x^2_{i_1 i_2 i_3}$	$x^3_{i_1 i_2 i_3}$	$x^1_{i_1 i_2 i_3}$	$x^2_{i_1 i_2 i_3}$	$x^3_{i_1 i_2 i_3}$
0,0,0	0.39484	0.67489	0.05749	0.26131	0.98417	0.90477
0,0,1	0.53665	0.71410	0.36458	0.58230	0.52242	0.08180
0,1,0	0.00574	0.51583	0.86179	0.73235	0.44415	0.98202
0,1,1	0.86002	0.21879	0.48213	0.30334	0.64329	0.52664
1,0,0	0.23829	0.68169	0.87376	0.74064	0.79921	0.97449
1,0,1	0.37713	0.79237	0.65927	0.98911	0.49795	0.76536
1,1,0	0.75112	0.64513	0.47855	0.81919	0.56288	0.92869
1,1,1	0.56661	0.86288	0.38965	0.01169	0.61644	0.24017

Table 9.4: Minimal value of the Jacobian

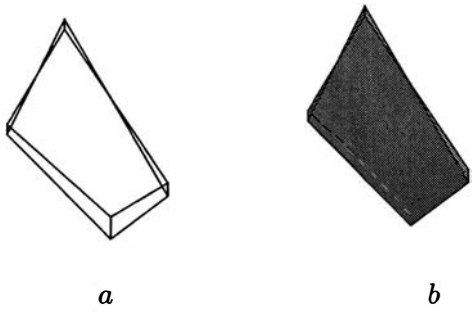
Figure	$\min J$	$\xi^1_{\min}$	$\xi^2_{\min}$	$\xi^3_{\min}$	$x^1_{\min}$	$x^2_{\min}$	$x^3_{\min}$
9.4	0.00234	0.00000	0.55907	1.00000	0.25761	0.46677	0.70549
9.5	-0.00008	0.66666	0.55555	1.00000	0.56036	0.70067	0.48295
9.6	-0.00031	0.88888	0.77777	0.62567	0.44557	0.59927	0.58074

9.3.4 Nondegeneracy Criteria in Practice of Structured Grid Generation

On the basis of above conditions, the program for testing three-dimensional structured grids on the nondegeneracy has been developed. The program checks each cell and gives the numbers of degenerate cells or reports that all cells are degenerate.

To recognize degenerate cells, necessary conditions are used. We give examples of degenerate cells for that necessary conditions were not satisfied already at the corners. Fore more complicated form of degenerate cells, other types of necessary conditions (on edges, faces and so on) or a special numerical algorithm can be used. Nondegeneracy of cells, in the most of considered cases, was established by means of sufficient conditions 1.

In Figure 9.7, there are structured grids for one domain: in cases a and b there is a degenerate grid that does not satisfy necessary conditions, in cases c and d there is a non-degenerate grid obtained by the global reconstruction algorithm described in chapter 10 (Figures 9.7 b and d show the fragments of a coordinate surfaces). In Fig. 9.7 a and b, a degenerate grid was obtained by numerical solving the Laplace equations in the computational domain. This algorithm is very often characterized by the property of producing degenerate grids for complicated configurations of the domains, for some simple configurations of domains, the algorithm produces nondegenerate grids. In Fig. 9.8 (a edges and b faces) there is an example of degenerate cell $15, 17, 20$ for a grid shown in Fig. 9.7 b. This cell does not satisfy necessary conditions at six corners of a cell since the mapping used for its generation has negative Jacobian at these corners.



a *b*

Figure 9.8: Degenerate hexahedral cell. Necessary conditions.

9.3.5 About Admissability of Degenerate Hexahedral Cells

It is necessary to note that, in some cases of complex configurations of domains, single degenerate ruled hexahedral cell can arise. As a rule, such cells can arise on the boundary of the domain (see the grid shown in Fig. 9.9 and generated by the global reconstruction algorithm from chapter 10). A grid shown in Fig. 9.9, along two edges of a curvilinear hexahedron, in this case one fourth of a sphere without cut inner central spherical part is represented (an example of a volume of revolution, see [16] and chapter 10), has degenerate hexahedral ruled cells with the Jacobian equal to zero in two vertices of a cells. Considered edges and two faces of a hexahedron of a physical domain are allocated in one plane. The cells allocated along this edges degenerate into curvilinear prisms (see Fig. 9.10). In this case, such degenerate grid can be considered admissible.

In the two-dimensional case, the discussion of analogous situation is contained in [17, 18, 19] and in chapter 5. In [17], triangular and nonconvex cells arising on the boundary for the configurations of the domains in the form of a semicircle or a sector of a circle, corresponding to an angle greater than 180 degree, are considered. Such cells are degenerate ruled quadrangles. Above cases demand additional modification of grid generation algorithms (see [17, 18, 19]) and numerical procedures for the solving physical problems. In these cases, cells are nonself-intersecting and can be considered as nondegenerate for another way of their construction (another mapping).

Namely because of admissability of such single cells as nondegenerate in [19] (in terms of another way of cell's construction or another mapping), not only convex quadrangles but also triangles and nonconvex quadrangles (nonself-intersecting cells) are considered. In the grid generation algorithm [19], the computation of nodes is carried out from the condition of nonself-intersection of nodes and from the condition of the minimum of grid quality functional [19]. The strategy for computation of nodes is different for convex and nonconvex special quadrangles determining the admissible set for the minimization of the

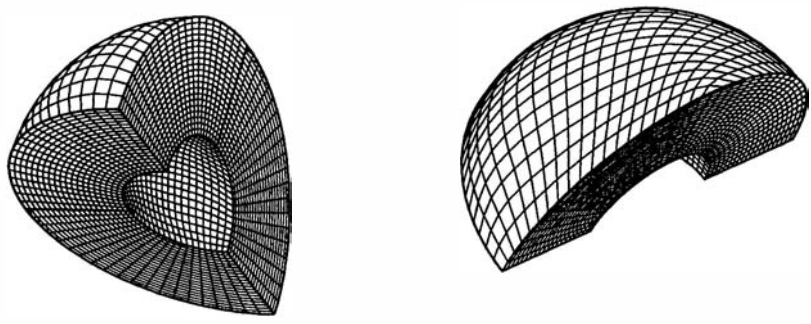


Figure 9.9: Examples of grids with degenerate hexahedral cells (different views).

a *b*

Figure 9.10: Hexahedral cell degenerating into prism.

functional.

To diagnose such degenerate cases, the following conditions of degeneracy for hexahedral cells have been elaborated.

1. Degeneration into the ruled prism with triangular base. Necessary conditions. If hexahedral cell degenerates into ruled prism with triangular base, then

$$\alpha_{i_1 i_2 i_3} = 0$$

only in two vertices of one edge ($i_k = 0, 1, \ k = 1, \ \text{or} \ k = 2, \ \text{or} \ k = 3$).

2. Degeneration into the polyhedron that is the union of two ruled prisms with triangular base. Necessary conditions. If hexahedral cell degenerates into the octahedron that is the union of two ruled prism with triangular base, then

$$\alpha_{i_1 i_2 i_3} \leq 0$$

only in two vertices of one edge ($i_k = 0, 1, \ k = 1, \ \text{or} \ k = 2, \ \text{or} \ k = 3$).

These conditions are only necessary. Full diagnostics of such cases can be carried out using the conditions obtained in the following section.

Since, in some cases of three-dimensional structured grid generation, other than hexahedral types of cells with curvilinear faces can arise, then, for these cases, conditions of nondegeneracy for other types of cells, having less number of faces, then six are needed. Necessity to have such conditions arises also in finite element analysis.

9.4 On the Nondegeneracy of Other Cells

Approximately at the same time when nondegeneracy conditions [6] for hexahedral cells have been obtained, in [9], conditions of the local invertibility of the mappings (in fact, due to theorems from [4, 5] nondegeneracy conditions) for pyramidal and prismatic finite elements and, later in [10], a numerical algorithm for checking the invertibility for hexahedral ruled cells have been published. These are the conditions of the Jacobian nonvanishing for the mappings of considered cells. The distinctive feature of investigations [9], [10] is that the mappings used for generation of finite elements are decomposed into a linear part $\mathbf{x}_T$ and a nonlinear part $\tilde{\mathbf{x}}$ and is considered in [9], [10] as a composition of two mappings

$$\mathbf{x} = \mathbf{x}_T \circ \tilde{\mathbf{x}}, \tag{9.28}$$

and properties of mapping are examined separately. If the Jacobian of a linear part is not equal to zero, then for verification of this fact for the general transformation (9.28), to check the Jacobian nonvanishing is needed for a nonlinear part that depends on a smaller number of parameters than the general mapping. So, obtained conditions also will depend on a smaller number of parameters.

9.4.1 Pyramidal Cell

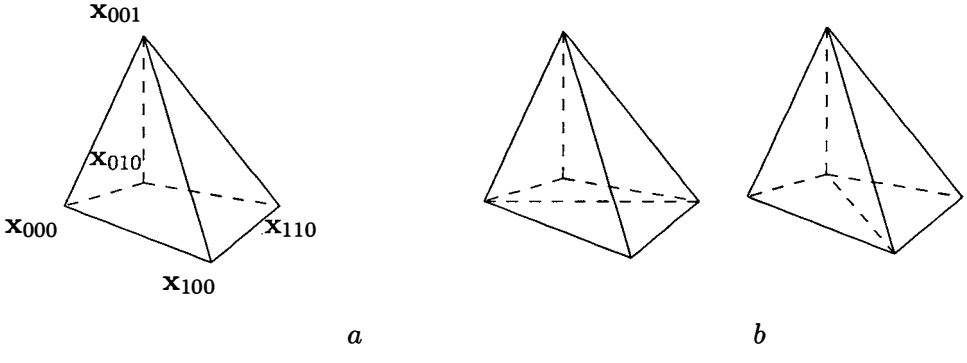


Figure 9.11: Pyramid and hexahedrons.

First in [9], it is considered the mapping of the unit pyramid P with the vertices $(0, 0, 0)$, $(1, 0, 0)$, $(0, 1, 0)$, $(1, 1, 0)$, that can be written in terms of notations (9.3) in the form

$$\mathbf{x}(\xi) = \mathbf{a}_{000} + \mathbf{a}_{100}\xi_1 + \mathbf{a}_{010}\xi_2 + \mathbf{a}_{001}\xi_3 + \mathbf{a}_{110}(\xi_1\xi_2 + \min(\xi_1, \xi_2)\xi_3). \tag{9.29}$$

This mapping is used for generation of pyramidal finite elements with planar triangular faces and ruled rectangular base (see Fig. 9.11 *a*). This mapping is nonsmooth. This singularity is separately discussed and it is shown that such singularity is not an obstacle in elaboration of numerical algorithms.

In [9, 10], the condition of positivity (negativity) of the Jacobian for the linear part

$$\mathbf{x}_T(\xi) = \mathbf{a}_{000} + \mathbf{a}_{100}\xi_1 + \mathbf{a}_{010}\xi_2 + \mathbf{a}_{001}\xi_3, \tag{9.30}$$

transforming the tetrahedron $T = \mathrm{conv}\{(0,0,0),(1,0,0),(0,1,0),(0,0,1)\}$ into the tetrahedron $\mathbf{x}_T(T) = \mathrm{conv}\{\mathbf{x}_{000}, \mathbf{x}_{100}, \mathbf{x}_{010}, \mathbf{x}_{001}\}$, is one of the necessary and sufficient conditions of invertibility and is checked above all. This condition in terms of (9.3),(9.6) can be written in the form

$$J_T = \det\mathbf{x}_T = [\mathbf{a}_{100}, \mathbf{a}_{010}, \mathbf{a}_{001}] = [\mathbf{p}, \mathbf{q}, \mathbf{r}]_{000} > 0\ (< 0). \tag{9.31}$$

If the Jacobian of a linear part $J_T \neq 0$, then it is possible to find $\tilde{\xi}_{110} = \mathbf{x}_T^{-1}(\mathbf{x}_{110})$ and to define the mapping corresponding to a nonlinear part of the mapping (9.29) according to the rule

$$\tilde{\mathbf{x}} = \xi + (\tilde{\xi}_{110} - \xi_{110})(\xi_1\xi_2 + \min(\xi_1, \xi_2)\xi_3), \quad \xi_{110} = (1,1,0)$$

and to examine its properties. Sufficient conditions of the positivity (negativity) of the Jacobian $J = \det\tilde{\mathbf{x}}$ are found in [9] in the form of following inequalities for the coordinates of the vector $\tilde{\xi}_{110}$:

$$\tilde{\xi}^1_{110} > 0, \quad \tilde{\xi}^2_{110} > 0, \quad \tilde{\xi}^1_{110} + \tilde{\xi}^2_{110} + \tilde{\xi}^3_{110} > 1.$$

These conditions and (9.31) give sufficient conditions of invertibility for the general mapping $\mathbf{x} = \mathbf{x}(\xi)$. In [9], their geometric interpretation (it will be given later in (9.33)) is obtained on the basis of the fact that $\det\tilde{\mathbf{x}}$ is a linear function in each of tetrahedrons $T_1 = \mathrm{conv}\{(0,0,0),(1,0,0),(1,1,0),(0,0,1)\}$ and $T_2 = \mathrm{conv}\{(0,0,0),(0,1,0),(1,1,0),(0,0,1)\}$ and it reaches the extremum in one of the vertices of tetrahedrons.

Let us obtain conditions of invertibility or nondegeneracy conditions for a pyramid using technique from [7]. Following the technique from [7] we do not separate linear and nonlinear parts of the mapping and investigate the Jacobian of a general mapping using its representation in terms of the volumes of a special tetrahedrons.

Theorem 10. *In order that the Jacobian* (9.29) *be positive negative it is necessary and sufficient that conditions*

$$\alpha_{000} > 0\ (< 0), \quad \beta^1_{100} > 0\ (< 0), \quad \bar{\gamma}^3_{110} = [\mathbf{p}, \mathbf{q}, \mathbf{d}]_{110} > 0\ (< 0), \quad \beta^2_{010} > 0\ (< 0)$$

be satisfied.

Proof. For a pyramid, the general mapping (9.29) is constructed by means of two its restrictions for tetrahedrons T_1 and T_2. It is seen that for each of the tetrahedrons the mapping is bilinear, and the Jacobians J_1 and J_2 have the form

$$\begin{aligned} J_1 &= [\mathbf{a}_{100} + \mathbf{a}_{110}\xi_2, \mathbf{a}_{010} + \mathbf{a}_{110}(\xi_1 + \xi_3), \mathbf{a}_{001} + \mathbf{a}_{110}\xi_2] \\ J_2 &= [\mathbf{a}_{100} + \mathbf{a}_{110}(\xi_2 + \xi_3), \mathbf{a}_{010} + \mathbf{a}_{110}\xi_1, \mathbf{a}_{001} + \mathbf{a}_{110}\xi_1]. \end{aligned} \tag{9.32}$$

The Jacobians are linear functions because the coefficients at quadratic terms after decomposing triple scalar products are vanishing since the coefficients are determinants of matrices with two equal columns or triple scalar products with two equal vectors. The Jacobians will be positive (negative) if and only if they are positive (negative) in the vertices of tetrahedrons. The last one, by virtue the vectors (9.3) are expressed in terms of vectors (9.6), is equivalent to the condition: tetrahedrons on the edges of a pyramid at the vertices $(0,0,0),(1,0,0),(1,1,0),(0,1,0)$ have nonvanishing volumes and one orientation (volumes are of the same sign), that is

$$\begin{aligned} J_1(0,0,0) &= J_2(0,0,0) = [\mathbf{p},\mathbf{q},\mathbf{r}]_{000} = 6V_{000}^{pqr} = \alpha_{000} > 0\ (<0), \\ J_1(1,0,0) &= J_1(0,0,1) = -[\mathbf{p},\mathbf{q},\mathbf{v}]_{100} = 6V_{100}^{pqv} = \beta_{100}^{1} > 0\ (<0), \\ J_1(1,1,0) &= J_2(1,1,0) = [\mathbf{p},\mathbf{q},\mathbf{d}]_{110} = 6V_{110}^{pqd} = \bar{\gamma}_{110}^{3} > 0\ (<0), \\ J_2(0,0,1) &= J_2(0,1,0) = -[\mathbf{p},\mathbf{q},\mathbf{u}]_{010} = 6V_{010}^{pqu} = \beta_{010}^{2} > 0\ (<0). \end{aligned} \tag{9.33}$$

The theorem is proved.

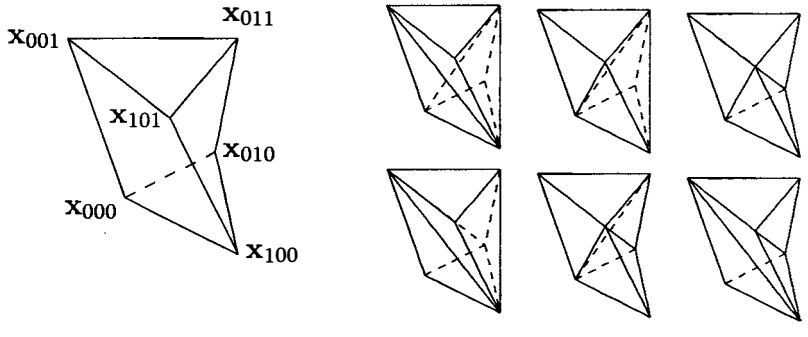
In [9], a similar geometrical interpretation is given for the obtained sufficient conditions.

9.4.2 Prismatic Cell

We consider the mapping

$$\mathbf{x}(\xi) = \mathbf{a}_{000} + \mathbf{a}_{100}\xi_1 + \mathbf{a}_{010}\xi_2 + \mathbf{a}_{001}\xi_3 + \mathbf{a}_{101}\xi_1\xi_3 + \mathbf{a}_{011}\xi_2\xi_3 \tag{9.34}$$

of the unit prism P with the vertices $(0,0,0),(1,0,0),(0,1,0),(0,0,1)(1,0,1),(0,1,1)$ into a prism G with the vertices $\mathbf{x}_{000},\mathbf{x}_{100},\mathbf{x}_{010},\mathbf{x}_{001},\mathbf{x}_{101},\mathbf{x}_{011}$ shown in Fig. 9.4.2 *a*, with two planar triangular faces and three ruled quadranqular faces. Vectors $\mathbf{a}_{000},\mathbf{a}_{100},\mathbf{a}_{010},\mathbf{a}_{001}$, $\mathbf{a}_{101},\mathbf{a}_{011}$ are also defined in relations (9.3).



a *b*

Figure 9.12: Prism and octahedrons.

In [9], the properties of the Jacobian are studied again indirectly but considering linear and nonlinear parts for (9.34) and the composition of mappings of the form (9.28). A linear part is defined according to (9.30). If its Jacobian does not vanish (condition (9.31) is satisfied), then points

$$\tilde{\xi}_{101} = \mathbf{x}_T^{-1}(\mathbf{x}_{101}),\ \tilde{\xi}_{011} = \mathbf{x}_T^{-1}(\mathbf{x}_{011})$$

are defined and the mapping corresponding to a nonlinear part is given in the following way

$$\tilde{\mathbf{x}} = \xi + (\tilde{\xi}_{101} - \xi_{101})\xi_1\xi_3 + (\tilde{\xi}_{011} - \xi_{011})\xi_2\xi_3,\ \xi_{101} = (1,0,1),\ \xi_{011} = (0,1,1). \quad (9.35)$$

Analogously to the case of a pyramid, in [9] the properties of the Jacobian $J = \det\tilde{\mathbf{x}}$ of the mapping (9.35) are examined. In the case of a prism, the Jacobian is a linear function in ξ_1, ξ_2 and a quadratic function in ξ_3. This implies, that in order the Jacobian be positive (negative) in the reference domain, it is necessary and sufficient that it be positive (negative) on the following three edges of the unit prism, that is, the inequalities

$$J(0,0,\xi_3),\ J(1,0,\xi_3),\ J(0,1,\xi_3) > 0 \ \text{ for all } \xi_3 \in [0,1] \quad (9.36)$$

be satisfied. (This property is valid for the Jacobian of the general transformation as well.)

On these edges, the Jacobian is a quadratic trinomial, and, in [9], three lemmas are obtained: sufficient conditions guaranteeing (9.36). These are inequalities for coordinates of vectors $\tilde{\xi}_{101},\ \tilde{\xi}_{011}$.

For $J(0,0,\xi_3)$, conditions have the form

$$\tilde{\xi}^1_{101}\tilde{\xi}^2_{011} > \tilde{\xi}^1_{011}\tilde{\xi}^2_{101}, \quad (9.37)$$

$$\tilde{\xi}^1_{101} + \tilde{\xi}^2_{011} \geq \tilde{\xi}^1_{101}\tilde{\xi}^2_{011} - \tilde{\xi}^1_{011}\tilde{\xi}^2_{101} \quad (9.38)$$

or (9.37) and

$$\tilde{\xi}^1_{101} + \tilde{\xi}^2_{011} - 2 > -2\sqrt{|(\tilde{\xi}^1_{101} - 1)(\tilde{\xi}^2_{011} - 1) - \tilde{\xi}^1_{011}\tilde{\xi}^2_{101})|}; \quad (9.39)$$

for $J(1,0,\xi_3)$, inequalities

$$\tilde{\xi}^3_{101} > 0, \quad (9.40)$$

$$\tilde{\xi}^2_{011}(\tilde{\xi}^1_{101} + \tilde{\xi}^3_{101} - 1) > \tilde{\xi}^2_{101}(\tilde{\xi}^1_{011} + \tilde{\xi}^3_{011} - 1), \quad (9.41)$$

$$\tilde{\xi}^3_{101} \geq (\tilde{\xi}^1_{101} - 1)(\tilde{\xi}^2_{011} - 1) - \tilde{\xi}^1_{011}\tilde{\xi}^2_{101} \quad (9.42)$$

or inequalities (9.40),(9.41) and such conditions:

$$\tilde{\xi}^1_{101} - 1 + \tilde{\xi}^3_{101}(\tilde{\xi}^2_{011} - 1)$$
$$-\tilde{\xi}^2_{101}(\tilde{\xi}^3_{011} - 1) > -2\sqrt{|\tilde{\xi}^3_{101}(\tilde{\xi}^1_{101} - 1)(\tilde{\xi}^2_{011} - 1) - \tilde{\xi}^1_{011}\tilde{\xi}^2_{101})|} \quad (9.43)$$

have to be satisfied, and for $J(0,1,\xi_3)$, conditions

$$\tilde{\xi}^3_{011} > 0, \tag{9.44}$$

$$\tilde{\xi}^1_{101}(\tilde{\xi}^2_{011} + \tilde{\xi}^3_{011} - 1) > \tilde{\xi}^1_{011}(\tilde{\xi}^2_{101} + \tilde{\xi}^3_{101} - 1), \tag{9.45}$$

$$\tilde{\xi}^3_{011} \geq (\tilde{\xi}^1_{101} - 1)(\tilde{\xi}^2_{011} - 1) - \tilde{\xi}^1_{011}\tilde{\xi}^2_{101} \tag{9.46}$$

or conditions (9.44),(9.45) and inequalities

$$\begin{aligned}&\tilde{\xi}^2_{011} - 1 + \tilde{\xi}^3_{011}(\tilde{\xi}^1_{101} - 1)\\ &-\tilde{\xi}^1_{011}(\tilde{\xi}^3_{101} - 1) > -2\sqrt{|\tilde{\xi}^3_{011}(\tilde{\xi}^1_{101} - 1)(\tilde{\xi}^2_{011} - 1) - \tilde{\xi}^1_{011}\tilde{\xi}^2_{101})|}\end{aligned} \tag{9.47}$$

have to be satisfied.

Conditions (9.37)—(9.47) give the following sufficient conditions of the positivity of the Jacobian $J = \det \tilde{\mathbf{x}}$ from [9].

Theorem 11. *The mapping $\tilde{\mathbf{x}}$ is invertible (corresponding prismatic element is nondegenerate), if conditions* (9.37) – (9.47) *are satisfied.*

In [9], other properties of the Jacobian have been examined. Three lemmas have been proved: necessary and sufficient conditions of the Jacobians $J(0,0,\xi_3)$, $J(1,0,\xi_3)$, and $J(0,1,\xi_3)$ vanishing for the general transformation (9.34) on the edges $\xi_1 = 0, \xi_2 = 0$, $\xi_1 = 1, \xi_2 = 0$, and $\xi_1 = 0, \xi_2 = 1$ in some $\xi_3 \in [0,1]$.

For comparison with (9.37)—(9.47), we obtain necessary and sufficient conditions of the invertibility of the mappings from [9] without separating linear and nonlinear parts and using the notations and technique from [7].

We have

$$\begin{aligned}\frac{\partial \mathbf{x}}{\partial \xi_1} &= \mathbf{a}_{100} + \mathbf{a}_{101}\xi_3 = \mathbf{p}_{000} + (\mathbf{p}_{001} - \mathbf{p}_{000})\xi_3 = \mathbf{p}_{000}(1 - \xi_3) + \mathbf{p}_{001}\xi_3,\\ \frac{\partial \mathbf{x}}{\partial \xi_2} &= \mathbf{a}_{010} + \mathbf{a}_{011}\xi_3 = \mathbf{q}_{000} + (\mathbf{q}_{001} - \mathbf{q}_{000})\xi_3 = \mathbf{q}_{000}(1 - \xi_3) + \mathbf{q}_{001}\xi_3,\\ \frac{\partial \mathbf{x}}{\partial \xi_3} &= \mathbf{a}_{001} + \mathbf{a}_{101}\xi_1 + \mathbf{a}_{011}\xi_2 = \mathbf{r}_{000} + (\mathbf{r}_{100} - \mathbf{r}_{000})\xi_1 + (\mathbf{r}_{010} - \mathbf{r}_{000})\xi_2\\ &= \mathbf{r}_{000}(1 - \xi_1 - \xi_2) + \mathbf{r}_{100}\xi_1 + \mathbf{r}_{010}\xi_2.\end{aligned}$$

For this mapping, the following theorem is valid.

Theorem 12. *The Jacobian of the mapping* (9.34) *is positive if and only if the conditions*

$$\begin{aligned}&\alpha_{000}, \alpha_{001} > 0, \quad \beta^3_{000} + \beta^3_{001} > -2\sqrt{\alpha_{000}\alpha_{001}},\\ &\beta^1_{000}, \beta^1_{0\,01} > 0, \quad \gamma^2_{100} + \gamma^2_{101} > -2\sqrt{\beta^1_{000}\beta^1_{001}},\\ &\beta^2_{000}, \beta^2_{001} > 0, \quad \gamma^1_{010} + \gamma^1_{011} > -2\sqrt{\beta^2_{000}\beta^2_{001}}\end{aligned} \tag{9.48}$$

are satisfied.

Proof. The Jacobians on faces are quadratic trinomials

$$\begin{aligned} J(0,0,\xi_3) &= [\mathbf{p}_{000}(1-\xi_3)+\mathbf{p}_{001}\xi_3, \mathbf{q}_{000}(1-\xi_3)+\mathbf{q}_{001}\xi_3, \mathbf{r}_{000}] \\ &= \alpha_{000}(1-\xi_3)^2+\alpha_{001}\xi_3^2+(\beta^3_{000}+\beta^3_{001})(1-\xi_3)\xi_3, \\ J(1,0,\xi_3) &= [\mathbf{p}_{000}(1-\xi_3)+\mathbf{p}_{001}\xi_3, \mathbf{q}_{000}(1-\xi_3)+\mathbf{q}_{001}\xi_3, \mathbf{r}_{100}] \\ &= \beta^1_{000}(1-\xi_3)^2+\beta^1_{001}\xi_3^2+(\gamma^2_{100}+\gamma^2_{101})(1-\xi_3)\xi_3, \\ J(0,1,\xi_3) &= [\mathbf{p}_{000}(1-\xi_3)+\mathbf{p}_{001}\xi_3, \mathbf{q}_{000}(1-\xi_3)+\mathbf{q}_{001}\xi_3, \mathbf{r}_{010}] \\ &= \beta^2_{000}(1-\xi_3)^2+\beta^2_{001}\xi_3^2+(\gamma^1_{010}+\gamma^1_{011})(1-\xi_3)\xi_3, \end{aligned} \tag{9.49}$$

similar to (9.20). It is easy to find for them conditions of positivity. Conditions of positivity will be analogous to (9.24). The theorem is proved.

In contrast to (9.38)—(9.46), conditions (9.48) have an obvious geometrical interpretation. This is the conditions for the volumes of special tetrahedrons with the vertices of a prism: the tetrahedrons constructed on edges of the prism for each vertex have to be positive and tetrahedrons constructed on two edges and one diagonal of a face can be negative and satisfy the quadratic inequalities.

By means of obtained in [9] conditions, the nondegeneracy of different finite elements are tested. Investigations [9] are continued in [10]. The trilinear mapping (9.2),(9.3) used for generation of hexahedral finite elements is investigated in [10]. In contrast to [9], due to the more complicated construction (more vertices of an element, higher degree of the Jacobian) authors state that they are not able to obtain algebraic relations that ensure the positivity of the Jacobian and elaborate the numerical algorithm for testing the invertibility of mappings. In [10], it is shown that the Jacobian is the polynomial of the fourth degree, in each of the variable the Jacobian is a quadratic trinomial. The last property is used in the numerical algorithm, in this connection authors follow the way of further simplification and try to get rid of quadratic nonlinearity of the Jacobian approximating it by linear functions along one coordinate direction on a finite mesh of nodes. The results of testing specially constructed cells are given. In conclusion, the questions remained open are listed:

Is it possible to obtain algebraic inequalities that ensure the positivity of the Jacobian of a trilinear mapping?

Is the local invertibility (positivity of the Jacobian) sufficient for its global invertibility?

Works [2, 4, 5] and [6] give the answers to posed questions.

9.4.3 Cells Defined by Bershtein-Bezier Polynomials

In [11] sufficient conditions are suggested for the invertibility of the polynomial functions that are, in particular, the generalization of the trilinear mapping and that are defined on a cube or simplex. Such constructions are used for generation of finite elements, and sufficient conditions are applicable to finite element analysis. In [11], the reference domain is a unit square $S^2 = \{(\xi_1,\xi_2) : 0 \le \xi_1,\xi_2 \le 1\}$ or triangle $T^2 = \{(\xi_1,\xi_2) : 0 \le \xi_1,\xi_2,\xi_1+\xi_2 \le 1\}$ in $\mathbb{R}^2$ or a unit cube $S^3 = \{(\xi_1,\xi_2,\xi_3) : 0 \le \xi_1,\xi_2,\xi_3 \le 1\}$ or

tetrahedron $T^3 = \{(\xi_1, \xi_2, \xi_3) : 0 \le \xi_1, \xi_2, \xi_3, \xi_1 + \xi_2 + \xi_3 \le 1\}$ in $\mathbb{R}^3$. The mapping $\mathbf{x} = \mathbf{x}(\xi)$ is considered as the polynomial function in the Bershtein-Bezier form defined on P, where P is either S^n or T^n, $n = 2, 3$. This form is a popular way to write polynomials in computer-aided geometric design [20]. A univariate polynomial of degree p in Bershtein-Bezier form is written

$$x(\xi) = \sum_{i=0}^{p} x_i (1-\xi)^{p-i} \xi^i \frac{p!}{i!(p-i)!}$$

and has natural parametric domain $\xi \in [0, 1]$.

A bivariate polynomial with maximum degree p individually in ξ_1, ξ_2 is written in the form

$$\mathbf{x}(\xi_1, \xi_2) = \sum_{i_1=0}^{p} \sum_{i_2=0}^{p} \mathbf{x}_{i_1 i_2} (1-\xi_1)^{p-i_1} {\xi_1}^{i_1} (1-\xi_2)^{p-i_2} {\xi_2}^{i_2} \frac{p!p!}{i_1!(p-i_1)!i_2!(p-i_2)!}$$

and has a natural parametric domain $(\xi_1, \xi_2) \in S^2$. A trivariate polynomial with the degree p individually has the form

$$\mathbf{x}(\xi_1, \xi_2, \xi_3) = \sum_{i_1=0}^{p} \sum_{i_2=0}^{p} \sum_{i_3=0}^{p} \mathbf{x}_{i_1 i_2 i_3} (1-\xi_1)^{p-i_1} {\xi_1}^{i_1} (1-\xi_2)^{p-i_2} {\xi_2}^{i_2} \times (1-\xi_3)^{p-i_3} {\xi_3}^{i_3} \frac{p!p!p!}{i_1!(p-i_1)!i_2!(p-i_2)!i_3!(p-i_3)!} \tag{9.50}$$

and is defined for $(\xi_1, \xi_2, \xi_3) \in S^3$. A trilinear mapping considered above is a special case of this polynomial for $p = 1$.

Analogously, the Bershtein-Bezier forms are defined on T^n, $n = 2, 3$. The bivariate polynomial of the total degree at most p is defined on T^2 in the form

$$\mathbf{x}(\xi_1, \xi_2) = \sum_{i_1=0}^{p} \sum_{i_2=0}^{p-i_1} \mathbf{x}_{i_1 i_2} {\xi_1}^{i_1} {\xi_2}^{i_2} (1-\xi_1-\xi_2)^{p-i_1-i_2} \frac{p!}{i_1!i_2!(p-i_1-i_2)!}.$$

The trivariate polynomial of the total degree at most p are defined on T^3 in the form

$$\mathbf{x}(\xi_1, \xi_2, \xi_3) = \sum_{i_1=0}^{p} \sum_{i_2=0}^{p-i_1} \sum_{i_3=0}^{p-i_1-i_2} \mathbf{x}_{i_1 i_2 i_3} {\xi_1}^{i_1} {\xi_2}^{i_2} {\xi_3}^{i_3} (1-\xi_1-\xi_2-\xi_3)^{p-i_1-i_2-i_3} \times \frac{p!}{i_1!i_2!i_3!(p-i_1-i_2-i_3)!}.$$

In all five cases, the vectors x_i, $\mathbf{x}_{i_1 i_2}, \mathbf{x}_{i_1 i_2 i_3}$ are called the control points. The proof of sufficient conditions from [11] is based on the properties of the Bershtein-Bezier forms used in the proof of their analogs in [7] and formulated in the following fundamental theorem from [20].

Theorem 13. *Let P be the natural parametric domain of Bershtein-Bezier form in five cases listed above. Then $\mathbf{x}(P)$ is contained in the convex hull of the control points.*

Really (see also [11]), all the coefficients, for example, ξ_1, $(1-\xi_1-\xi_2)$ and so on, inside the parametric domain, are positive (vanishing only on the boundary of the domains). More over, all sums, if the control points are excluded, are equal to 1.

Another property of the Bershein-Bezier form also used in the proof of sufficient conditions is the following property: the derivatives of the Bershtein-Bezier form can be written in Bershtein-Bezier form with control points that are finite differences of the control points of the considered polynomial in the direction of the variable being differentiated. For example, for the generalization of a trilinear mapping with the domain S^3 we have

$$\begin{aligned}
\frac{\partial \mathbf{x}}{\partial \xi_1} &= \sum_{i_1=0}^{p-1}\sum_{i_2=0}^{p}\sum_{i_3=0}^{p} p(\mathbf{x}_{i_1+1,i_2,i_3}-\mathbf{x}_{i_1,i_2,i_3})(1-\xi_1)^{p-i_1-1}{\xi_1}^{i_1} \\
&\times (1-\xi_2)^{p-i_2}{\xi_2}^{i_2}(1-\xi_3)^{p-i_3}{\xi_3}^{i_3}\frac{(p-1)!p!p!}{i_1!(p-i_1-1)!i_2!(p-i_2)!i_3!(p-i_3)!}, \\
\frac{\partial \mathbf{x}}{\partial \xi_2} &= \sum_{i_1=0}^{p}\sum_{i_2=0}^{p-1}\sum_{i_3=0}^{p} p(\mathbf{x}_{i_1,i_2+1,i_3}-\mathbf{x}_{i_1,i_2,i_3})(1-\xi_1)^{p-i_1}{\xi_1}^{i_1} \\
&\times (1-\xi_2)^{p-i_2-1}{\xi_2}^{i_2}(1-\xi_3)^{p-i_3}{\xi_3}^{i_3}\frac{(p-1)!p!p!}{i_1!(p-i_1)!i_2!(p-i_2-1)!i_3!(p-i_3)!}, \\
\frac{\partial \mathbf{x}}{\partial \xi_3} &= \sum_{i_1=0}^{p}\sum_{i_2=0}^{p}\sum_{i_3=0}^{p-1} p(\mathbf{x}_{i_1,i_2,i_3+1}-\mathbf{x}_{i_1,i_2,i_3})(1-\xi_1)^{p-i_1}{\xi_1}^{i_1} \\
&\times (1-\xi_2)^{p-i_2}{\xi_2}^{i_2}(1-\xi_3)^{p-i_3-1}{\xi_3}^{i_3}\frac{(p-1)!p!p!}{i_1!(p-i_1)!i_2!(p-i_2)!i_3!(p-i_3-1)!}.
\end{aligned} \tag{9.51}$$

(The analogous form for a trilinear mapping is obtained in (9.5)). In the case of (9.51) vectors $p(\mathbf{x}_{i_1+1,i_2,i_3}-\mathbf{x}_{i_1,i_2,i_3})$ are control points for $\partial\mathbf{x}/\partial\xi_1$, $p(\mathbf{x}_{i_1,i_2+1,i_3}-\mathbf{x}_{i_1,i_2,i_3})$ are control points for $\partial\mathbf{x}/\partial\xi_2$ and $p(\mathbf{x}_{i_1,i_2,i_3+1}-\mathbf{x}_{i_1,i_2,i_3})$ are control points for $\partial\mathbf{x}/\partial\xi_3$. Let us denote these lists of control points by $G_{\xi_1}, G_{\xi_2}, G_{\xi_3}$, respectively. In [11] the following theorem is proved.

Theorem 14. *The Jacobian matrix for the Bershtein-Bezier form is invertible on the entire reference element under the condition: in the case $n=2$, the matrix $[\mathbf{p},\mathbf{q}]$ is invertible for any vector $\mathbf{p}$ from the convex hull of G_{ξ_1} and for any vector from the convex hull of G_{ξ_2}. Analogously, in the case $n=3$ under the condition: the matrix $[\mathbf{p},\mathbf{q},\mathbf{r}]$ is invertible for any $\mathbf{p}$ from the convex hull of G_{ξ_1}, and for any $\mathbf{q}$ from the convex hull of G_{ξ_2} and for any $\mathbf{r}$ from the convex hull of G_{ξ_3}.*

Note, that for the case of a trilinear mapping of the cube S^3, obtained sufficient conditions are more restrictive than sufficient conditions 1. Really, the check of the invertibility of matrix is carried out for the convex hulls of sets, it means that for the vectors of sets. All triple scalar products of vectors from $G_{\xi_1}, G_{\xi_2}, G_{\xi_3}$ give 64 coefficients (9.8) of the polynomial in the form (9.10). Conditions of positivity of these 64 coefficients are more restrictive conditions than sufficient conditions 1 in which we demand the positivity of 27 sums composed of above 64 coefficients corresponding to similar terms. Numerical experiment with random numbers showed that these conditions involve only 3–4 % of cells with positive Jacobian.

It should be noted also that we do not need to check conditions of the theorem 14 for all vectors of convex hulls of sets $G_{\xi_1}, G_{\xi_2}, G_{\xi_3}$. It is sufficient to check the invertibility of matrix only for the vectors of sets $G_{\xi_1}, G_{\xi_2}, G_{\xi_3}$: matrices will be invertible for all vectors from convex hulls. It follows from the properties of a convex hull of a set and rules of calculation of a determinant. The following theorem is obvious.

Theorem 15. *The Jacobian of considered mappings defined by the polynomials in above Bershtein-Bezier forms is positive (negative) on the reference elements, if the determinants of matrices* $[\mathbf{p}, \mathbf{q}]$ *are positive (negative) for all vectors* $\mathbf{p} \in G_{\xi_1}$, $\mathbf{q} \in G_{\xi_2}$ *in the case* $n = 2$, *and if the determinants of matrices* $[\mathbf{p}, \mathbf{q}, \mathbf{r}]$ *(triple scalar products) are positive (negative) for all vectors* $\mathbf{p} \in G_{\xi_1}$, $\mathbf{q} \in G_{\xi_2}$, $\mathbf{r} \in G_{\xi_3}$ *in the case* $n = 3$.

Let us prove this theorem without using conditions of the theorem 14.

Proof. We shall use the properties of the Bershtein-Bezier form for the derivatives. Substitute the derivatives (9.51) into (9.4) and calculate the determinant of Jacobian matrix decomposing it into the sum of the determinants of matrices composed of vectors of control points for the derivatives with factors in the form of polynomials. Taking the common factors out of the signs of determinants we obtain that the Jacobian of each of the mappings will be the sum of the polynomials positive in the natural parametric domain with the coefficients proportional to the determinants of matrices composed of all vectors from $G_{\xi_1}, G_{\xi_2}, G_{\xi_3}$ (in a three-dimensional case, proportional to the triple scalar products). For example, for the considered case of a generalization of a trilinear mapping, the formula of the Jacobian (9.4) after substitution in it (9.51) and decomposition of the triple scalar products has the form

$$J(\xi_1, \xi_2, \xi_3) = \sum_{i_1=0}^{p-1} \sum_{i_2=0}^{p} \sum_{i_3=0}^{p} \sum_{j_1=0}^{p} \sum_{j_2=0}^{p-1} \sum_{j_3=0}^{p} \sum_{k_1=0}^{p} \sum_{k_2=0}^{p} \sum_{k_3=0}^{p-1} \alpha_{i_1 i_2 i_3 j_1 j_2 j_3 k_1 k_2 k_3} \times \Xi_{i_1 i_2 i_3 j_1 j_2 j_3 k_1 k_2 k_3} P_{i_1 i_2 i_3 j_1 j_2 j_3 k_1 k_2 k_3}, \quad (9.52)$$

where

$$
\begin{aligned}
\alpha_{i_1 i_2 i_3 j_1 j_2 j_3 k_1 k_2 k_3} &= [\mathbf{p}, \mathbf{q}, \mathbf{r}], \\
\mathbf{p} &= p(\mathbf{x}_{i_1+1,i_2,i_3} - \mathbf{x}_{i_1,i_2,i_3}), \\
\mathbf{q} &= p(\mathbf{x}_{j_1,j_2+1,j_3} - \mathbf{x}_{j_1,j_2,j_3}), \\
\mathbf{r} &= p(\mathbf{x}_{k_1,k_2,k_3+1} - \mathbf{x}_{k_1,k_2,k_3})], \\
\Xi_{i_1 i_2 i_3 j_1 j_2 j_3 k_1 k_2 k_3} &= (1-\xi_1)^{p-i_1-1}{\xi_1}^{i_1}(1-\xi_2)^{p-i_2}{\xi_2}^{i_2}(1-\xi_3)^{p-i_3}{\xi_3}^{i_3} \\
&\times (1-\xi_1)^{p-j_1}{\xi_1}^{j_1}(1-\xi_2)^{p-j_2-1}{\xi_2}^{j_2}(1-\xi_3)^{p-j_3}{\xi_3}^{j_3} \\
&\times (1-\xi_1)^{p-k_1}{\xi_1}^{k_1}(1-\xi_2)^{p-k_2}{\xi_2}^{k_2}(1-\xi_3)^{p-k_3-1}{\xi_3}^{k_3}, \\
P_{i_1 i_2 i_3 j_1 j_2 j_3 k_1 k_2 k_3} &= \frac{(p-1)!p!p!}{i_1!(p-i_1-1)!i_2!(p-i_2)!i_3!(p-i_3)!} \\
&\times \frac{(p-1)!p!p!}{j_1!(p-j_1)!j_2!(p-j_2-1)!j_3!(p-j_3)!} \\
&\times \frac{(p-1)!p!p!}{k_1!(p-k_1)!k_2!(p-k_2)!k_3!(p-k_3-1)!}.
\end{aligned}
$$

If coefficients $\alpha_{i_1 i_2 i_3 j_1 j_2 j_3 k_1 k_2 k_3}$ (determinants of matrices) are positive, the Jacobian will be positive in the parametric domain since the polynomials $\Xi_{i_1 i_2 i_3 j_1 j_2 j_3 k_1 k_2 k_3}$ are positive. It is easy to show that the Jacobian will be positive on the boundary of a domain. So, the theorem is proved.

The conditions of theorem 15 are simpler computationally than conditions of theorem 14.

The conditions of theorem 15 are analogous to corollary 1. If we summarize similar terms containing $\Xi_{i_1 i_2 i_3 j_1 j_2 j_3 k_1 k_2 k_3}$ in (9.29), and then demand the positivity of their coefficients, we shall obtain more general conditions of the Jacobian's positivity for the generalization of a trilinear mapping analogous to sufficient conditions 1 of the nondegeneracy of hexahedral ruled cells.

The following theorem is valid.

Theorem 16. *In order that the Jacobian of the mapping (9.50) be positive in the whole cube P including its boundary it is sufficient that the following conditions be satisfied*

$$
\begin{aligned}
\Lambda_{I_1 I_2 I_3} &= \sum_{\substack{j_1=0 \\ j_1+k_1 \le I_1}}^{p} \sum_{k_1=0}^{p} \sum_{\substack{i_2=0 \\ i_2+k_2 \le I_2}}^{p} \sum_{k_2=0}^{p} \sum_{\substack{i_3=0 \\ i_3+k_3 \le I_3}}^{p} \sum_{j_3=0}^{p} \alpha_{I_1-j_1-k_1 j_1 k_1 i_2 I_2-i_2-k_2 k_2 i_3 j_3 I_3-i_3-j_3} \\
&\times P_{I_1-j_1-k_1 j_1 k_1 i_2 I_2-i_2-k_2 k_2 i_3 j_3 I_3-i_3-j_3} > 0, \quad I_l = 0, 1, ..., 3p-1, \quad l = 1, 2, 3.
\end{aligned} \tag{9.53}
$$

Proof. Among $p^3(p+1)^6$ polynomials $\Xi_{i_1 i_2 i_3 j_1 j_2 j_3 k_1 k_2 k_3}$ in (9.52), only $27p^3$ polyno-

mials are different. Really, writing out the polynomials in the form

$$\begin{aligned}\Xi_{i_1 i_2 i_3 j_1 j_2 j_3 k_1 k_2 k_3} &= (1-\xi_1)^{3p-i_1-j_1-k_1-1}{\xi_1}^{i_1+j_1+k_1} \\ &\times (1-\xi_2)^{3p-i_2-j_2-k_2-1}{\xi_2}^{i_2+j_2+k_2} \\ &\times (1-\xi_3)^{3p-i_3-j_3-k_3-1}{\xi_3}^{i_3+j_3+k_3}\end{aligned}$$

and introducing the new indices $I_1 = i_1 + j_1 + k_1$, $I_2 = i_2 + j_2 + k_2$, $I_3 = i_3 + j_3 + k_3$, we can rewrite the polynomials as

$$\begin{aligned}\Xi_{i_1 i_2 i_3 j_1 j_2 j_3 k_1 k_2 k_3} &= \tilde{\Xi}_{I_1 I_2 I_3} \\ &= (1-\xi_1)^{3p-I_1-1}\xi_1^{I_1}(1-\xi_2)^{3p-I_2-1}\xi_2^{I_2}(1-\xi_3)^{3p-I_3-1}\xi_3^{I_3}\end{aligned}$$

and obtain (9.52) in the form

$$J(\xi_1,\xi_2,\xi_3) = \sum_{I_1=0}^{3p-1}\sum_{I_2=0}^{3p-1}\sum_{I_3=0}^{3p-1}\Lambda_{I_1 I_2 I_3}\tilde{\Xi}_{I_1 I_2 I_3},$$

where $\Lambda_{I_1 I_2 I_3}$ are from (9.53). It is easy to see that if conditions (9.53) are satisfied, the Jacobian will be positive in the cube P.

Necessary conditions of the Jacobian's positivity for the mapping (9.50), analogous to the necessary conditions 1 of the trilinear mapping, can be found by computing the Jacobian of (9.50) at the vertices of a cell and at the midpoints on edges, faces and in the interior of the cube P and by demanding the positivity of these expressions.

With the purpose of substitution of theorem 14 by more constructive ones, in [11], the following computational characterization is suggested.

Theorem 17. *In the case of* $\mathbb{R}^2$, *conditions 1 are equivalent to the following conditions: There exists a vector* $\mathbf{h} \in \mathbb{R}^2$ *such that for all* $\mathbf{f} \in G_{\xi_1} \bigcup G_{\xi_2}$, $\mathbf{h}^T\mathbf{f} > 0$ *and there exists a vector* $\mathbf{h}_1 \in \mathbb{R}^2$ *such that for all* $\mathbf{f} \in G_{\xi_1}$, ${\mathbf{h}_1}^T\mathbf{f} > 0$ *and for all* $\mathbf{f} \in G_{\xi_2}$, ${\mathbf{h}_1}^T\mathbf{f} < 0$.

Analogous interpretation is given for $\mathbb{R}^3$. Each of the obtained computational conditions is suggested to check by linear programming method [21] (which is linear time in three dimensions) or, for checking conditions of the theorem 14, to use convex hull methods [22] (estimated in time as $O(n \log n)$, where n is a parameter characterizing the number of elements of sets $G_{\xi_1}, G_{\xi_2}, G_{\xi_3}$.)

In [11], the theorem about global invertibility of considered mapping is proved.

9.5 Formulas of Volumes of Ruled Cells

9.5.1 Hexahedral Cell

Formulas of a volume of a ruled hexahedral cell published in [12, 23] are rather complex and demand a large amount of computations. Efficient volume computation and another formula of a cell volume were suggested in [24]. The formula obtained in [7] and given in this work is similar to [24] but requires computation of volumes of ten tetrahedrons.

Theorem 18. *The volume of the ruled hexahedral cell (9.1) has the form*

$$V = \int_{0\le\xi_l\le 1} J d\xi_1 d\xi_2 d\xi_3 = \frac{1}{12}\left(\sum_{i_1,i_2,i_3=0}^{1} \alpha_{i_1 i_2 i_3} + \bar{\kappa}_{000} + \bar{\kappa}_{111}\right).$$

It is possible to see that the volume of the ruled hexahedral cell is equal to one-half of a sum of volumes of two dodecahedrons with planar faces, with the same corners $\mathbf{x}_{i_1 i_2 i_3}$, edges $\mathbf{p}_{i_1 i_2 i_3}$, $\mathbf{q}_{i_1 i_2 i_3}$, $\mathbf{r}_{i_1 i_2 i_3}$, and with different $\mathbf{u}$, $\mathbf{v}$, $\mathbf{w}$ (or faces) shown in Figure 9.1 *b*. In chapter 12, it was proved that the volume of the ruled hexahedral cell is equal to one-half of a sum of volumes of any two dodecahedrons with the same corners but different in the way of constructing triangular planar faces by means of diagonal vectors $\mathbf{u}$, $\mathbf{v}$, $\mathbf{w}$.

Theorem 19. *The volume of the ruled hexahedral cell (9.1) has the form*

$$V = \int_{0\le\xi^l\le 1} J d\xi^1 d\xi^2 d\xi^3 = \frac{1}{24}\sum_{i_1,i_2,i_3=0}^{1}\sum_{k=1}^{3} \beta^k_{i_1 i_2 i_3}.$$

In this formula, the volume of a hexahedral cell is expressed in terms of the volumes of twenty four hexahedrons corresponding to coefficients $\beta^k_{i_1 i_2 i_3}$ from (9.8).

The proofs of theorem 18 and 19 are given in [7].

Analogous formulas can be obtained for ruled pyramids and prisms with a triangular base.

9.5.2 Pyramidal Cell

Theorem 20. *The volume of a ruled pyramid (9.29) has the form*

$$V = \int_P J d\xi^1 d\xi^2 d\xi^3 = \frac{1}{12}\left(\alpha_{000} + \beta^1_{100} + \beta^2_{010} + \gamma^3_{110}\right), \tag{9.54}$$

where P is a unit pyramid with the vertices $(0,0,0), (1,0,0), (0,1,0), (1,1,0)$.

Proof. We have

$$V = \int_P J d\xi^1 d\xi^2 d\xi^3 = \int_{T_1} J_1 d\xi^1 d\xi^2 d\xi^3 + \int_{T_2} J_2 d\xi^1 d\xi^2 d\xi^3,$$

where J_1 and J_2 are defined in (9.32). Formulas (9.32) can be written in the form

$$J_1 = A_1 + B_1(\xi_1 + \xi_3) + C_1\xi_3, \quad J_2 = A_2 + B_2(\xi_2 + \xi_3) + C_2\xi_1, \tag{9.55}$$

where

$$\begin{aligned} A_1 &= A_2 = J_1(0,0,0), \quad B_1 = J_1(1,0,0) - J_1(0,0,0), \quad B_2 = J_2(0,1,0) - J_1(0,0,0), \\ C_1 &= J_1(1,1,0) - J_1(1,0,0), \quad C_2 = J_1(1,1,0) - J_2(0,1,0). \end{aligned}$$

Integrating J_1 from (9.55) over T_1 and J_2 from (9.55) over T_2 we have

$$\begin{aligned}\int_0^1\left(\int_0^{\xi_1}\left(\int_0^{1-\xi_1} J_1 d\xi_3\right)d\xi_2\right)d\xi_1 &= \frac{A_1}{6}+\frac{C_1}{24}+\frac{B_1}{8}\\ &= \frac{J_1(0,0,0)}{24}+\frac{J_1(1,1,0)}{24}+\frac{J_1(1,0,0)}{12},\\ \int_0^1\left(\int_0^{\xi_2}\left(\int_0^{1-\xi_2} J_2 d\xi_3\right)d\xi_1\right)d\xi_2 &= \frac{A_1}{6}+\frac{C_2}{24}+\frac{B_2}{8}\\ &= \frac{J_1(0,0,0)}{24}+\frac{J_1(1,1,0)}{24}+\frac{J_2(0,1,0)}{12}.\end{aligned}$$

Using (9.33) we get (9.54).

In this case, the volume of a pyramid is equal to one-half of volumes of four tetrahedrons constructed on the edges with the vertices $x_{000}, x_{100}, x_{110}, x_{010}$ or to one-half of two hexahedrons with the same vertices but with different triangular faces (see Fig. 9.11).

9.5.3 Prismatic Cell

Now we shall give the formula of a volume for a prism with a triangular base and three ruled faces that requires the computation of twelve tetrahedrons

Theorem 21. *The volume of a ruled prism (9.34) has the form*

$$\begin{aligned}V &= \int_P Jd\xi_1 d\xi_2 d\xi_3 = \frac{1}{12}[\frac{2}{3}(\alpha_{000}+\alpha_{001}+\beta^1_{000}+\beta^1_{001}+\beta^2_{000}+\beta^2_{001})\\ &+ \frac{1}{3}(\beta^3_{000}+\beta^3_{001}+\gamma^1_{010}+\gamma^1_{011}+\gamma^2_{100}+\gamma^2_{101})],\end{aligned} \tag{9.56}$$

where P is a unit prism with the vertices $(0,0,0),(1,0,0),(0,1,0),(0,0,1)(1,0,1),$ $(0,1,1)$.

The volume of such a prism is equal to one sixth of the volume of six octahedrones with the same vertices but with different triangular faces shown in Fig. 9.4.2. It is easy to see from Fig. 9.4.2 and formulas (9.10), since a volume of each of octahedrons is equal to a sum of two volumes of tetrahedrons constructed on three edges with common vertex (corresponding to summands with coefficient $\frac{2}{3}$) and a volume of tetrahedron constructed on two edges and one "diagonal" of a face (corresponding to summands with coefficient $\frac{1}{3}$).

Proof. Really,

$$V=\int_P Jd\xi^1 d\xi^2 d\xi^3=\int_0^1\left(\int_0^1\left(\int_0^{1-\xi_1} Jd\xi_2\right)d\xi_1\right)d\xi_3,$$

where

$$J=A+(B-A)\xi_1+(C-A)\xi_2,$$

and $A = J(0,0,\xi_3)$, $B = J(1,0,\xi_3)$, $C = J(0,1,\xi_3)$ from (9.49). Since

$$\int_0^1 \left(\int_0^{1-\xi_1} J d\xi_2 \right) d\xi_1 = \int_0^1 (A(1-\xi_1) + (B-A)\xi_1(1-\xi_1) + \frac{1}{2}(C-A)(1-\xi_1)^2 \Big) d\xi_1 = \frac{1}{6}(A+B+C), \tag{9.57}$$

and each of A, B, C is a quadratic trinomial of the form [7]

$$-a\xi_3^2 + (a + \alpha_1 - \alpha_0) + \alpha_0,$$

where $a = \beta_1 + \beta_0 - \alpha_1 - \alpha_0$, $\alpha_i = \text{const}$, $\beta_i = \text{const}$, $i = 0, 1$ we have

$$\int_0^1 A d\xi_3 = \frac{\beta_1 + \beta_0}{6} + \frac{\alpha_1 + \alpha_0}{3}.$$

Substituting values of α_0, α_1, β_0, β_1 for each of A, B, C from (9.49) into (9.57), we get (9.56).

In all three considered cases, the volume of ruled cells having in general case non-planar faces and complicated form is expressed in terms of the volumes of the polyhedrons with the same vertices and with planar triangular faces. This gives an opportunity of the essential simplification of numerical algorithms (see chapter 12).

Analogous formulas can be obtained for the volume of cells constructed by means of functions defined by Bershtein-Bezier forms since the Jacobians for them will be also the polynomials with the coefficients proportional to the volumes of tetrahedrons (see, for example, (9.52)).

9.6 Conclusion

Obtained theorems, conditions of nondegeneracy and formulas of volumes of cells can be applied in the numerical algorithms of grid generation and in numerical analysis using different types of curved meshes.

The following directions of further investigation of considered problem are viewed.

It is expedient for testing the nondegeneracy of the hexahedral cells to compare the reliability of the special numerical algorithms described in this paper and in [10].

Above investigations would promote to liquidate the gap between necessary and sufficient conditions of nondegeneracy of grids and to find reliable criteria. These investigations are interesting from the scientific point of view and undoubtedly essential for the practice of grid generation.

However, to have nondegeneracy conditions being both necessary and sufficient or criteria is not always obligatory because of the following reasons. The construction of a grid is an initial and important stage of the solution of a physical problem, this stage is followed by

the approximation of differential equations or another relations modelling the physical process on the constructed grid. This process does not allow exotic form of cells even they are nondegenerate. (Because of this reason a number of main requirements to a grid mentioned in the introduction are considered.) By virtue of these circumstances, easy testable sufficient conditions are also important for the practice of computations, so further development of investigations is useful to concentrate on finding the simple sufficient conditions that are handly in numerical realization of the algorithms of grid generation of three-dimensional grids. Some experience of this can be found in [16] and in chapter 10.

Acknowledgment

This work was supported by the Russian Foundation for Basic Research, project 02–01–00236.

Bibliography

[1] P. M. Knupp and S. Steinberg, *Fundamentals of Grid Generation*, CRC Press, Boca Raton, FL, 1994.

[2] S. A. Ivanenko, Harmonic mappings, *Handbook of Grid Generation*, J. F. Thompson, B. K. Soni, and N. P. Weatherill, eds., CRC Press, Boca Raton, FL, 1999, pp. 8-1–8-43.

[3] S. A. Ivanenko, *Adaptive-harmonic grids*, Moscow: Computing Center of Russian Academy of Science, 1997.

[4] N. A. Bobylev, S. A. Ivanenko, I. G. Ismailov, A few notes about homeomorphic mapping, *Mat. Zametki*, **60**, 4 (1996), pp. 593–596.

[5] N. A. Bobylev, S. A. Ivanenko, A. V. Kazunin, On the piecewise homeomorphic mappings of the bounded domains and their applications to the theory of grids, Grid Generation: Theory and Applications, *Proceedings of the workshop organized by the Computing Center RAS and R & D Company Tesis*, June 24-28, 2002, Dorodnicyn Computing Center, Moscow, Russia, edited by S.A. Ivanenko, V.A.Garanzha, Moscow, Computing Center RAS, 2002, pp. 26–42. (http://www.ccas.ru/gridgen/ggta02/papers/Bobylev.pdf)

[6] O. V. Ushakova, Conditions of nondegeneracy of three-dimensional cells: A formula of a volume of cells, in *Numerical Grid Generation in Computational Field Simulations*, B. K. Soni, J. Haeuser, J. F. Thompson, and P. Eiseman, eds. International Society of Grid Generation (ISGG), Mississippi State, MS, 2000, pp. 659–668.

[7] O.V.Ushakova, Conditions of nondegeneracy of three- dimensional cells, A formula of a volume of cells, *SIAM J. Sci. Comp*, **23**, 4, 2001, pp. 1273–1289.

[8] O.V.Ushakova, Nondegeneracy criteria for 3-D grid cells. Formulas for a cell volume, Grid Generation: New trends and applications in real-world simulations. *Proceedings of the minisymposium in the International conference "Optimization of finite-element approximations, splines and wavelets"*, June 25-29, 2001, St.Petersburg, Russia, Edited by S.A.Ivanenko, V.A.Garanzha, 115–128.(http://www.ccas.ru/gridgen/Ushakova.pdf)

[9] P.Knabner, G.Summ, The invertibility of the isoparametric mapping for pyramidal and prismatic finite elements, *Numerical mathematics*, **88**, 2001, pp. 661-681.

[10] P.Knabner, S.Korotov, G.Summ, Conditions for the invertibility of the isoparametric mapping for hexahedral finite elements, *Finite elements in analysis and design*, **2**, 2003, pp. 159-172.

[11] S.A. Vavasis, A Bernstein–Bezier Sufficient Condition for Invertibility of Polynomial Mapping Functions, November 3, 2001. (http://www.cs.cornell.edu/home/vavasis)

[12] J. Killeen, *Controlled Fusion*, Academic Press, New York, San Francisco, London, 1976.

[13] G. Strang and G. Fix, An Analysis of the Finite Element Method, New York: Prentice–Hall, New York, 1973.

[14] S. A. Ivanenko and A. A Charakhch'yan, Curvilinear grids of convex quadrilaterals, U.S.S.R. Comput. *Math. and Math. Phys*, **28** (1988), pp. 126–133.

[15] P. M. Knupp. On the invertibility of isoparametric map, *Comput. Methods Appl. Mech. Engrg.*, **78** (1990), pp. 313–329.

[16] T. N. Bronina, I. A. Gasilova , O. V. Ushakova, Algorithms for Three-Dimensional Structured Grids Generation. Zh. Vychisl. *Mat. Mat. Fiz.*, **6**, 2003, pp. 875-883. (http://www.ccas.ru/gridgen/ggta02/papers/Ushakova.pdf)

[17] G. P. Prokopov, On the organizing the comparison of algorithms and programs of generation of regular two-dimensional difference grids, Voprosy Atomnoi Nauki i Tekhniki. *Ser.: Matem. Modelirovanie Fizicheskikh Processov*, 1988, No. 3, pp. 98-108.

[18] G. P. Prokopov, Variational methods for calculations of two-dimensional grids for the solution of nonstationaty problems. Keldysh institute of applied mathematics, Preprint, 4, 2003.

[19] O. V. Ushakova, Algorithm of two-dimensional optimal grid generation, in *Numerical Grid Generation in Computational Field Simulation* , B. K. Soni and J. F. Thompson, eds., Mississippi State University, Mississippi State, MS, 1996, pp. 37–46.

[20] G.Farin, Curves and Surfaces for Computer Aided Geometric Design, A Practical Guide, Academic Press, 1997.

[21] N.Megiddo, *SIAM J. Computing*, **12**, (1983), pp. 759–776.

[22] H.Edelsbrunner, Algorithms in combinatorial geometry, Springer-Verlag, New York. 1987.

[23] A. S. Shvedov. *Mat. Zametki*, **39** (1986), pp. 597–605.

[24] J. K. Dukowicz, *Journal of Computational Physics*, **74**, 2, (1988), pp. 493–496.

Chapter 10

APPLICATION OF OPTIMAL GRID GENERATION ALGORITHMS TO THE VOLUMES OF REVOLUTION

Tatjana N. Bronina and Olga V. Ushakova
Institute of Mathematics and Mechanics
Ural Branch of the Russian Academy of Sciences

New algorithms for generation of three-dimensional structured optimal grids are developed. Their application to calculation of grids in the volumes of revolution are described.

10.1 Introduction

We describe the development of the algorithms [1] for generation of three-dimensional structured grids in the volumes of revolution. Suggested algorithms are designed for multi-material hydrodynamic simulation and for solving other physical and engineering problems (see chapter 8). The grid generation algorithms are not be reduced to the rotation of a two-dimensional grid about the axis. Rotational algorithms produce O-type grids containing degenerate cells on the axis of rotation, moreover these degenerate cells can become too small for small angles of rotation. Such grid quality is not desirable.

Section 10.2 is devoted to formulation of the problem.

The algorithms are developed within the variational approach of construction of optimal curvilinear grids [2]. Section 10.3 is devoted to a theoretical background of the approach in a three-dimensional case. In this section, a variational principle for construction of three-dimensional optimal grids close to a uniform and orthogonal ones in complex configurations of the domains is introduced. The distinctive features of this approach are enumerated.

The method realizing the approach consists of two stages. At the first stage, an initial grid satisfying some required properties is constructed. For generating an initial grid, a new algorithm based on the geometric approach is suggested. The requirement to the initial grid are the following. The algorithm constructs a structured nondegenerate grid close in the sense of [2] to a uniform grid. The initial grid satisfies the requirement of closeness of linear sizes of cells [1] (see further section 10.6). A nondegeneracy of the initial grid is

understood in terms of chapter 9. For some configurations of the domains, it is admissible that the initial grid can contain degenerate hexahedral cells degenerating into prisms with triangular bases or into a union of such prisms. Suggested algorithm permits the joining of different volumes of revolutions (for a grid, from a node to a node) that gives the opportunity to consider multi-block constructions of such configurations of domains. The algorithm ensures the correspondence of the symmetries of a grid to the symmetries of a domain. Such algorithm for construction of an initial grid is described in section 10.4.

The algorithm generating an initial grid has restricted possibilities in regulating the grid quality, therefore at the next stage some optimization algorithms are used. These algorithms are described in section 10.5. The optimization algorithms are developed on the basis of algorithms of the global reconstruction (rezoning) [1]. The optimization algorithms reconstruct an initial grid, in the general case, nondegenerate (in some cases, degenerate) into a nondegenerate and optimal in the sense of [2] curvilinear grid that is smooth and close to a uniform orthogonal grid. It is admissible that for some configurations of volumes of revolution, an optimal grid can contain, on the boundary, the hexahedral cells degenerating into prisms with a triangular base. An optimal grid satisfies also the criterion of closeness of linear sizes of cells [1] (section 10.6). The optimization of a grid can take place not only inside the body of revolution but also on its boundary. In this situation, the nodes of a grid are moving on the surface composed of ruled surfaces of cell faces of an initial grid. In order to fix the sharp turn of a boundary the special algorithms are constructed. New algorithms providing the orthogonality of grid lines to the boundary are also developed.

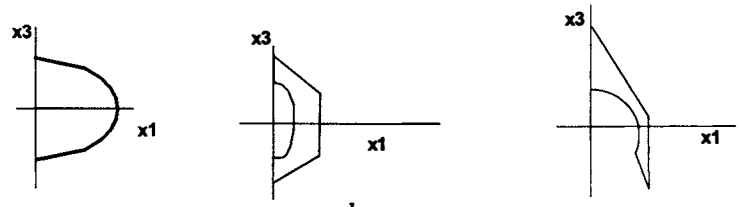
In section 10.6, some tests examples are given along with the estimations of grids quality.

10.2 Formulation of the Problem

The problem consists in constructing a structured three-dimensional grid in a three-dimensional domain G of the variables x_1, x_2, x_3 obtained by the rotation of a generatrix curve (further also called the main generatrix curve), given in the plane x_1, x_3, about the axis x_3 through the angle φ, $(\varphi_0 \leq \varphi \leq \pi)$. The generatrix curve can consist of straight line segments, arcs of circles and ellipses.

Types of generatrix curve are shown in Fig. 10.1 a — c. Corresponding configurations of domains very often arise in multimaterial hydrodynamics simulations (see chapter 8). It is supposed that the algorithm is not reduced to the rotation of a two-dimensional grid about the axis. It is already mentioned in Introduction that when constructing three-dimensional grids by the rotation of two-dimensional grids about the axis, singularities (degenerate cells) arise on the axis, more over degenerate cells become small for small angles of rotations. Such grid quality is not desirable. Because of this, described algorithms were suggested.

The generation of a structured grid in such domains is carried out by mapping approach (see the detailed description of this approach in chapter 9). According to the approach the domain G is represented as a curvilinear hexahedron with the vertices numbered from 1 to 8, and to construct a grid we look for a continuous mapping of an additional parametric



a *b* *c*

Figure 10.1: Types of generatrix curves:
a generatrix curve for the body of revolution;
b generatrix curve for the body of the "shell" type;
c generatrix curve for the body of the "cut shell" type.

parallelepiped $P = \{\xi = (\xi_1, \xi_2, \xi_3) : 0 \le \xi_1 \le N,\ 0 \le \xi_2 \le M,\ 0 \le \xi_3 \le L,\}$, where N, M, L are positive integers determining the number of grid nodes for each of coordinate directions. The values of the mapping $\mathbf{x} : P \longrightarrow G$

$$\mathbf{x} = \mathbf{x}(\xi_1, \xi_2, \xi_3) = \{x_1(\xi_1, \xi_2, \xi_3), x_2(\xi_1, \xi_2, \xi_3), x_3(\xi_1, \xi_2, \xi_3)\}$$

for $\xi_i, \xi_j, \xi_k = i, j, k,\ i = 0, 1, ..., N,\ j = 0, 1, ..., M,\ k = 0, 1, ..., L$ give the coordinates of nodes of a three-dimensional grid $H_{ijk} = \mathbf{x}_{ijk} = \mathbf{x}(i, j, k)$. We look for a mapping at the inner points and also at the boundary points of the parallelepiped P.

For the domain G, first we define the vertices of the curvilinear hexahedron and curvilinear edges (thus we define faces and interior part of the hexahedron), then we have to distribute grid points (nodes) in the interior of the hexahedron G, on its faces and edges. Vertices of G coincide with grid nodes corresponding to indices $i = 0, N,\ j = 0, M,\ k = 0, L$.

The grid has to satisfy some requirements. First of all, it should be a nondegenerate grid (see a definition in chapter 9). It means that a mapping from a parallelepiped P into a hexahedron G has to be one-to-one and onto. In this case, all cells will be nondegenerate hexahedrons with the one orientation of edges. As an exception, single cells degenerating into prisms with a triangular base arising as a rule on the boundary of domains are admissible. Other requirements imposed on a grid are the requirements of geometric optimality of a grid that is smoothness of grid lines and closeness of a curvilinear grid to a uniform and orthogonal grid. The requirements imposed on the algorithm are minimum of an initial data and reliability and efficiency of a numerical procedure of grid generation.

10.3 Variational Principles for Generating Optimal Grids

Variational methods are the most natural for generating optimal grids. To generate grids in the volumes of revolution we use approach [2].

To give the main idea of this approach in the three-dimensional case, we consider, in the space of variables x_1, x_2, x_3, the simply connected curvilinear hexahedron G with the

12 edges l_i $(i=1,...,12)$, 6 curvilinear faces S_k $(k=1,2,...,6)$ and 8 vertices. It is required to construct a grid in G with the coordinate surfaces λ_i, μ_j, ν_k $(i=0,1,...,N;$ $j=0,1,...,M;$ $k=0,1,...,L)$. For $i=0,N;$ $j=0,M;$ $k=0,L$, the coordinate surfaces coincide with the faces S_k. The intersection of three coordinate surfaces (see Figure 10.2) yields a grid node H_{ijk}. Let $r_{i\pm1,j,k}=|\overrightarrow{H_{ijk}H_{i\pm1,j,k}}|=|\mathbf{h_{i\pm1}}|$. Similarly, $r_{i,j\pm1,k}$, $r_{i,j,k\pm1}$, $\mathbf{h_{j\pm1}}$, $\mathbf{h_{k\pm1}}$ are defined. Denote by φ^l_{ij} the angles between the vectors $\mathbf{h_{i\pm1}}$ and $\mathbf{h_{j\pm1}}$ $(l = 1,2,3,4)$, and denote by $\Phi(x_1,x_2,x_3)$ the function for which the grid should be adapted.

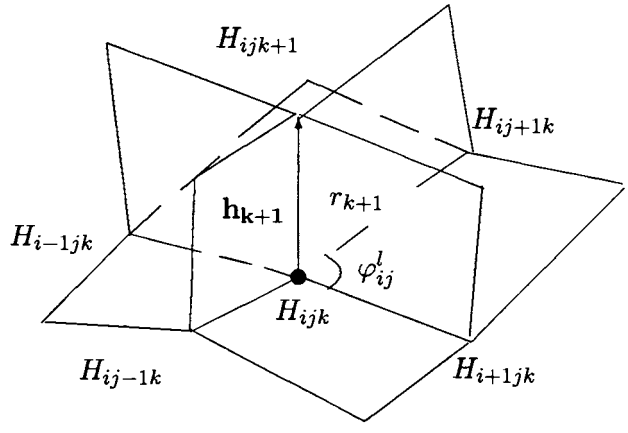


Figure 10.2: Grid nodes.

Let us introduce the functionals

$$D_U = \sum_{ijk} \left\{ [r_{i+1,j,k} - r_{i-1,j,k}]^2 \left(\frac{1}{r^2_{i+1,j,k}} + \frac{1}{r^2_{i-1,j,k}} \right) + \right. \tag{10.1}$$

$$+ [r_{i,j+1,k} - r_{i,j-1,k}]^2 \left(\frac{1}{r^2_{i,j+1,k}} + \frac{1}{r^2_{i,j-1,k}} \right) +$$

$$\left. + [r_{i,j,k+1} - r_{i,j,k-1}]^2 \left(\frac{1}{r^2_{i,j,k+1}} + \frac{1}{r^2_{i,j,k-1}} \right) \right\},$$

$$D_O = \sum_{ijk} \sum_{p=1}^{4} \left(\frac{1}{\sin^2 \varphi^p_{ij}} + \frac{1}{\sin^2 \varphi^p_{ik}} + \frac{1}{\sin^2 \varphi^p_{jk}} \right), \tag{10.2}$$

$$D_A = \sum_{ijk} \left\{ r^2_{i+1,j,k} [\Phi(H_{i+1,j,k}) - \Phi(H_{ijk})]^2 + \right. \tag{10.3}$$

$$\left. + r^2_{i,j+1,k} [\Phi(H_{i,j+1,k}) - \Phi(H_{ijk})]^2 + r^2_{i,j,k+1} [\Phi(H_{i,j,k+1}) - \Phi(H_{ijk})]^2 \right\},$$

$$D = A_U D_U + A_O D_O + A_A D_A. \tag{10.4}$$

The summation is taken over all inner grid points.

These functionals arise naturally as the generalization of their analogs for one and two-dimensional cases [2]. The functionals D_U, D_O, and D_A formalize criteria of grid closeness to uniform (U), orthogonal (O) and adaptive grids (A), respectively. These criteria, especially (U) and (A), are contradictory. So, they are applied by means of non-negative weight parameters A_U, A_O, and A_A determining the values of optimality criteria. Minimizing the functional D, one can obtain an optimal structured grid close to a uniform orthogonal grid, if the weight A_A is equal to zero.

Grid nodes on the boundary of the curvilinear hexahedron G can be given in different ways. It is possible to fix them or leave them free. We also can fix the angles between the coordinate (grid) lines and the boundary of the hexahedron G. These different cases for boundary node arrangement are also admissible to consider in the continuous formulation of the problems [2].

To get the continuous formulation of the problem, we can write out the continuous analogs I_U, I_O, I_A, I [2] of discrete functionals D_U, D_O, D_A, D

$$I_U = \iiint\limits_P \sum_{i=1}^{3} \left(\frac{\partial}{\partial \xi_i} \ln \sqrt{g_{ii}} \right)^2 d\xi_1 d\xi_2 d\xi_3,$$

$$I_O = \iiint\limits_P \frac{1}{J^2} \left(\frac{G_1 G_2}{g_{33}} + \frac{G_1 G_3}{g_{22}} + \frac{G_2 G_3}{g_{11}} \right)^2 d\xi_1 d\xi_2 d\xi_3,$$

$$I_A = \iiint\limits_P \left[\sum_{i=1}^{3} \left(\frac{\partial \Phi}{\partial \xi_i} \right)^2 g_{ii} \right] d\xi_1 d\xi_2 d\xi_3, \quad \text{where}$$

$$g_{ii} = \sum_{i=1}^{3} \left(\frac{\partial x_k}{\partial \xi_i} \right)^2, \quad J = \det \left\{ \frac{\partial x_i}{\partial \xi_k} \right\}, \quad \frac{\partial \Phi}{\partial \xi_i} = \sum_{j=1}^{3} \frac{\partial \Phi}{\partial x_j} \frac{\partial x_j}{\partial \xi_i},$$

$$G_i = g_{kk} g_{ll} - g_{kl}^2, \quad i \neq k, l,$$

$$I = A_U I_U + A_O I_O + A_A I_A.$$

We can find for them corresponding boundary conditions (fixed or natural [2]) and can consider the associated variational problems. For continuous variational functionals, we also can write out corresponding Euler equations (with proper boundary conditions) and can elaborate some numerical method for their solving.

Since we shall utilize the discrete functionals, we omit the corresponding formulations in continuous case and refer the reader to [2, 3].

We only note the main feature of arising variational problems. The main features of the approach are associated with the special way of formalization of criterion (U) which gives a nonlinear continuous variational functional I_U containing both first and second partial derivatives of the functions of grid node coordinates realizing the mapping. This continuous functional arises naturally as a measure of a relative error of a non-uniform grid in

comparison with a uniform grid. Such formalization leads to a system of Euler equations of the fourth order, hyperbolic in general sense. Second order of the derivatives in the minimized functional allows more arbitrariness in posing boundary conditions. It has enabled consideration of all types of boundary conditions listed above. Unknown functions of grid node coordinates and their derivatives can be given and fixed or can be found from the natural boundary conditions.

The construction of the functional of orthogonality I_O (D_O) also plays very important role. Due to this construction, the objective function includes the Jacobian of the desired mapping in the denominator; therefore, D (J) goes to infinity if the discrete analog of the Jacobian (or the Jacobian) of the mapping vanishes. Grid generation methods based on such functionals now are popular to call the barrier methods [4] since they put a barrier for degenerate elements. The functional of orthogonality influences also on the smoothness of grids.

The functional of adaptivity provides the condensing of grid nodes in the region where Φ (solution of the physical problems) has high gradients. In this paper, we do not use such feature of the functional, however such opportunity exists.

The analysis of Euler equations and arising boundary value problems as well as a number of problems inherent in grid generation methods based only on the solution of Euler equations, are also can be found in [2]. We will not follow the way of numerical solution of Euler equations because of arising problems and a necessity to overcome a number of difficulties (see [2]) influencing on the efficiency of the algorithms. These are the following difficulties:

1. The bulky form of the Euler equations results in large number of arithmetical operations.

2. For stability of calculations in the iterative schemes, the small time step should be selected and that has an effect on the number of iterations required to reach the steady–state condition.

3. The contradictoriness of the requirements included in the basis of a variational method leads to natural difficulties in the choice of control parameters defining the value of one or another criterion of optimality. Variation of the weight coefficients in a wide range can cause instability of the numerical procedure in the solution of the equations.

In [2], for any positive weight coefficients the type of the Euler system does not vary. However, since for A_U=0, $A_O \neq 0$ the system becomes of a mixed elliptic-hyperbolic type (see [2], in this case the problem can be incorrectly formulated), then for stability of calculations in the solution of Euler equations the weight coefficient A_U should be positive ($A_U = 1$) and other weight coefficients should be selected so that the contribution of summands corresponding to D_O and D_A does not exceed D_U. Otherwise the problem can turn out to be unstable.

For implementation of the variational principles, we shall elaborate, relying on our experience in the two-dimensional case [2],[5] and our first experience in three-dimensional case [1], a direct geometrical methods of the discrete functional D minimization.

While constructing grids in the volume of revolution, we consider only two optimality

criteria responsible for the closeness of a curvilinear grid to a uniform and orthogonal grid and construct geometrically optimal grids disregarding the adaptation criterion.

The algorithm consists of two independent parts: construction of an initial grid which plays the role of an initial guess and optimization of an initial grid with respect to given criteria of optimality.

Construction of an initial grid is realized by a geometrical method. Optimization of an initial grid represents the procedure of minimization of the functional D with $A_A = 0$.

10.4 Generation of the Initial Grid

We describe the algorithm of initial grid generation based on the geometric approach taking into account natural symmetry with respect to the axis of rotation and lines of sticking together separate surfaces of revolution. At the stage of the generation of the initial grid, the obtaining the non-self-intersecting families of coordinate lines and surfaces is considered as the main requirement to a grid. In [1], algorithms for generation of initial grids are described. Examples of grids obtained by these algorithms are shown in Fig 10.3.

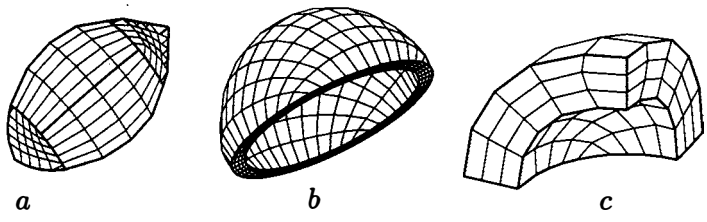


Figure 10.3: Grids in the domains of revolution:
a a grid in the body of revolution;
b a grid in the body of revolution — " shell";
c a grid in the body of revolution — "cut shell" type.

While working with grids generated by the algorithms from [1] the following disadvantages were revealed:

Grids for the body of revolution (Fig. 10.3 *a*) do not preserve the symmetry of the body.

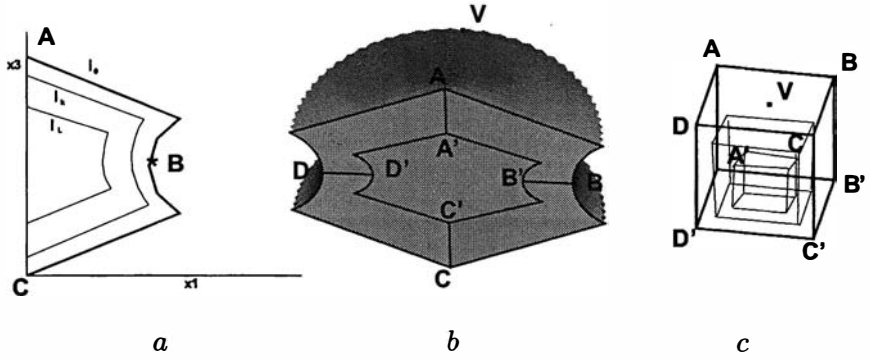
In the bodies of revolution, in some zones, cells of very small size and cells with triangular faces arise (Fig. 10.3 *b*, *c*).

Besides, for the computation of the processes in constructions heterogeneous in materials, it is desirable to have block grids, that is for grids generated in bodies of revolution and shells (types of generatrix curves *a* and *b* in Fig. 10.1) the possibility of nodes joining on one of the faces of a curvilinear hexahedrons has to be provided under the condition of identical description of generatrix curves in blocks of "sticking regions". The joining of grids on one of the faces is understood as coinciding the coordinates of nodes and their numeration.

In this chapter, we suggest the algorithm avoiding to some extent indicated disadvantages of [1].

For generating structured grid, the domain of revolution is represented as curvilinear hexahedron with 8th vertices A, B, C, D, A', B', C', D' of the type shown in Fig. 10.4. The representation of the body of revolution in the form of curvilinear hexahedron of the type shown in Fig. 10.4 was suggested by O.M.Kozyrev. Despite of such exotic representation of a hexahedron in which five faces lie in one plane, this configuration allows to solve the question about preserving the symmetry on the surfaces of revolution and allows to compute block constructions consisting of the body of revolution and a few shells.

Taking into account that a grid is constructed for the body of revolution, we organize the algorithm in such a way that a three-dimensional grid is a set of Lth grids on the surfaces of revolution, obtained by the rotation through angle φ of some lines of the main generatrix curve in the plane x_1x_3. In Fig. 10.4, the arrangement of vertices and faces of a curvilinear hexahedron on the body of revolution (Fig. 10.4 b) and correspondence of them to vertices and faces of a parametric parallelepiped (Fig. 10.4 c) are schematically shown. (The point V on the face $ABCD$ is shown for the clearness of representation.)



a b c

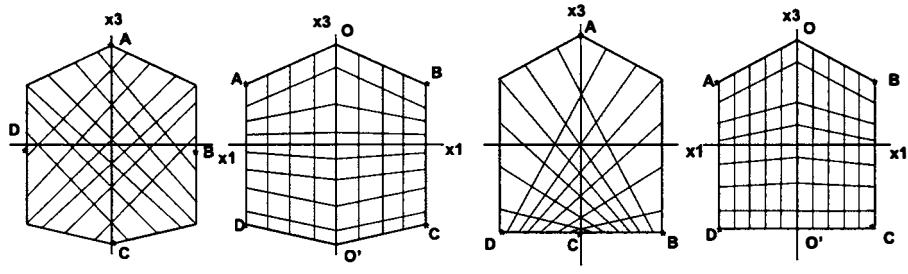
Figure 10.4: Correspondence of vertices and faces of a curvilinear hexahedron to the vertices and faces of a parametric parallelepiped for the body of revolution:
a generatrix curve for the body of revolution;
b curvilinear hexahedron;
c parametric parallelepiped.

Let us describe the algorithm for generation of a three-dimensional structured grid $\{x_{1_{ijk}}, x_{2_{ijk}}, x_{3_{ijk}}\}, i = 0, .., N, j = 0, .., M, k = 0, .., L, M = N$ in the body of revolution. Let in the plane x_1x_3 (the axis x_3 — axis of rotation) the main generatrix curve AC be given (Fig. 10.4 a), vertices of a curvilinear hexahedron A and C lie on the axis of rotation. Let us choose on the generatrix curve the point B (from the symmetry condition, from the equality of lengths of curves AB and BC, or the features of generatrix curve AC). In the plane x_1x_3, we construct curves l_k, $k = 1, .., L$, similar to the main generatrix curve

ABC (Fig. 10.4 a).

Using each curve l_k as a generatrix curve and rotating it through $180°$ and thus constructing on each of Lth surfaces of rotation the grid with the $(N+1)*(N+1)$ number of nodes, we obtain the required three-dimensional grid in the domain of revolution.

Let us describe the generation of a grid on the surface of rotation. Two of the possible arrangements of the families of the coordinate lines on the faces of revolution is schematically shown in Fig. 10.5 (Fig. 10.5 a, b for the configuration cosisting of the volume of revolution and the shell, Fig. 10.5 c, d for the "cut shell").



a b c d

Figure 10.5: Arrangements of the coordinate lines on the surface of revolution:
a, b for the surfaces in configuration consisting of the body of revolution and "shell";
c, d for the surfaces in the "cut shell".

Taking into account the fact that the nodes are lying on the surface of revolution, it is natural to use the formulas

$$\begin{aligned} x_{1_{ijk}} &= \overline{x}_{1_{nmk}} \cos(\alpha_{ij}), \\ x_{2_{ijk}} &= \overline{x}_{1_{nmk}} \sin(\alpha_{ij}), \\ x_{3_{ijk}} &= \overline{x}_{3_{nmk}}, \end{aligned} \tag{10.5}$$

where $\{\overline{x}_{1_{nmk}}, \overline{x}_{3_{nmk}}\}$ are coordinate of nodes lying on the generatrix curve l_k.

The values of angles α_{ij} for computing coordinate of nodes for the scheme in Fig. 10.5 a for the grid on the surface of revolution are suggested to compute in the following way. For the coordinates of nodes on the diagonal AC (Fig. 10.4 a, c) $\alpha_{ij} = \pi/2$. For the coordinates of nodes in the curvilinear triangle DAB (coordinates of nodes on the generatrix curve l_k, we have indices $m = j, n = N$ from (10.5)) with ijk , where $i = j-1,\ j = 1, .., j_1,\ j_1 = 2, .., N$

$$\alpha_{ij} = \pi j_1/2(N - j_1), \tag{10.6}$$

and with ijk, where $i = j-1, ..., 1,\ j = 2, .., N-1$,

$$\alpha_{ij} = i\pi/2(N - i). \tag{10.7}$$

Coordinates of nodes in the triangle DBC is calculated analogously by the symmetric substitution of indices (ij) by (ji). As the result, we obtain grids on Lth surfaces of revolution. To obtain the grid on the face $A'B'C'D'$, we can define $x_{2_{ijk}} = 0$ for the nodes with $k = L$, and further we can improve (optimize) the grid by one of the algorithms for construction of optimal grids [2].

Described above algorithm allows to compute the initial grids in the constructions consisting of some layers of shells or for the body of revolution with some shells put on it. Considered construction of grids is an example of block-structured grids joining on the boundaries of blocks.

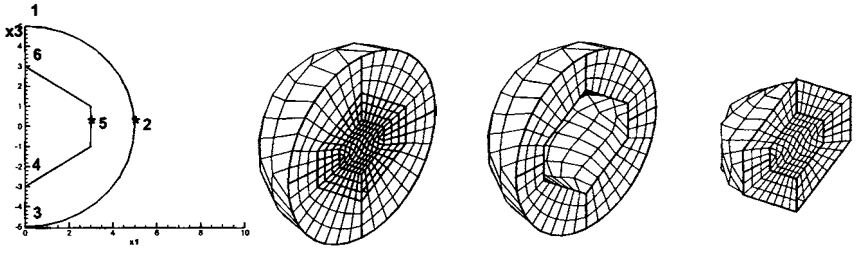


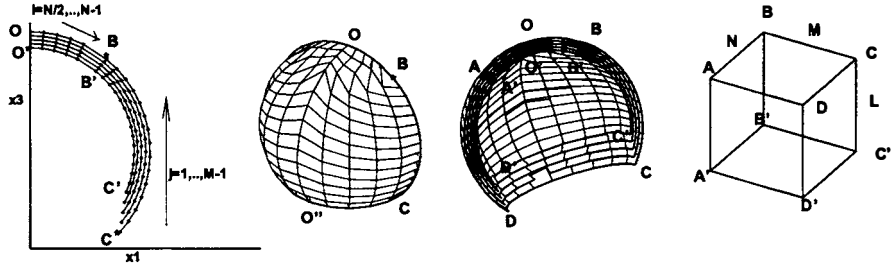
Figure 10.6: Example of joining grids :
a generatrix curves;
b joining grids;
c grid for the shell;
d grid for the volume of revolution.

An example of such construction is shown in Fig. 10.6 *b*. In Fig. 10.6 *a*, the generatrix curves for two bodies are shown: the line 1-2-3-4-5-6-1 is the generatrix curve for the shell, the line 6-5-4 is the generatrix curve for the body of revolution. The line 6-5-4, under the rotation, forms the surface common for the body of revolution and the shell. The representation of the body of revolution in the form of curvilinear hexahedron in the considered case allows to obtain the face $ABCD$ (Fig. 10.4 *b*) coinciding with one of the faces of a shell the generatrix curve of which partially consists of a generatrix curve for the body of revolution (Fig. 10.6 *c*, *d*).

The arrangement of parametric lines on the surface of revolution obtained by using (10.6) and (10.7) (Fig. 10.5 *c*) leads, as mentioned above, to emerging of cells of small size and with small angles (Fig. 10.3 *c*). Because of this, for obtaining grids on the surface of revolution for the body of "cut shell" type (generatrix curve is shown in Fig. 10.1 *c*), it is obvious to use the algorithms with the arrangements of coordinate lines shown in Fig. 10.5 *d*.

The idea of generation of a three-dimensional grid for the body of revolution of the "cut shell" type is analogous to the idea described above, that is a grid is constructed by Lth surface grids arranged on the surfaces of revolution. The arrangement of vertices and faces

of a curvilinear hexahedron for this case and correspondence to the vertices and faces of a parametric cube is schematically shown in Fig. 10.7 b, d. Let us describe, first, generation of a grid for the case, when for the main generatrix curve $OCC'O'$ (Fig. 10.7 a) in the curvilinear hexahedron $OCC'O'$ we manage to construct the two-dimensional grid by linear interpolation using points on the curves OC and $O'C'$. In this case, generating two grids on the surfaces of rotation (generatrix curves OBC and $O'B'C'$), inner nodes of a three-dimensional grid are possible to find also by linear interpolation.



a b c d

Figure 10.7: The scheme of grid generation for the "cut shell":
a generatrix curve;
b,c grids on the boundary surfaces; d parametric parallelepiped; N, M, L are numbers of grid intervals for coordinate directions.

The grid on the surface of rotation is suggested to construct in the following way. Calculating $M + 1$ nodes (for example, uniformly in the arc length) on the line OBC and using (10.5) for $\alpha_{ij} = \pi/2$, let us obtain nodes of a three-dimensional grid on the line OO''. Coordinates of nodes lying on the part OB of the main generatrix curve (Fig. 10.7 a) we calculate using (10.5) for

$$\alpha_{ij} = \pi/(2n_\varphi), \tag{10.8}$$

where $(((i = N_0 + N_1, ..., N_0 + 2N_1, j = M - 2, .., M - 2 - N_1), n_\varphi = 1, .., N_1)$, $N_1 = 1, .., N_0), N_0 = (N + 1)/2$.

Then, we calculate the coordinates $\{(x_{1_{ij0}}, x_{3_{ij0}}), j = i, .., M\}$ $(M - i)$ of the additional nodes on the line BC and use (10.5) for

$$\alpha_{ij} = \pi/2 + \pi(N_0 - i)/2N_0. \tag{10.9}$$

If the domain $OCC'O'$ has more complicated configuration, then for constructing generatrix curve in the plane x_1x_3, it is necessary to construct a two-dimensional grid $\{x_{1_{ks}}, x_{3_{ks}}\}$, $k = 0, ..., L$; $s = 0, ..., L_1$; $L_1 = (N + 1)/2 + M$ with the number of grid intervals $((N + 1)/2 + M) * L$. Then, let us define the generatrix curve with the number k_f as a line l_{k_f}, given by the points $\{x_{1_{k_fs}}, x_{2_{k_fs}}\}$, $s = 1, .., L1$ (Fig. 10.7 a). Then on the constructed generatrix curves we calculate nodes using (10.8),(10.9).

10.5 Algorithm for Minimization of the Functional D. (Global Reconstruction Algorithm)

The algorithm for the functional D minimization initially emerged as a global reconstruction algorithm designed for reconstructing the bad quality grid into a grid of a good quality. Bad quality grids can be obtained using the Lagrangian method. On each iteration in the procedure of the solution of a physical problem, the grid is constructed independently in each of the coordinate directions. For some moment of time, the grid can become distorted and cannot satisfy by its quality. It can possess sharp changing of the size of cells, it cannot be smooth and it can be very close to degenerate. But it would be desirable to continue the computation of the main problem. In such situations, the rezoning is required. It is required on the basis of "bad" grid, the grid which does not give desirable quality, to develop the algorithms for reconstruction of the grid with the aim to obtain "good" grid, close to the uniform one, orthogonal one and smooth. For this purpose, the global reconstruction algorithm was developed. To start the computations of the physical problem by the Lagrangian method was supposed on the initial grid constructed by the method described in the previous section. However the quality of the initial grid, since it was constructed by simple geometrical procedures, did not satisfy the requirements of the numerical method because it often turned out to be nonsmooth and some times contained nondegenerate hexahedral cells degenerating into prisms (see chapter 9). (For constructing hexahedral cells, we use a trilinear mapping of the unit cube and for estimating grid nondegeneracy, we use nondegeneracy criteria also of chapter 9.) Because of above reason a global reconstruction algorithm was begun to use as the algorithm generating an optimal grid on the basis of some initial grid. From that time this algorithm underwent a number of changes. Its present description will be done taking into account these changes.

Let a domain G be given by a structured nondegenerate three-dimensional grid. At this stage, the grid is considered as an initial one. It is allowed that some hexahedral cells (usually along edges of a curvilinear hexahedron G or at its vertices) degenerate into prisms with triangular bases or into a union of such prisms. Such cases are characterized by zero or nonpositive values of the Jacobian of a trilinear mapping of corresponding cells in two vertices (see chapter 9). The practice of computations shows that described algorithm allowed single cells to have nonpositive values of the Jacobian maximum at three or four vertices.

The algorithm of functional D minimization can be carried out both in the whole domain G including its boundary except vertices or in its subdomain selected from the given domain by the starting and ending values of indices of nodes. For this case, boundary nodes of the subdomain are considered to be fixed. In the case of reconstruction of the grid on the boundary, it is important to preserve the form of the domain. The grid points on the boundary of the physical domain move on the surface "woven" from ruled surfaces (see chapter 9), which are the faces of the initial grid cells.

At present time the following five algorithms are elaborated:

Algorithm 1. Fixed boundary nodes. The reconstruction of a grid is carried out on the basis of the above criteria of optimality inside the physical domain or in its subdomain.

In this case, boundary nodes are considered to be fixed (both for the whole domain or its subdomain).

Algorithm 2. Free boundary nodes. The grid generation is carried out in the whole domain. This algorithm assumes grid node reconstruction from the above criteria of optimality as on the boundary of the domain G (on edges and faces) and inside the domain. Only vertices of a hexahedron are fixed.

Algorithm 3. Conditions of orthogonality to faces. The reconstruction of nodes is carried out also for the whole domain. The reconstruction of nodes on the faces is carried out from the condition of grid lines orthogonality to faces of the domain, and on edges and inside the domain from the optimality criteria.

Algorithm 4. Conditions of orthogonality to faces and edges. The reconstruction of nodes is carried out for the whole domain, on faces and edges from the conditions of grid line orthogonality to faces and edges, inside the domain from the optimality criteria. In this case, because of concordance conditions for nodes on edges, adjoining faces have to be orthogonal to each other.

Algorithm 5. Reconstruction of selected nodes. The reconstruction of the grid is carried out in the nodes selected by the values of indices. This algorithm allows to fix the node arrangement on the single faces, to fix the location of single nodes on the boundary for the preservation of such constructive features of the form of the domain as sharp turn of the boundary, and to define the location of single grid nodes from the condition of grid line orthogonality to the boundary.

Without loss of generality we consider that a subdomain in which it is required to reconstruct the grid is the domain G.

A numerical algorithm is a three-dimensional analog of the algorithm [5] and is developed within the approach [2]. This algorithm is the algorithm of direct geometric minimizing the discrete functional D with $A_A = 0$. This minimization will ensure construction of the optimal grid close to uniform and orthogonal ones and with smooth grid lines.

Here, we describe the basic ideas underlying the algorithm.

Iterations start from a nondegenerate initial grid. As an exception degenerate grids containing single degenerate cells (usually on the boundary ∂G) are allowed. Admissible cases are listed above.

At each iteration, while defining the inner node i, j, k, the other nodes are fixed and the location of a node is found from the condition of nondegeneracy of a grid and the condition of a minimum of the functional D on a special set of points determining a feasible set. For each inner node, we apply the local procedure for optimizing the grid. We consider eight grid cells which have node i, j, k as a vertex. At nth iteration, the grid points $H^n_{i+\Delta_i, j+\Delta_j, k+\Delta_k}$, $\Delta_i, \Delta_j, \Delta_k = -1, 0, 1$ which belong to the union of eight cells (hexahedrons) form a so-called 27 point stencil (Fig. 10.8). We fix exterior vertices of this union while allowing the location of the central node to vary. For boundary nodes, the stencil is constructed analogously (see Fig. 10.15). It is important to note the following main features of this procedure.

The displacement of any node does not effect the stencil geometry of any other node;

hence it cannot change the values of the Jacobian at other nodes and therefore the properties of a grid at those nodes.

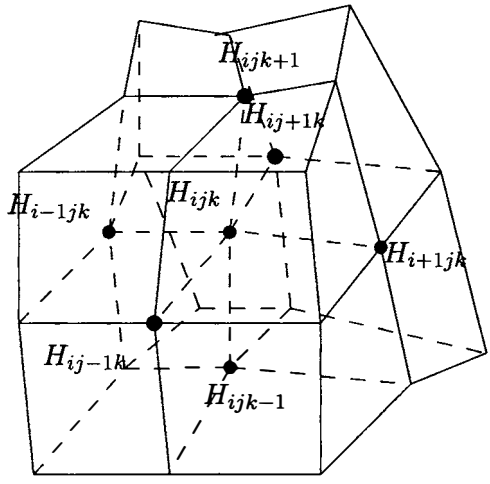


Figure 10.8: Stencil geometry.

Many strategies may be used to define the displacement $\overrightarrow{H^n_{ijk}H^{n+1}_{ijk}}$ of any node H^n_{ijk} for minimizing the functional and preserving a nondegeneracy of a grid. The feasible set is the octagon $H^n_{i,j-1,k}$, $H^n_{i,j+1,k}$, $H^n_{i-1,j,k}$, $H^n_{i+1,j,k}$, $H^n_{i,j,k-1}$, $H^n_{i,j,k+1}$, (see Figures 10.8,10.9) if this octagon is convex.

It was already mentioned that the objective function contains the determinant of the Jacobian matrix in the denominator; therefore D will become infinite if discrete analog of the Jacobian (six volumes of tetrahedron at vertices of a cell) vanishes. It was shown in [4] that such an objective function (containing the Jacobian in denominator) will not produce degenerate elements provided that the initial grid of the iterative procedure has no degenerate elements. Such construction for grid generation are now referred to as barrier methods possessing a barrier against degenerate elements.

10.5.1 Optimizing Procedure for Inner Nodes

We describe the computational realization of five algorithms mentioned above.

We consider the stencil of nodes $H^n_{i+\Delta_i,j+\Delta_j,k+\Delta_k}$, Δ_i, Δ_j, $\Delta_k = -1, 0, 1$, in order to find the new location H^{n+1}_{ijk} of the inner node H^n_{ijk}. The geometry of this stencil will define the new location of the node H^n_{ijk} by means of the value of the functional D. We fix the nodes different from H^n_{ijk} and allow the node H^n_{ijk} to move. The location of the node is found from the condition of the nondegeneracy of a grid and minimum of the functional D

on the special set of points H_+ or H_-. In the process of computations, grid node coordinates are replaced by new ones.

Sets of points for minimizing the functional D. Sets H_+ and H_- are constructed by means of special points C_+ and C_-. While finding these points, we choose the criteria of a closeness of a grid to a uniform one in the distance between neighboring nodes as a governing criteria.

We consider the octahedron Γ (with plane faces) and the vertices $H^n_{i,j-1,k}$, $H^n_{i,j+1,k}$, $H^n_{i-1,j,k}$, $H^n_{i+1,j,k}$, $H^n_{i,j,k-1}$, $H^n_{i,j,k+1}$ (See Fig. 10.8, 10.9, 10.10. In Fig. 10.8, a superscript n of a number of iteration is omitted). For brevity, we use also the notations

$$I_m = H^n_{i-1jk},\ I_p = H^n_{i+1jk},$$

$$J_m = H^n_{ij-1k},\ J_p = H^n_{ij+1k},$$

$$K_m = H^n_{ijk-1},\ K_p = H^n_{ijk+1}.$$

If a grid is nondegenerate at each iteration, then octahedron will be nondegenerate, since such octahedron can be divide into eight tetrahedrons corresponding to coefficients α equal to six volumes of hexahedrons constructed on the edges of a cell (see chapter 9, here we omitted the subscripts in the notation of the coefficients α). Tetrahedrons are considered as the images of corresponding tetrahedrons of the unit cubes for linear mappings and the whole octahedron as the image of the octahedron of the computational space for a piecewise-linear mapping. By virtue of positivity of the coefficients α at each node, a union of tetrahedrons — octahedron — will be nondegenerate in the sense of chapter 9 for the considered mapping.

If octahedron Γ is convex, then a point C_+ is defined. It coincides with the center of gravity of this octahedron (center of gravity of its three diagonals). The point C_- is defined, if Γ is nonconvex. In this case, C_- is defined as a center of gravity of one or two inner diagonals in dependence of how many diagonals this octahedron has. For convex polyhedrons, the functional D is minimized on the set H_+, for nonconvex polyhedrons on the set H_-.

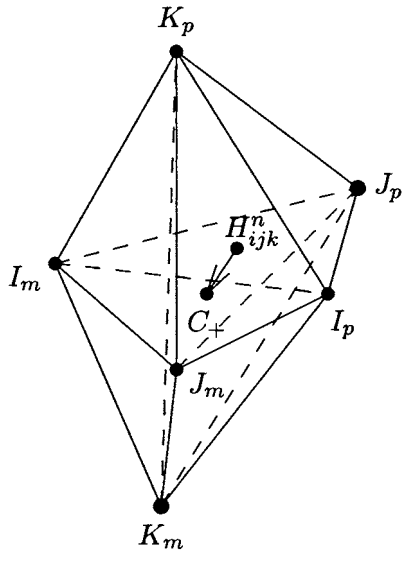


Figure 10.9: Convex feasible set.

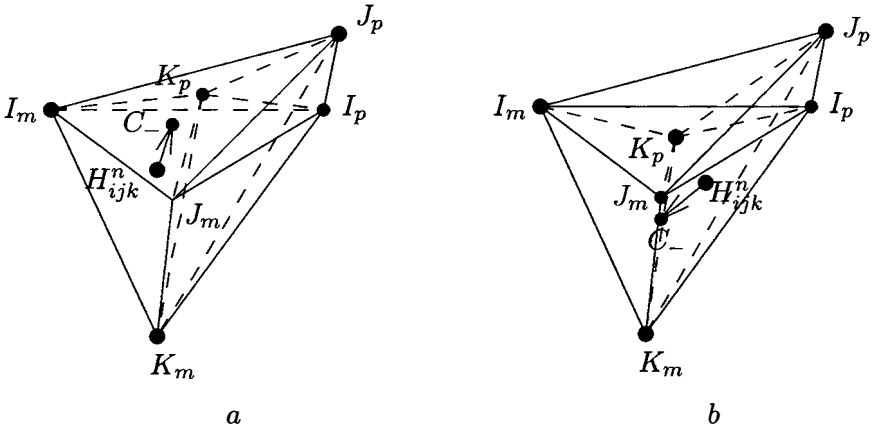


Figure 10.10: Nonconvex feasible sets.

The sets H_+ and H_- are the sets of points $H^{n+1}_{ijk}(m)$ such that

$$\overrightarrow{H^n_{ijk}H^{n+1}_{ijk}(m)} = \overrightarrow{H^n_{ijk}C}\frac{m}{3}, \quad m = 0, 1, 2, 3,$$

where C coincides with C_+ or C_-.

Note that for the center of gravity of the octahedron ($H^{n+1}_{ijk} = C$), (for a uniform and orthogonal grid) a contribution to the functionals D_U and D_O is minimal (equal to zero for

D_U). We use this point for convex octahedrons Γ for curvilinear grids too. For nonconvex octahedrons for the center of gravity of one inner diagonal, we have zero contribution to the functional D_U for the direction corresponding to the inner diagonal of the octahedron. The necessity to consider convex and nonconvex octahedrons Γ is caused by a necessity to consider complicated configurations of the domains with such a quality of an initial grid and admissability of a single degenerate cells in the initial grid. In fact, nonconvex octahedrons Γ can arise even if all eight coefficients α (see chapter 9), corresponding to the node i, j, k and considering eight cells, composing the main stencil are positive (see Fig. 10.10 *a*). In Fig. 10.10 *b*, there is an example of nonconvex octahedron Γ with four negative coefficients α. Examples *a* and *b* can arise for complex configurations of the domains, an example *b* for grid with degenerate cells. In the considered cases of octahedrons Γ, if we choose, as the new location H_{ijk}^{n+1} of the node H_{ijk}^{n}, any point on the segment $H_{ijk}^{n}C$, we will not change the signs of the eight coefficients α and thus the new grid node will satisfy necessary nondegeneracy conditions (for sufficiently small displacements, sufficient nondegeneracy conditions provided that the initial grid was nondegenerate). It is clear that if, for one node, we have more than four nonpositive coefficients α, (this characterizes a degenerate self- intersecting grid), then we can not draw a conclusion that octahedron is nondegenerate. In such situations the octahedron is self-intersecting and is not of the form shown in Fig. 10.9, 10.10. So because of this, suggested algorithm assumes only single degenerate cells with a number of nonpositive coefficients α listed above. If, in an initial grid, degenerate self-intersecting octahedrons Γ arise, the algorithm and corresponding computer code diagnoses such cases and computation of a grid stops.

In the cases of initial nondegenerate grids, the considered way of construction of new nodes will not produce degenerate cells.

The testing of octahedron Γ on convexity is carried out on the basis of one of definitions of polyhedron convexity, namely of testing the following: for any face of a polyhedron, all its vertices (except vertices of a chosen face) are arranged from one side of a plane of a face. Computationally, this fact is tested by the positivity of the volumes of special tetrahedrons. In these conditions, it is easy to see the analogy with nondegeneracy conditions for a grid from chapter 9 which is, in fact, also some conditions on the volumes of tetrahedrons, so in testing the octahedron convexity we operate again with the volumes of tetrahedrons and this is very convenient for organization of computations.

To verify the convexity of the octahedron Γ, we need to check the positivity of volumes of the following twelve tetrahedrons or triple scalar products:

$$[\overrightarrow{I_mJ_m}, \overrightarrow{I_mI_p}, \overrightarrow{I_mK_p}],\ [\overrightarrow{I_mI_p}, \overrightarrow{I_mJ_p}, \overrightarrow{I_mK_p}],$$

$$[\overrightarrow{I_mI_p}, \overrightarrow{I_mJ_m}, \overrightarrow{I_mK_m}],\ [\overrightarrow{I_mJ_p}, \overrightarrow{I_mI_p}, \overrightarrow{I_mK_m}],$$

$$[\overrightarrow{J_pI_m}, \overrightarrow{J_pJ_m}, \overrightarrow{J_pK_p}],\ [\overrightarrow{J_pI_p}, \overrightarrow{J_pJ_m}, \overrightarrow{J_pK_p}],$$

$$[\overrightarrow{J_pJ_m}, \overrightarrow{J_pI_m}, \overrightarrow{J_pK_m}],\ [\overrightarrow{J_pI_p}, \overrightarrow{J_pJ_m}, \overrightarrow{K_mJ_p}],$$

$$[\overrightarrow{K_mK_p}, \overrightarrow{K_mI_m}, \overrightarrow{K_mJ_m}],\ [\overrightarrow{K_mK_p}, \overrightarrow{K_mK_p}, \overrightarrow{K_mJ_m}],$$
$$[\overrightarrow{K_mI_m}, \overrightarrow{K_mK_p}, \overrightarrow{K_mJ_p}],\ [\overrightarrow{K_mK_p}, \overrightarrow{K_mI_p}, \overrightarrow{K_mJ_p}].$$

The positivity of first four triple scalar products means that the diagonal $[I_m, I_p]$ is an inner diagonal of the octahedron Γ, the positivity of second four triple scalar products means that the diagonal $[J_m, J_p]$ is an inner one, and the positivity of the last four triple scalar products guarantees this for the $[K_m, K_p]$.

If three diagonals of the octahedron Γ are its inner diagonals, then octahedron is convex, if one or two diagonals are outer diagonals, then octahedron is nonconvex. If octahedron is convex, then

$$C = C_+ = \frac{1}{6}(I_m + I_p + J_m + Jp + K_m + K_p). \tag{10.10}$$

If only the diagonal $[I_m, I_p]$ is an inner one, then

$$C = C_- = \frac{1}{2}(I_m + I_p). \tag{10.11}$$

If only the diagonal $[J_m, J_p]$ is an inner one, then

$$C = C_- = \frac{1}{2}(J_m + J_p). \tag{10.12}$$

If this is valid for the diagonal $[K_m, K_p]$, then

$$C = C_- = \frac{1}{2}(K_m + K_p).$$

In the cases of two inner diagonals, we have

$$C = C_- = \frac{1}{4}(I_m + I_p + J_m + J_p)$$

for $[I_m, I_p]$ and $[J_m, J_p]$,

$$C = C_- = \frac{1}{4}(I_m + I_p + K_m + K_p)$$

for $[I_m, I_p]$ and $[K_m, K_p]$,

$$C = C_- = \frac{1}{4}(J_m + J_p + K_m + K_p)$$

for $[J_m, J_p]$ and $[K_m, K_p]$.

If, for all stencils, octahedrons Γ are convex and if $H_{ijk}^{n+1} = C = C_+$, then such a procedure is equivalent to the numerical solution of the system of Laplace equations of the form

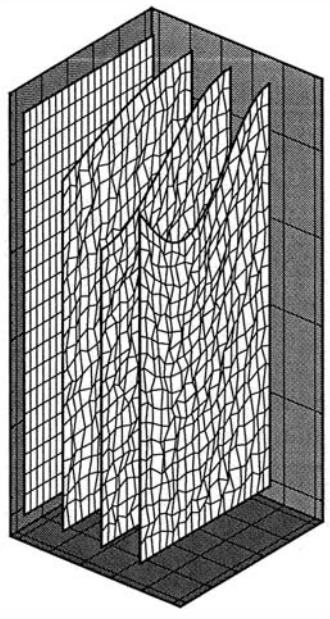
$$x_{1\xi_1\xi_1} + x_{1\xi_2\xi_2} + x_{1\xi_3\xi_3} = 0$$

$$x_{2\xi_1\xi_1} + x_{2\xi_2\xi_2} + x_{2\xi_3\xi_3} = 0$$

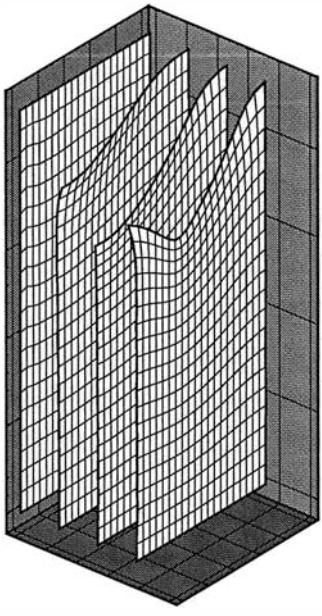
$$x_{3\xi_1\xi_1} + x_{3\xi_2\xi_2} + x_{3\xi_3\xi_3} = 0$$

with fixed boundary conditions on the boundary of the domain P (functions of grid node coordinate x_i, $i = 1, 2, 3$ are given and fixed on the boundary of the domain P). Numerical procedure for this case is the explicit scheme (10.10) for the solution of an auxiliary nonstationary problem for such equations. However, this simple procedure will not work efficiently even for simple configurations of the domains, since it produces degenerate cells especially for the domain with sharp turn of the boundary (see, an example in Fig. 9.7 a, b of chapter 9).

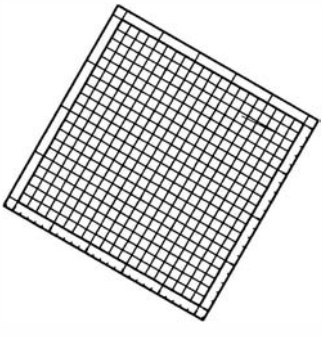
Arising degenerate cells is because this form of Laplace equations does not guarantees the nondegeneracy of its solution giving us the sought mapping (see, for example, chapter 1, 5 and [6]). Above deficiency occurs for nonconvex octahedrons Γ. Since, for nonconvex octahedrons Γ, we modify (see (10.11) and (10.12)) the scheme (10.10) excluding one or two directions or components in each of the equations in dependence of the number of inner diagonals of the Γ. In this case $C = C_-$. Such modification does not allow us to get rid of degeneracy inherent in the initial grid and caused by the configuration of the domain. So, if we have prismatic cells with triangular base in the initial grid (this occurs usually on the boundary), we have in the same places of a domain G such prismatic cells in the optimal grid too (see further Fig. 10.21). However, if nondegeneracy is not caused by the configurations features of the domain and is a characteristic only of the initial grid, as the experiments show, the algorithm allows to get rid of degenerate hexahedral cells of the types mentioned above (see for example, Fig. 10.11). The practice of computation shows that our algorithm can improve an initial grid composed of nondegenerate hexahedral cells and degenerate hexahedral cells being not self-intersecting polyhedrons (or nondegenerate cells of other types than hexahedral). Such limitation for the initial grids defines now one of the requirements to them.



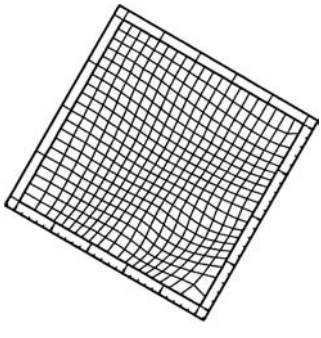
a



b



c



d

Figure 10.11: Example of reconstruction of a degenerate grid: coordinate surfaces $i = 0, 5, 10, 15$, *a* initial grid, *b* optimal grid; coordinate surface $k = 21$, top bound of the domain, *c* initial grid, *d* optimal grid.

When the point C is found, we organize the local optimization process for the functional D on the set of point H_+ or H_-. We choose the new node as a point from above set giving the functional D its minimal value. As the result of such choice, we obtain the new stencil consisting of nondegenerate cells, if the initial grid is nondegenerate, or a stencil containing some degenerate cells of the type listed above and in chapter 9, if the initial grid is degenerate. This is allowed as an exception. When the new node is found, we replace the old one by the new node and go into the next stencil.

The order of computations. Initially, in [1] the computation of inner nodes was organized in the following order. First, the index i is changed, then j and then k. However, in order to obtain the grid with the symmetry corresponding to the symmetry of a domain we have to organize the procedure analogous to that which was organized in the two-dimensional case [5]. In the two-dimensional case the computation of nodes was organized in the clock-wise direction on the bordering lines in the order of moving away from the boundary. In the three-dimensional case, to provide the desired symmetry, the computations have to be organized on the bordering surfaces. Zero bordering surface is a boundary of the domain G, the first bordering surface is formed by the nodes neighboring with the boundary nodes and so on. On the bordering surface, the computation can be organized, for example, first on the faces corresponding to constant values of index i, then j and then k. On each face of the bordering surface, the computation is organized on the bordering lines in the clockwise direction in the order of moving away from the boundary (see Fig. 10.12, bordering lines on faces $i = 0$ and $i = N$). The computation of inner nodes begins from the first bordering surface.

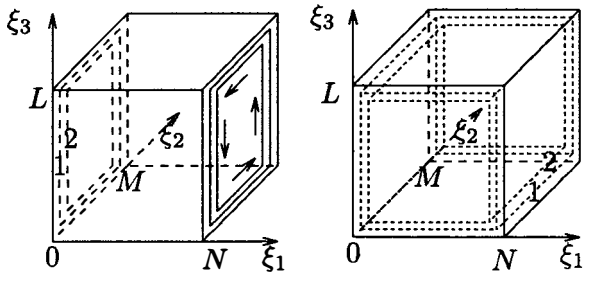


Figure 10.12: Bordering lines and surfaces.

Computation of the functional D. While computing coordinates of inner nodes we estimate the contribution D_{ijk} to the functional D. This contribution is evaluated as

$$D_{ijk} = D_{U\,ijk} + A_O D_{O\,ijk}.$$

The contribution to the functional of uniformity for each inner node is computed as the component of the corresponding sum (10.1).

The contribution to the functional of orthogonality for each inner node is computed as

$$D_{O_{ijk}} = \sum_{p=1}^{4} \left(\frac{1}{\sin^2 \varphi_{ij}^p} + \frac{1}{\sin^2 \varphi_{ik}^p} + \frac{1}{\sin^2 \varphi_{jk}^p} \right) + \sum_{p=1}^{8} \left(\frac{1}{\sin^2 \varphi_i^p} + \frac{1}{\sin^2 \varphi_j^p} + \frac{1}{\sin^2 \varphi_k^p} \right),$$

where angles $\varphi_{ij}^p, \varphi_{ik}^p, \varphi_{jk}^p, \varphi_i^p, \varphi_j^p, \varphi_k^p$ are shown in Fig. 10.13, and angles $\varphi_{ij}^p, \varphi_{ik}^p, \varphi_{jk}^p$ are introduced in (10.2). Further, we shall use shorthand notation φ for above angles.

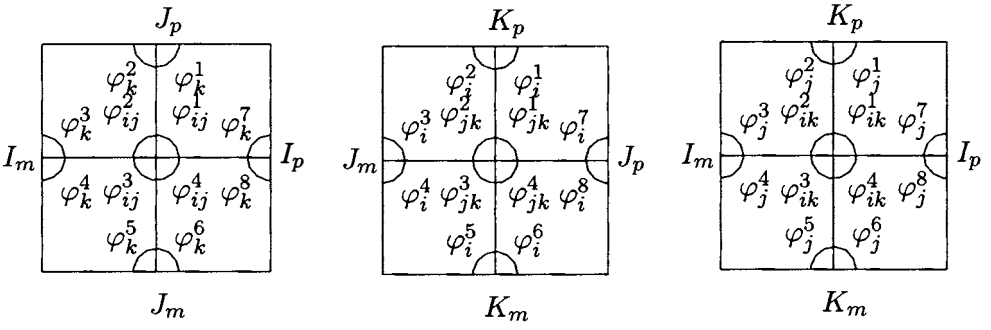


Figure 10.13: Scheme for definition of angles between coordinate lines.

If a sine of one of the angles φ in its absolute value is less than or equal to 0.01

$$|\sin \varphi| \leq 0.01,$$

then we put a barrier substituting the corresponding summand by 10^4, that is

$$\frac{1}{\sin^2 \varphi} = 10^4.$$

After computing the coordinate of all nodes at each iteration we compute the general functional D by formulas (10.1)—(10.4) with $A_A = 0$. Here, we also use an above barrier.

Formulas (10.1)—(10.4) do not contain directly the magnitudes of the volumes of tetrahedrons with the vertices of a hexahedral cells that are one sixth of the Jacobian of a trilinear mapping used for constructing a cell with such vertices (see chapter 9), however sines of angles between coordinate lines are expressed in terms of these values since the volume of tetrahedron is equal to the one third of the product of the area of a base of a tetrahedron and a height of a tetrahedron. The area of a base, according to the sine theorem, is expressed in terms of sines. By this, we do not allow the node to move in the direction characterized by the values of the Jacobian close to zero. This also prevents degenerate cell formation.

10.5.2 Computing Boundary Nodes

Algorithm 1. Fixed boundary nodes. In this algorithm, nodes on the boundary of the whole domain or its subdomain are given and fixed. The location of the boundary nodes are given by the location of the corresponding nodes of the initial grid.

Algorithm 2. Free boundary nodes. In this algorithm, nodes on the boundary of the domain are free. At each iteration, their location on each of the faces are found from the minimum of the functional D with $A_A = 0$. A boundary of the domain G is given by the distribution of the boundary nodes of the initial grid and boundary surface is defined as a surface obtained by the ruled surfaces of corresponding faces of boundary cells. To construct such a surface at each iteration, we keep coordinates of initial grid boundary nodes in three two-dimensional arrays for each of the six faces of the curvilinear hexahedron G. For each of the faces, we also introduce two two-dimensional arrays of values of parameters p and q defining the coordinates of grid nodes at each iteration for the initial parametrization of the face that is the parametrization of the surface woven of the ruled faces of cells of the initial grid.

For example, for the face $i = 0$ we keep three arrays of initial boundary node coordinates $x^0_{i_k}(0, i, j)$ shown in Fig. 10.14 *a* and arrays of parameters $p^n_{j,k}$ and $q^n_{j,k}$ corresponding to the coordinates of grid nodes lying on this surfaces for the current nth iteration (see Fig. 10.14 *b*). While constructing a surface of this face, we use the following parametrization

$$x_{i_k} = S_{i_k}(p, q) = x^0_{i_k}(0, p, q).$$

The surface is given by the following values of parameters

$$p^0_{j,k} = j, \quad q^0_{j,k} = k, \quad j = 0, 1, ..., M, \; k = 0, 1, ...L,$$

and corresponding values of coordinates

$$x_{i_k} = S_{i_k}(i, j) = x^0_{i_k}(0, i, j).$$

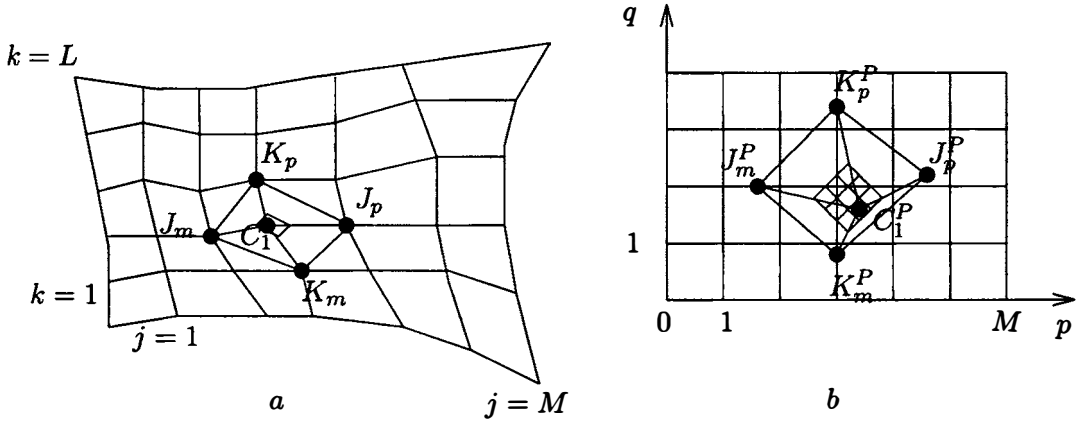


Figure 10.14: Interpolation technique for definition of face nodes: *a* physical space and *b* space of parameters.

To find the position of grid nodes we use the following interpolation technique. We fill up the functions of the surface parametrization by the ruled surfaces of the initial grid.

Suppose, we already have found the node H_{1jk}^{n+1} at nth iteration. To find the node H_{0jk}^{n+1}, $j = 1, ..., M-1$, $k = 1, ..., L-1$, we shall change the parameters and find for them corresponding points on the surface woven by the ruled faces of cells of the initial grid. Thus we shall construct admissible set of boundary nodes for minimization of the functional D. We choose a new node H_{0jk}^{n+1} as a point from the admissible set and bringing the functional D its minimal value. We utilize the following algorithm. First, we define the following points (see Fig. 10.14 a).

$$C_1 = H_{0jk}^n,\ J_m = H_{0j-1k},\ J_p = H_{0j+1k},\ K_m = H_{0jk-1},\ K_p = H_{0jk+1}.$$

Here J_m, J_k, K_m, K_p can be for the nth iteration or for the iteration $n+1$, it depends of whether this node is already computed or not. Above points belong to the surface of the face $i = 0$ woven of the ruled faces of the initial grid cells. Corresponding points in the space of parameters are denoted by the same notations with superscript P. They are defined in the following way

$$C_1^P = (p_{jk}^n, q_{jk}^n),\ J_m^P = (p_{j-1k}^n, q_{j-1k}^n),\ J_p^P = (p_{j+1k}^n, q_{j+1k}^n),$$

$$K_m^P = (p_{jk-1}^n, q_{jk-1}^n),\ K_p^P = (p_{jk+1}^n, q_{jk+1}^n).$$

Now in the space of parameters we define four points for $l = 3$

$$J_{m-}^P = C_1^P - (C_1^P - J_m^P)/l,$$
$$J_{p-}^P = C_1^P + (J_p^P - C_1^P)/l,$$
$$K_{m-}^P = C_1^P - (C_1^P - K_m^P)/l,$$
$$K_{p-}^P = C_1^P + (K_p^P - C_1^P)/l,$$

contained in the quadrangle $J_m^P K_p^P J_p^P K_m^P$ (see Fig. 10.14 b). In the quadrangle with the vertices at new points $J_{m-}^P, K_{p-}^P, J_{p-}^P, K_{m-}^P$, we consider 16 points

$$\begin{aligned} P_- &= (p_-, q_-) = J_{m-}^P + (K_{m-}^P - J_{m-}^P)a \\ &+ (K_{p-}^P - J_{m-}^P)b + (J_{p-}^P - K_{m-}^P - K_{p-}^P + J_{m-}^P)ab \\ &= J_{m-}^P(1-a)(1-b) + K_{m-}^P a(1-b) + K_{p-}^P b(1-a) + J_{p-}^P ab, \end{aligned}$$

where

$$a = h(i-1),\ b = h(j-1),\ h = 1/(l_1 - 1),\ l_1 = 4, i, j = 1, 2, ..., l_1.$$

When we find the points in the space of parameters, we define corresponding points on the surface of the face $i = 0$ in the following way. First, for each of 16 point, we define parameters

$$i_- = [p_-],\ j_- = [q_-],$$

where $[p_-]$, $[q_-]$ are integer parts of p_-, q_-. Then, we define points

$$X_{00} = H^0_{0j_-k_-},\ X_{01} = H^0_{0j_-k_-+1},$$

$$X_{10} = H^0_{0j_-+1k_-},\ X_{11} = H^0_{0j_-+1k_-+1},$$

as the corresponding nodes of the initial grid. By this, for each of 16 points corresponding to $[p_-]$, $[q_-]$ on the considered surface of curvilinear hexahedron G, we find corresponding cell faces (their vertices coincide with $X_{00}, X_{01}, X_{10}, X_{11}$) containing that point. For each point, it will be its own cell face. Then we find the points

$$\begin{aligned} C_2 &= X_{00} + (X_{10} - X_{00})(p_- - j_-) + (X_{01} - X_{00})(q_- - k_-) \\ &+ (X_{11} - X_{01} - X_{10} - X_{00})(p_- - j_-)(q_- - k_-). \end{aligned}$$

Between all points C_2, we find that point which gives the minimal contribution D_{0jk} to the functional D and choose it as the new boundary node H^{n+1}_{0jk} and corresponding values of parameters as new values p^{n+1}_{jk}, q^{n+1}_{jk}, that we keep in the arrays of parameters. Spaces between nodes and angles between coordinate lines used in computing the contribution to the functional D are shown in Fig. 10.12, Fig. 10.15 *b*, and Fig. 10.16.

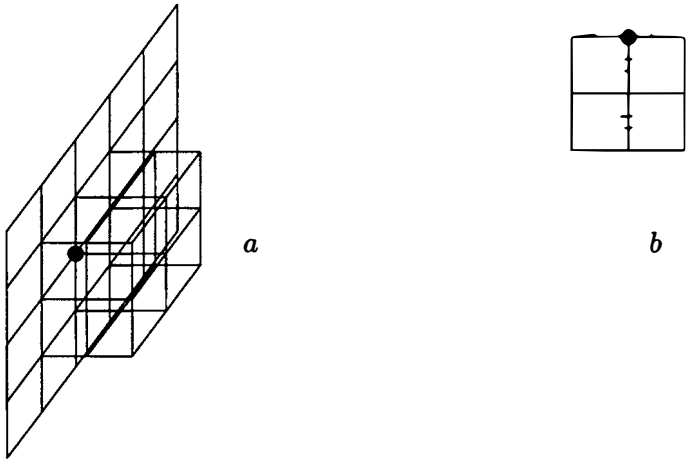


Figure 10.15: Main stencil for the face $i = 0$ *a* and spaces between nodes *b* for D_{0jk} for one of the directions.

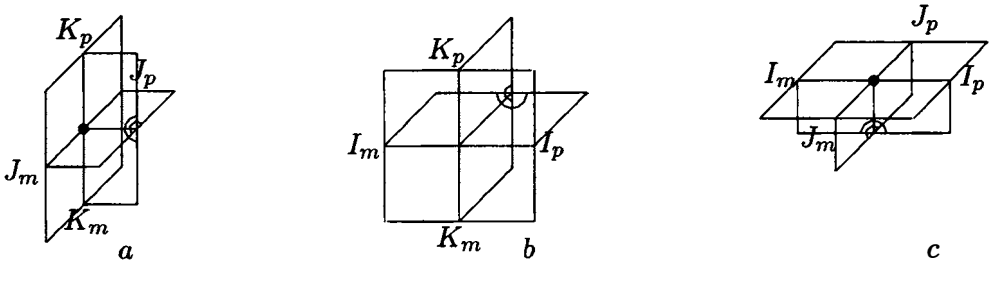


Figure 10.16: Angles between coordinate lines: a for face $i = 0$; b for face $j = 0$ and c for face $k = 0$.

Above we described the local minimization of the functional D on the faces (excluding edges).

Analogous minimization of the functional D is organized on the edges numbered from 1 up to 12. We shall describe this minimization on the edge $i = 0,\ k = 0$ numbered by 1 (see Fig. 10.17 a). We assume that at nth iteration, inner nodes on the faces are computed. We define the following points

$$C_1 = H_{0j0},\ \ J_m = H_{0j-10},\ J_p = H_{0j+10}.$$

Above points belong to the edge $i = 0,\ k = 0$. Corresponding points in the space of parameters are denoted by the same notations with superscript P. They are defined in the following way

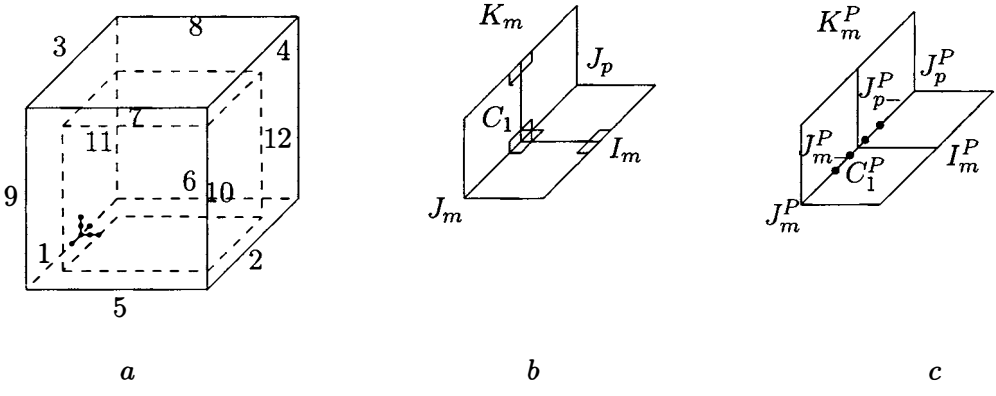
$$C_1^P = p_{j0}^n,\ J_m^P = p_{j-10}^n,\ J_p^P = p_{j+10}^n.$$

Again, in the space of parameters, we define two points for $l = 3$

$$J_{m-}^P = C_1^P - (C_1^P - J_m^P)/l,$$

$$J_{p-}^P = C_1^P + (J_p^P - C_1^P)/l$$

contained on the edge between points J_m^P and J_p^P (see Fig. 10.17 b, c).



a b c

Figure 10.17: Algorithm for the edge $i = 0, k = 0$: a spaces, b angles and c points P_-.

On the segment $[J_{m-}^P, J_{p-}^P]$, we consider 4 points (Fig. 10.17 c)

$$P_- = (p_-, q_-) = (p_-, 0) = J_{m-}^P(1 - a) + J_{p-}^P a, \tag{10.13}$$

where

$$a = h(j-1),\ h = 1/(l_1 - 1),\ l_1 = 4, j = 1, 2, 3, l_1.$$

When we found the points in the space of parameters, we define corresponding points on the edge $i = 0,\ k = 0$ in the following way. First, for each of 4 points, we define parameters

$$j_- = [p_-],$$

where $[p_-]$ is an integer part p_-. Then, we define points

$$X_{00} = H^0_{0j_-0},\ X_{10} = H^0_{0j_-+10},$$

as the corresponding nodes of the initial grid. By this, for each of 4 points corresponding to $[p_-]$ on the considered edge of curvilinear hexahedron G, we find corresponding edge of a cell containing the points

$$C_2 \ = \ X_{00} + (X_{10} - X_{00})(p_- - j_-).$$

Between all points C_2, we find that point which gives the minimal contribution D_{0j0} to the functional D and choose it as the new boundary node H^{n+1}_{0j0} and corresponding value of parameters as p^{n+1}_{j0} for the face $i = 0$ and q^{n+1}_{0j} for the face $k = 0$. Points P_-, spaces between nodes and angles between coordinate lines which are used in computing the contribution to the functional D are shown in Fig. 10.17 for the considered edge. On other edges, the algorithm is analogous.

So, at each iteration, when we found inner nodes, we find inner nodes on each of the faces and then nodes on the edges by the above procedure.

Algorithm 3. Conditions of orthogonality to faces. In this algorithm, we also represent the boundary of a domain as a surface woven by ruled faces of the initial grid. At each iteration, we compute inner nodes, and then define the position of nodes on the faces from the condition of orthogonality of grid lines to the boundary. Suppose, that at nth iteration we found the node $H^{n+1}_{1jk},\ j = 1, ..., M-1,\ k = 1, ..., L-1$, then we define the position of the nodes $H^{n+1}_{0jk},\ j = 1, ..., M-1,\ k = 1, ..., L-1$ on the face $i = 0$ from above orthogonality conditions by the following algorithm (see Fig. 10.18).

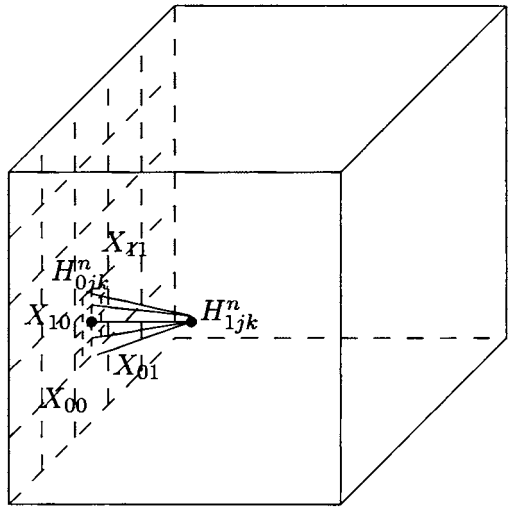


Figure 10.18: Algorithm 3 for the face $i = 0$.

We sort out all faces of initial grid cells forming the face $i = 0$. The algorithm of sorting is the following

$$X_{00} = H^0_{0j_1k_1}, \quad X_{01} = H^0_{0j_1k_1+1},$$

$$X_{10} = H^0_{0j_1+1k_1}, \quad X_{01} = H^0_{0j_1+1k_1+1},$$

where $j_1 = 0, 1, ...M-1$, $k_1 = 0, 1, ..., L-1$. On each cell face $X_{00}, X_{10}, X_{11}, X_{01}$ we change points

$$X_{p_lq_l} = (1-p_l)(X_{00}(1-q_l) + X_{01}q_l) + p_l(X_{10}(1-q_l) + X_{11}q_l)$$

for the following values of parameters $h = 1/(l_- - 1)$, $p_l = h(l-1)$, $q_l = h(l-1)$, $l = 1, ..., l_-$, (in test examples $l_- = 21$), and among points choose H^{n+1}_{0jk} as that point $X_{p_lq_l}$ which gives to the distance $|X_{p_lq_l}, H^{n+1}_{1jk}|$ its minimal value (shortest distance). On other faces of a curvilinear hexahedron G, we apply the similar algorithm.

Nodes on edges are found by the algorithm 3 of minimizing functional D on edges.

Algorithm 4. Conditions of orthogonality to faces and edges. Nodes on the faces (excluding those belonging to the edges) are found by the algorithm 3. On the edges, the nodes are found by the following algorithm similar to algorithm 3: from the conditions of orthogonality to edges of coordinate lines lying on faces. Suppose, on each face, we found inner nodes. Then we consider nodes H^{n+1}_{ijk} lying on the following coordinate lines

$$i = 0, j = 1, \quad i = 0, j = M-1, \quad i = N, j = 1, \quad i = N, j = M-1,$$

$$i = 1, k = 0, \quad i = N-1, k = 0, \quad i = 1, k = L, \quad i = N-1, k = L,$$

$$j = 1, k = 0, \quad j = M-1, k = 0, \quad j = 1, k = L, \quad j = M-1, k = L$$

shown in Fig. 10.19.

Using nodes H_{ijk}^{n+1} lying on this coordinate lines we find the nodes on the edges

$$i = 0, j = 0, \quad i = 0, j = M, \quad i = N, j = 0, \quad i = N, j = M,$$

$$i = 0, k = 0, \quad i = N, k = 0, \quad i = 0, k = L, \quad i = N, k = L,$$

$$j = 0, k = 0, \quad j = M, k = 0, \quad j = 0, k = L, \quad j = M, k = L,$$

correspondingly, by the following algorithm.

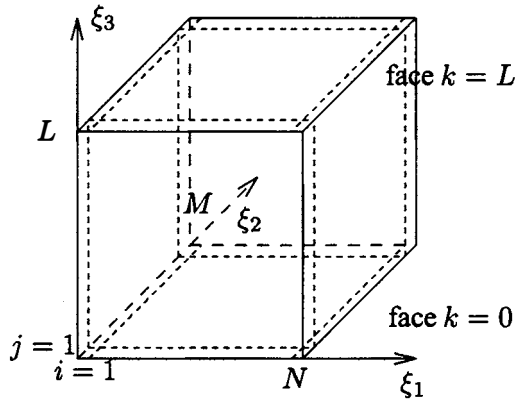


Figure 10.19: Coordinate lines from the algorithm 4.

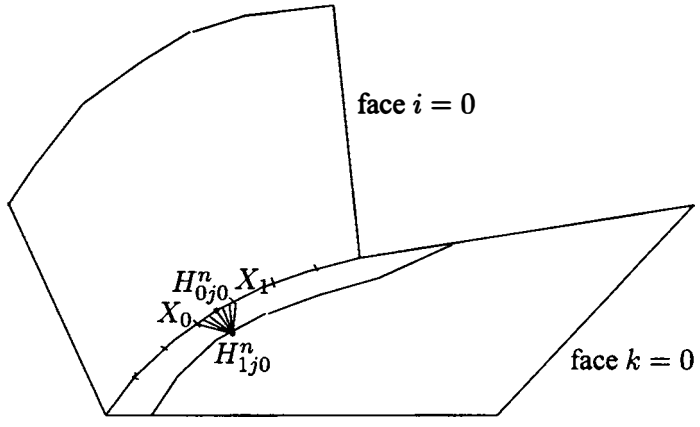


Figure 10.20: Algorithm 4 for the edge $i = 0, k = 0$.

Let us consider the line $i = 1, k = 0$ and nodes H_{1j0}^{n+1}, $j = 1, ..., M-1$ (see Fig. 10.20). For each node H_{1j0}^{n+1}, we find the node H_{0j0}^{n+1} from the conditions of orthogonality of the corresponding coordinate lines $j = j_1$, $j_1 = 1, ..., M-1$, $k = 0$ to the edge

$i = 0, k = 0$. The above condition of orthogonality will be fulfilled, if we choose the node H^{n+1}_{0j0k} as the point on the edge $i = 0, k = 0$ giving the distance $|H^{n+1}_{1j0}, H^{n+1}_{0j0}|$ its minimal value (shortest distance). We find this point sorting out the points on the edge. First, we sort edges of cells. Here we write out their end points

$$X_0 = H^0_{0j0}, \quad X_1 = H^0_{0j+10}, \; j = 0, ..., M-1,$$

and then, on the each cell edge, we sort points

$$X_{q_l} = X_0(1 - pq_l) + X_1 q_l$$

for the following values of parameters $h = 1/(l_- - 1)$, $q_l = h(l-1)$, $l = 1, ..., l_-$, (in test examples $l_- = 21$), and among points we choose H^{n+1}_{0j0} as that point X_{q_l} giving to the distance $|X_{q_l}, H^{n+1}_{1j0}|$ its minimal value. On other edges of a curvilinear hexahedron G, we apply the similar algorithm.

Algorithms 3 and 4 are also examples of free node algorithms. In the present computer code (algorithms 3 and 4), nodes on the boundary are reconstructed only for each fifth iteration. Note also, that condition of orthogonality is satisfied not exactly but approximately. It means that the angle between the edges or faces and the coordinate lines for complicated configurations of domains are usually close to $\pi/2$. It can be equal to $\pi/2$, if the point on the edge (face) giving the minimal distance coincides with the point giving, in fact, the shortest distance.

Algorithm 5. Reconstruction of selected nodes. To organize a reconstruction only of selected nodes we introduce a three-dimensional array $I(i, j, k)$. If $I(i, j, k) = 0$, the node i, j, k is reconstructed, if $I(i, j, k) = 1$, the node i, j, k is not reconstructed. The standard filling of this array is provided. It is the following filling

$$I(i, j, k) = 0, \; i = 0, ..., N, \; j = 0, ..., M, \; k = 0, ..., L$$

for the free boundary nodes and, for the fixed boundary nodes, boundary elements of this array are filled as follows

$$I(i, j, k) = 1, \; i = 0, N, \; j = 0, ..., M, \; k = 0, ..., L,$$

for left and right boundaries,

$$I(i, j, k) = 1, \; i = 0, ..., N, \; j = 0, M, \; k = 0, ..., L,$$

for front and back ones, and

$$I(i, j, k) = 1, \; i = 0, ..., N, \; j = 0, ..., M, \; k = 0, L,$$

for top and bottom ones. To choose the standard filling of the array or the particular one, we have to define an another additional parameter which is equal to zero for a standard filling and to one for a particular filling.

10.6 Test Examples

In Fig. 10.21, there are examples of the grids for the shell. These examples illustrate algorithms 1 and 2 by means of realization in algorithm 5. On the plane faces, nodes are fixed (algorithm 1) and, on spherical faces, nodes are free (algorithm 2). The initial grid is symmetrical and the number of nodes is $31 * 31 * 4$. The parameter $A_O = 0.3$. The optimal grid is obtained from the initial one for 10 iterations.

In Fig. 10.22, the reconstruction of the grid is carried out both on the boundary of the cube and inside the cube from the condition of orthogonality of the coordinate lines to the boundary (only faces), nodes on the edges are found from the minimum of the functional D (algorithm 3). The initial grid is constructed by means of the uniform grid for a cube inscribed in the given cube asymmetrically. Nodes on the edges are arranged uniformly, on the faces from the condition of orthogonality to faces. The number of iterations is equal to 601. The parameter $A_O = 0.1$. (For increasing the influence of the uniformity criteria, the parameter A_O is decreased and visa versa for decreasing the uniformity of a grid, A_O is increased. The parameter A_O is decreased and increased in the value of the order, for example, $A_O = 10^l$, $l = -3, -2, -1, 0, 1, 2$.)

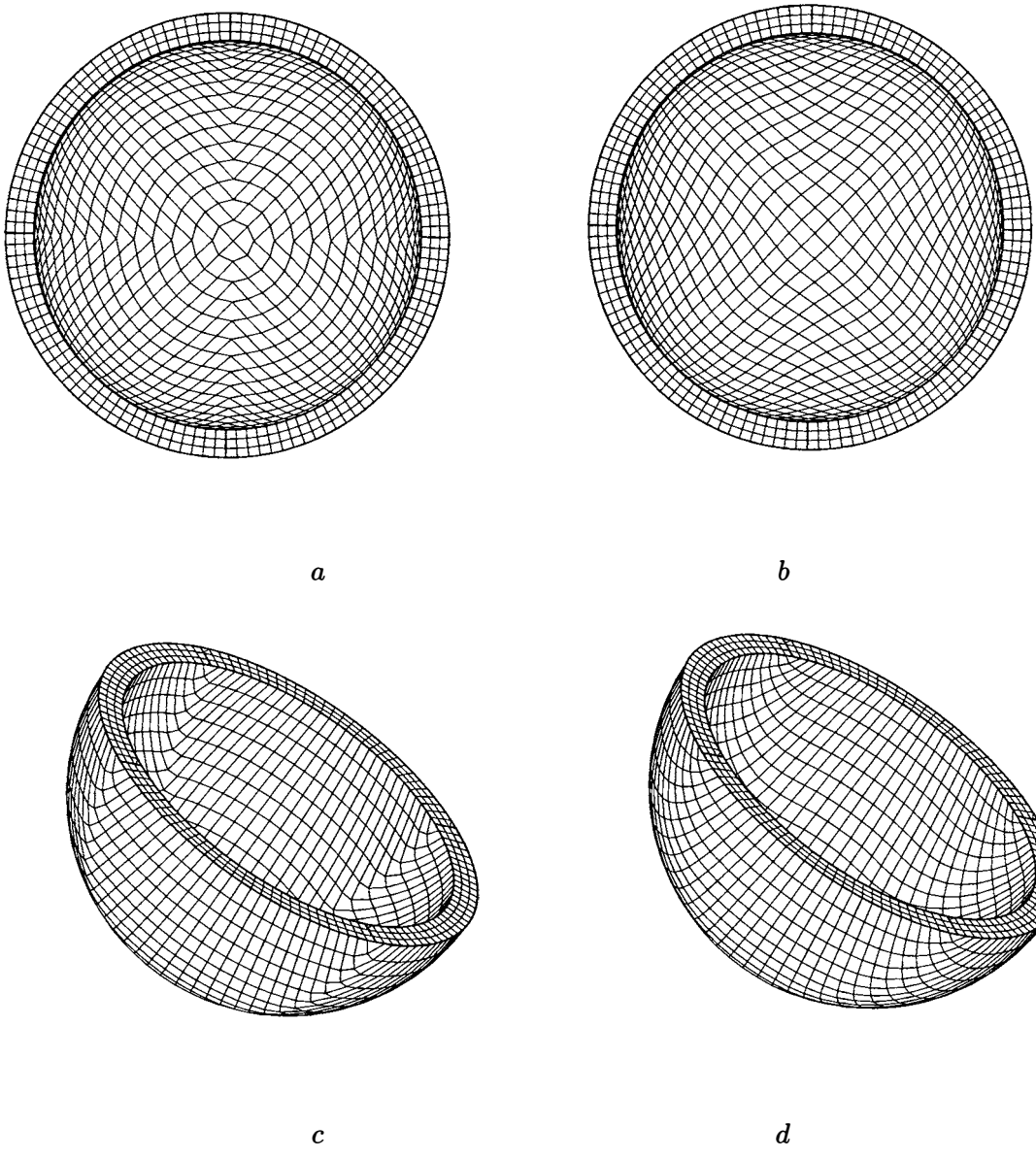
In the next examples (Fig. 10.23), the reconstruction of nodes is carried out in the semicylinder for fixed nodes on the boundary (algorithm 1), and from the condition of orthogonality of the coordinate lines to faces (algorithm 3) and of orthogonality of coordinate lines to faces and edges (algorithm 4). The number of iterations is equal to 101. The parameter $A_O = 0.001$ in the algorithm 1 (Fig. 10.23 *a*) and 3 (Fig. 10.23 *b*, *c*) and $A_O = 0.1$ in the case of algorithm 4 (Fig. 10.23 *d*, *e*).

Estimation of the grid quality. To estimate the quality of the generated grids, we used the values of the discrete functionals D_U and D_O. The less these values, the closer is the grid to the uniform one (with respect to the distance between adjacent grid points) or to the orthogonal one, respectively.

The another estimation of a grid quality is the ratio of the maximal and minimal linear sizes of cells over the whole computation domain. The linear size of a cell along one parametric direction (i, j or k) is defined as the ratio of the cell volume to the area of its mean cross-section in the direction orthogonal to the considered parametric direction. We estimate linear sizes for all directions and obtained the required value. This criterion was suggested by O.M.Kozyrev. The cell size is used in constraints imposed on the time step of the iteration procedure for solving physical problems of chapter 8. A grid has a good quality, if the ratio of the maximal and minimal linear size of cells over the whole computation domain is close to unity.

In considered tests, first we estimate whether the grid is degenerate or not. In Fig. 10.21, all grid cells besides 12 cells along 4 edges are nondegenerate. The cells along edges degenerate into prisms with triangular base (this is diagnosed by the computer code), such grid quality is allowed in computations described in chapter 8. Then we estimate D_U and D_A. For the initial grid in Fig.10.21, $D_U = 74$, $D_O = 63771$, and the ratio of the maximal and minimal linear sizes of cells over the entire domain is 2.5. For the optimal grid, we

have $D_U = 42$, $D_O = 62088$, and the ratio of the maximal and minimal linear sizes of cells over the entire domain is 3.2.



a *b*

c *d*

Figure 10.21: Grids for a shell:
a, c initial grid; b, d optimal grid.

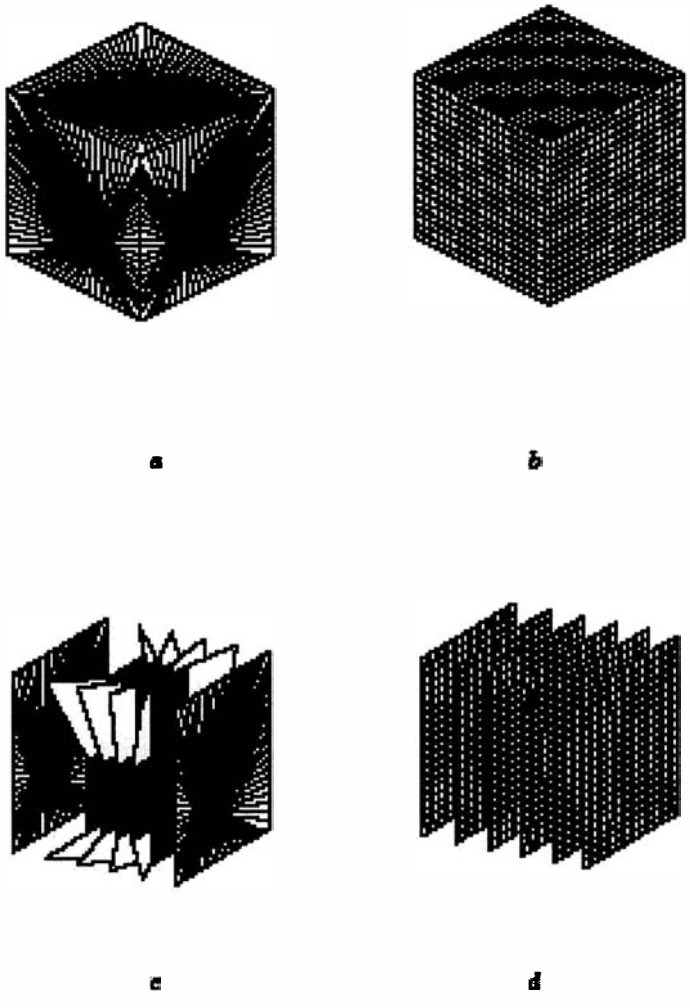
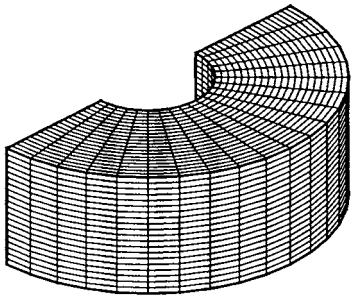
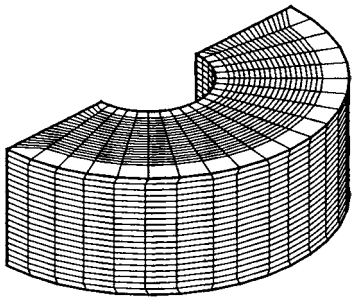


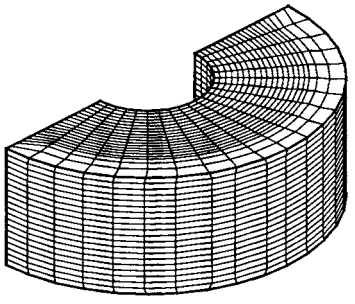
Figure 10.22: Reconstruction of a grid by algorithm 3:
a, c initial grid; b, d optimal grid.



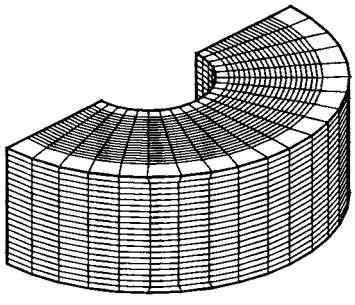
a



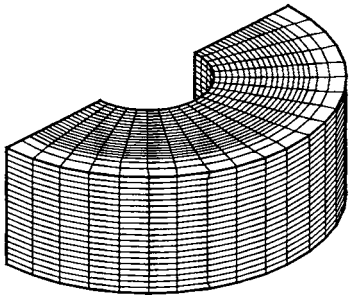
b



c



d



e

Figure 10.23: Reconstruction of a grid: *a* Fixed nodes;
b—*e* condition of orthogonality of coordinate lines to the boundary;
b—*c* algorithm 3; *d*—*e* algorithm 4;
b, *d* initial grids; *c*, *e* optimal grids.

In Fig. 10.22, initial and final grids are nondegenerate, for the initial grid in Fig. 10.22 a, c, $D_U = 656530$, $D_O = 265370$ and for the optimal grid in Fig. 10.22 b, d, we have $D_U = 2$, $D_O = 246980$. The ratio of maximal and minimal linear sizes of cells in this example is equal to 18.87 for the initial grid and 1.21 for the optimal grid in Fig. 10.22 b, d.

In Fig. 10.23, initial and final grids are also nondegenerate, for the initial grid in Fig. 10.23 b, $D_U = 8335$, $D_O = 258570$ and for the optimal grid in Fig. 10.23 c, we have $D_U = 273$, $D_O = 250140$. The ratio of maximal and minimal linear sizes of cells is equal to 15.7 for the initial grid and 6.5 for the optimal grid in Fig. 10.23 c. For figures d and e, above values are similar.

10.7 Conclusion

Suggested algorithms discovered themselves as efficient algorithms. They are applied in mathematical modelling by the computer code described in chapter 8 and are developing now according to its demands.

Acknowledgement

This work was supported by the Russian Foundation for Basic Research, project 02–01–00236.

Bibliography

[1] T. N. Bronina, I. A. Gasilova, and O. V. Ushakova, Algorithms for Three-Dimensional Structured Grids Generation. Zh. Vychisl. *Mat. Mat. Fiz.*, **6**, 2003, pp. 875-883.

[2] O. B. Khairullina, A. F. Sidorov, and O. V. Ushakova, Variational methods of construction of optimal grids. *Handbook of Grid Generation*, J. F. Thompson, B. K. Soni, and N. P. Weatherill, eds., CRC Press, Boca Raton, FL, 1999, pp. 36-1–36-25.

[3] A.F.Sidorov, O.B.Khairullina, and A.F.Khairullin. Parallel algorithms of generation of optimal multi-block-structed two-dimensional and three-dimensional grids of large size. *Numerical Grid Generation in Computational Field Simulation*, M. Cross, B. K. Soni and J. F. Thompson, J.Hauser, P.R.Eiseman eds., ISGG, MS, 1998, pp. 759–769.

[4] S. A. Ivanenko and A. A Charakhch'yan, Curvilinear grids of convex quadrilaterals, *USSR Comput. Maths. Math. Phys*, **28**(2), 1988, pp. 126–133.

[5] O. V. Ushakova. Algorithm of two-dimensional optimal grid generation. *Numerical Grid Generation in Computational Field Simulation* . B. K. Soni and J. F. Thompson, eds., Mississippi State University, Mississippi State, MS, 1996, pp. 37–46.

[6] G. P. Prokopov, Some general problems in constructing grid generation algorithms. Keldysh Institute of Applied Mathematics RAS, Preprint No. 98, Moscow, 1987.

Chapter 11

AN ALGORITHM OF CONSTRUCTING OPTIMAL THREE-DIMENSIONAL GRIDS IN DOMAINS OF THE "PIPELINE" TYPE

Irina A. Gasilova
Institute of Mathematics and Mechanics
Ural Branch of the Russian Academy of Sciences

In the present work, an algorithm of constructing optimal three-dimensional grids in domains of the "pipeline" type is suggested. Such configurations of the domains can arise while modelling the physical field phenomena by the computer code described in chapter 8 and in many other applications. The cross section of a "pipeline" is a simply connected two-dimensional domain G of the star type. The plane directrix T of the "pipeline" and the boundary Γ of the domain G consist of finitely many segments of straight lines and arcs of circles.

The idea of the algorithm is to construct a two-dimensional optimal grid in the cross section G. The latter uniformly moves along the directrix T and rotates through an angle γ depending on the geometry of the directrix T, forming thus a three-dimensional grid. In order to construct a two-dimensional optimal grid in the cross section G by the iterative method [1] we need to specify its initial approximation. As a basis we take the algorithm [2], which automatically generates an initial grid, using the technique of R-functions [3, 4], and automatically checks whether the two-dimensional domain G belongs to the star type. This results in an initial approximation of the two-dimensional grid without self-intersecting cells.

Complete three-dimensional grids for different "pipelines" are shown in Figs. 11.4 — 11.8.

11.1 Problem Formulation

The calculation and construction of three-dimensional grids are carried out in the right rectangular coordinate system in the space of variables (x, y, z). In the plane (x, y), a

two-dimensional simply connected domain G of the star type with boundary Γ consisting of finitely many segments of straight lines and arcs of circles is given. In the plain (y, z) the directrix T, which is a curve obtained by "gluing" finitely many segments of straight lines and arcs of circles together, is specified. The motion of the domain G along the directrix T forms a three-dimensional simply connected body Q. To construct a curvilinear grid, we consider the domain Q as a curvilinear hexahedron. Its vertices $A, B, C, D, A_1, B_1, C_1, D_1$ are arranged so that we moves along the boundary from $A(A_1)$ to $D(D_1)$ through the vertices $B(B_1)$ and $C(C_1)$ clockwise. The size of the three-dimensional grid is determined by the number of its nodes: N on the edges $AD(A_1D_1)$ and $BC(B_1C_1)$, M on the edges $AB(A_1B_1)$ and $DC(D_1C_1)$, and L on the edges AA_1, BB_1, CC_1, and DD_1.

It is required to generate a three-dimensional grid of the size $N * M * L$. To this end, we need

- to determine whether the domain G is of the star type;
- to generate in G an optimal two-dimensional grid with nodes (x_{ij}, y_{ij}), $i = 1, .., N$, $j = 1, .., M$, if the domain is of the star type;
- to generate in the domain Q an optimal three-dimensional grid with nodes $(x_{ijk}, y_{ijk}, z_{ijk})$, $i = 1, .., N$, $j = 1, .., M$, $k = 1, .., L$.

1.2 Description of the Algorithm

The proposed algorithm of constructing an optimal three-dimensional grid in the given domain Q consists of several independent parts.

1.2.1 Determining Whether the Domain G is of the Star Type

An indication that enables us to determine whether a two-dimensional simply connected domain G is of the star type is the existence in G of some starry domain G^*; in the general case, it is a continual set of all possible locations of the pole for which all segments connecting the pole with all boundary points of the domain G belong to this domain. It is known that the starry domain G^* is convex. Obviously, any convex domain is of the star type.

Checking whether the domain G is convex. The boundary Γ of the domain G can consist of segments of straight lines and arcs of circles. We represent every arc as a broken line and obtain a new boundary of the domain G consisting of N_q segments—polygon $\Gamma_1\Gamma_2...\Gamma_{N_q}$. If every interior angle of the polygon is less than or equal to π, then the polygon is convex. If at least one interior angle is greater than π, then the polygon is nonconvex. The interior angle $\angle\Gamma_\nu\Gamma_{\nu+1}\Gamma_{\nu+2}$ $(\nu = 1, 2, ..., N_q;\ \Gamma_{N_q+1} = \Gamma_1,\ \Gamma_{N_q+2} = \Gamma_2)$ is formed by rotation of the segment $\Gamma_\nu\Gamma_{\nu+1}$ about the point $\Gamma_{\nu+1}$ counter-clockwise until it coincides with the segment $\Gamma_{\nu+1}\Gamma_{\nu+2}$ (see Fig. 1.1a). Segments linking vertices Γ_ν and $\Gamma_{\nu+1}$ are

parts of oriented straight lines described by the equation

$$f_\nu(x,y) = (y_{\nu+1} - y_\nu)x + (x_\nu - x_{\nu+1})y + (x_{\nu+1}y_\nu - x_\nu y_{\nu+1}) = 0. \qquad (11.1)$$

To determine the value of the interior angle $\angle\Gamma_\nu\Gamma_{\nu+1}\Gamma_{\nu+2}$, we substitute the coordinates of the vertex $\Gamma_{\nu+2}$ into equation (11.1) and check the sign of the obtained expression [5]. The condition $f_\nu(x_{\nu+2}, y_{\nu+2}) > 0$ means that the vertex $\Gamma_{\nu+2}$ lies on the right of the straight line $\Gamma_\nu\Gamma_{\nu+1}$ and the interior angle is less than π. If $f_\nu(x_{\nu+2}, y_{\nu+2}) < 0$, then the vertex $\Gamma_{\nu+2}$ lies on the left of the straight line $\Gamma_\nu\Gamma_{\nu+1}$ and the interior angle is greater than π.

The polygon $\Gamma_1\Gamma_2...\Gamma_{N_q}$ will be convex and, therefore, the domain G will be of the star type if all vertices $\Gamma_{\nu+2}$ satisfy the condition $f_\nu(x_{\nu+2}, y_{\nu+2}) \geq 0$. For the pole of the star domain G we take the center of gravity of the convex polygon $\Gamma_1\Gamma_2...\Gamma_{N_q}$ with coordinates

$$x^* = \frac{1}{N_q}\sum_{j=1}^{N_q} x_j; \quad y^* = \frac{1}{N_q}\sum_{j=1}^{N_q} y_j .$$

If there exists at least one vertex $\Gamma_{\nu+2}$ for which $f_\nu(x_{\nu+2}, y_{\nu+2}) < 0$, then it is obvious that the polygon, as well as the domain G, is nonconvex.

Finding the starry domain. To decide whether a nonconvex domain belongs to the star type, we take as a basis the algorithm [2] that automatically determines whether the domain G is of the star type. This algorithm is based on simple geometrical constructions and uses the technique of R-functions [3, 4]. The gist of the algorithm is in the following.

Assume that a nonconvex domain G is of the star type. Then there exists a convex starry domain G^* inside the domain G (see Fig. 11.1*b*), which can be represented as the intersection of some set of half-planes U_ν containing G^* [6]. We transform the boundary Γ, consisting of segments and arcs of circles (convex or concave with respect to the domain G), in the following way. We replace every concave arc by two segments tangent to the arc at the endpoints and represent every convex arc as a broken line. As a result, we obtain a new boundary consisting of N_p segments, which are parts of oriented straight lines (11.1). The half-planes U_ν, ν=1,2,..,N_p, are determined by the inequalities $f_\nu(x,y) \geq 0$, where $f_\nu(x,y)$ is the expression from (11.1). Assume that the intersection of the half-planes U_ν is nonempty. We will consider it as a geometrical object whose analytical image is the function $f(x,y)$ given in the form of a united analytical expression and possessing the following properties:

$$f(x,y) = 0 \text{ on the boundary of } G^*; \quad f(x,y) > 0 \text{ inside } G^*;$$
$$f(x,y) < 0 \text{ outside } G^*.$$

We have such a possibility due to the technique of R-functions [4]:

$$f(x,y) = (...(f_1 \wedge_\alpha f_2) \wedge_\alpha f_3) \wedge_\alpha ...) \wedge_\alpha f_{N_p} = \bigwedge_{\nu=1}^{N_p}{}_\alpha\, f_\nu(x,y) \geq 0, \qquad (11.2)$$

where

$$f_\nu \wedge_\alpha f_{\nu+1} = \frac{1}{1+\alpha}\left(f_\nu + f_{\nu+1} - \sqrt{f_\nu^2 + f_{\nu+1}^2 - 2\alpha f_\nu f_{\nu+1}}\right)$$

is the R-conjunction operation ($0 < \alpha \leq 1$).

It is known that the intersection of finitely many half-planes U_ν is a convex polygon [6] (see Fig. 11.1*b*). The vertices T_j, $j = 1, 2, ..., N^*$, of this polygon belong to the set of intersection points of every straight line (11.1) and the remaining straight lines of the family of oriented straight lines. The coordinates of the intersection points are calculated by the formulas

$$x_{m,n} = \frac{(B_m C_n - B_n C_m)}{(A_m B_n - A_n B_m)}, \quad y_{m,n} = \frac{(A_n C_m - A_m C_n)}{(A_m B_n - A_n B_m)}, \tag{11.3}$$

where

$$A_m = y_{m+1} - y_m\,, \; B_m = x_m - x_{m+1}\,, \; C_m = x_{m+1}y_m - x_m y_{m+1}\,,$$

$$A_n = y_{n+1} - y_n\,, \; B_n = x_n - x_{n+1}\,, \; C_n = x_{n+1}y_n - x_n y_{n+1}\,,$$

$$m = 1, 2, ..., (N_p - 1)\,, \; n = (m+1), (m+2), ..., N_p\,.$$

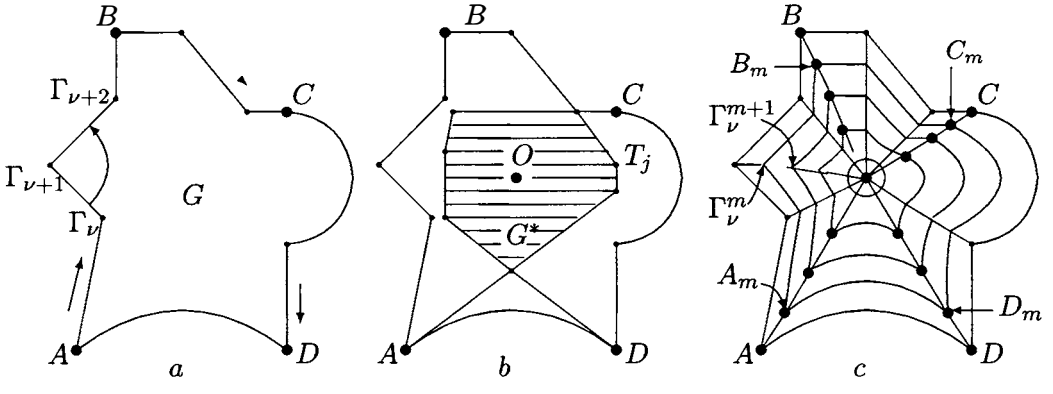


Figure 11.1: *a* Domain G with boundary Γ; vertices Γ_ν of the boundary, $\nu = 1, ..., 10$.
b starry domain G^* with the pole at point O; the starry domain, $j = 1, ..., N^*$, $N^* = 7$.
c vertices Γ_ν^m of the mth border line, here $m = 1, ..., (P+1)$, $P = 4$ ($N = M = 11$).

Successively substituting the coordinates of all intersection points (11.3) into inequality (11.2), we choose N^* points with coordinates (x_j^*, y_j^*), $j = 1, 2, ..., N^*$, for which $f(x_j^*, y_j^*) = 0$. If such points exist, then they determine the boundary of the starry domain

G^* and therefore the domain G is a star domain. For the pole of this star domain, we choose the point O with the coordinates

$$x^* = \frac{1}{N^*}\sum_{j=1}^{N^*} x_j^*, \quad y^* = \frac{1}{N^*}\sum_{j=1}^{N^*} y_j^* .$$

If $f(x_{m,n}, y_{m,n}) < 0$ for all points (11.3), then the domain G^* is the empty set and the domain G is complicated and not of the star type.

The subsequent construction of the initial grid approximation in the star domain G depends on the number of points T_j, $j = 1, 2, ..., N^*$, determining the boundary of the starry domain G^* and on the value of the ratio of the area of the domain $G^* = S_{G^*}$ to the area of the domain $G = S_G$.

The following three variants are possible:

(1) $N^* \leq 2$, i.e., the starry domain G^* consists of one point or the segment T_1T_2;

(2) $N^* \geq 3$, $S_{G^*}/S_G \leq \epsilon$;

(3) $N^* \geq 3$, $S_{G^*}/S_G > \epsilon$ $(\epsilon = 0.01)$.

The algorithm of constructing an initial grid approximation automatically chooses the third variant and rejects the first two because they result in a grid containing cells "sticking together" whose areas are close or equal to zero.

An Algorithm of Constructing an Initial Curvilinear Grid Approximation for Star Domains

The algorithm consists of two stages:

— border lines [2, 7], which are the base of a grid, are constructed inside the domain G;

— on every border line, a curvilinear quadrangle is determined and the grid nodes (x_{ij}, y_{ij}), $i = 1, .., N$, $j = 1, .., M$, are arranged on its sides.

Constructing border lines. The boundary Γ of the domain G consists of a finite number N_Γ of segments of straight lines and arcs of circles. We link the vertices Γ_ν of the boundary with coordinates (x_ν, y_ν), $\nu = 1, ..., N_\Gamma$, and the pole of the star domain (the point $O\,(x^*, y^*)$) and calculate the lengths

$$l_\nu = \sqrt{(x^* - x_\nu)^2 + (y^* - y_\nu)^2}\,, \; \nu = 1, ..., N_\Gamma,$$

of segments $O\Gamma_\nu$ on which points Γ_ν^m with coordinates

$$X_{\Gamma_\nu^m} = \frac{x_\nu + \lambda x^*}{1+\lambda}\,, \quad Y_{\Gamma_\nu^m} = \frac{y_\nu + \lambda y^*}{1+\lambda}\,,$$

where

$$\lambda = \frac{(m-1)\Delta_\nu}{l_\nu - (m-1)\Delta_\nu}\,,\ \Delta_\nu = \frac{l_\nu - R}{P}\,,\ R = \frac{l_{min}}{P+1}\,,\ l_{min} = \min l_\nu\,,$$

are uniformly arranged with an interval Δ_ν. Subsequently linking points $\Gamma_\nu^m\,(X_{\Gamma_\nu^m}, Y_{\Gamma_\nu^m})$ by a segment or broken line (if the corresponding part of the boundary is an arc of a circle), we obtain the mth border line, $m = 2, 3, ..., (P+1)$. The number of border lines inside the domain G is determined by the value $P = [\min(N, M)/2] - 1$, where N and M is the grid size. The family of all border lines together with the boundary Γ is a base for constructing an initial grid approximation (see Fig. 11.1*c*).

Arrangement of the grid nodes. To construct a two-dimensional grid, we consider the domain G as a curvilinear quadrangle. Its vertices A, B, C, D are arranged on the boundary Γ so that we move along the boundary from A to D through the vertices B and C clockwise. The size of the grid is determined by the number of its nodes: N on the sides BC and DA and M on the sides AB and CD.

For vertices A_m, B_m, C_m, D_m of the curvilinear quadrangle on the mth border line, we choose points Γ_ν^m of segments $O\Gamma_\nu$, $\nu = 1, 2, ..., N_\Gamma$, which link the pole of the star domain (the point $O\,(x^*, y^*)$) and the vertices A, B, C, D. The number of the grid nodes on the sides of the curvilinear quadrangle $A_mB_mC_mD_m$ equals

$$K_m = \begin{cases} M - 2(m-1) & \text{for } A_mB_m \text{ and } C_mD_m\,, \\ N - 2(m-1) & \text{for } B_mC_m \text{ and } D_mA_m\,, \end{cases}$$

$$m = 1, 2, ..., (P+1)\,.$$

We consider only one side, for example, A_mB_m, since the algorithm of arrangement of nodes is the same for every side of the curvilinear quadrangle $A_m\,B_m\,C_m\,D_m$. Knowing on the side A_mB_m the number of nodes K_m and the number K_{AB} of links $\Gamma_\eta^m\,\Gamma_{\eta+1}^m$, $\eta = 1, \ldots, K_{AB}$, we calculate the coordinates of the grid nodes by the formulas

$$X_{AB}^n = \frac{X_{\Gamma_\eta^m} + \lambda X_{\Gamma_{\eta+1}^m}}{1+\lambda}\,,\quad Y_{AB}^n = \frac{Y_{\Gamma_\eta^m} + \lambda Y_{\Gamma_{\eta+1}^m}}{1+\lambda}\,,$$

where

$$\lambda = \frac{n\Delta_\eta}{l - n\Delta_\eta}\,,\quad \Delta_\eta = \frac{l}{P-1}\,,\quad n = 1, 2, \ldots, (K_m - 2),$$

$$l = \sum_{\eta=1}^{K_{AB}} \sqrt{(X_{\Gamma_\eta^m} - X_{\Gamma_{\eta+1}^m})^2 + (Y_{\Gamma_\eta^m} - Y_{\Gamma_{\eta+1}^m})^2}\,.$$

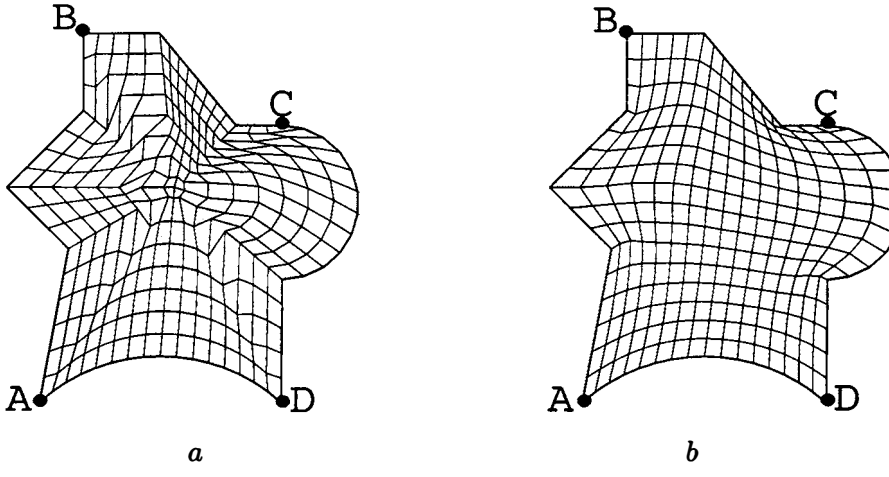
In particular,

- if when arranging nodes on the m_1th border line the number of vertices of the outline is large than that of the grid nodes, then the broken line is replaced by the straight line $A_{m_1}B_{m_1}$. It is verified whether they intersect. If there is an intersection, then all vertices A_j, B_j, C_j, D_j, $j = m_1, (m_1+1), ..., (P+1)$, are shifted along the corresponding segments $O\Gamma_\nu$ closer to the point $O(x^*, y^*)$ and the coordinates of the grid nodes on A_jB_j are calculated by the same formulas;

- if $N = M$ is an odd number, then the point $O(x^*, y^*)$—the pole of the star domain—is placed instead of the lacking node;

- if $N < M$ and N is an odd number or $M < N$ and M is an odd number, then the lacking nodes are additionally calculated by the linear interpolation formulas using the coordinates of the grid nodes on the sides of the curvilinear quadrangle $A_{P+1}, B_{P+1}, C_{P+1}, D_{P+1}$.

Arranging in this way nodes on all sides of the curvilinear quadrangle A_m, B_m, C_m, D_m, $m = 1, 2, ..., (P + 1)$, we obtain the required initial approximation of the curvilinear grid of the size $N * M$ in the domain G (see Fig. 11.2*a*).



a *b*

Figure 11.2: *a* initial grid, *b* optimal grid, $N = M = 17$.

Optimization of the Initial Grid Approximation

The generation of a curvilinear optimal two-dimensional grid in the domain G using an initial approximation is carried out by the algorithm "LADA" [1], which is based on a special minimization procedure of the functional of the quality of a grid. The functional to be minimized is represented as the sum of the measures of closeness of a curvilinear grid to a uniform and orthogonal grids and can be found in chapter 6, formula (6.2). The generalization of this functional for three-dimensional case is considered in chapter 10. As a result, we obtain an optimal two-dimensional grid with nodes (x_{ij}, y_{ij}), $i = 1, .., N$, $j = 1, .., M$, in the domain G (see Fig. 11.2*b*).

11.2.2 Generation of an Optimal Three-Dimensional Grid

The optimality criterion of a three-dimensional grid is the closeness of the grid to a uniform and orthogonal one (chapter 10) and nondegeneracy of cells (chapter 9). This criterion is a base for the algorithm of generation of an optimal two-dimensional grid in the domain G and plays a key role in geometrical construction of three-dimensional grids in domains of the "pipeline" type.

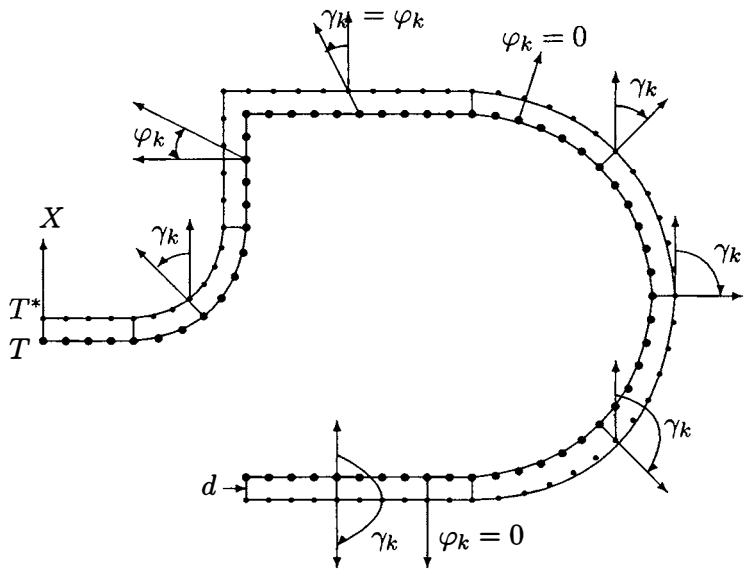


Figure 11.3: directrix T (T^*) [6 links]; $k = 1, ..., L$ ($L = 61$)

The directrix T, lying in the plane (y, z), is composed of N_T "links", which may be segments of straight lines and arcs of circles (when constructing the directrix T, one should take into account that segments of straight lines and arcs of circles join tangentially). The length of each link is calculated and summarized, which results in the length D_T of the directrix T. The nodes (y_k, z_k), $k = 1, ..., L$, on the directrix T are arranged with the equal step $h = D_T/(L-1)$. Simultaneously, the number of nodes n_μ, $\mu = 1, ..., N_T$, is calculated on every link. To determine the angle γ_k between the plane (x, y) and the directrix T, we carry out some geometrical construction at every node (y_k, z_k), $k = 1, ..., L$.

Similar to the directrix T, a new directrix T^*, consisting of N_T links, is constructed parallel to T at a distance d (see Fig. 11.3). Uniformly arranging nodes on every link of the new directrix T^* in accordance with n_μ, $\mu = 1, ..., N_T$, we calculate their coordinates (y_k^*, z_k^*), $k = 1, ..., L$. Linking (y_k, z_k) and (y_k^*, z_k^*), we obtain directed segments of straight lines and, with the help of them, calculate the angles γ_k, $k = 1, ..., L$, between the plane (x, y) and the directrix T at every node and the angles φ_k, $k = 1, ..., L$, of deviation of the directed segment from the perpendicular to the link which contains the given node.

Thus, a three-dimensional domain (see Fig. 11.4) is obtained as a result of the uniform

motion of the domain G, lying in the plane (x, y), along the directrix T with the step h and the rotation of the plane (x, y) about the x-axis at every node (y_k, z_k) through the angle γ_k, $k = 1, ..., L$, taking into account the value of the angle φ_k, $k = 1, ..., L$:

$$
\begin{aligned}
x_{ij\,k} &= x_{ij}, \\
y_{ij\,k} &= y_k + (y_{ij} \cos\gamma_k - z_{ij} \sin\gamma_k) \setminus \cos\varphi_k, \\
z_{ij\,k} &= z_k + y_{ij} \sin\gamma_k + z_{ij} \cos\gamma_k,
\end{aligned}
$$

where $i = 1, ..., N$, $j = 1, ..., M$.

The calculations result in a matrix of the node coordinates of a three-dimensional optimal grid of the size $N * M * L$:

$$(x_{ij\,k}, y_{ij\,k}, z_{ij\,k}) \ , \ i = 1, ..., N, \ j = 1, ..., M, \ k = 1, ..., L.$$

11.3 Conclusion

Minimum input information, automation of computations, high speed and graphic visualization of the process of the grid generation enabled us to carry out numerous test calculations of three-dimensional grids [8] and come to the conclusion that the algorithm described above makes it possible to calculate an optimal grid practically for any three-dimensional domain of the "pipeline" type with section G in the form of a star domain. Several examples of three-dimensional grids are shown in Figs. 11.4 — 11.8.

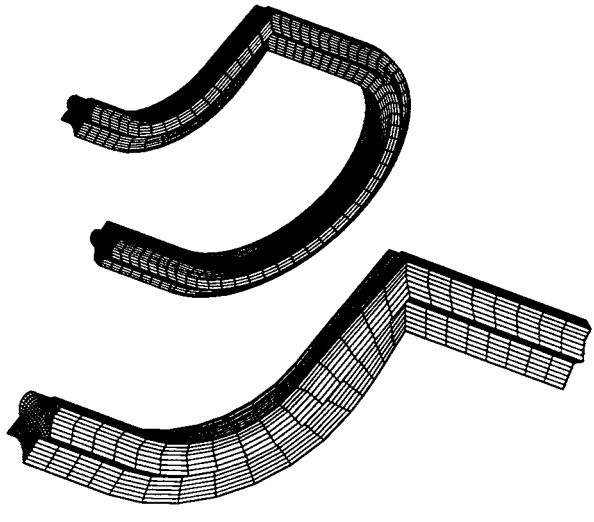


Figure 11.4:

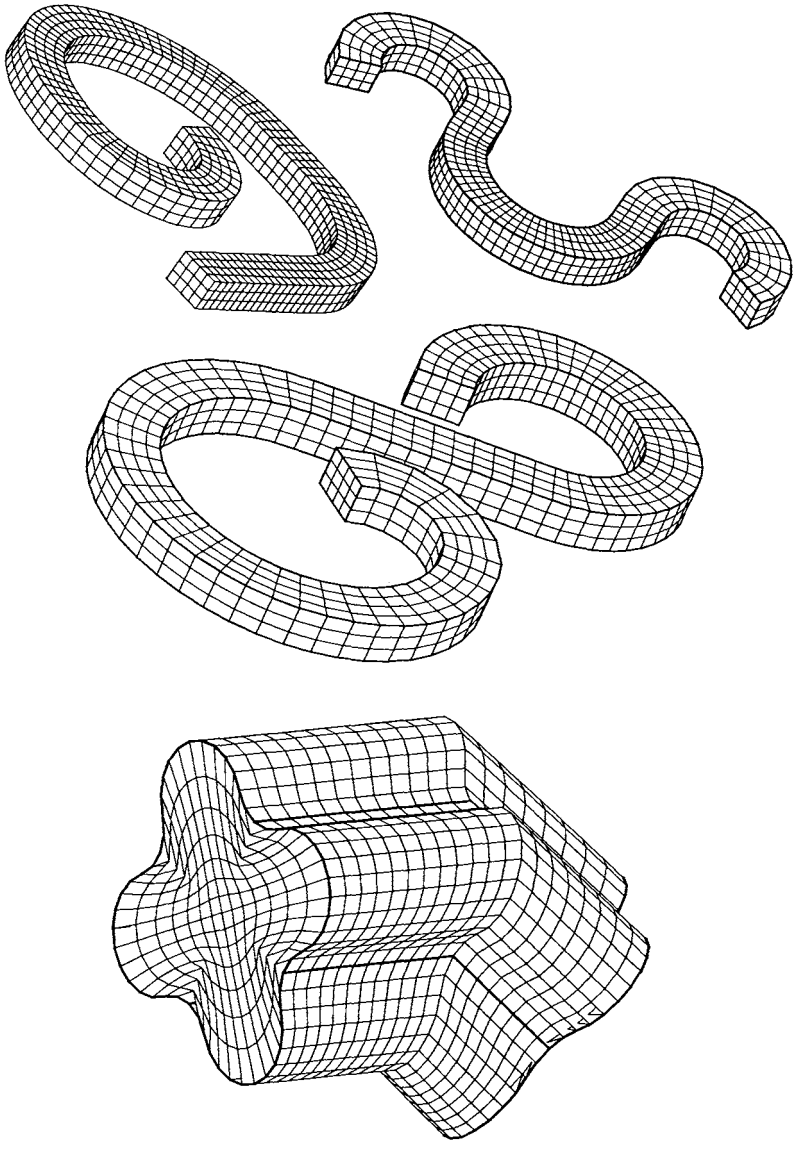


Figure 11.5:

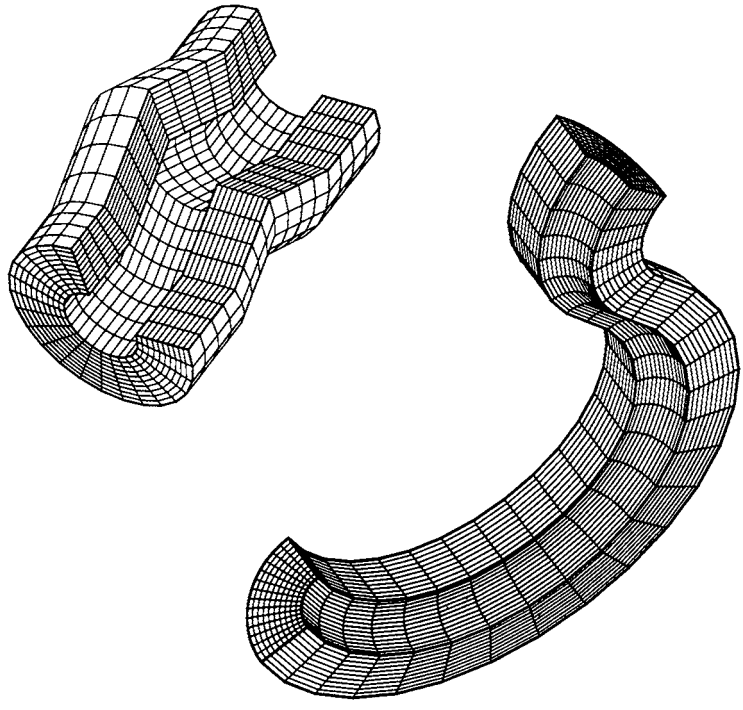


Figure 11.6: The cross sections of "pipelines" shown in this figure, are not of star type, but algorithm provides the possibility to generate grids for the domains of these types too.

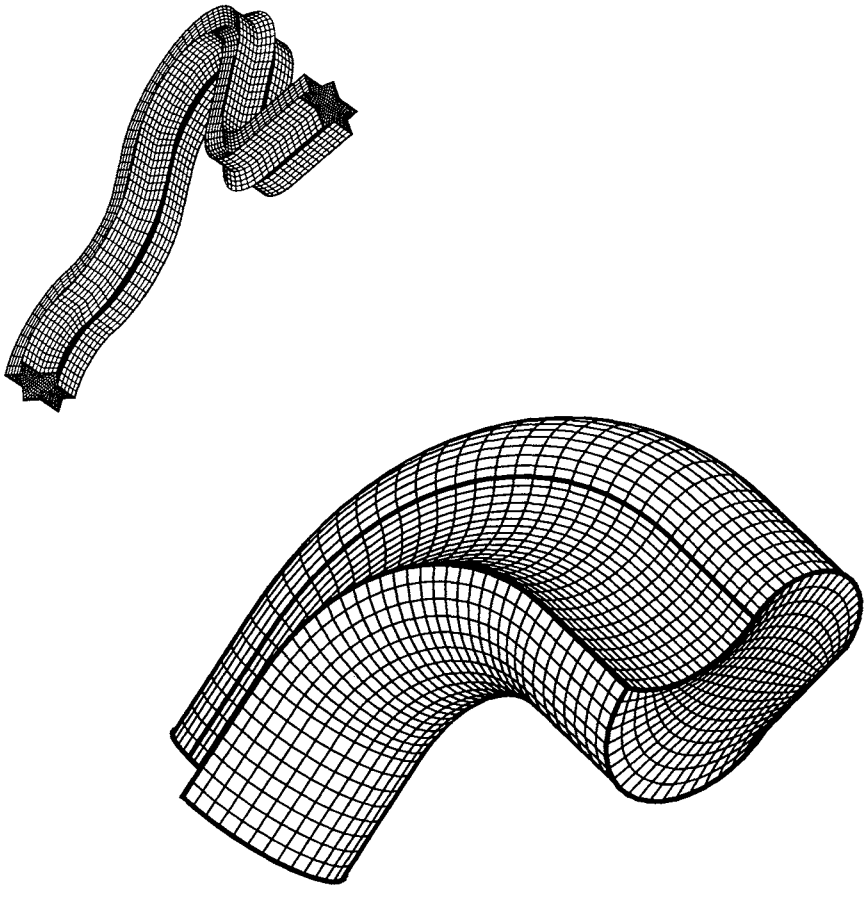


Figure 11.7:

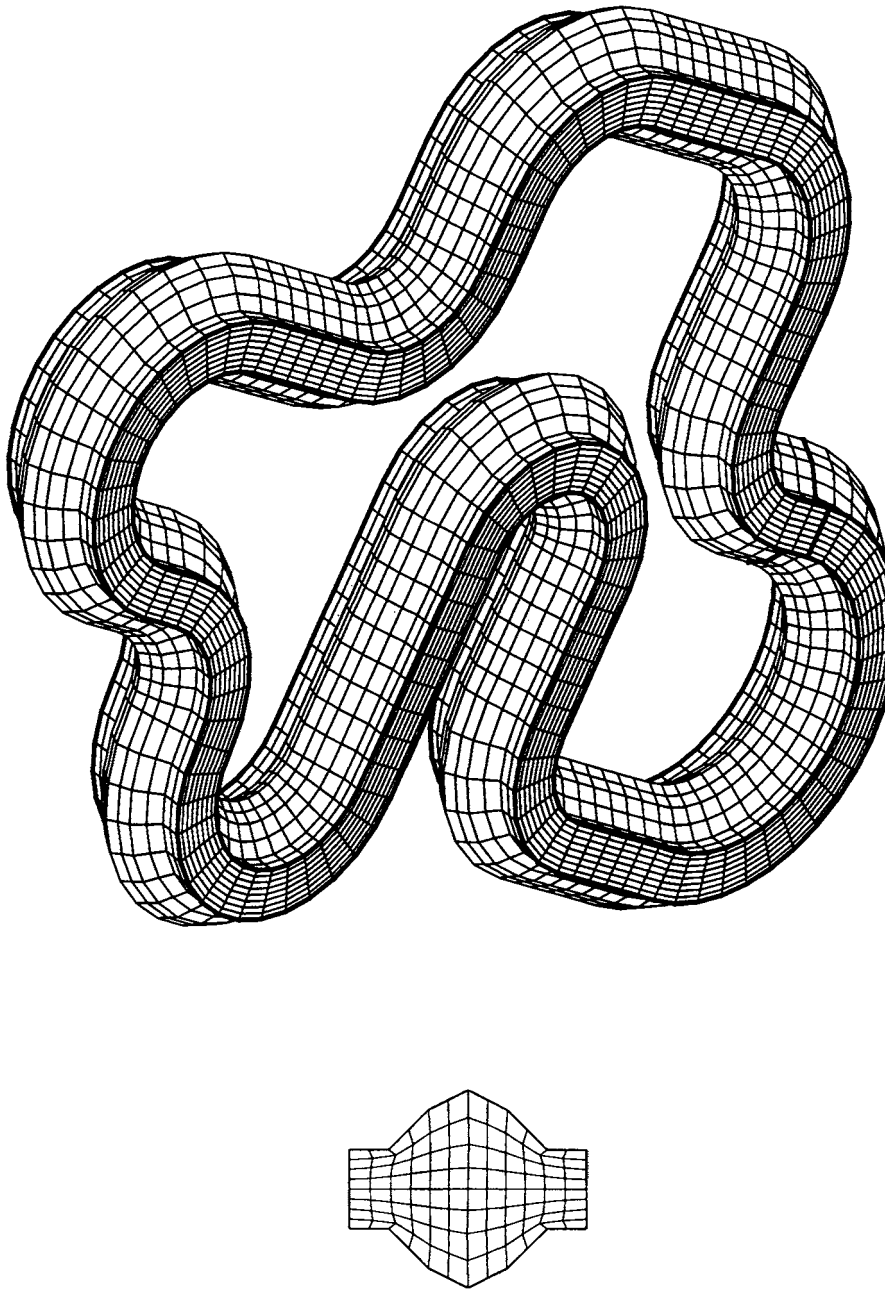


Figure 11.8:

Acknowledgment

This work was supported by the Russian Foundation for Basic Research, project 02–01–00236.

Bibliography

[1] Ushakova, O.V., Algorithm of two-dimensional optimal grid generation, *Numerical Grid Generation in Computational Field Simulation*, B. K. Soni and J. F. Thompson, eds., Mississippi State University, Mississippi State, MS, 1996, pp. 37–46.

[2] Gasilova, I.A., An algorithm of automatic generation of the initial approximation of a curvilinear grid for the star type domains, *Voprosy Atomnoi Nauki i Tekhniki. Ser.: Mat. Model. Fiz. Protsessov*, 1994, issue 3, pp. 33–40.

[3] Rvachyov, V.L., *Geometrical applications of the algebra of logics*, Tekhnika, Kiev, 1967.

[4] Rvachyov, V.L., *The theory of R-functions and some of its applications*, Naukova Dumka, Kiev, 1982.

[5] Stoyan, Yu. G. and Gil', N.I., *Methods and algorithms of arrangement of plane geometrical objects*, Naukova Dumka, Kiev, 1976.

[6] Boltyanskii, V.G. and Yaglom, I.M., *Convex figures and bodies: Encyclopaedia of elementary mathematics*, Nauka, Moscow, 1966, vol. 5, pp. 181–269.

[7] Sidorov, A.F. and Shabashova, T.I., A method of calculating optimal difference grids for multi-dimensional domains, *Chislennye metody mekhaniki sploshnoi sredy*, Novosibirsk, 1981, vol. 12, no. 5, pp. 106–123.

[8] Bronina, T.N., Gasilova, I.A., and Ushakova, O.V., Algorithms of constructing three-dimensional structured grids, *Zh. Vychisl. Mat. Mat. Fiz.*, 2003, vol. 43, no. 6, pp. 875–883.

Chapter 12

CONSERVATIVE REMAPPING ON HEXAHEDRAL MESHES

Boris N. Azarenok
Dorodnicyn Computing Center of the Russian Academy of Sciences

An algorithm for the first-order interpolation of the hydrodynamical parameters from the old hexahedral mesh onto the new hexahedral mesh is presented. Conservative remapping is reduced to determining the volume of the overlapping figure between the old mesh cells and new mesh cells. The ruled cell of the hexahedral mesh is substituted for two dodecahedrons with planar faces, and, thus, the problem of constructing the overlapping figure between the dodecahedrons is considered. For the underlying new mesh cell, the optimal algorithm of selecting the old mesh cells with nontrivial intersection is suggested. Interpolation error estimate is performed. Examples of remapping are presented.

12.1 Introduction

The task of remapping is encountered when at hydrodynamical modeling we need to pass from modeling on one mesh to another grid. For example, in the frame of arbitrary Lagrangian-Eulerian calculations we may run into the problem of grid tangling. To avoid breaking one should interpolate the flow parameters from the "bad" old mesh onto the "good" new one allowing to continue simulation. Particularly cases require the remapping algorithm to be conservative. It implies that amount of the conserved quantity in the same local volume must remain unchangeable after the interpolation procedure. We focus on the spatial case of hexahedral meshes.

In the second-order remapping algorithm, suggested in [2], the intersection between the ruled cells is considered. They extract the intersection curves in $\mathbb{R}^3$ by solving a differential equations which have singularities in the regions where some faces of the old and new cells are close and nearly parallel. In [3] the ruled cell is replaced with the 24-faceted triangular polyhedron, a tetrakis hexahedron (TH). The TH is constructed by adding one point at the center of every of six cell faces. The ruled face is replaced to four triangles. The TH volume

is the same as the hexahedron volume. The TH is divided into 48 tets and one seeks the intersection volumes between every "new" tetrahedron with every "old" one. Note that some tet volumes may be negative meanwhile the hex cell is non-degenerate, however this possibility is not discussed in [3].

We present the first-order remapping algorithm where a hex cell is substituted for two dodecahedrons with planar facets. The determination of the overlapping figure between two dodecahedrons is reduced to constructing the intersection polygonal line $\mathcal{L}_{on}$ between the boundaries of these 12-faceted cells. The polygonal line $\mathcal{L}_{on}$ is a closed contour, every segment of which is the intersection line between two triangular facets. The overlapping figure is a polyhedron with planar facets. Each its facet is the polygon, carved by $\mathcal{L}_{on}$ on a triangular facet of the dodecahedron. A union of all such polygons is the overlapping figure boundary. The overlapping figure volume is calculated through the surface integral over its boundary. For the particular cell of the new mesh, we apply the optimal filtering algorithm of the old grid cells so as to proceed only old cells with nontrivial intersection.

12.2 Problem Formulation

Let two regular non-folded grids, the old ω_o and new ω_n, be given in a domain $G \in \mathbb{R}^3$. On the mesh ω_o, the values of the conservative hydrodynamical parameters[1] are defined. We need to remap these parameters onto the mesh ω_n. Each mesh consists of hexahedral cells (also referred to as ruled or hex cells), specified by the trilinear transformation in space $\mathbb{R}^3$, see Fig. 12.1,

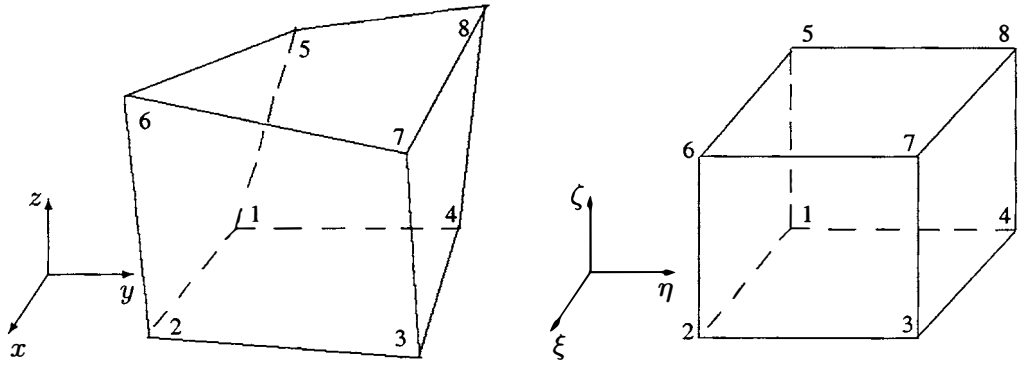


Figure 12.1: The ruled cell in physical space (x, y, z) to which the unit cube $I^3=\{(\xi, \eta, \zeta): 0\leq\xi, \eta, \zeta\leq 1\}$ from parametric space (ξ, η, ζ) is transformed via (12.1).

$$\begin{aligned} \boldsymbol{r} = (1-\zeta)\{\xi[(1-\eta)\boldsymbol{r}_2 + \eta\boldsymbol{r}_3] + (1-\xi)[\eta\boldsymbol{r}_4 + (1-\eta)\boldsymbol{r}_1]\} \\ + \zeta\{\xi[(1-\eta)\boldsymbol{r}_6 + \eta\boldsymbol{r}_7] + (1-\xi)[\eta\boldsymbol{r}_8 + (1-\eta)\boldsymbol{r}_5]\}\,, \end{aligned} \tag{12.1}$$

[1]The conservative remapping may be used in numerical modeling of another physical problem as well.

where $\boldsymbol{r}_i=(x_i, y_i, z_i)$, $i=1,\ldots,8$ are the coordinates of the cell vertices; ξ, η, ζ specify the unit cube $I^3=\{(\xi,\eta,\zeta) : 0 \le \xi,\eta,\zeta \le 1\}$ in parametric space $\mathbb{R}^3$. The lateral faces of the hexahedron in general are ruled surfaces. For instance, the face 1234 ($\zeta=0$) is given by

$$\boldsymbol{r} = \xi[(1-\eta)\boldsymbol{r}_2 + \eta\boldsymbol{r}_3] + (1-\xi)[\eta\boldsymbol{r}_4 + (1-\eta)\boldsymbol{r}_1] \,. \tag{12.2}$$

Remapping is conservative, i.e., amount of the conserved quantity in the same local volume must remain invariable after interpolation. For instance, remapping of the mass is given by the formula

$$m_n = \int_{V_n} \rho_o(\boldsymbol{r}) dV \,, \tag{12.3}$$

where $\rho_o(\boldsymbol{r})$ is the known density distribution on the mesh ω_o, V_n is the volume occupied by a particular cell of ω_n, and m_n is the resulting mass of this new cell. Therefore, the average density in the new cell is

$$\rho_n = \frac{m_n}{V_n} \,.$$

Further we will use the mass m as the conservative parameter to be remapped, meaning that it can represent any other conserved quantity, such as the momentum components, total energy, etc. Since the density $\rho_o(\boldsymbol{r})$ is specified locally, i.e., in every cell of ω_o a particular density distribution is given, so as to calculate the integral (12.3) one needs to determine all nontrivial intersection domains between the underlying new cell with old cells. We consider the first-order remapping, therefore, $\rho_o(\boldsymbol{r})$ is the piece-wise constant function, and it is necessary to determine the volume of the intersection domains.

Construction of the overlapping domain between two hexahedrons is rather cumbersome problem, see [2]. In [4], by straightforward evaluating the volume integral it was shown that the ruled cell volume is equal to one-half of a sum of two dodecahedron volumes with planar faces, which have the same vertices, see Fig. 12.2. Each dodecahedron consists of five tetrahedrons, four corner at vertices and one internal. internal.

Noting the results of [4], instead of the ruled cell we will use the dodecahedrons. It is more convenient to calculate a domain volume using the surface integral, since integration over the planar faces is rather an easy procedure.

We will derive the formula of the dodecahedron volume applying the surface integral. In vector analysis, we know the formula for calculating the domain volume

$$V = \frac{1}{3}\int\int_S \boldsymbol{r}\cdot d\boldsymbol{S} = \frac{1}{3}\int\int_S \boldsymbol{r}\cdot\boldsymbol{n}\, dS \,, \tag{12.4}$$

where S is the domain surface, $\boldsymbol{n}$ is the outward unit vector normal to S. Instead of the ruled cell, depicted in Fig. 12.1, we consider the dodecahedron where, for instance, the ruled face 1234 is replaced by two triangles 124 and 234, see Fig. 12.2 *a*. Let us calculate the contribution of these two triangular facets to the dodecahedron volume. The triangle 124 is defined

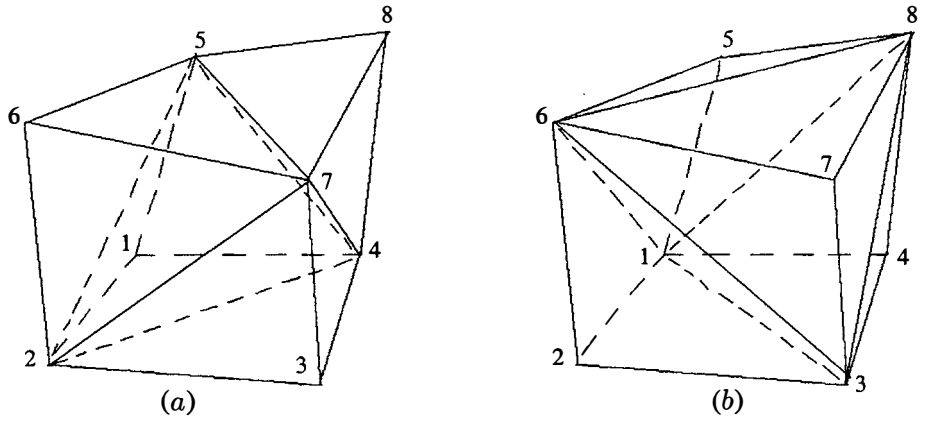


Figure 12.2: Pair of dodecahedrons, for which one-half of a sum of volumes is equal to the ruled cell volume. Each dodecahedron consists of five tetrahedrons.

by the linear transformation of the parametric triangle $\triangle^2=\{(\xi,\eta) : 0\le\xi,\eta;\ \xi+\eta\le 1\}$ as follows

$$\boldsymbol{r} = (\boldsymbol{r}_2 - \boldsymbol{r}_1)\xi + (\boldsymbol{r}_4 - \boldsymbol{r}_1)\eta + \boldsymbol{r}_1 ,$$

and the element of the triangular surface is

$$d\boldsymbol{S} = (\boldsymbol{r}_{14}\times\boldsymbol{r}_{12})d\eta d\xi = (\boldsymbol{r}_4 - \boldsymbol{r}_1)\times(\boldsymbol{r}_2 - \boldsymbol{r}_1)d\eta d\xi .$$

One can readily obtain that the scalar product of the vectors $\boldsymbol{r}$ and $d\boldsymbol{S}$ is

$$\boldsymbol{r}\cdot d\boldsymbol{S} = (\boldsymbol{r}_1\boldsymbol{r}_2\boldsymbol{r}_4)d\eta d\xi ,$$

where $(\boldsymbol{r}_1\boldsymbol{r}_2\boldsymbol{r}_4) = \boldsymbol{r}_1\cdot(\boldsymbol{r}_2\times\boldsymbol{r}_3)$ is the triple product. Therefore, the contribution of the facet 124 to the dodecahedron volume is given by

$$V_{124} = \frac{1}{3}\int\!\!\int_{S_{124}} \boldsymbol{r}\cdot d\boldsymbol{S} = \frac{1}{3}(\boldsymbol{r}_2\boldsymbol{r}_1\boldsymbol{r}_4)\int_0^1\int_0^{1-\xi} d\eta d\xi = \frac{1}{6}(\boldsymbol{r}_2\boldsymbol{r}_1\boldsymbol{r}_4) . \qquad (12.5)$$

Accordingly for the triangle 234 we have

$$\boldsymbol{r} = (\boldsymbol{r}_2 - \boldsymbol{r}_3)\xi + (\boldsymbol{r}_4 - \boldsymbol{r}_3)\eta + \boldsymbol{r}_3 ,$$
$$d\boldsymbol{S} = (\boldsymbol{r}_{32}\times\boldsymbol{r}_{43})d\eta d\xi = (\boldsymbol{r}_2 - \boldsymbol{r}_3)\times(\boldsymbol{r}_4 - \boldsymbol{r}_3)d\eta d\xi ,$$

and the contribution of this facet to the dodecahedron volume is given by

$$V_{234} = \frac{1}{3}\int\!\!\int_{S_{234}} \boldsymbol{r}\cdot d\boldsymbol{S} = \frac{1}{3}(\boldsymbol{r}_4\boldsymbol{r}_3\boldsymbol{r}_2)\int_0^1\int_0^{1-\xi} d\eta d\xi = \frac{1}{6}(\boldsymbol{r}_4\boldsymbol{r}_3\boldsymbol{r}_2) .$$

Two facets 124 and 234 provide the contribution

$$V'_{1234} = \frac{1}{6}(\boldsymbol{r}_2\boldsymbol{r}_1\boldsymbol{r}_4) + \frac{1}{6}(\boldsymbol{r}_4\boldsymbol{r}_3\boldsymbol{r}_2) .$$

In a similar way one can derive that if the ruled face 1234 of the hexahedron is replaced by triangles 123 and 134, see Fig. 12.2 *b*, then the contribution of these facets to the volume is given by

$$V''_{1234} = \frac{1}{6}(\boldsymbol{r}_3\boldsymbol{r}_2\boldsymbol{r}_1) + \frac{1}{6}(\boldsymbol{r}_1\boldsymbol{r}_4\boldsymbol{r}_3) .$$

Taking the mean of V'_{1234} and V''_{1234}, we get the formula for the contribution of the ruled cell face 1234, derived in [1]

$$V_{1234} = \frac{1}{2}(V'_{1234} + V''_{1234}) = \frac{1}{12}[(\boldsymbol{r}_1\boldsymbol{r}_4\boldsymbol{r}_3) + (\boldsymbol{r}_2\boldsymbol{r}_1\boldsymbol{r}_4) + (\boldsymbol{r}_3\boldsymbol{r}_2\boldsymbol{r}_1) + (\boldsymbol{r}_4\boldsymbol{r}_3\boldsymbol{r}_2)] .$$

Thus, the ruled face contribution is equal to the mean of two faces contribution, one consists of the facets 124, 234 and the other of 123, 134. We make conclusion that the ruled cell volume is equal to one-half of the volume of two dodecahedrons, depicted in Fig. 12.2. One gets immediately the consequence:

There are 32 pairs of such dodecahedrons. In every pair, the respective triangular facets are obtained by drawing the opposite diagonals.

One more pair is depicted in Fig. 12.3, where each a dodecahedron consists of two tetrahedrons and two prisms.

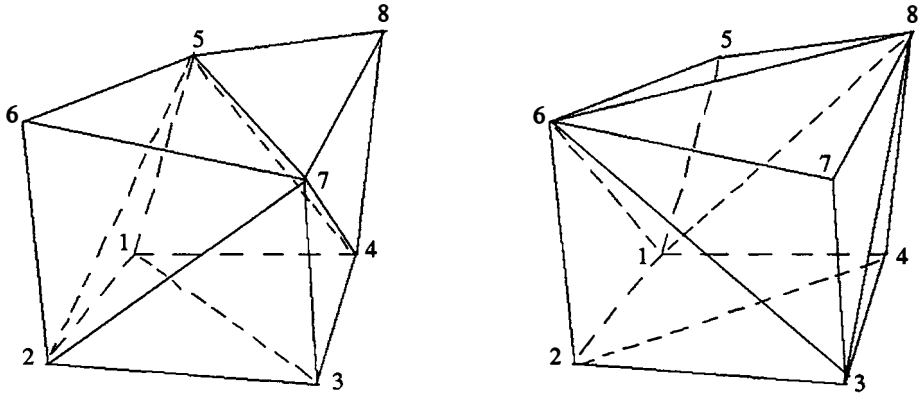


Figure 12.3: Pair of dodecahedrons, for which one-half of a sum of volumes is equal to the ruled cell volume. Each consists of two tetrahedrons and two prisms.

To perform remapping of the hydrodynamical parameters from one hex mesh onto another we use the following approach. Instead the ruled cell we take two dodecahedrons with structure depicted in Fig. 12.2. Thus, the integral (12.3) is calculated four times over the overlapping domain of two dodecahedrons, one taken from the pair corresponding to the new ruled cell, and the other from the pair corresponding to the old ruled cell. The contribution of the old ruled cell mass to the new ruled cell mass m_n is the mean of four overlapping figure masses.

12.3 Description of the Algorithm

As the term "cell" we will mean both the hex cell and dodecahedron that can be clear of context. We introduce some notations to be used further in the paper:

Ω_n and Ω_o denote the new and old dodecahedron, respectively;
Ω_n^{ijk} is the cell with vertex global indices $i+p$, $j+s$, $k+t$, where p, s, t=0, 1, see Fig. 12.19;
Ω_{on} is the overlapping figure of the dodecahedrons Ω_o and Ω_n;
$\mathcal{L}_{on}$ is the polygonal intersection line between the boundaries of Ω_o and Ω_n, if there are several lines we use $\mathcal{L}_{on}^i$;
$\triangle_n^{l_1}$ and $\triangle_o^{l_2}$ are the triangular facets of the dodecahedrons Ω_n and Ω_o, respectively, l_1, l_2=1, ..., 12;
$\boldsymbol{r}_i$ and $\tilde{\boldsymbol{r}}_i$ are the coordinates of the facet $\triangle_n^{l_1}$ and $\triangle_o^{l_2}$ vertices, respectively, i=1, 2, 3.

All vertex coordinates should be normalized with respect to a specific length, e.g. minimal diagonal length over all old or new cells, etc. The grids ω_o and ω_n must be non-folded and all dodecahedral cells are non-degenerate.

Next we should enumerate the vertices in every of twelve triangular facets. Consider two adjacent facets, e.g. 124 and 234 in Fig. 12.2 *a*. We set that the opposite (with respect to the diagonal 24) vertices 1 and 3 in the triangles have number 1. The other two vertices are numbered so as to specify the facet outward normal vector if to go round triangle in a vertex numbering. In other words we have the following accordance between vertex numeration in the dodecahedron, see Fig. 12.2 *a*, and local numeration in the triangular facet, see Fig. 12.4:

vertex number in dodecahedron	vertex number in triangle
1 2 4	1 3 2
2 3 4	3 1 2

Similarly we enumerate the other ten facets. Note the numeration rule for the second dodecahedron, depicted in Fig. 12.2 *b*, will be different. For the triangles 123 and 134 we have the following accordance:

vertex number in dodecahedron	vertex number in triangle
1 2 3	2 1 3
1 3 4	3 2 1

The remapping algorithm can be divided into four stages.

Stage I. Construction of the polygonal intersection line $\mathcal{L}_{on}$ between the boundaries of two 12-faceted cells, dodecahedrons Ω_n and Ω_o. We draw the polygonal line $\mathcal{L}_{on}$, a closed contour, where each segment is the intersection line between two triangular facets. There can be several lines $\mathcal{L}_{on}^i$.

Stage II. Construction of the overlapping figure Ω_{on} between Ω_n and Ω_o, a polyhedron with planar faces, in other words simplex. This stage is performed parallel to the Stage I, after a new segment of the line $\mathcal{L}_{on}$ has been obtained. On every triangular facet of the cells

Ω_n and Ω_o we construct the polygon, carved by $\mathcal{L}_{on}$. For several lines $\mathcal{L}_{on}^i$ there can be several polygons on the triangular facet (but not more than two). A union of all polygons is the boundary of the overlapping figure Ω_{on}. Given the simplex Ω_{on} we determine a particular mass of the cell Ω_o contributed to Ω_n. Note that Ω_{on} may consist of two polyhedrons.

Stage III. Calculation of the Ω_{on} volume and mass.

Stage IV. Algorithm of selecting the cells. One needs to find the environment of the underlying new cell Ω_n, i.e., all the old cells Ω_o^j, j=1, . . ., J, with nontrivial intersection between Ω_o^j and Ω_n. For the particular Ω_n we apply the optimal filtering algorithm for the mesh ω_o so as to proceed only cells Ω_o with nontrivial intersection. Next the mass m_n and density ρ_n of the new hex cell is calculated.

12.3.1 Stage I. Construction of an Intersection Line

Algorithm of constructing the intersection line $\mathcal{L}_{on}$ includes two steps.

Step 1. We sequently look through twelve triangular facets of Ω_n and for everyone twelve facets of underlying Ω_o are checked until the pair of facets with nonempty intersection will be found.

Step 2. We sequently draw the segments of the polygonal line $\mathcal{L}_{on}$ using two properties: a) $\mathcal{L}_{on}$ is a closed contour, and b) both segment ends are on the triangle $\triangle_n^{l_1}$ or $\triangle_o^{l_2}$ sides.

Step 1. Determination of intersection points between facets $\triangle_n^{l_1}$ and $\triangle_o^{l_2}$.

Consider the facets $\triangle_n^{l_1}$ and $\triangle_o^{l_2}$ with local vertex numeration 1, 2, 3, see Fig. 12.4. To construct the intersection line between these two facets one needs to write the equation for planes passing through the triangle vertices.

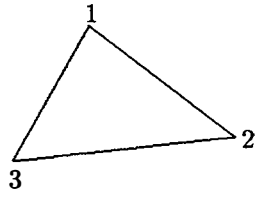


Figure 12.4: Local vertex numeration in the triangular facet $\triangle_n^{l_1}$ or $\triangle_o^{l_2}$.

For the sake of simplicity we transform the coordinates x, y, z so that in the new coordinate system x', y', z'

1) vertex 1 of the triangle $\triangle_n^{l_1}$ has the coordinates $(0, 0, 0)$,

2) x'-axis passes through the side 12 of $\triangle_n^{l_1}$,

3) triangle $\triangle_n^{l_1}$ lies in plane $x'-y'$.

To this end, the coordinate system x, y, z is translated and rotated. The translation is defined by $\boldsymbol{r}_1$, a radius-vector of vertex 1 of $\triangle_n^{l_1}$. Let us express $\boldsymbol{i}'$, $\boldsymbol{j}'$, $\boldsymbol{k}'$ (the unit vectors

in the x', y', z'-directions) in terms of $\boldsymbol{i}$, $\boldsymbol{j}$, $\boldsymbol{k}$ (the unit vectors in the x, y, z-directions). We have

$$\boldsymbol{i}' = \frac{\boldsymbol{r}_2 - \boldsymbol{r}_1}{|\boldsymbol{r}_2 - \boldsymbol{r}_1|}\,, \quad \boldsymbol{k}' - \frac{\boldsymbol{i}' \times (\boldsymbol{r}_3 - \boldsymbol{r}_1)}{|\boldsymbol{i}' \times (\boldsymbol{r}_3 - \boldsymbol{r}_1)|}\,, \quad \boldsymbol{j}' = \boldsymbol{k}' \times \boldsymbol{i}'\,. \tag{12.6}$$

Thus, from (12.6) we get the expansion for the unit vectors

$$\boldsymbol{i}' = \alpha_{11}\boldsymbol{i} + \alpha_{12}\boldsymbol{j} + \alpha_{13}\boldsymbol{k}, \;\; \boldsymbol{j}' = \alpha_{21}\boldsymbol{i} + \alpha_{22}\boldsymbol{j} + \alpha_{23}\boldsymbol{k}, \;\; \boldsymbol{k}' = \alpha_{31}\boldsymbol{i} + \alpha_{32}\boldsymbol{j} + \alpha_{33}\boldsymbol{k}, \tag{12.7}$$

where α_{ij} are the matrix $\mathcal{A}$ elements defining the coordinate system rotation. The coordinates of the triangle $\triangle_n^{l_1}$ vertices in the new coordinate system are

$$\boldsymbol{r}_1' = \mathcal{A}(\boldsymbol{r}_1 - \boldsymbol{r}_1) = 0, \;\; \boldsymbol{r}_2' = \mathcal{A}(\boldsymbol{r}_2 - \boldsymbol{r}_1) = (x_2', 0, 0), \;\; \boldsymbol{r}_3' = \mathcal{A}(\boldsymbol{r}_3 - \boldsymbol{r}_1) = (x_2', y_2', 0).$$

For brevity we omit the prime in writing, implying (if specially not specified) that one works in the new coordinate system. The equation for the plane, passing through the triangle $\triangle_n^{l_1}$ vertices, is

$$z = 0\,. \tag{12.8}$$

Now consider the old cell facet $\triangle_o^{l_2}$. In the new coordinate system the coordinates of the triangle vertices are given by the transformation

$$\tilde{\boldsymbol{r}}_i' = \mathcal{A}(\tilde{\boldsymbol{r}}_i - \boldsymbol{r}_1)\,,$$

where $\boldsymbol{r}_1$ is the radius-vector of vertex 1 of $\triangle_n^{l_1}$ in the old coordinate system. Again, for brevity, we omit the prime in writing, implying that one works in the new coordinate system. The equation for the plane, passing through the triangle $\triangle_o^{l_2}$ vertices, is

$$\left\| \begin{array}{ccc} x - \tilde{x}_1 & y - \tilde{y}_1 & z - \tilde{z}_1 \\ \tilde{x}_2 - \tilde{x}_1 & \tilde{y}_2 - \tilde{y}_1 & \tilde{z}_2 - \tilde{z}_1 \\ \tilde{x}_3 - \tilde{x}_1 & \tilde{y}_3 - \tilde{y}_1 & \tilde{z}_3 - \tilde{z}_1 \end{array} \right\| = 0\,,$$

or

$$Ax + By + Cz + D = 0\,, \tag{12.9}$$

where

$$A = (\tilde{y}_2 - \tilde{y}_1)(\tilde{z}_3 - \tilde{z}_1) - (\tilde{y}_3 - \tilde{y}_1)(\tilde{z}_2 - \tilde{z}_1)\,, \;\; B = -(\tilde{x}_2 - \tilde{x}_1)(\tilde{z}_3 - \tilde{z}_1) + (\tilde{x}_3 - \tilde{x}_1)(\tilde{z}_2 - \tilde{z}_1)\,,$$

$$C = (\tilde{x}_2 - \tilde{x}_1)(\tilde{y}_3 - \tilde{y}_1) - (\tilde{x}_3 - \tilde{x}_1)(\tilde{y}_2 - \tilde{y}_1)\,, \;\; D = -A\tilde{x}_1 - B\tilde{y}_1 - C\tilde{z}_1\,.$$

To find the intersection segment between the triangles $\triangle_n^{l_1}$ and $\triangle_o^{l_2}$, we seek the intersection line between the planes (12.8) and (12.9). Substitution of (12.8) in (12.9) gives the equation of the line

$$Ax + By + D = 0\,. \tag{12.10}$$

First one should check whether the line (12.10) intersects the triangle $\triangle_n^{l_1}$ sides. Accordingly to the number of sides, three cases are possible.

1. Consider possibility that the line (12.10) intersects the side 12 for which the equation is $y = 0$. Substituting it in (12.10) we obtain

$$Ax + D = 0\,. \tag{12.11}$$

Here there are three variants:

1.1. Variant I. The coefficients are A, D=0. Then the line (12.10) coincides with the side 12 of the triangle $\triangle_n^{l_1}$. Therefore, we seek the intersection point between the side 12 and side $\tilde{1}\tilde{2}$, $\tilde{1}\tilde{3}$ or $\tilde{2}\tilde{3}$ of the triangle $\triangle_o^{l_2}$.

1.1.1. Check whether the sides 12 and $\tilde{1}\tilde{2}$ intersect. If yes, then the intersection point Q lies at these two sides. To find Q coordinates we write the equation of the sides in the parametric form

$$\boldsymbol{r} = t\boldsymbol{r}_2\,, \tag{12.12}$$

and

$$\boldsymbol{r} = (1-\tilde{t})\tilde{\boldsymbol{r}}_1 + \tilde{t}\tilde{\boldsymbol{r}}_2\,, \tag{12.13}$$

where the parameters are $0 \le t, \tilde{t} \le 1$. Equating the right parts of the above vector equations and by virtue of $\boldsymbol{r}_2$=$(x_2, 0, 0)$, we get the relation to the y-component

$$0 = (1-\tilde{t})\tilde{y}_1 + \tilde{t}\tilde{y}_2\,,$$

and, therefore,

$$\tilde{t} = \frac{\tilde{y}_1}{\tilde{y}_1 - \tilde{y}_2}\,.$$

The equation for the x-component

$$x_2 t = (1-\tilde{t})\tilde{x}_1 + \tilde{t}\tilde{x}_2\,,$$

serves to determine the parameter t

$$t = \frac{(1-\tilde{t})\tilde{x}_1 + \tilde{t}\tilde{x}_2}{x_2}\,.$$

If the both parameters are within the interval

$$0 \le t, \tilde{t} \le 1\,, \tag{12.14}$$

then $Q(tx_2, 0, 0)$ is the intersection point between the sides 12 and $\tilde{1}\tilde{2}$, and we take it as the next node of the polygonal line $\mathcal{L}_{on}$. If the condition (12.14) fails, then

1.1.2. Check whether the sides 12 and $\tilde{1}\tilde{3}$ intersect. The equation for the side $\tilde{1}\tilde{3}$ is

$$\boldsymbol{r} = (1-\tilde{t})\tilde{\boldsymbol{r}}_1 + \tilde{t}\tilde{\boldsymbol{r}}_3\,. \tag{12.15}$$

Likewise in item 1.1.1 we write the vector equation, equating the right parts of the equations (12.12) and (12.15), and find the parameters

$$\tilde{t} = \frac{\tilde{y}_1}{\tilde{y}_1 - \tilde{y}_3}\,, \quad t = \frac{(1-\tilde{t})\tilde{x}_1 + \tilde{t}\tilde{x}_3}{x_2}\,.$$

If the condition (12.14) is fulfilled, then $Q(tx_2, 0, 0)$ is the intersection point between the sides 12 and $\tilde{1}\tilde{3}$, and we take it as the next node of $\mathcal{L}_{on}$. If (12.14) fails, then

1.1.3. Check whether the sides 12 and $\tilde{2}\tilde{3}$ intersect. The equation of the side $\tilde{2}\tilde{3}$ is

$$r = (1 - \tilde{t})\tilde{r}_2 + \tilde{t}\tilde{r}_3 \,. \tag{12.16}$$

We write the vector equation, equating the right parts of (12.12) and (12.16), and find the parameters

$$\tilde{t} = \frac{\tilde{y}_2}{\tilde{y}_2 - \tilde{y}_3} \,, \quad t = \frac{(1 - \tilde{t})\tilde{x}_2 + \tilde{t}\tilde{x}_3}{x_2} \,.$$

If the condition (12.14) is fulfilled, then $Q(tx_2, 0, 0)$ is the intersection point of the sides 12 and $\tilde{2}\tilde{3}$, and we take it as the next node of $\mathcal{L}_{on}$. If (12.14) fails, then the side 12 does not intersect the sides of the triangle $\triangle_o^{l_2}$ and we should consider the side 13 of the triangle $\triangle_n^{l_1}$, see item 2 below.

1.2. Variant II. The coefficients in (12.11) are $A{=}0$, $D{\neq}0$. Then the planes (12.8) and (12.9) have no intersection and, therefore, the triangular facets as well. We should consider the side 13 of $\triangle_n^{l_1}$, see item 2 below.

1.3. Variant III. The coefficient $A{\neq}0$, and from (12.11) we get $x{=}{-}D/A$. Using the projection onto the x-axis of the equation of the side 12 in the parametric form $x{=}tx_2$, we find the parameter

$$t = -\frac{D}{Ax_2} \,.$$

If $0{\leq}t{\leq}1$, then one needs to check whether the intersection point $Q(tx_2, 0, 0)$ lies inside the triangle $\triangle_o^{l_2}$ (including sides) or not. We apply the algorithm A (see below) for the check. If Q lies inside the triangle or on the boundary, we take it as the next node of $\mathcal{L}_{on}$. If no, the side 12 does not intersect the triangle $\triangle_o^{l_2}$ sides and we should consider the side 13 of $\triangle_n^{l_1}$, see item 2 below.

Algorithm A. Fig. 12.5 represents two possible cases for the point Q. Let $\boldsymbol{r}_{Q\tilde{1}}, \boldsymbol{r}_{Q\tilde{2}}$, $\boldsymbol{r}_{Q\tilde{3}}$ be the vectors directed from Q to points $\tilde{1}, \tilde{2}, \tilde{3}$, respectively. Make up two vector products $\boldsymbol{a} = \boldsymbol{r}_{Q\tilde{1}} \times \boldsymbol{r}_{Q\tilde{2}}$, $\boldsymbol{b} = \boldsymbol{r}_{Q\tilde{2}} \times \boldsymbol{r}_{Q\tilde{3}}$. If Q lies inside or on the boundary of the triangle, see Fig. 12.5 *a*, the scalar product $\boldsymbol{a} \cdot \boldsymbol{b} \geq 0$. If Q lies outside of the triangle, see Fig. 12.5 *b*, then $\boldsymbol{a} \cdot \boldsymbol{b} < 0$.

2. Next we consider possibility that the line (12.10) intersects the side 13 with the equation

$$x{=}ky \,,$$

where $k{=}x_3/y_3$. Substituting it in (12.10) we obtain

$$(Ak + B)y + D = 0 \,. \tag{12.17}$$

Here again there are three variants:

2.1. Variant I. The coefficients satisfy the conditions $Ak{+}B{=}0$, $D{=}0$. Then the intersection line (12.10) coincides with the side 13 of $\triangle_n^{l_1}$. Therefore, we seek the intersection point between the side 12 and side $\tilde{1}\tilde{2}$, $\tilde{1}\tilde{3}$ or $\tilde{2}\tilde{3}$ of $\triangle_o^{l_2}$.

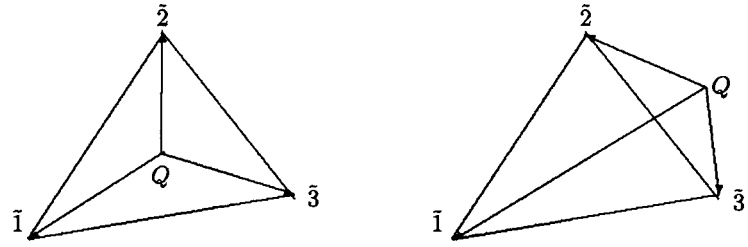


Figure 12.5: Point Q lies inside of the triangle $\triangle_o^{\tilde{l}\tilde{2}}$ (a) and outside of it (b).

2.1.1. First check whether the sides 13 and $\tilde{1}\tilde{2}$ intersect. If yes, then the intersection point Q lies at these two sides. To find it, we write the equation of the side 13

$$\boldsymbol{r} = t\boldsymbol{r}_3 \,, \tag{12.18}$$

and recall (12.13) for the side $\tilde{1}\tilde{2}$. Equating the right parts of (12.18) and (12.13), we obtain

$$t\boldsymbol{r}_3 = (1-\tilde{t})\tilde{\boldsymbol{r}}_1 + \tilde{t}\tilde{\boldsymbol{r}}_2 \,.$$

Writing the x- and y-components of this equation, we get the system for determining the parameters $t, \tilde{t}$

$$tx_3 = (1-\tilde{t})\tilde{x}_1 + \tilde{t}\tilde{x}_2 \,, \quad ty_3 = (1-\tilde{t})\tilde{y}_1 + \tilde{t}\tilde{y}_2 \,,$$

and from it we have

$$\tilde{t} = \frac{\tilde{x}_1 y_3 - \tilde{y}_1 x_3}{x_3(\tilde{y}_2 - \tilde{y}_1) - y_3(\tilde{x}_2 - \tilde{x}_1)} \,, \quad t = \frac{(1-\tilde{t})\tilde{y}_1 + \tilde{t}\tilde{y}_2}{y_3} \,.$$

If the both parameters satisfy the condition (12.14), then $Q(tx_3, ty_3, 0)$ is the intersection point between the sides 13 and $\tilde{1}\tilde{2}$, and we take it as the next node of the line $\mathcal{L}_{on}$. If (12.14) fails, then

2.1.2. Check whether the sides 13 and $\tilde{1}\tilde{3}$ intersect. Equating the right parts of the equations (12.18) and (12.15), we obtain

$$t\boldsymbol{r}_3 = (1-\tilde{t})\tilde{\boldsymbol{r}}_1 + \tilde{t}\tilde{\boldsymbol{r}}_3 \,.$$

Similarly to that in item 2.1.1 we resolve this vector equation and get

$$\tilde{t} = \frac{\tilde{x}_2 y_3 - \tilde{y}_1 x_3}{x_3(\tilde{y}_3 - \tilde{y}_1) - y_3(\tilde{x}_3 - \tilde{x}_1)} \,, \quad t = \frac{(1-\tilde{t})\tilde{y}_1 + \tilde{t}\tilde{y}_3}{y_3} \,.$$

If the condition (12.14) is fulfilled, then $Q(tx_3, ty_3, 0)$ is the intersection point between the sides 13 and $\tilde{1}\tilde{3}$, and we take it as the next node $\mathcal{L}_{on}$. If (12.14) fails, then

2.1.3. Check whether the sides 13 and $\tilde{2}\tilde{3}$ intersect. Equating the right parts of the equations (12.18) and (12.16), we obtain

$$t\boldsymbol{r}_3 = (1-\tilde{t})\tilde{\boldsymbol{r}}_2 + \tilde{t}\tilde{\boldsymbol{r}}_3 \,.$$

Resolving it, we have

$$\tilde{t} = \frac{\tilde{x}_2 y_3 - \tilde{y}_2 x_3}{x_3(\tilde{y}_3 - \tilde{y}_2) - y_3(\tilde{x}_3 - \tilde{x}_2)} \,, \quad t = \frac{(1-\tilde{t})\tilde{y}_2 + \tilde{t}\tilde{y}_3}{y_3} \,.$$

If the condition (12.14) is fulfilled, then $Q(tx_3, ty_3, 0)$ is the intersection point between the sides 12 and $\tilde{2}\tilde{3}$. If (12.14) fails then the side 13 does not intersect sides of the triangle $\triangle_o^{l_2}$ and we should consider the side 23 of $\triangle_n^{l_1}$, see item 3.

2.2. Variant II. The coefficients in (12.11) satisfy the conditions $Ak+B=0$, $D\neq 0$. Then the planes (12.8) and (12.9) do not intersect and, therefore, the triangular facets as well. We should consider the side 23 of $\triangle_n^{l_1}$, see item 3.

2.3. Variant III. The coefficients satisfy the condition $Ak+B\neq 0$ and from (12.17) we get $y=-D/(Ak+B)$. Using the projection onto the y-axis of the equation of the side 13 in the parametric form $y=ty_3$, we find the parameter

$$t = -\frac{D}{(Ak+B)y_3} \,.$$

If $0\leq t\leq 1$, then one needs to check whether the point $Q(tx_3, ty_3, 0)$ lies inside the triangle $\triangle_o^{l_2}$ (including the sides) or not. We apply the algorithm A for the check, see item 1.1.3. If Q lies inside the triangle, we take it as the next node of $\mathcal{L}_{on}$. If no, the side 13 does not intersect the sides of $\triangle_o^{l_2}$ and we should consider the side 23 of $\triangle_n^{l_1}$, see item 3.

3. Next we consider possibility that the line (12.10) intersects the side 23 with the equation

$$x = ky + p \,,$$

where

$$k = \frac{x_3 - x_2}{y_3 - y_2} \,, \quad p = \frac{y_3 x_2 - y_2 x_3}{y_3 - y_2} \,.$$

Substituting it in (12.10) we get

$$(Ak + B)y + Ap + D = 0 \,. \tag{12.19}$$

Here again there are three variants:

3.1. Variant I. The coefficients satisfy the conditions $Ak+B=0$, $D=0$. Then the intersection line (12.10) coincides with the side 23 of $\triangle_n^{l_1}$. Therefore, we seek the intersection point between the side 23 and side $\tilde{1}\tilde{2}$, $\tilde{1}\tilde{3}$ or $\tilde{2}\tilde{3}$ of $\triangle_o^{l_2}$.

3.1.1. First check whether the sides 23 and $\tilde{1}\tilde{2}$ intersect. If yes, then the intersection point Q lies at these two sides. To find it, we write the equation of the side 23

$$\boldsymbol{r} = (1-t)\boldsymbol{r}_2 + t\boldsymbol{r}_3 \,, \tag{12.20}$$

and recall (12.13) for $\tilde{1}\tilde{2}$. Equating the right parts of (12.20) and (12.13), we obtain the system

$$(1-t)\boldsymbol{r}_2 + t\boldsymbol{r}_3 = (1-\tilde{t})\tilde{\boldsymbol{r}}_1 + \tilde{t}\tilde{\boldsymbol{r}}_2 .$$

Resolving it, we have

$$\tilde{t} = \frac{(\tilde{x}_1 - x_2)(y_3 - y_2) - (\tilde{y}_1 - y_2)(x_3 - x_2)}{(\tilde{y}_2 - \tilde{y}_1)(x_3 - x_2) - (\tilde{x}_2 - \tilde{x}_1)(y_3 - y_2)}, \quad t = \frac{(1-\tilde{t})\tilde{y}_1 + \tilde{t}\tilde{y}_2 - y_2}{y_3 - y_2} .$$

If the both parameters satisfy (12.14), then $Q((1-t)x_2+tx_3,(1-t)y_2+ty_3,0)$ is the intersection point between the sides 23 and $\tilde{1}\tilde{2}$, and we take it as the next node of $\mathcal{L}_{on}$. If (12.14) fails, then

3.1.2. Check whether the sides 23 and $\tilde{1}\tilde{3}$ intersect. Equating the right parts of the equations (12.20) and (12.15), we obtain

$$(1-t)\boldsymbol{r}_2 + t\boldsymbol{r}_3 = (1-\tilde{t})\tilde{\boldsymbol{r}}_1 + \tilde{t}\tilde{\boldsymbol{r}}_3 .$$

Resolving it, we have

$$\tilde{t} = \frac{(\tilde{x}_1 - x_2)(y_3 - y_2) - (\tilde{y}_1 - y_2)(x_3 - x_2)}{(\tilde{y}_3 - \tilde{y}_1)(x_3 - x_2) - (\tilde{x}_3 - \tilde{x}_1)(y_3 - y_2)}, \quad t = \frac{(1-\tilde{t})\tilde{y}_1 + \tilde{t}\tilde{y}_3 - y_2}{y_3 - y_2} .$$

If the condition (12.14) is fulfilled, then $Q((1-t)x_2+tx_3, (1-t)y_2+ty_3, 0)$ is the intersection point between the sides 23 and $\tilde{1}\tilde{3}$, and we take it as the next node of $\mathcal{L}_{on}$. If (12.14) fails, then

3.1.3. Check whether the sides 23 and $\tilde{2}\tilde{3}$ intersect. Equating the right parts of the equations (12.20) and (12.16), we obtain

$$(1-t)\boldsymbol{r}_2 + t\boldsymbol{r}_3 = (1-\tilde{t})\tilde{\boldsymbol{r}}_2 + \tilde{t}\tilde{\boldsymbol{r}}_3 .$$

Resolving it, we have

$$\tilde{t} = \frac{(\tilde{x}_2 - x_2)(y_3 - y_2) - (\tilde{y}_2 - y_2)(x_3 - x_2)}{(\tilde{y}_3 - \tilde{y}_2)(x_3 - x_2) - (\tilde{x}_3 - \tilde{x}_2)(y_3 - y_2)}, \quad t = \frac{(1-\tilde{t})\tilde{y}_2 + \tilde{t}\tilde{y}_3 - y_2}{y_3 - y_2} .$$

If the condition (12.14) is fulfilled, then $Q((1-t)x_2+tx_3, (1-t)y_2+ty_3, 0)$ is the intersection point between the sides 23 and $\tilde{2}\tilde{3}$. If (12.14) fails, then the side 23 does not intersect the sides of the triangle $\triangle_o^{l_2}$ and we check whether the line (12.10) intersects the triangle $\triangle_o^{l_2}$ sides, see item 4.

3.2. Variant II. The coefficients in (12.11) satisfy the conditions $Ak+B=0$ and $Ap+D\neq 0$, then the planes (12.8) and (12.9) do not intersect and, therefore, the triangular facets as well. We should check whether the line (12.10) intersects the triangle $\triangle_o^{l_2}$ sides, see item 4.

3.3. Variant III. The coefficients satisfy the condition $Ak+B\neq 0$, and from (12.19) we get $y=-(Ap+D)/(Ak+B)$. Using the y-projection of (12.20) we find the parameter

$$t = -\left(\frac{Ap+D}{Ak+B} + y_2\right) \Big/ (y_3 - y_2) .$$

If $0 \le t \le 1$, then we check whether $Q((1-t)x_2+tx_3, (1-t)y_2+ty_3, 0)$ lies inside the triangle $\triangle_o^{l_2}$ (including the sides) or not. We apply the algorithm A for the check, see item 1.1.3. If Q lies inside the triangle, we take it as the next node of the line $\mathcal{L}_{on}$. If no, the side 13 does not intersect the sides of the triangle $\triangle_o^{l_2}$, and we turn to item 4.

Next we should check whether the line (12.10) intersects the triangle $\triangle_o^{l_2}$ sides. Accordingly to the number of sides, three cases are possible.

4. Consider possibility that the line (12.10) intersects the side $\tilde{1}\tilde{2}$ with the equation (12.13). The line (12.10) and side $\tilde{1}\tilde{2}$ can be parallel or not. Therefore, there are two cases.

4.1. They are parallel. Here two variants are possible.

4.1.1. If the coordinates $\tilde{z}_1 = \tilde{z}_2 \neq 0$, then these lines do not have common points. Further the side $\tilde{1}\tilde{3}$ should be considered.

4.1.2. If $\tilde{z}_1 = \tilde{z}_2 = 0$, then the line (12.10) and side $\tilde{1}\tilde{2}$ coincide and one should check whether there are common points of the side $\tilde{1}\tilde{2}$ and triangle $\triangle_n^{l_1}$.

4.1.2.1. We seek intersection between the sides $\tilde{1}\tilde{2}$ and 12, see item 1.1.1. If the condition (12.14) is fulfilled, then $Q(tx_2, 0, 0)$ is the intersection point between the sides $\tilde{1}\tilde{2}$ and 12, and we take it as the next node of $\mathcal{L}_{on}$. If (12.14) fails, then

4.1.2.2. We seek intersection between the sides $\tilde{1}\tilde{2}$ and 13, see item 2.1.1. If the condition (12.14) is fulfilled, then $Q(tx_3, ty_3, 0)$ is the intersection point between the sides $\tilde{1}\tilde{2}$ and 13, and we take it as the next node of $\mathcal{L}_{on}$. If (12.14) fails, then

4.1.2.3. We seek intersection between the sides $\tilde{1}\tilde{2}$ and 23, see item 3.1.1. If the condition (12.14) is fulfilled, then $Q((1-t)x_2+tx_3, (1-t)y_2+ty_3, 0)$ is the intersection point between the sides $\tilde{1}\tilde{2}$ and 23, and we take it as the next node of $\mathcal{L}_{on}$. If (12.14) fails, then we consider the side $\tilde{1}\tilde{3}$ of $\triangle_o^{l_2}$, see item 5.

4.2. The line (12.10) and side $\tilde{1}\tilde{2}$ are not parallel. Equating the z-projection of (12.13) to zero (since the triangle $\triangle_n^{l_1}$ is in plane $z=0$), we obtain

$$(1-\tilde{t})\tilde{z}_1 + \tilde{t}\tilde{z}_2 = 0\,,$$

and, therefore,

$$\tilde{t} = \frac{\tilde{z}_1}{\tilde{z}_1 - \tilde{z}_2}\,.$$

If the parameter $\tilde{t}$ satisfies the condition $0 \le \tilde{t} \le 1$, we check whether $Q((1-\tilde{t})\tilde{x}_1+\tilde{t}\tilde{x}_2, (1-\tilde{t})\tilde{y}_1 + \tilde{t}\tilde{y}_2, 0)$ is inside the triangle $\triangle_n^{l_1}$ applying the algorithm A. If Q is inside $\triangle_n^{l_1}$, we take it as the next of $\mathcal{L}_{on}$. If it is outside, we consider the side $\tilde{1}\tilde{3}$ of $\triangle_o^{l_2}$, see item 5.

5. Consider possibility that the line (12.10) intersects side $\tilde{1}\tilde{3}$ with the equation (12.15). The line (12.10) and side $\tilde{1}\tilde{3}$ can be parallel or not. Therefore, there are two cases.

5.1. They are parallel. Here two variants are possible.

5.1.1. If the coordinates $\tilde{z}_1 = \tilde{z}_3 \neq 0$, then these lines do not have common points. We turn to the side $\tilde{2}\tilde{3}$.

5.1.2. If $\tilde{z}_1 = \tilde{z}_3 = 0$, then the line (12.10) and side $\tilde{1}\tilde{3}$ coincide and one should check whether there are common points of the side $\tilde{1}\tilde{3}$ and triangle $\triangle_n^{l_1}$.

5.1.2.1. We seek intersection between the sides $\tilde{1}\tilde{3}$ and 12, see item 1.1.2. If the condition (12.14) is fulfilled, then $Q(tx_2, 0, 0)$ is the intersection point between the sides $\tilde{1}\tilde{3}$ and 12, and we take it as the next node of $\mathcal{L}_{on}$. If (12.14) fails, then

5.1.2.2. We seek intersection between the sides $\tilde{1}\tilde{3}$ and 13, see item 2.1.2. If the condition (12.14) is fulfilled, then $Q(tx_3, ty_3, 0)$ is the intersection point between the sides $\tilde{1}\tilde{3}$ and 13, and we take it as the next node of $\mathcal{L}_{on}$. If (12.14) fails, then

5.1.2.3. We seek intersection between the sides $\tilde{1}\tilde{3}$ and 23, see item 3.1.2. If the condition (12.14) is fulfilled, then $Q((1-t)x_2+tx_3, (1-t)y_2+ty_3, 0)$ is the intersection point between the sides $\tilde{1}\tilde{3}$ and 23, and we take it as the next node of $\mathcal{L}_{on}$. If (12.14) fails, then we consider the side $\tilde{2}\tilde{3}$ of $\triangle_o^{l_2}$, see item 6.

5.2. The line (12.10) and side $\tilde{1}\tilde{3}$ are not parallel. Equating the z-projection of (12.15) to zero, we obtain

$$(1-\tilde{t})\tilde{z}_1 + \tilde{t}\tilde{z}_3 = 0\,,$$

and, therefore,

$$\tilde{t}=\frac{\tilde{z}_1}{\tilde{z}_1-\tilde{z}_3}\,.$$

If $0\le\tilde{t}\le 1$, we check whether $Q((1-\tilde{t})\tilde{x}_1+\tilde{t}\tilde{x}_2, (1-\tilde{t})\tilde{y}_1+\tilde{t}\tilde{y}_2, 0)$ is inside the triangle $\triangle_n^{l_1}$ applying the algorithm A. If Q is inside $\triangle_n^{l_1}$, we take it as the next node of $\mathcal{L}_{on}$. If it is outside, we consider the side $\tilde{2}\tilde{3}$ of $\triangle_o^{l_2}$, see item 6.

6. Finally, consider possibility that the line (12.10) intersects side $\tilde{2}\tilde{3}$ with the equation (12.16). The line (12.10) and side $\tilde{2}\tilde{3}$ can be parallel or not. Therefore, there are two cases.

6.1. They are parallel. Here two variants are possible.

6.1.1. If the coordinates $\tilde{z}_2=\tilde{z}_3\neq 0$, then these lines do not have common points. We consider the next triangular facet $\triangle_o^{l_2+1}$ (if $l_2=12$ we proceed to the next facet $\triangle_n^{l_1+1}$).

6.1.2. If $\tilde{z}_2=\tilde{z}_3=0$, then the line (12.10) and side $\tilde{2}\tilde{3}$ coincide and one should check whether there are common points of the side $\tilde{2}\tilde{3}$ and triangle $\triangle_n^{l_1}$.

6.1.2.1. We seek intersection between the sides $\tilde{2}\tilde{3}$ and 12, see item 1.1.3. If the condition (12.14) is fulfilled, then $Q(tx_2, 0, 0)$ is the intersection point between the sides $\tilde{2}\tilde{3}$ and 12, and we take it as the next node of $\mathcal{L}_{on}$. If (12.14) fails, then

6.1.2.2. We seek intersection between the sides $\tilde{2}\tilde{3}$ and 13, see item 2.1.3. If the condition (12.14) is fulfilled, then $Q(tx_3, ty_3, 0)$ is the intersection point between the sides $\tilde{2}\tilde{3}$ and 13, and we take it as the next node of $\mathcal{L}_{on}$. If (12.14) fails, then

6.1.2.3. We seek intersection between the sides $\tilde{2}\tilde{3}$ and 23, see item 3.1.3. If the condition (12.14) is fulfilled, then $Q((1-t)x_2+tx_3, (1-t)y_2+ty_3, 0)$ is the intersection point between the sides $\tilde{2}\tilde{3}$ and 23, and we take it as the next node of $\mathcal{L}_{on}$. If (12.14) fails, then we consider the next triangular facet $\triangle_o^{l_2+1}$ (if $l_2=12$ we proceed to the next facet $\triangle_n^{l_1+1}$).

6.2. The line (12.10) and side $\tilde{2}\tilde{3}$ are not parallel. Equating the z-projection of (12.16) to zero, we obtain

$$(1-\tilde{t})\tilde{z}_2 + \tilde{t}\tilde{z}_3 = 0\,,$$

and, therefore,

$$\tilde{t}=\frac{\tilde{z}_2}{\tilde{z}_2-\tilde{z}_3}\,.$$

If $0\le\tilde{t}\le 1$, we check whether $Q((1-\tilde{t})\tilde{x}_1+\tilde{t}\tilde{x}_2,\,(1-\tilde{t})\tilde{y}_1+\tilde{t}\tilde{y}_2,\,0)$ is inside the triangle $\triangle_n^{l_1}$ applying the algorithm A. If Q is inside $\triangle_n^{l_1}$, we take it as the next node of the line $\mathcal{L}_{on}$. If it is outside, we consider the next triangular facet $\triangle_o^{l_2+1}$ (if $l_2=12$ we proceed to the next facet $\triangle_n^{l_1+1}$).

After finding the point Q coordinates we return to the old coordinate system using inverse rotation and translation. With this purpose we apply the matrix $\mathcal{A}^\top$, transposed to the rotation matrix $\mathcal{A}$. The Q coordinates in the old coordinate system are

$$\boldsymbol{r}_Q=\mathcal{A}^\top\boldsymbol{r}'_Q+\boldsymbol{r}_1\,,$$

where $\boldsymbol{r}'_Q$ are the Q coordinates in the new coordinate system, $\boldsymbol{r}_1$ is referred to vertex 1 of the triangle $\triangle_n^{l_1}$ in the old coordinate system.

Step 2. Determination of the line $\mathcal{L}_{on}$ segments.

To determine the line $\mathcal{L}_{on}$ segments we need to know the rule of transition from one triangular facet to another depending on what triangle side the line $\mathcal{L}_{on}$ crosses. We use the variable, indicating the number of the next triangular facet. In the computer code this is a two-dimensional array $Ar(l,m)$ with $l=1,\ldots,12$ and $m=1,2,3$. Here l specifies the facet number, and m is the side number in the triangle, which the line $\mathcal{L}_{on}$ crosses. We accept that $m=1,2,3$ for the sides 12, 13, and 23, respectively, in the triangle $\triangle_n^{l_1}$ and similarly in $\triangle_o^{l_2}$.

After determining a new point Q at the *Step 1*, at the *Step 2* we go to the adjacent triangular facet of the cell, new or old, depending on what side the line $\mathcal{L}_{on}$ crosses. Next we repeat the *Step 1*, etc., until some point Q^i coincides with the first point Q^1 of $\mathcal{L}_{on}$. All these intersection points are entered into the vertex list of $\mathcal{L}_{on}$. Further we examine unchecked facets with the purpose to find a new intersection line. Since two triangular facets, new $\triangle_n^{l_1}$ and old $\triangle_o^{l_2}$, can not intersect twice, we exclude such facets, participating in forming the intersection lines $\mathcal{L}_{on}^1,\ldots,\mathcal{L}_{on}^i$, when constructing the line $\mathcal{L}_{on}^{i+1}$. In total we need to check $12\times12=144$ facet pairs of the new and old cells (dodecahedrons).

Further we will sometimes verify whether some point O is inside the 12-faceted cell or not. Consider such a procedure, referred to as Algorithm B.

Algorithm B. We divide the cell into five tets as shown in Fig. 12.2. Consider one of tets with vertices 1234, see Fig. 12.6. Joining the point O with every of four tet vertices, we obtain four new tets: $O234$, $O134$, $O124$, and $O123$. If O is inside the tet 1234, see Fig. 12.6 *a*, its volume is equal to the sum of four tet volumes

$$V_{1234}=V_{O234}+V_{O134}+V_{O124}+V_{O123}\,,$$

where, for instance, $V_{1234} = \frac{1}{6}(\boldsymbol{r_{12}} \boldsymbol{r_{13}} \boldsymbol{r_{14}})$. Otherwise, see Fig. 12.6 *b*,

$$V_{1234} < V_{O234} + V_{O134} + V_{O124} + V_{O123} \,.$$

Checking all five tets, which form the cell, we determine whether the point O is inside one of them. If yes, O is inside the cell. If O is not inside one of them, then O is outside of the cell.

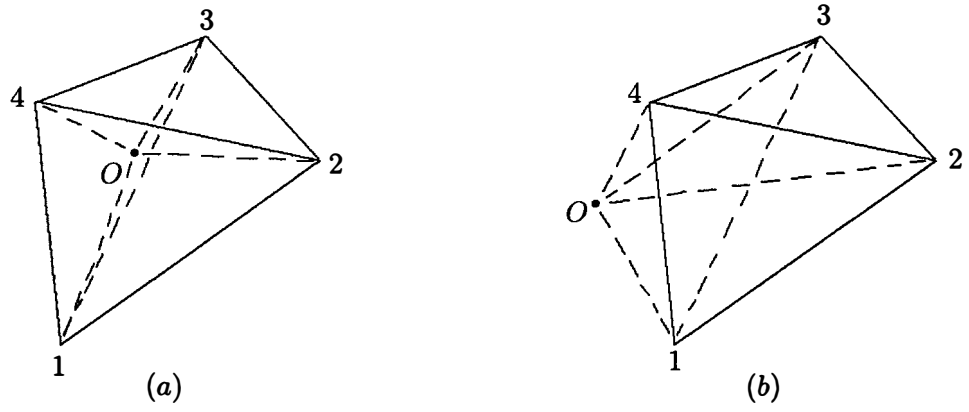


Figure 12.6: One of five tets, forming the dodecahedron. The point O is inside (*a*) and outside (*b*) of the tet.

12.3.2 Stage II. Construction of Overlapping Figure Ω_{on}

Forming of the overlapping figure Ω_{on} is executed simultaneously with the intersection lines $\mathcal{L}^i_{on}$. On the triangular facets $\triangle^{l_1}_n$ and $\triangle^{l_2}_o$ we construct the polygons, carved by the lines $\mathcal{L}^i_{on}$. On a facet it can be 0,1 or 2 polygons (this is only assumption, confirmed by practice, but is not strictly proved). A union of all polygons $\bigcup_{i=1}^{m} P^i$ forms the boundary of the overlapping figure Ω_{on}. The algorithm of constructing a polygon P^i is the following. When the line $\mathcal{L}^i_{on}$ intersects, for instance, the facet $\triangle^{l_1}_n$ (it may be one segment with ends placed at two sides of the triangle, see Fig. 12.7 *a*, or may be a polygonal line with several segments, see Fig. 12.7 *b*) and divides it into two subdomains, the subdomain, located inside the cell Ω_o (a hatched region in Fig. 12.7 *a*, *b*), is kept for constructing the polygon P^i. It does not mean that the remained part of the facet is abandoned at further treating. In this part, another polygon can be carved by the underlying line $\mathcal{L}^i_{on}$ or another intersection line. As well as the hatched region in Fig. 12.7 *b* can be partly cut off when further forming P^i, see Fig. 12.7 *c*. In Fig. 12.7 *c*, the lines $\mathcal{L}^{i_1}_{on}$, $\mathcal{L}^{i_2}_{on}$ may denote the same line $\mathcal{L}^i_{on}$, shown in Fig. 12.7 *b*, or different lines.

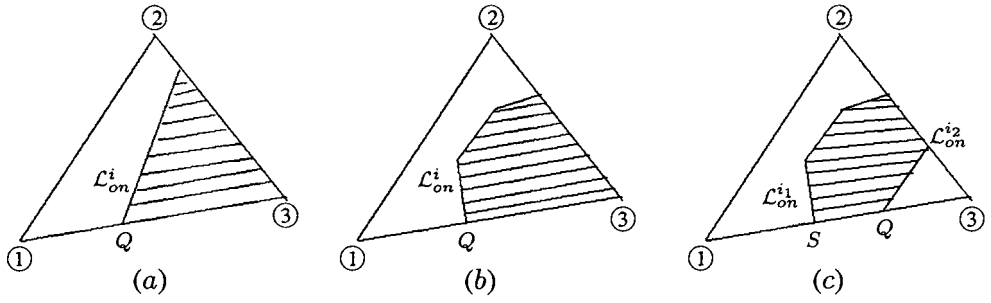


Figure 12.7:

Similarly, when the line $\mathcal{L}^i_{on}$ intersects the facets of Ω_o, the domain, located inside the cell Ω_n, is kept for constructing the polygon P^i. Note that the figure Ω_{on} may consist of one or two polyhedrons.

Consider the formation of the polygons P^i, carved on the triangular facets of Ω_n. We introduce several notations. The values $V_x^{ij}, V_y^{ij}, V_z^{ij}$ are the coordinates of the jth vertex of the ith polygon P^i. Since on a facet there can be 0, 1 or 2 polygons, the polygon number is given by the rule

$$i = l_1 + 12m\,, \tag{12.21}$$

where l_1 is the facet $\triangle_n^{l_1}$ ordinal number, m=0, $N_{pol}^{l_1} - 1$, $N_{pol}^{l_1}$=0, 1, 2 is the number of existing polygons on $\triangle_n^{l_1}$. The value N_{tot}^i is the number of the polygon P^i vertices. The value N_{nod}^{ik} specifies the number of vertices located at the kth side of $\triangle_n^{l_1}$, k=1, 2, 3. As noted above j is the vertex ordinal number in P^i, and N_v^{ikl}=j specifies the ordinal number of the vertex located at the kth side of $\triangle_n^{l_1}$. Here the indices l and l_1 have different sense, l is the ordinal number of the vertex at the kth side. At the side there can be several vertices of P^i and, therefore, l=1, ..., N_{nod}^{ik}. The values Q_x, Q_y, Q_z are the coordinates of the underlying intersection point Q, being the break point of the polygonal line $\mathcal{L}^i_{on}$. The point Q may be either at the side or inside the triangle $\triangle_n^{l_1}$. The points R^1, R^2 are the end-points of the underlying facet edge. The point R^1 is such a point, that vicinity of Q in the segment $[QR^1]$ is located inside the old cell Ω_o. In Fig. 12.7 *a*, *b*, for the side 13, R^1 is the triangle vertex 3, and in Fig. 12.7 *c* it is vertex 1. As we see in Fig. 12.7 *c*, R^1 itself, i.e., the triangle vertex 1, may be located outside of Ω_o. The point R^2 is the opposite side end-point. It is vertex 1 for the side 13 in Fig. 12.7 *a*, *b*, and vertex 3 for this side in Fig. 12.7 *c*.

The order of determining intersection points does not coincide with vertex numbering in the polygon. Thus, when considering the point Q, there are three operations:

1) to continue forming one of existing polygons on the facet $\triangle_n^{l_1}$,

2) to partition the polygon into two polygons and to enter Q into the vertex list of one of these two polygons,

3) to merge two polygons into one and to enter Q into the vertex list of this polygon.

The basic steps of constructing P^i on the facet $\triangle_n^{l_1}$ are given bellow.

1. If, when $\mathcal{L}_{on}^i$ passes from one old facet to another, a new polygon vertex arises inside the current facet of Ω_n, then a break of $\mathcal{L}_{on}^i$ is inside of the triangle, see Fig. 12.7 *b*. Then we enter Q into the vertex list of this polygon as follows:

1.1. The index j gets increment by 1 (we use notations applied in computer codes)

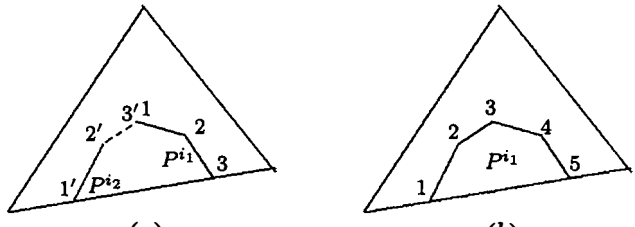
$$j{=}j{+}1,\ \ N_{tot}^i = N_{tot}^i{+}1,\ \ V_x^{ij} = Q_x,\ \ V_y^{ij} = Q_y,\ \ V_z^{ij} = Q_z\,.$$

If Q is the first point of $\mathcal{L}_{on}^i$, we begin to form a new polygon P^i on this facet, i.e., first we change

$$N_{pol}^{l_1} = N_{pol}^{l_1} + 1,\ \ i = l_1 + 12(N_{pol}^{l_1} - 1)\,, \tag{12.22}$$

and later on Q is entered into the vertex list of P^i.

1.2. Construction of P^i can begin inside of the facet $\triangle_n^{l_1}$. For instance, in Fig. 12.8, first, we construct vertices 1, 2, 3 of the polygon P^{i_1}. Further the line $\mathcal{L}_{on}^i$ goes out the facet, and later comes on it again, forming vertices $1'$, $2'$ of the new polygon P^{i_2} (conditions, whether at the second entering on the facet the line $\mathcal{L}_{on}^i$ forms P^{i_2} or continues to form P^{i_1}, are considered in item 2). The determined point $3'$ of P^{i_2} coincides with point 1 of P^{i_1}, see Fig. 12.8 *a* (the segment $2'3'$ is indicated by the dashed line). Therefore, it is necessary to merge the polygons P^{i_1}, P^{i_2} into one P^{i_1} by renumbering the vertices as shown in Fig. 12.8 *b*. In Fig. 12.8 *b*, the interiority of the polygon 12345 is not hatched because it may turn out that its exterior is located inside of Ω_o, and then we need to add the triangle $\triangle_n^{l_1}$ vertices into the vertex list of P^{i_1}.



(*a*) (*b*)

Figure 12.8:

2. The line $\mathcal{L}_{on}^i$ enters on the facet $\triangle_n^{l_1}$ through the kth side.

2.1. A new polygon is formed in the following cases:

2.1.1. There are no vertices of P^i at the kth side of $\triangle_n^{l_1}$ inside of the segment $[QR^1]$, see Fig. 12.7 *a*, *b*, i.e., when $N_{nod}^{ik}{=}0$.

2.1.2. There are already vertices of P^i at the kth side of $\triangle_n^{l_1}$ inside of the segment $[QR^2]$. In Fig. 12.9 *a*, it is the point S (intersection point between $\mathcal{L}_{on}^{i_2}$ and side 13). Knowing the number j (remind that $j{=}N_v^{ikl}$) we set $Q_x'{=}V_x^{ij}$ and determine the parameter t

$$t = \frac{Q_x'{-}Q_x}{R_x^1{-}Q_x}\,,$$

which should satisfy the inequality $t \le 0$. If $R_x^1 = Q_x$, we calculate t using y or z-coordinate.

2.1.3. There are more complicated cases presented in Fig. 12.9 *b*. Let the polygon P^{i_2} with vertices $1'2'3'4'5'$ be formed after points $1, 6$ of P^{i_1} have been found. If P^{i_2} is formed beginning with vertex $1'$, then the algorithm of item 2.1.2 specifies correctly that Q, i.e., vertex $1'$, is referred to P^{i_2}. However, if P^{i_2} is formed beginning with vertex $5'$, then Q will be incorrectly entered into the vertex list of P^{i_1}. Therefore, an additional check is required. Among points $1, 6$ we should select the closest one to point $5'$ (it is point 1), within segment $[5'1]$ take some point Q' in the vicinity of point 1, and check whether Q' is inside of Ω_o (using the algorithm B). As we see in Fig. 12.9 *b*, a vicinity of point 1 is not inside of Ω_o, and, therefore, $5'$ is entered into the vertex list of P^{i_2}.

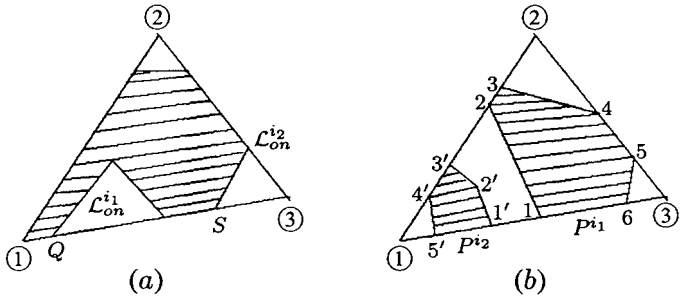


Figure 12.9:

Note that in items 2.1.1–2.1.3, when forming a new polygon, its number is given by (12.22).

Besides Q, additionally we should check whether the facet vertices are the polygon vertices. Here two variants are possible.

(A) If the point R^1, being the facet vertex, is outside of Ω_o (in Fig. 12.9 *b* it is the facet vertex 1), therefore, only Q is entered into the vertex list of the new polygon

$$N_{tot}^i = 1,\ N_{nod}^{ik} = 1,\ l{=}N_{nod}^{ik},\ N_v^{ikl}{=}1,\ j{=}N_v^{ikl},\ V_x^{ij}{=}Q_x,\ V_y^{ij}{=}Q_y,\ V_z^{ij}{=}Q_z.$$

(B) If R^1 is inside of Ω_o (in Fig. 12.9 *a* it is facet vertex 1), therefore, the both points R^1 and Q are entered into the vertex list of the new polygon

$$1)\ l{=}1,\ \ N_v^{ikl}{=}1,\ j{=}N_v^{ikl},\ V_x^{ij}{=}R_x^1,\ V_y^{ij}{=}R_y^1,\ V_z^{ij}{=}R_z^1\,,$$

$$2)\ N_{tot}^i = 2,\ N_{nod}^{ik} = 2,\ l{=}N_{nod}^{ik},\ \ N_v^{ikl}{=}2,\ j{=}N_v^{ikl},\ V_x^{ij}{=}Q_x,\ V_y^{ij}{=}Q_y,\ V_z^{ij}{=}Q_z,$$

and it is necessary to enter R^1 into the vertex list of P^i as located simultaneously at the adjacent facet edge k_1 (in Fig. 12.9 *a* for vertex 1 it is the facet edge 12 with number $k_1{=}1$), i.e., set

$$N_{nod}^{ik_1} = 1,\ N_v^{ik_1 1} = 1\,.$$

2.2. We continue to form the current polygon P^i in the following cases:

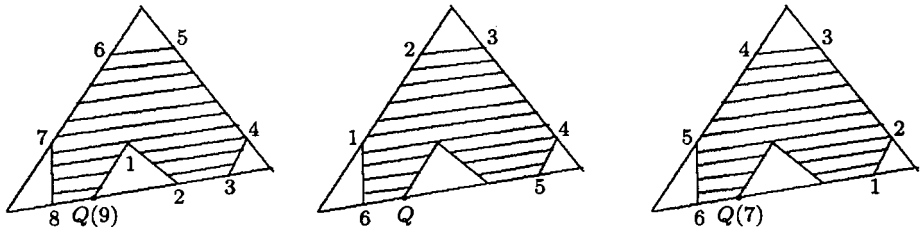


Figure 12.10:

2.2.1. There is a vertex of P^i within the segment $[QR^1]$. In Fig. 12.10 *a*, it is point 8. We enter Q into the vertex list with number 9. However, in the case presented in Fig. 12.10 *b*, where not numbered polygon vertices are considered as not determined yet, we can not enter Q with number 7 into the list, since the order of vertex numbering would be violated. First we need to change vertex numbering as shown in Fig. 12.10 *c*, and next to enter Q into the list with number 7. In the both cases the jth vertex, located in the R^1-direction from Q (in Fig. 12.10 *a* it is point 8 and in Fig. 12.10 *b* it is point 6), is the last point in the list, i.e., $j{=}N^i_{tot}$. The difference is that in Fig. 12.10 *a* two already found points at the current facet edge with numbers 8 and 2 (2 is closest to Q from R^2) are not the neighboring vertices in the list, and in Fig. 12.10 *b* points 6 and 5 are the neighboring vertices. Thus, in the second case, first, we change vertex numbering to inverse for all points, except the last point 6. Accordingly, the values of N^{ikl}_v to the polygon P^i are changed as well.

2.2.2. It may happen that the jth vertex of P^i, closest to Q from R^1, is not the last point in the vertex list, i.e., $j{<}N^i_{tot}$. Fig. 12.11 *a* presents an example when, first, we shift vertex numbering by two in clockwise sense (so that the jth vertex, $j{=}4$, becomes the last point in the vertex list), and then enter Q into the list, see Fig. 12.11 *b*. In the case, presented in Fig. 12.11 *c*, first, we change vertex numbering to inverse, next shift numbering by two in clockwise sense, enter Q into the list, and get the polygon with numbering depicted in Fig. 12.11 *b*. The difference between Fig. 12.11 *a* and *c* is the following. In Fig. 12.11 *a*, the point, closest to Q in the R^1-direction, has number $j{=}4$, and point, closest to Q in the R^2-direction, has number $j{=}5$. Therefore, vertex numbering in the polygon P^i specifies the bypass in the same anticlockwise sense as that specified by "4Q" numbering. In Fig. 12.11 *c*, vertex numbering in P^i specifies the bypass in clockwise sense, being inverse to that specified by "3Q" numbering (point 3 is in the R^1-direction from Q).

3. The line $\mathcal{L}^i_{on}$ goes out the facet $\triangle^{l_1}_n$ through the kth side.

3.1. There is one polygon on $\triangle^{l_1}_n$.

3.1.1. If there is no intersection point between $\mathcal{L}^i_{on}$ and the kth side, see Fig. 12.7 *a*, *b*, we continue to form the current polygon P^i and enter Q into the P^i vertex list by setting

$$j{=}j{+}1,\ V^{ij}_x{=}Q_x,\ V^{ij}_y{=}Q_y,\ V^{ij}_z{=}Q_z,\ N^i_{tot} = N^i_{tot}{+}1,\ N^{ikl}_v = j,\ N^{ik}_{nod} = 1.$$

In addition, if R^1 is inside of Ω_o (in Fig. 12.7 *a*, *b* it is facet vertex 3), we enter R^1 into the

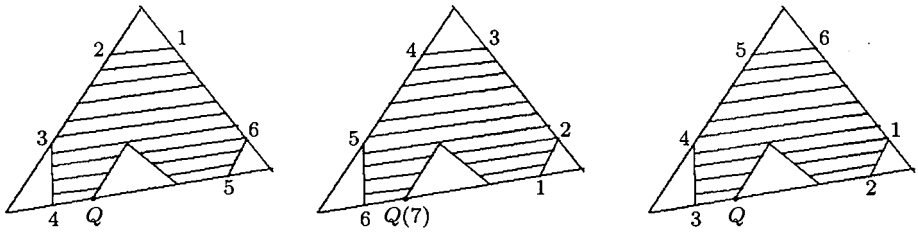


Figure 12.11:

list as well

$$j\text{–}\not{j}\text{+}1,\ V_x^{ij}\ \text{–}R_x^1,\ V_y^{ij}=R_y^1,\ V_z^{ij}=R_z^1,\ N_{tot}^i=N_{tot}^i\text{+}1,\ N_v^{ikl}=j,\ N_{nod}^{ik}=2.$$

and it is necessary to enter R^1 into the vertex list of P^i as located simultaneously at the adjacent side k_1 of the facet, i.e., to set

$$N_{nod}^{ik_1}=1,\ N_v^{ik_1 1}=1.$$

3.1.2. If there is already intersection points between $\mathcal{L}_{on}^i$ and kth side, then

3.1.2.1. If there is only one point, the point S in Fig. 12.7 *c* (in other words N_{nod}^{ik}=1), therefore, we enter Q into the vertex list.

3.1.2.2. If there are several points, i.e., $N_{nod}^{ik}>1$, one needs to know whether to continue forming the current polygon P^i or to begin constructing a new polygon on this facet. In Fig. 12.12 *a*, all determined vertices 1, 3, 4 at the current facet edge are in the R^1-direction from Q (since N_{nod}^{ik}=3 it is sufficient to check this condition only for the first j=N_v^{ik1}=1 and last j=N_v^{ik3}=4 point). Thus, we enter Q into the list with number 9. In Fig. 12.12 *b*, Q is between the already determined points 1 and 2. The left polygon is not completed yet and point 1 is still in the list of the current polygon P^i. Since point 2 is in the R^1-direction from Q, therefore, the current polygon is divided into two, P^{i_1} with vertex 1′ and P^{i_2} with vertices 1234 as shown in Fig. 12.12 *c*.

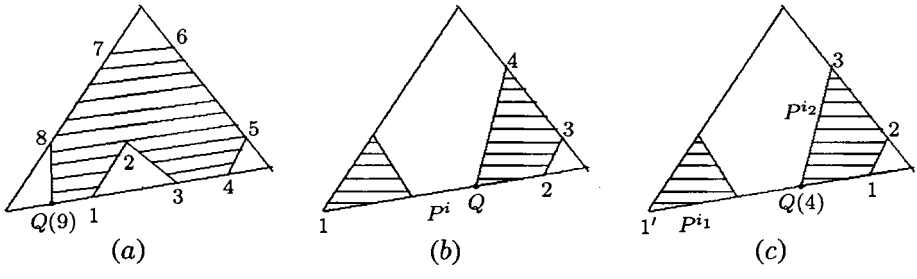


Figure 12.12:

3.2. There are two polygons on the facet $\triangle_n^{l_1}$. Here, in addition to the above cases, it may be required to merge two polygons into one. In Fig. 12.13 *a*, there are two polygons: current P^{i_2} with vertices 1, 2, 3, 4, and P^{i_1} with vertices 1′, 2′. The point Q is referred to P^{i_2}. Point 2′ is in the R^1-direction from Q, and a vicinity of 2′ within the segment $[Q2']$ is inside of the cell Ω_o. Therefore, Q and 2′ belong to one polygon. One needs to merge P^{i_1} and P^{i_2} into one polygon P^{i_1} and exclude the current polygon P^{i_2} from the polygon list, see Fig. 12.13 *b*. Note that in the final variant, Q and 1 can be not the adjacent vertices of P^{i_1} (and even not belong to one polygon), but it will be clear at further forming the polygon.

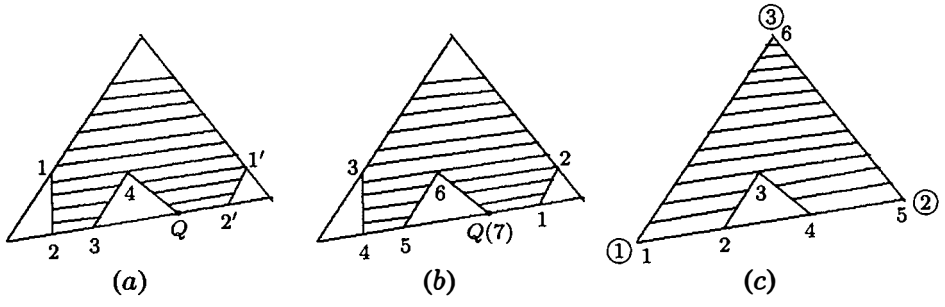


Figure 12.13:

It is possible the case depicted in Fig. 12.13 *c*. The line $\mathcal{L}_{on}^i$ intersects only one facet edge 12. Then only vertices 1, 2, 3, 4, 5 are entered into the list, meanwhile the last point 6 is not. To avoid the error in such a case, after completing construction of Ω_{on} we should check all facets of the cell Ω_n as follows. If three conditions are satisfied:

[1] there is at least one polygon on the facet $\triangle_n^{l_1}$;

[2] two vertices of the triangle $\triangle_n^{l_1}$ (vertices 1, 2 in Fig. 12.13 *c*) are inside of the cell Ω_o;

[3] there is one polygon vertex on every of two sides joining to the third facet vertex, in Fig. 12.13 *c*, it is the sides 13 and 23 (with corresponding numbers k=2, 3), and, therefore, the variables N_{nod}^{i2}=1, N_{nod}^{i3}=1, since the facet vertices 1, 2, entered into the polygon list with numbers 1, 5, are located at the facet edges 13 and 23 as well;

then the third facet vertex (in Fig. 12.13 *c*, it is with number 3) is also entered into the polygon vertex list.

Even if the facets $\triangle_n^{l_1}$ and $\triangle_o^{l_2}$ do not intersect each other, it can be that the entire facet $\triangle_n^{l_1}$ is inside of the cell Ω_o. Therefore, the entire facet $\triangle_n^{l_1}$ is the part of the Ω_{on} boundary.

Thus, we described all basic cases, which can arise when constructing the simplex Ω_{on}. All other cases are combinations of the basic ones. Similarly, we consider formation of the polygons for the cell Ω_o facets.

After constructing polygons on the facets of Ω_n and Ω_o, we should verify that vertex numbering in every polygon specifies the outward normal vector to the surface of Ω_{on}.

To this end, we consider the kth triangular facet edge, where there are at least two polygon vertices, i.e., $N_{nod}^{ik} \geq 2$, and verify whether bypass of the facet vertices $1 \rightarrow 2 \rightarrow 3$ is performed in the same sense as that specified by polygon vertex numbering. If they are in the opposite senses, we change the polygon vertex numbering to inverse. However, there is possible a situation when all polygon vertices are inside of the facet, say $\triangle_n^{l_1}$, see Fig. 12.14 *a*. Then we verify the bypass direction of vertex numbering as follows. Assume the polygon P^i to be the "star domain"[2]. We take a central point O, see Fig. 12.14 *b*, join it with two consecutive polygon vertices, for instance 2 and 3, and obtain two vectors, $\boldsymbol{r}_2$ and $\boldsymbol{r}_3$. The vector product $\boldsymbol{r_2} \times \boldsymbol{r_3}$ should specify the outward normal vector to the facet $\triangle_n^{l_1}$ (with respect to Ω_n). If it specifies the inward normal vector, we change polygon vertex numbering to inverse. Numerical experiments have shown that as the central point O it can be used the point with coordinates

$$\boldsymbol{r}_o = \frac{1}{M} \sum_{m=1}^{M} \boldsymbol{r}_m \,,$$

where $\boldsymbol{r}_m = (x_m, y_m, z_m)$ are the vertex coordinates, M is the number of polygon vertices, $M = N_{tot}^i$.

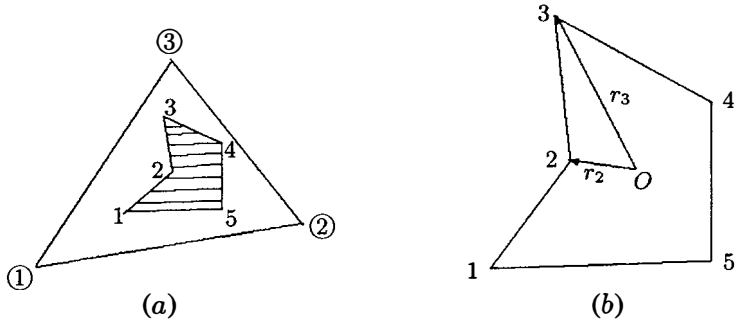


Figure 12.14:

We present two examples of constructing the intersection lines $\mathcal{L}_{on}^i$. First example, see Fig. 12.15, demonstrates intersection between two identical cubes, one translated with respect to the other by $\triangle \boldsymbol{r}$. In this rather a simple case some polygons are triangles and the other pentagons. In the next example, we construct the intersection lines between two convex dodecahedrons, Fig. 12.16 *a*, *b*. The overlapping figure is rather complicated: there are three intersection lines, see Fig. 12.16 *c*.

[2]Some region is a star domain if inside of it there is a point, referred to as the central point (or pole), which can be joined with any boundary point by a straight line without intersection the boundary at another point.

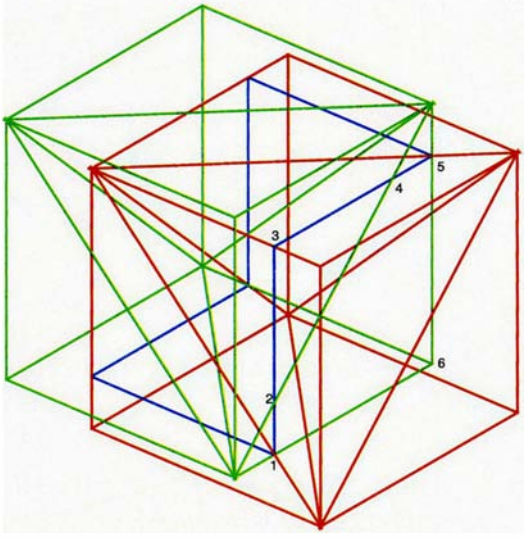


Figure 12.15: Two cubes, the first is drawn with red color and second with green. The overlapping figure is the cube as well. For instance, its facet 123456 consists of the triangle 234 and pentagon 12456, carved by the intersection line L_{on} of dark blue color on the triangular facets of the green dodecahedron.

12.3.3 Stage III. Calculation of Ω_{on} Volume and Mass

The volume of the overlapping figure Ω_{on} is calculated via the surface integral (12.4)

$$V = \frac{1}{3}\int\int_S \boldsymbol{r}\cdot\boldsymbol{n}\, dS\,.$$

Therefore, we need to integrate over all polygonal facets forming the simplex Ω_{on} boundary. A contribution of a polygonal facet to the Ω_{on} volume is determined via (12.5), a contribution of a triangular facet. Therefore, first, one need to divide the polygon into triangles as shown in Fig. 12.17. A contribution of the polygon depicted in Fig. 12.17 to the volume is

$$V_{12345678} = V_{123} + V_{134} + V_{145} + V_{156} + V_{167} + V_{178}\,.$$

Here the negative volume, contributed by triangle 123, is taken into account automatically.

Since every hex cell is represented as a pair of dodecahedrons, we determine the volumes V^{kl} and masses m_{on}^{kl} of four overlapping figures Ω_{on}^{kl}, where k, l=1, 2. Here Ω_{on}^{kl} is the overlapping figure between the kth dodecahedron of the new hex cell and lth dodecahedron of the old hex cell. After determining four volumes V^{kl}, we can calculate the part of the mass of the jth hex old cell contributed to the mass of the new hex cell

$$\tilde{m}_{on}^{j} = \frac{1}{4}\sum_{k,l=1}^{2} m_{on}^{kl} = \frac{1}{4}\rho_o^j \sum_{k,l=1}^{2} V^{kl}\,,$$

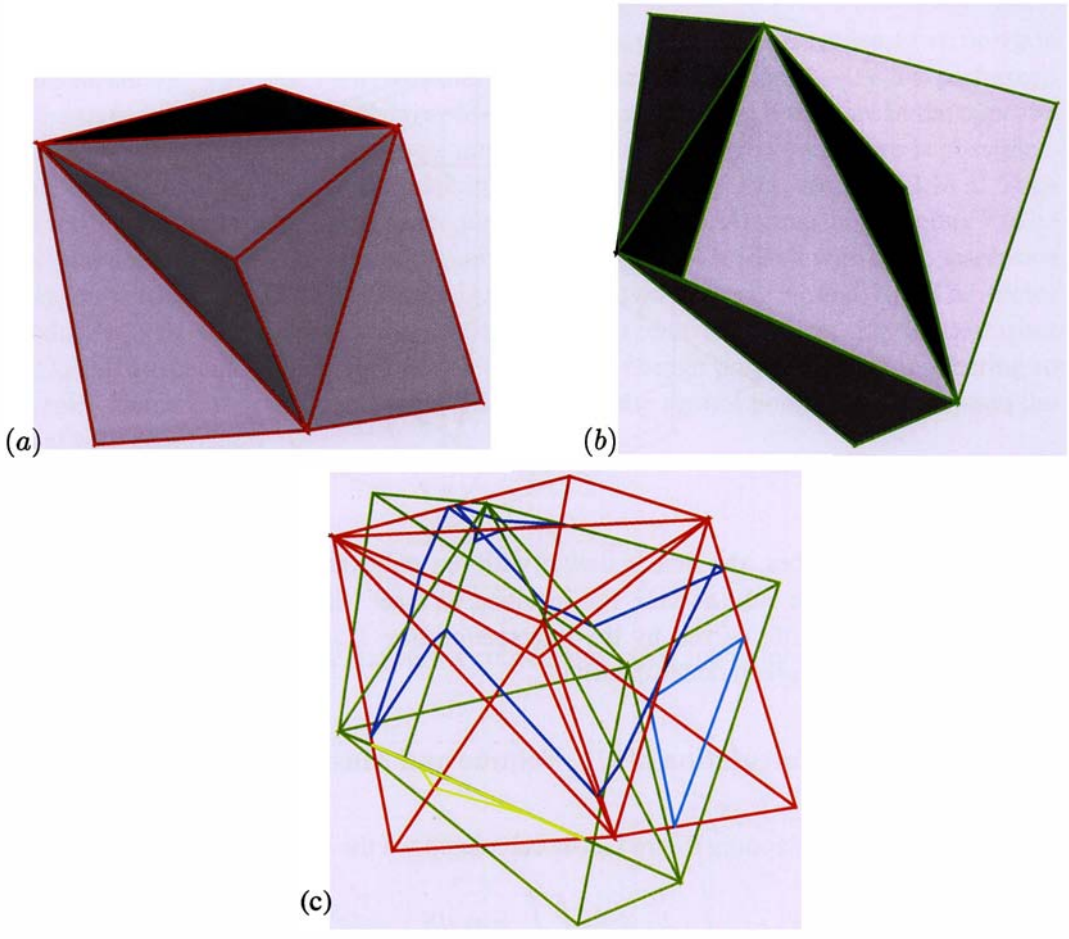


Figure 12.16: Two convex dodecahedrons (a), (b) and their intersection (c). There are three intersection lines of dark blue, blue, and yellow color.

where ρ_o^j is the density in the jth hex old cell. To calculate the mass of the underlying new cell we need to find the set of old cells giving nonempty intersection with this new cell.

One can see that the algorithm can treat non-convex dodecahedrons and, therefore, non-convex hex cells. The only requirement is that old and new mesh, both hex and dodecahedral, must be non-degenerate. Conditions of nondegeneracy for a hex cell were considered in [4] and chapter 9.

12.3.4 Stage IV. Algorithm of Selecting Cells

We need to find all old cells Ω_o^j, j=1, . . ., J with nontrivial intersection relating to current Ω_n. The set of such old cells we refer to as the Ω_n environment, see the example of two-dimensional grids in Fig. 12.18. If to check all cells of the mesh ω_o, the running time of the

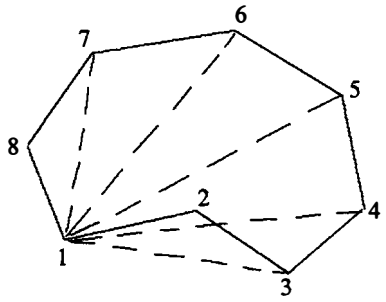


Figure 12.17: Polygon 12345678 is divided into triangles with the purpose to calculate the contribution to the volume of Ω_{on}.

interpolation procedure will be too large. We apply the optimal filtering algorithm of the mesh ω_o so as to proceed only Ω_o^j with nontrivial intersection.

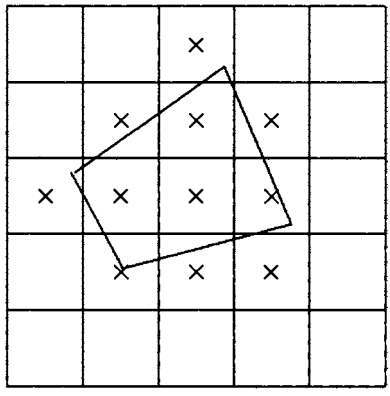


Figure 12.18: Example of two-dimensional grids. Old cells, specified by the sign "×", form the environment of the current new cell.

To find the Ω_n environment we use the selection algorithm as a tree. Let for Ω_n we know some old cell $\Omega_o^{i_c j_c k_c}$ (we use the global numeration) so that

$$\Omega_n \bigcap \Omega_o^{i_c j_c k_c} \neq \emptyset\,, \tag{12.23}$$

i.e. their intersection is a nonempty set. The cell $\Omega_o^{i_c j_c k_c}$ is referred to as the center of the cell Ω_n environment. As the center we can take any old cell satisfying the condition (12.23).

In the first dodecahedron, depicted in Fig. 12.2a, two triangular facets 124 and 234 correspond to the direction of decreasing the index k (referred to as the direction $k-$), see Fig. 12.19, and facets 567, 578 to the direction of increasing the index k (direction $k+$). Such a conformity is drawn from the original hex cell with faces 1234 and 5678. Similarly, the facets 347 and 487 specify the direction $j+$, 125 and 265 indicate the direction $j-$. The pairs 237, 276 and 158, 184 specify the directions $i+$ and $i-$, respectively. Thus, every pair of facets corresponds to six directions of changing the indices

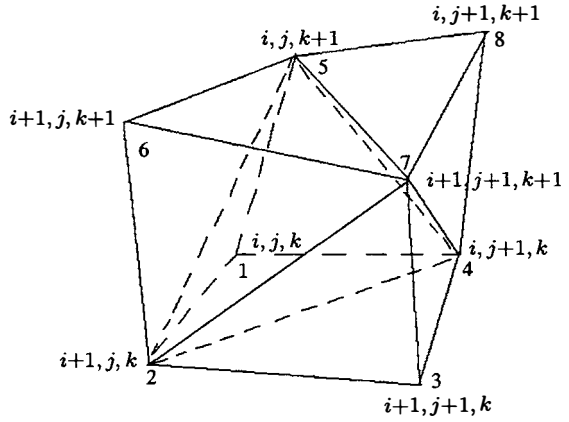


Figure 12.19: Global and local index numeration.

direction l	1	2	3	4	5	6
index change	$k-$	$k+$	$j-$	$j+$	$i-$	$i+$

When constructing Ω_{on} we find those facets of $\Omega_o^{i_c j_c k_c}$ which intersect facets of Ω_n, and, therefore, the index change directions where we should check the old cells. We write the found directions into the integer array $Inters^{i_c j_c k_c l}$. Here the index l specifies the directions of selecting the old cells, see Fig. 12.20. The value $Inters^{i_c j_c k_c l}$=0 indicates that we need not select the old cells in the l-direction (the facets of $\Omega_o^{i_c j_c k_c}$, corresponding to the l-direction, do not intersect the boundary of Ω_n). The value $Inters^{i_c j_c k_c l}$=1 indicates that we should select the old cells in the l-direction. However, there is an exception. It may be that two facets of Ω_o, corresponding to the l-direction, do not intersect the boundary of Ω_n, but they are inside of this cell. Therefore, even if $Inters^{i_c j_c k_c l}$=0 we should verify

(*) Whether any of four vertices of these two facets is inside of the cell Ω_n. If it is so, we set $Inters^{i_c j_c k_c l}$=1 and should select the old cells in the l-direction.

We introduce the integer arrays $Indic^{ijk}$ and $Increm^{ijk}$. The former indicates whether the cell Ω_o^{ijk} has been checked (value "1") or not yet (value "0"). The latter specifies the direction of returning to the up tree level, i.e., to the cell from which we came to the current cell Ω_o^{ijk} (in Fig. 12.20 it is the l_1-direction). For instance, if we arrived at the current cell from the up level cell in the direction l=3, then we return in the direction l=4. We should return from the current old cell to the up level cell if

1) $Inters^{ijkl}$=0 for five l-directions (in Fig. 12.20, it is the directions $l_2, \ldots, l_6$); here we also should execute the above check (*) for the facet vertices of all five directions; or

2) all five low level cells have been already checked for intersection, in other words when for all these cells $Indic^{ijk}$=1.

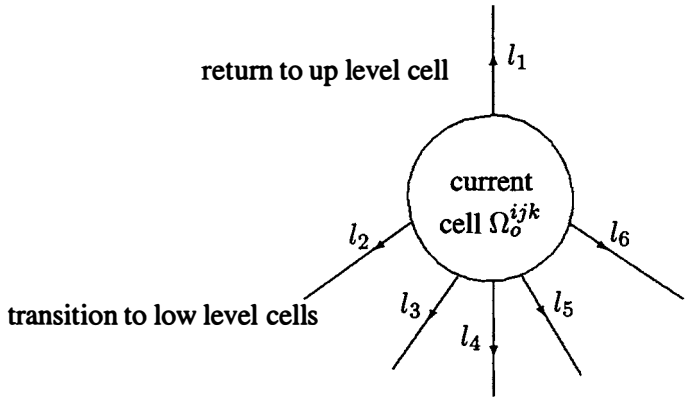


Figure 12.20: For the current cell Ω_o^{ijk} the parameter l specifies six directions of changing the indices i, j, k. Here $l{=}l_1{=}Increm_{ijk}$ is the direction of returning to the up tree level, $l{=}l_2, \ldots, l_6$ are the directions of transition to the low tree level.

Besides, if the current cell Ω_o^{ijk} is the environment center, i.e., $\Omega_o^{i_c j_c k_c}$, therefore, we have selected all old cells, satisfying the condition (12.23), and determined the current cell Ω_n environment.

When treating the current cell Ω_o^{ijk}, we verify whether its boundary intersects any two triangular facets of Ω_n, specifying the cell selection direction for the new grid ω_n. If yes, this cell Ω_o^{ijk} will be the environment center for the next new cell, and for the next Ω_n we define $i_c{=}i$, $j_c{=}j$, $k_c{=}k$. There can be several such old cells. We store the number of the last one.

If no of two Ω_n facets, specifying the selection direction for the new grid, has intersection with Ω_o^{ijk}, as the environment center we use the old cell inside of which there is any of four vertices forming these two facets of Ω_n.

If Ω_n is inside of the underlying old cell, the indices i_c, j_c, k_c remain the same for the next new cell.

Due to the check (*) this algorithm will also verify the old cells entirely located inside of Ω_n, see the example of two-dimensional grids in Fig. 12.18, where one old cell is inside of the new cell.

Such an algorithm of finding the environment center requires that we perform consecutive selection of the new cells, as shown for the ith odd cell layer in Fig. 12.21. Transition from the ith cell layer to $i{+}1$th should be through the cell $\Omega_n^{ij_{max}k_{max}}$. Selection is performed in reverse order in the $i{+}1$th cell layer.

The cell environment, found for the first of two dodecahedrons Ω_n, related to the new hex cell, which consists of the first 12-faceted cells Ω_o^j (we again use the local numeration of the old cells $j = 1, \ldots, J$, related to the underlying new cell), is also used for the other three combinations: first Ω_n and second Ω_o^j cells, etc.

The mass of the underlying new cell is obtained by summation of the environment cell

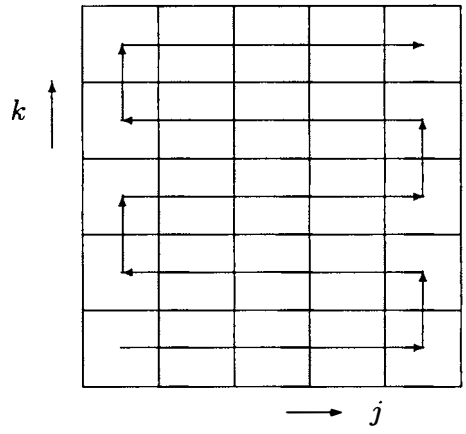


Figure 12.21: Selection order in the ith odd cell layer of ω_n.

parts

$$m_n = \sum_j \tilde{m}_{on}^j \, .$$

Finally, the density in the new hex cell is given

$$\rho_n = \frac{m_n}{V_n} \, .$$

12.4 Construction of Fictitious Cell Layers

The facets of the new and old cell should not coincide, otherwise we will get division by zero when trying to find the intersection between two coinciding planes. To avoid such situations we slightly shift the mesh ω_n with respect to ω_o by $\triangle \boldsymbol{r}=(\triangle x, \triangle y, \triangle z)$. Since the reference domain for the old and new meshes is the same (we interpolate in one space domain G), due to the shift some boundary cells of ω_n go outside of G. In order not to lose the mass when remapping, we extend the old mesh ω_o by adding one fictitious cell layer on the G boundary in all of six l-directions [2]. Such an extended old mesh is referred to as ω_o^{ext}.

Generally, construction of ω_o^{ext} is not difficult. In parametric space $\mathbb{R}^3$, where the mesh is cubic and every cell is a unit cube, we add one more layer of such cubes on the boundary for all of six sides. If to consider the trilinear mapping of the unit cube onto a hex cell (12.1), then, for instance for the left boundary in the ξ-direction, the coordinates of the fictitious cell four vertices can be obtained from (12.1) by setting $\xi=-1$ and $\eta, \zeta=0, 1$. For the left fictitious node $\boldsymbol{r}_{0,j,k}$ one obtains

$$\boldsymbol{r}_{0,j,k} = 2\boldsymbol{r}_{1,j,k} - \boldsymbol{r}_{2,j,k} = \boldsymbol{r}_{1,j,k} + p_i^l(\boldsymbol{r}_{1,j,k} - \boldsymbol{r}_{2,j,k}) \, , \qquad (12.24)$$

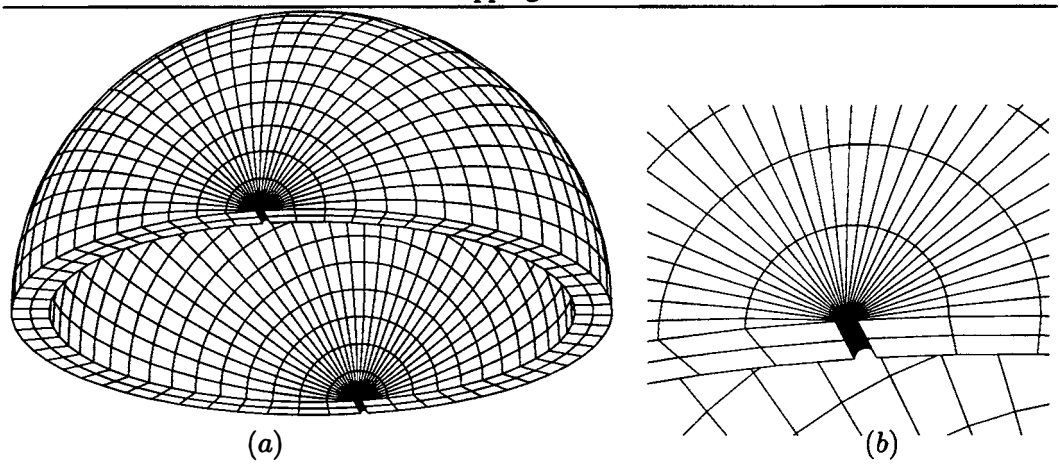


Figure 12.22: Cylindrical mesh ω_o : 3×31×31 (a); close-up near the symmetry axis (b).

where p_i^l=1 is the extrapolation parameter for the left boundary. Similarly, on the right boundary in the ξ-direction the fictitious node coordinates are calculated with the extrapolation parameter p_i^r=1, and on the other four boundaries we construct the fictitious nodes with p_j^l=p_j^r=p_k^l=p_k^r=1.

However, sometimes we need to vary the extrapolation parameters. The cylindrical mesh ω_o : 3×31×31 is shown in Fig. 12.22. Construction of the fictitious cells by formulae similar to (12.24), with all extrapolation parameters equal to 1, gives the extended mesh ω_o^{ext} : 5×33×33 depicted in Fig. 12.23. One can see the fictitious cells in the layers j=0, J+1 (where J=30) are degenerate. We set the parameters p_j^l (p_j^r) so that the fictitious nodes $\boldsymbol{r}_{i,0,k}$ ($\boldsymbol{r}_{i,J+1,k}$) locate between the nodes $\boldsymbol{r}_{i,1,k}$ ($\boldsymbol{r}_{i,J,k}$) and symmetry axis, but not to the left (right) from the symmetry axis as shown in Fig. 12.23 b. We define p_j^l=p_j^r=0.05 and obtain the non-folded mesh, see Fig. 12.24. If the extrapolation parameters are too small then the shifted mesh ω_n can go outside of the boundary of ω_o^{ext} and this leads to the interpolation error.

One more example concerns the regular mesh, see Fig. 12.25. On the x- and y-axis, the cells are prismatic (two hex faces are in one plane). Using the formulae similar to (12.24) leads to result presented in Fig. 12.26. In the ith layer, three fictitious cells, joining the prismatic ones, are degenerate. To generate suitable fictitious cells, the node coordinates of three adjacent nodes, joining the corner node $\boldsymbol{r}_{i,1,1}$, should be given as

$$\boldsymbol{r}_{i,0,0} = (\boldsymbol{r}_{i,0,2} + \boldsymbol{r}_{i,2,0} + \boldsymbol{r}_{i,1,1})/3\,, \quad \boldsymbol{r}_{i,1,0} = (\boldsymbol{r}_{i,0,2} + \boldsymbol{r}_{i,0,0})/2\,,$$
$$\boldsymbol{r}_{i,1,0} = (\boldsymbol{r}_{i,2,0} + \boldsymbol{r}_{i,0,0})/2\,, \qquad (12.25)$$

where the fictitious nodes $\boldsymbol{r}_{i,0,2}$, $\boldsymbol{r}_{i,2,0}$ are considered as already found. Similarly we obtain the other set of three nodes, joining every corner node in the ith layer, such as $\boldsymbol{r}_{i,J+1,1}$, $\boldsymbol{r}_{i,1,K+1}$, and $\boldsymbol{r}_{i,J+1,K+1}$ (where K=30). The resulted non-folded mesh is presented in Fig. 12.27.

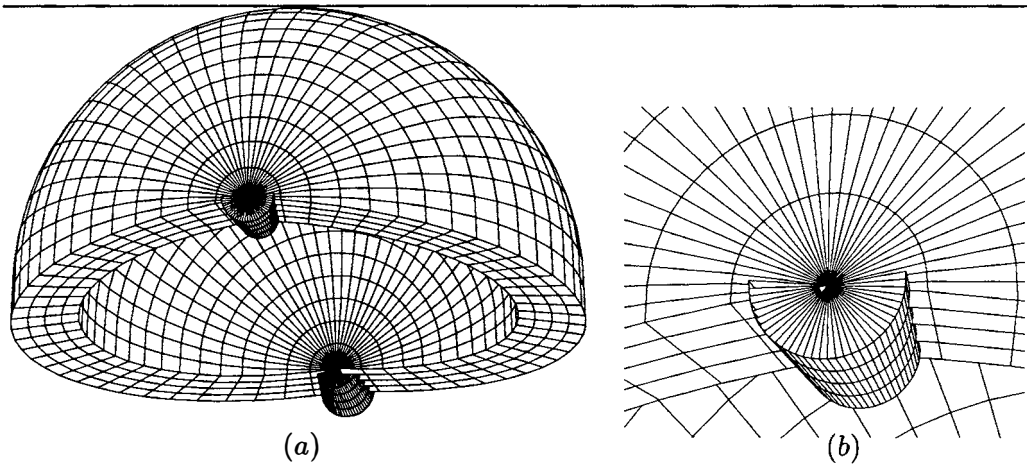


Figure 12.23: Folded extended mesh ω_o^{ext} : 5×33×33 with p_j^l=p_j^r=1 (a). Close-up near the symmetry axis (b); boundary nodes $\boldsymbol{r}_{i,0,k}$ are to the left from the symmetry axis, that is why the cells of the left fictitious layer in the j-direction are degenerate.

12.5 Interpolation Error

The interpolation error is the result of two type errors. The first error is due to round-off during calculations. When increasing computer digit capacity this error tends to zero. The second, systematic error, is due to we need to shift the mesh ω_n by $\triangle\boldsymbol{r}$=$(\triangle x, \triangle y, \triangle z)$ with respect to ω_o. Let $G_n, G_o \in \mathbb{R}^3$ denote the domains related to the meshes ω_n, ω_o, respectively (G_o coincides with the initial domain G). To understand why the systematic error arises, we consider an example of remapping from the parallelepipedal mesh ω_o with I×J×K cells and spacings h_x, h_y, h_z onto the same mesh ω_n, shifted by $\triangle\boldsymbol{r}$, see Fig. 12.28, provided that $|\triangle\boldsymbol{r}|\ll h_x, h_y, h_z$. We assume that $\triangle x, \triangle y, \triangle z>0$. If $\rho_{i,j,k}$ is the density in the old cell Ω_o^{ijk}, the mass of the domain G_o is

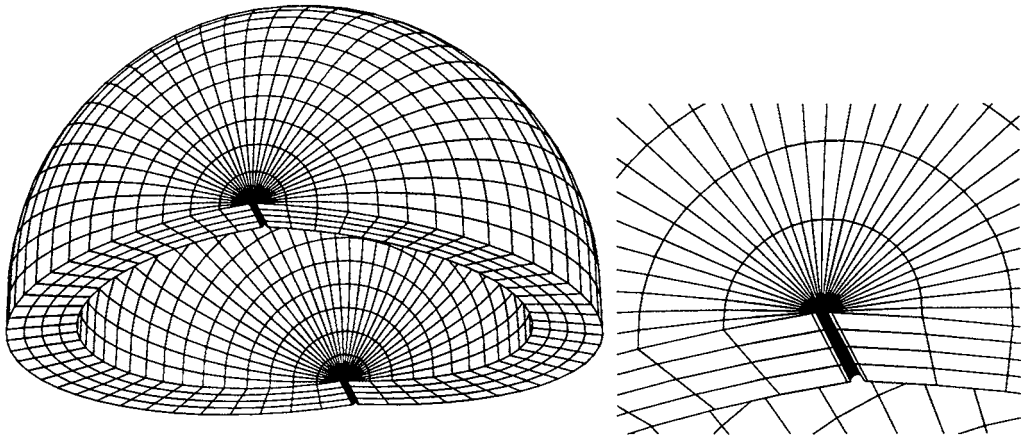
$$m_o = h_x h_y h_z \sum_{i,j,k=1}^{I,J,K} \rho_{i,j,k} \,.$$

The mass of G_n is a combination of the domain G_{on} mass (where G_{on}=$G_o \cup G_n$)

$$m_{on} = m_o - h_x h_y \triangle z \sum_{i,j=2}^{I,J} \rho_{i,j,1} - h_x h_z \triangle y \sum_{i,k=2}^{I,K} \rho_{i,1,k} - h_y h_z \triangle x \sum_{j,k=2}^{J,K} \rho_{1,j,k}$$

$$- h_x \triangle y \triangle z \sum_{i=2}^{I} \rho_{i,1,1} - h_y \triangle x \triangle z \sum_{j=2}^{J} \rho_{1,j,1} - h_z \triangle x \triangle y \sum_{k=2}^{K} \rho_{1,1,k} - \triangle x \triangle y \triangle z \rho_{1,1,1} \,,$$

and mass m' of the additional domain G'=$G_n \backslash G_{on}$, occurring due to ω_n is shifted with



(a) (b)

Figure 12.24: Non-folded extended mesh ω_o^{ext} : 5×33×33 with p_j^l=p_j^r=0.05 (a). Close-up near the symmetry axis (b); boundary nodes $\boldsymbol{r}_{i,0,k}$ locate between the nodes $\boldsymbol{r}_{i,1,k}$ ($\boldsymbol{r}_{i,J,k}$) and symmetry axis.

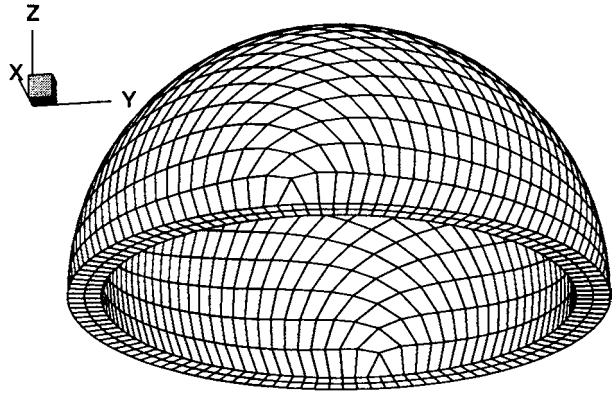


Figure 12.25: Regular mesh ω_o : 3×31×31. On the x- and y-axis the cells are prismatic.

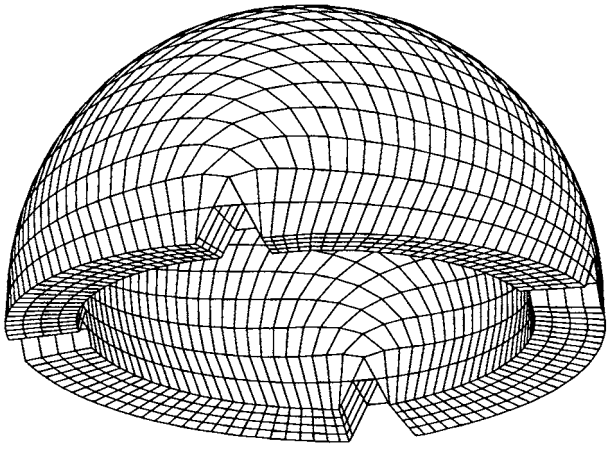


Figure 12.26: Extended mesh ω_o : $5\times33\times33$. Fictitious cells, adjacent to prismatic ones, and generated via formulae (12.24), are degenerate.

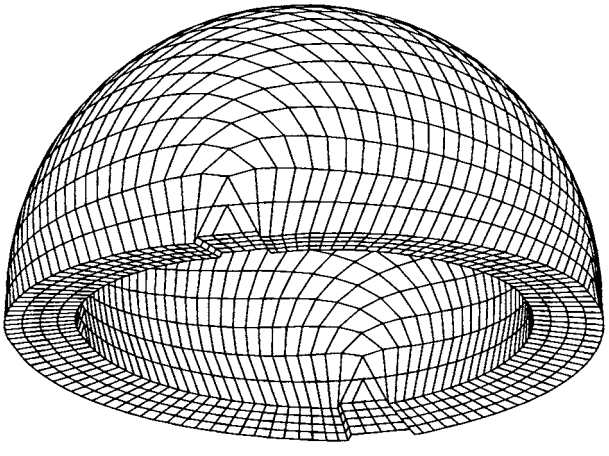


Figure 12.27: Extended mesh ω_o : $5\times33\times33$. Fictitious cells, adjacent to prismatic ones, and generated via formulae (12.25) are non-folded.

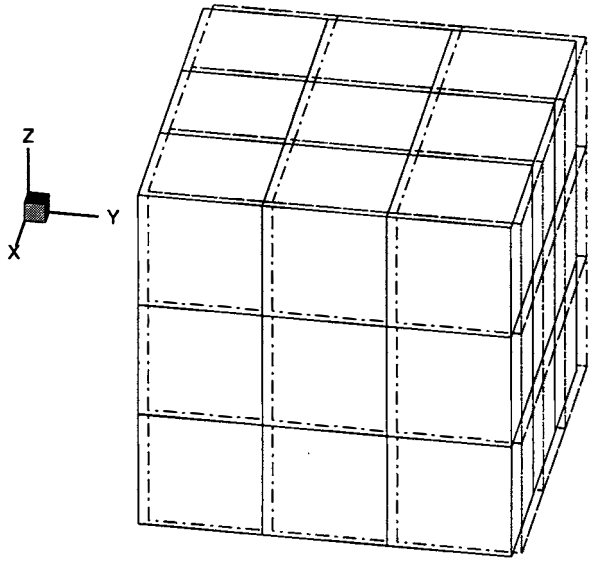


Figure 12.28: Parallelepipedal mesh $\omega_o : I \times J \times K$ and the same new mesh ω_n, shifted by $\triangle r$ with respect to ω_o.

respect to ω_o

$$m' = h_x h_y \triangle z \sum_{i,j=2}^{I,J} \rho_{i,j,K+1} + h_x h_z \triangle y \sum_{i,k=2}^{I,K} \rho_{i,J+1,k} + h_y h_z \triangle x \sum_{j,k=2}^{J,K} \rho_{I+1,j,k}$$

$$+ h_x \triangle y \triangle z \sum_{i=2}^{I} \rho_{i,J+1,K+1} + h_y \triangle x \triangle z \sum_{j=2}^{J} \rho_{I+1,j,K+1} + h_z \triangle x \triangle y \sum_{k=2}^{K} \rho_{I+1,J+1,k}$$

$$+ \triangle x \triangle y \triangle z \rho_{I+1,J+1,K+1} \cdot$$

Here $\rho_{i,j,K+1}$, $\rho_{i,J+1,k}$, $\rho_{I+1,j,k}$ are the density in the fictitious cells on the right boundary in the i, j, k-directions. Summing m_{on} and m' we get m_n, the mass of the domain G_n. The change of mass after remapping, the systematic error $\triangle m_s$, is

$$\triangle m_s = m_n - m_o = h_x h_y \triangle z \sum_{i,j=2}^{I,J} (\rho_{i,j,K+1} - \rho_{i,j,1}) + h_x h_z \triangle y \sum_{i,k=2}^{I,K} (\rho_{i,J+1,k} - \rho_{i,1,k})$$

$$+ h_y h_z \triangle x \sum_{j,k=2}^{J,K} (\rho_{I+1,j,k} - \rho_{1,j,k}) + h_x \triangle y \triangle z \sum_{i=2}^{I} (\rho_{i,J+1,K+1} - \rho_{i,1,1})$$

$$+ h_y \triangle x \triangle z \sum_{j=2}^{J} (\rho_{I+1,j,K+1} - \rho_{1,j,1}) + h_z \triangle x \triangle y \sum_{k=2}^{K} (\rho_{I+1,J+1,k} - \rho_{1,1,k})$$

$$+ \triangle x \triangle y \triangle z (\rho_{I+1,J+1,K+1} - \rho_{1,1,1}) \, .$$

The systematic error $\triangle m_s$ depends on the density value in the fictitious cells. It is reasonable to extrapolate smoothly ρ in the fictitious cells, say, to set ρ equal to the value in the adjacent cell of ω_o, e.g. in the i-direction $\rho_{I+1,j,k}{=}\rho_{I,j,k}$. If the density in the opposite boundary cell layers in every direction is the same, i.e., $\rho_{1,j,k}{=}\rho_{I,j,k}$, etc., then the systematic error is equal to zero, and the total error is equal to the round-off error.

Suppose on ω_o we have $\rho_{i,1,k}{=}\rho_{i,J,k}$ and $\rho_{i,j,1}{=}\rho_{i,j,K}$. Then the relative systematic error is

$$\varepsilon_s \approx \frac{h_y h_z \triangle x \sum\limits_{j,k=2}^{J,K} (\rho_{I+1,j,k} - \rho_{1,j,k})}{h_x h_y h_z \sum\limits_{i,j,k=1}^{I,J,K} \rho_{i,j,k}} = O(\triangle x / h_x) \, , \tag{12.26}$$

Therefore, ε_s is proportional to the ω_n relative shift in the x-direction. In general case ε_s is a sum of three relative systematic errors in three coordinate directions.

We can define ρ to be the same in the fictitious cells in the opposite boundary cells, for instance, in the i-direction as $\rho_{I+1,j,k}{=}\rho_{1,j,k}$. However, if $\rho_{1,j,k} \neq \rho_{I,j,k}$, then, keeping global conservation of remapping, we obtain a local discontinuity of the density in the right cell layer (in the i-direction) of the mesh ω_n. Thus, we can talk about local conservation of remapping only in the domain G_{on}. Failure of global conservation (change of the domain G_n mass relatively G_o mass) is due to the following. When shifting ω_n by $\triangle \boldsymbol{r}$, we cut off a volume with *some* density from the left (in the i, j, k-directions) of G_n and add to the right the same volume with *another* density. Similar occurs on the curvilinear meshes.

If on the old mesh the density everywhere is equal to unity, i.e., $\rho_o(\boldsymbol{r}){\equiv}1$, after remapping on the new mesh we must have $\rho_n(\boldsymbol{r}){\equiv}1$. This criteria was suggested in [2] for the check whether the remapping is successful. We use it as follows. If after treating a particular new hex cell with indices i, j, k, the condition

$$|\rho_n^{ijk} - 1| < \epsilon,$$

where ϵ is small enough, fails, we slightly shift only this new cell and execute interpolation in this cell again.

12.6 Numerical Examples

First consider mass remapping from the curvilinear mesh onto the parallelepipedal one, see Fig. 12.29. The part of ω_o in the domain, bounded by the surfaces

$$x = 0 \, , \quad x = 1 \, , \quad y = 0 \, , \quad y = 1 \, , \quad z = 0 \, , \quad z = z_b = 2 + 0.6 \, Cos(2\pi xy)$$

with $15{\times}15{\times}25$ nodes, is generated using the algorithm in [5]. The top part $z_b{\leq}z{\leq}4$ supplements the initial domain to the parallelepiped $0{\leq}x, y{\leq}1$, $0{\leq}z{\leq}4$. Next, by the symmetric reflection about the planes $x{=}0$ and $y{=}0$ we supplement it to the parallelepiped

$-1 \le x, y \le 1$, $0 \le z \le 4$. The mesh ω_o is of $29 \times 29 \times 49$ nodes. The density on ω_o is given as

$$\rho(x,y,z) = \begin{cases} \rho_b\,, & \text{if } z < z_b - 0.43 \\ \rho_b + (\rho_t - \rho_b)\{1 - exp[-\alpha(z - z_b + 0.43)^2]\}\,, & \text{otherwise,} \end{cases}$$

where $\rho_b=1$, $\rho_t=100$, $\alpha=4$. Fig. 12.30 presents the density on the z-axis. The new mesh is parallelepipedal $\omega_n : 31 \times 31 \times 51$. We execute two variants of remapping. In the first variant the density is extrapolated smoothly in the fictitious cells, e.g. in the top layer $K+1$ (where $K=48$) we set $\rho_{i,j,K+1}=\rho_{i,j,K}$, and in the bottom layer $\rho_{i,j,0}=\rho_{i,j,1}$. In the i,j-directions due to symmetry we extrapolate the density smoothly as well. The shift of ω_n with respect to ω_o is given as

$$\triangle \boldsymbol{r} = h_{sc}(\triangle x, \triangle y, \triangle z) = h_{sc}(-0.006, -0.005, -0.007)\,,$$

where $h_{sc}=0.0714$ is the specific length. The projection of the coordinate surfaces $j=7$ and $j=1$ onto the plane $x-z$ is shown in Figs. 12.31 *a*, 12.32 *a*. Figs. 12.31 *b*, 12.32 *b* present the density plots in the layers $j=7, 1$ of ω_o, and Figs. 12.31 *c*, 12.32 *c* present in the layers $j=7, 1$ of ω_n. Relative errors to the mass and volume are

$$\varepsilon_m = \frac{|\triangle m|}{m_o} = 2.464 \cdot 10^{-4}\,, \quad \varepsilon_v = \frac{|\triangle V|}{V_o} = 1.518 \cdot 10^{-6}\,.$$

Note that $\varepsilon_v=\varepsilon_m$ when $\rho \equiv 1$.

In the second variant to exclude the systematic error we define the density in the fictitious cells as follows: $\rho_{i,j,K+1}=\rho_{i,j,1}$, $\rho_{i,j,0}=\rho_{i,j,K}$. In this case the relative error for the mass is

$$\varepsilon_m = 2.802 \cdot 10^{-6}\,,$$

and for the volume it is the same as that of the first variant. We see that presence of the systematic error in the first variant leads the total error to increase by factor of ≈ 90. On the other hand in the second variant, the density in the bottom layer of ω_n is equal to 1.618 instead of 1 as it was on the mesh ω_o (see the value of ρ at $z=0$ in Fig. 12.30). Thus, keeping global conservation, we obtain a nonphysical discontinuity of ρ in the bottom cell layer of ω_n.

Since the mesh ω_o in the bottom and top cell layers in the z-direction is close to parallelepipedal and uniform, one can expect that the estimate (12.26) is valid for the systematic error. To verify it we execute two more calculations when varying the shift component $\triangle z$ and with fixed $\triangle x$, $\triangle y$. The density is extrapolated smoothly as that of the first variant. The errors for all three calculations are shown in the table below.

$\triangle z / h_{sc}$	ε_m	ε_v
-0.0035	$1.161 \cdot 10^{-4}$	$1.5692 \cdot 10^{-6}$
-0.0070	$2.464 \cdot 10^{-4}$	$1.5181 \cdot 10^{-6}$
-0.0140	$5.685 \cdot 10^{-4}$	$3.1981 \cdot 10^{-5}$

In the first and second cases, ε_v is nearly constant, meanwhile ε_m, in accordance with (12.26), increases by factor of ≈ 2. The error ε_v characterizes the round off error, and, since $\varepsilon_v \ll \varepsilon_m$, the error ε_m characterizes the systematic one. In the third case, ε_v increases by factor of 20 owing to the round off error enlarges. The round off error is casual, it occurs when, for instance, some old and new triangular facets are closed and nearly parallel. Nevertheless, the dependence (12.26) is kept, ε_m increases a bit less than by factor of 2, since the systematic error is greater than the round off error by two orders of magnitude. Note that here the curvilinear mesh geometry can also distort the dependence (12.26).

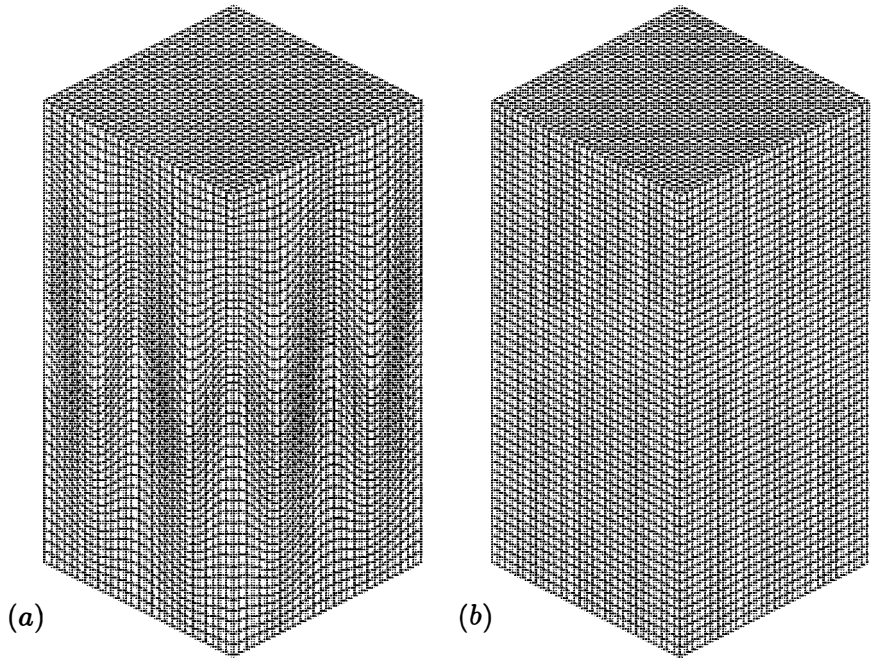


Figure 12.29: Curvilinear mesh ω_o: $29\times29\times49$ (a), parallelepipedal mesh ω_n: $31\times31\times51$ (b). The reference domain is $-1\leq x, y\leq 1$, $0\leq z\leq 4$.

The second example is remapping from the cylindrical mesh ω_o : $21\times61\times61$ onto regular ω_n : $21\times61\times61$, see Fig. 12.33. The reference domain is the hull enclosed between two half-spheres with radiuses R_1=7, R_2=8 and center at the point $(0, 0, 0)$. The density on the mesh ω_o is given as

$$\rho = 100 - 99\exp\left[-0.25(y - \sin 0.5\pi xy)^2\right] .$$

In the fictitious cells of the extended old mesh the density is extrapolated smoothly. Density plots in the top cell layer i=20 of ω_o are presented in Fig. 12.34 a, and after remapping in the same layer of the mesh ω_n in Fig. 12.34 b. Relative change of the mass and volume is

$$\varepsilon_m = 4.563\cdot 10^{-5} , \quad \varepsilon_v = 3.472\cdot 10^{-5} .$$

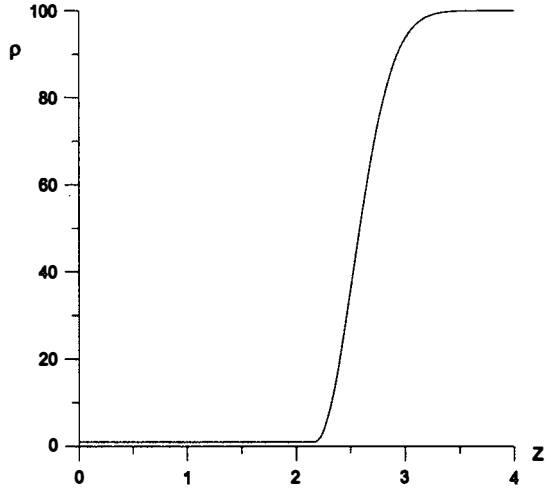


Figure 12.30: Density on the z-axis (x, y=0) on the mesh ω_o.

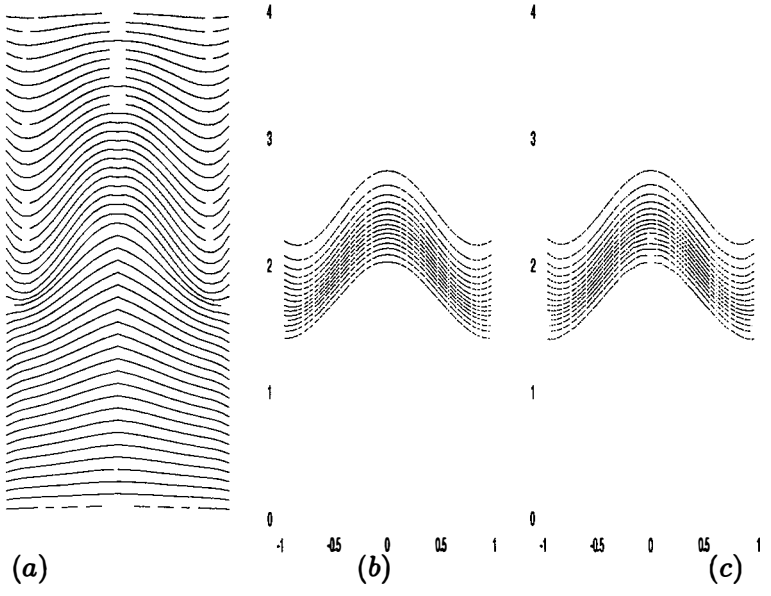


Figure 12.31: Coordinate surface j=7 of ω_o (a); density plots in layer j=7 of ω_o (b) and ω_n (c) meshes.

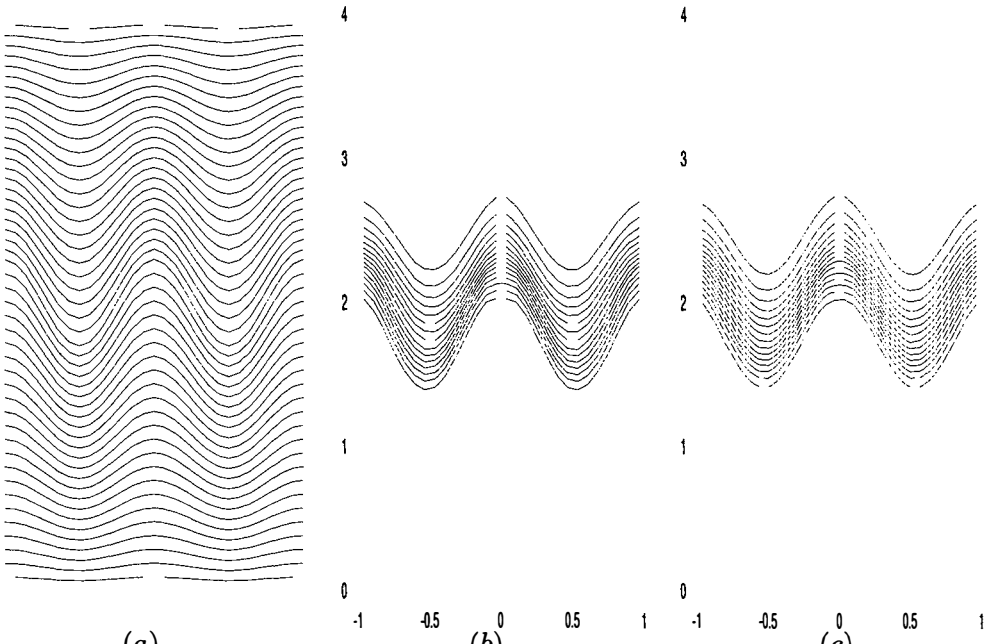


Figure 12.32: Coordinate surface j=1 of ω_o (a); density plots in layer j=1 of ω_o (b) and ω_n (c) meshes.

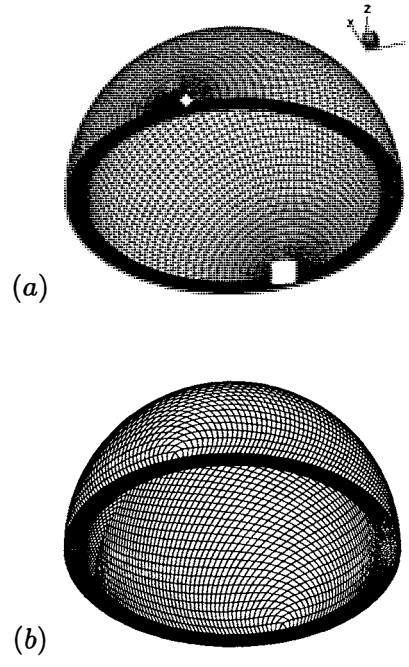
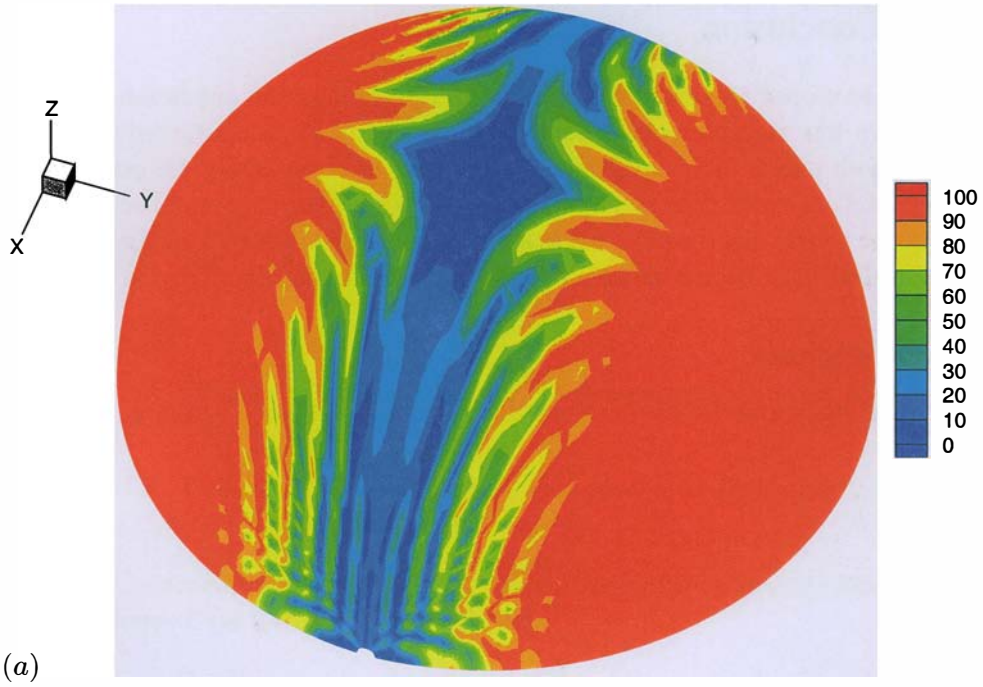
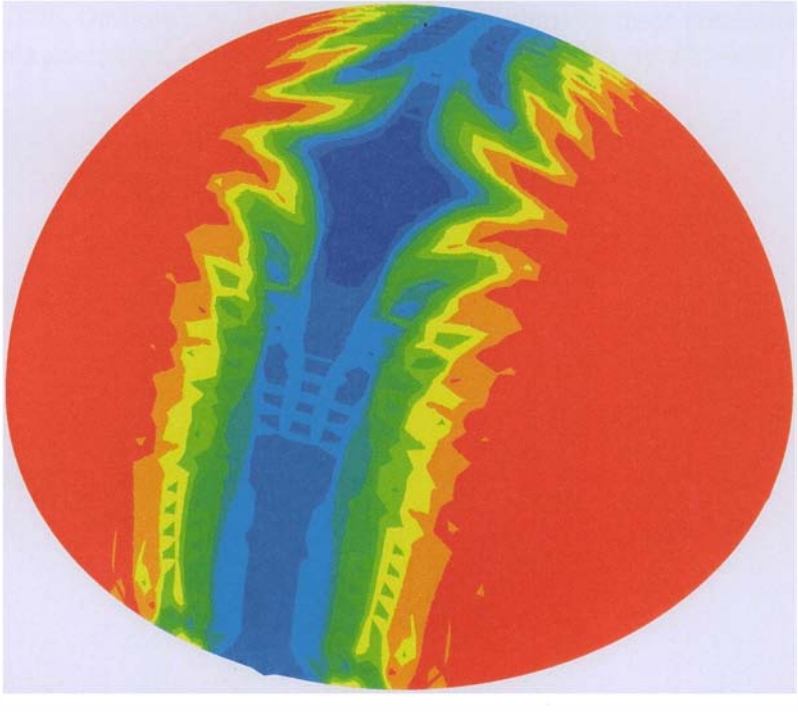


Figure 12.33: Old cylindrical mesh ω_o : 21×61×61 (a) and new regular ω_n : 21×61×61.



(a)



(b)

Figure 12.34: Density plots in top the layer i=20 of the old cylindrical mesh ω_o (a) and after remapping in the same layer of the new regular mesh ω_n.

12.7 Conclusion

We have developed a geometric algorithm for remapping from one hexahedral mesh onto another. In this algorithm every hex cell with ruled faces is substituted for two dodecahedrons with planar faces and, thus, the problem of constructing the overlapping figure between the old mesh cells and new mesh cells is significantly simplified. The mesh cell need not be convex and the only requirement to the meshes is that all grid cells, both hex and dodecahedral, must be non-degenerate. The optimal filtering algorithm has been suggested.

Acknowledgment

This work was supported by the Russian Foundation for Basic Research, project 02–01–00236.

Bibliography

[1] Dukowicz J.K. Efficient volume computation for three-dimensional hexahedral cells, *J. Comp. Phys.*, **74**(1988), pp. 493–496.

[2] Dukowicz J.K., Padial N.T. REMAP3D: A conservative three-dimensional remapping code, LA-12136-MS, Los Alamos report, 1991.

[3] Grandy J. Conservative remapping and regions overlays by intersecting arbitrary polyhedra, *J. Comp. Phys.* **148**(1999), pp. 433-466.

[4] Ushakova O.V. Conditions of nondegeneracy of three-dimensional cells. A formula of a volume of cells, *SIAM J. Sci. Comp*, **23** (2001), No. 4, pp. 1273–1289.

[5] Bronina T.N., Gasilova I.A., Ushakova O.V., Algorithms for three-dimensional structured grids generation, *Comput. Math. Math. Phys.*, **6** (2003), pp. 875–883.

INDEX

G

I

L

M

N

O

P

R

S

T

Advances in Grid Generation

Olga V. Ushakova

Editor

ISBN 10: 1-59454-273-2
ISBN 13: 978-1-59454-273-2

www.novapublishers.com