Mathematical models and numerical algorithms for the viscous flows
(for the flows of viscous fluids)

In the presented paper the following topics shall be discussed:

i) Mathematical properties of Navier Stokes equations

ii) Numerical algorithms for solving the problems of viscous flows

iii) Modular analysis of the numerical algorithms

iv) Correspondence between the structure of global algorithm and architecture of an computer

1. In cartesian coordinates the N.-S. equations look in a generalized form as follows

$$\frac{\partial w^i}{\partial t} + \frac{\partial \Sigma^{i\alpha}}{\partial x^\alpha} = F^i \qquad (1.1)$$

where
$i = 0-4$, $\alpha = 1-3$ is the index of summing

$$\bar{W} = \{ w^i \} = \{ \rho, \rho u^1, \rho u^2, \rho u^3, \rho E \} \qquad (1.2)$$

$$E = e(\rho, p) + \frac{\bar{u}^2}{2} \quad ; \qquad (1.3)$$

$\bar{w}$ is the vector of the state of the flow

$\Sigma^{i\alpha}$ is the matrix of the fluxes (fluxes)

$\bar{F} = \{ F^i \}$ is the vector of the mass forces

The ~~rheological~~ constitutional relations

$$\Sigma^{i\alpha} = \Sigma^{i\alpha} \left(w^k, \frac{\partial w^e}{\partial x^\beta} \right) \qquad (1.4)$$

make the system (1.1) closed

All the notations are conventional

In the case of linear constitutional laws relations (1.4) look as follows

f 15

*

In the general nonlinear case the relations

f 1.6

are valid **

The system (1.1) - (1.3) allows particular solutions, which play an essential role in the theory of N.-S. equations

The first one is the shock wave transition solution, which gives a

continous steady state profile of the shock wave.
The second one is the shear flow that
represents the transition between two
parallel ~~flows~~ uniform flows in the zone
with constant pressure
If the viscosity coefficients converge to 0
both zones of transition (theoretically of
infinite width) get narrow and
the gradient of state vector $\bar{w}$ grows to
infinity (contact transition layer)
(The Boundary layer may be considered as a
special case of the contact transition layer
The system of N.-S. equations is
? non fully parabolic, i.e. the dissipative
matrix has at least one of the ~~s~~ eigenvalues
equal to zero, whereas the corresponding
eigenvector is non zero one. ****

f. 17
and text

Of special interest is the stationary
boundary value problem. ***

f. 18
and text

If the stationary solution $\bar{w}(x)$ is uniquely determined, then all solutions $w(x,t)$ of the initial value problem 11-18 are converging to this solution

$$w(x,t) \to \bar{w}(x) \qquad (19)$$

independently of the initial value function $\bar{w}(x,0)$.

This property is very important for practical applications, because it makes possible to get stationary solution $\bar{w}(x)$ by some iteration process based on the initial value problem 11-18.

This process can be accelerated if we construct, in a rather arbitrary fashion, a nonstationary system (stationarising) possessing the same property (19). Indeed, we consider, along with the equation (11), the equations of the form

$$B\frac{\partial \bar{w}}{\partial t} + \frac{\partial \bar{\Sigma}^{\alpha}}{\partial x^{\alpha}} = \bar{F} \qquad (110)$$

correspondingly

$$C_1\frac{\partial \bar{w}}{\partial t} + C_2\frac{\partial^2 \bar{w}}{\partial t^2} + \dots C_p\frac{\partial^p \bar{w}}{\partial t^p} = \bar{F} \qquad (111)$$

If the steady state solution $\bar{w}(x)$ is uniquely determined, then all solutions $w(x,t)$ of the initial boundary value

By a suitable choice of the operators $B, C_1, C_2 \ldots C_p$ we can accelerate iteration process and replace system (1.1) by systems (1.10) or (1.11)

Algorithmically it may be more convenient

We give here two examples of useful applications

We determine B as some factorized operator

$$B = B_1 \cdot B_2 \ldots B_q \qquad (1.12)$$

where B_s have the property of easy inverting in such a way that every equation

$$B_s \bar{w} = \varphi_s \qquad (1.13)$$

is easily solvable

In this case it is possible to construct an iterative process based on equation (1.10), which is converging more rapidly than (?) that one based on equation (1.1)

To get a second example we set in (1.11)

$$C_s = C_p = 0 \qquad (1.14)$$

As a result (1.1) becomes a hyperbolic system which can be solved by

methods inherent to this class of equations, in particular by the schemes of running computation. [1], [2] (cf. R, I; MFS)

ref > To conclude we describe another useful approach, to accelerate the convergence. We make it clear by very simple example

For Burger's equation

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \nu \frac{\partial^2 u}{\partial x^2}, \quad \nu = \nu_0 > 0 \qquad (1.15)$$

we set the stationary boundary conditions

$$u(0,t) = u_0, \quad u(1,t) = u_1 \qquad (1.16)$$

and arbitrary initial value function

$$u(x,0) = u_0(x) \qquad (1.17)$$

For small ν_0 the convergence of nonstationary solution $u(x,t)$ of (1.15)–(1.17) to the stationary one $u(x,\infty)$ is slow.

We replace equation (1.15) by another one with artificial viscosity coefficient

$$\frac{\partial \bar u}{\partial t} + \bar u \frac{\partial \bar u}{\partial x} = \bar\nu \frac{\partial^2 \bar u}{\partial x^2}, \quad \bar\nu = \nu_0 + c\left(\frac{\partial \bar u}{\partial t}\right)^2 \qquad (1.18)$$

The nonstationary solution $\bar u(x,t)$ of (1.18), (1.16), (1.17) converges to the same stationary one

$$\bar u(x,t) \to \bar u(x,\infty) = u(x,\infty) \qquad (1.19)$$

more rapidly. This approach can be easily applied to N.-S. equations (see ref. [3])

NB! ref.

The above-mentioned mathematical properties shall be used in section two of our paper for constructing efficient numerical schemes

2 Numerical algorithms, here presented, for solving N.-S. equation, shall be based on splitting up schemes and adaptive moving mesh (see ref. [1] K, I)

NS comm.

We present at first splitting up in differential from in cartesian coordinates as a kind of a weak approximation, (see ref [2] NFS)

We use three kinds of splitting up (see ref [4]):

1 geometrical one, that is decomposing an equation into simpler ones, determined on subspaces (or manifolds) of inferior dimensionality.

2 physical one, that is decomposing an equation into simpler ones, each describing specific physical process

3. analytical one, that is decomposing an equation into simpler ones, each part fulfilling special function.

The differential splitting for N.-S. equations can be presented in the following form

§2.1
2.3

The requirement of maximal or, at least uniform, accuracy for a given number of mesh points and the existence of the flow zones with large gradients of $\bar{w}$ make unavoidable the application of a movable mesh.

The movable mesh is governed by some functional, expressing the requirements of accuracy and other mathematical properties (see ref [5])

~~The Euler's equations~~

As a consequence the velocity vector of mesh satisfies to a set of equations, which are Euler's equations, corresponding to the functional

To realize splitting up along coordinate directions of a curvilinear net, we transform the components of the flow velocity into contravarian form. As a result we get the coupled system of equations for material and informational media

If we introduce the compound vector of the state

$$\bar{z} = \{ \rho, u^1 - v^1, u^2 - v^2, u^3 - v^3, E, v^1, v^2, v^3 \} \qquad (2.4)$$

where

$$\bar{v} = \{ v^1, v^2, v^3 \} \qquad (2.5)$$

is the velocity vector of the mesh, we obtain the equations

$$\frac{\partial \bar{z}}{\partial t} + \frac{\partial \bar{\varphi}^\alpha}{\partial q^\alpha} = \bar{R} \qquad (2.6)$$

where

$\{ \bar{\varphi}^\alpha \}$ is the generalized matrix of the fluxes

$\bar{R}$ - generalized vector of the forces

Accordingly to the system (1.1) the system (2.6) may be written in the form (2.7-2.9)

After these transformations we

The obtained system (2.7-2.9) can be discretized in different ways

For nonstationary flow one can apply predictor corrector scheme with splitting up

We make it clear on the same example of initial value problem (1.16-a) for Burgers equation (1.15)

NBCC

Predictor corrector scheme looks as follows

$$\frac{u^{n+\frac{1}{2}} - u^n}{\tau} + u^n \frac{\Delta u^{n+\frac{1}{2}}}{h} = \nu \frac{\Delta\Delta^-}{h^2} u^{n+\frac{1}{2}} \qquad 2.10$$

?

$$\frac{u^{n+1} - u^n}{\tau} + \frac{\Delta_+\Delta_-}{h} u^{n+\frac{1}{2}} = \nu \frac{\Delta\Delta^-}{h^2} \frac{u^{n+1} + u^{n+1}}{2} \qquad 2.11$$

where

where $\operatorname{sign}\Delta \cdot \operatorname{sign} n \geq 0$ 2.12

For the full N.-S. equation it is necessary to apply at first splitting up, then predictor scheme and as last step corrector scheme, which reestablishes conservation laws. (see ref. [3])

NBCC

In stationary case we dispose of many mathematical models (see section 1, formulas) which garantee good convergence to the steady state solution:

NBCC

$$\bar{w}(x,t) \to \bar{w}(x,\infty)$$

It should be noted here, that in stationary case the application of the so called marching algorithm in supersonic domains is effective, because reducing the dimensionality of the problem.

The marching algorithm is in general use for ~~for simplified models of N.-S. equation: the boundary layer equation,~~ the simplified N.-S. equations, which neglect viscous force due to longitudinal gradients of velocity. and, as a special case, for the boundary layer equation.

The most difficult problem in mixed-iteration and marching-algorithms is the joining ~~matching~~ ~~of~~ stable and unstable solutions.

The most typical example is joining the elliptic solution near the front hyperbolic solution in edge of a wing and down stream region (see dias N°)

dias N°

The first subsonic ~~elliptical~~ domain can be effectively tracked by a nonstationary algorithm (see formulas [1]) and matching with supersonic solution can be

N cc

realized by iterations with overlapping of sub- and supersonic domains (see dias 1

dias N

If we use instead of N-S. equation the simplified model, there arises no difficulties in joining sub- and supersonic domain because in downstream direction we have always parabolic equation. But as we come to detached flow the parabolic equation gets antiparabolic and we have to join stable and unstable solution

This can be done by applying convenient regularization (see ref. [6])

To this purpose we set

$$\bar{u} = u + h^{\alpha}\left(\frac{\partial u}{\partial y}\right)^{\beta} \qquad \alpha > 0,\ \beta > 0$$

where u is longitudinal velocity

$\bar{u}$ is regularised velocity

By suitable choice of α, β we can make $\bar{u}$ positive

To get exact solution we have to recalculate the solution of detached flow in inverse direction.

3. In the first two sections we discussed mainly the features of the local algorithm, which was uniform, that is based on uniform and uniquely determined

i/ original continual model

ii/ difference scheme

iii/ mesh

In general, the pattern of the flow is not uniform, there arises many singularities and irregularities of the flow, and because of it the global algorithm becomes non uniform

There are two approaches to describe the flow

The first one consists in dissecting the domain of integration into subdomains, each domain being characterized by specific mathematical and/or physical properties (supersonic or subsonic domain, boundary layer, front ~~sh~~ shock zone, suspended

shock fronts, wakes, laminar-turbulent transitient regions) For every subdomain corresponding mathematical model is introduced. The interior boundaryres satisfig corresponding matching conditions This approach is mostly efficient as regards the number of arithmetic operations; but logically very complicated and hard to be programmed.

The second approach consists in applying one universal continual model for the whole field of the flow To make the model efficient and adaptive the movable mesh is to be used. The dissection of the domain into subdomains is not excluded, but it is mainly conditioned by cybernetic considerations (dynamical memory allocation, architecture of the computer and so on) Theoretically the second approach is preferable, but it requires good mathematical model and an computer

with high speed of execution and large memory

Practically the choice between two alternatives is conditioned by economical use of computer's resources.

The lack of good unifying mathematical model and of adequate serial computer makes necessary the subdividing the global algorithm into a sequence of simple and uniform algorithms.

This decomposing of the global algorithm denoted usually as modular analysis facilitates programming on serial computers, makes the programs more flexible and adaptive and ideally reduces the programming to the assemblage of already prepared beforehand prepared modules.

In such a way it is possible to reduce the global programming to that one on

cc) modular level. (see ref. [1])

It is not easy to give an objective definition of the module

One can say, formally, that the module is an element of some set of programming units.

To make definition of the module more

cc) effective we have introduced in [7] the notion of simple (homogenous) boundary value problem and corresponding algorithm (simple module)

The simple module, corresponding to some simple b.v. problem, has to satisfy to following requirements of homogenity

i) homogenous original model, boundary conditions including

ii) homogenous algorithm

iii) " " mesh

iv) imbedding into homogenous piece of operative memory

v) ~~imbedding~~ some properties of stability (restriction on condition number a.s.o.)

vi) maximal possible degree of generality

~~In most cases~~ Not always is the module an ~~X~~ autonomous algorithmic and programming unit

In the case of nonlinear intermodular connections, which is the typical situation, in- and outputs of a module depend on the state of neighbouring ones.

> ~~In~~ In this case the upper level ~~superior~~ module ~~X~~ joining upper > lower. the lower level ~~minor~~ ones is necessary.

ref. In ~~xxxxxxx~~ [8], it ~~has~~ been demonstrated that for some important classes of algorithmes the construction of joining module is possible both for explicit and implicit schemes

cc) It has ~~been~~ stressed in [ ] that good modularized programm is also good for execution on ~~computers with vector processors~~ and vector- and especially ~~array computers~~ arraycomputers

4.) It has been stressed in [8] that good modulized programm is good also for implementing on vector and array computers Now we can formulate additional requirement for the simple module:

VII) it can be executed in one of the processing units of the array computer It is to be noted here, that joining module, as being global, cannot be executed in single processing unit.

The last requirement is necessary but not sufficient for good parallelized execution on the array processor.

To achieve higher effiency of the array processor the set of modules has to be executed concurrently has to be homogenous, that is all modules $M_i \in M$ have to be identical in the structure and differ only by coefficients.

In this case the array computer can work in SIMD fashion, that is single instruction allows to execute

in parallel way all modules $M_i \in M$

Splitting up schemes suit well to this requirement because on every fractional step the set of executed modules is homogeneous, that is all modules got by splitting up have identical structure and differ only by coefficients.

Splitting up schemes suits well to the requirement of homogenity, because all modules simple modules produced by splitting up of a homogenous discretized b.v. problem have at every fractional step identical structure and differ only by coefficients

~~It follows from that~~

The question is how to transform arbitary boundary value problem into ~~hogeneg~~ homogeneous one.

There are many ways to achieve this aim

i) the use of homogeneous algorithmes and schemes

ii) approximating nonhomogeneous mathematical model in the interior of the domain and on the boundary by homogeneous ones

iii) the use of regular single mesh

iv) The use of fictitious domains

cc See the references T.C., Kon, Ine, In, Wan

The rapid development of vector and array computers has stimulated the development and reevaluation of the algorithms.

Many algorithms, which are good for implementing on scalar computers proved to be inefficient for vector and array computers.

The splitting up schemes possess the property of being good both for scalar and array computers.

This is due to the homogenity of the set of modules generated by splitting up.

Among the fractional steps schemes those of splitting up method are most efficient.

The competitive properties of the implicit schemes as compared to explicit ones remain on the same

~~text~~

The [illegible] implicit schemes conserve their efficiency as compared to ~~esti~~ explicit ones ~~[illegible]~~ both for scalar and array processors

That is especially true for stationary problems and as a consequence for a main problem of aerodynamics — stationary flow of gas past arbitrary shaped three dimensional body

The conformity between the structure of ~~[illegible]~~ the algorithm and that of array computer is evident in the case of splitting up schemes. This virtual accordance has stimulated several projects of array computer oriented on the problems ~~and~~ of mechanics and mathematical physics

cc. (see ref. ?...)

Another way to establish the conformity between numerical methods and effective decomposing is the modular analysis of the algorithms of linear algebra

~~effective decomposing of large classes~~

These two approaches - decomposing on the level of problems of mathematical physics and that one on the level of linear algebra - constitute the foundation for the constructing vector and array computer in the next future.