A high order method for simulation of fluid flow in complex geometries

by

Erik Stålberg

May 2005
Technical Reports from
KTH Mechanics
SE-100 44 Stockholm, Sweden

Typsatt i $\mathcal{AMS}$-LaTeX.

Akademisk avhandling som med tillstånd av Kungliga Tekniska Högskolan i Stockholm framlägges till offentlig granskning för avläggande av teknologie licentiatexamen tisdagen den 24:e maj 2005 kl 10.15 i sal E 3, Osquars Backe 14, Kungliga Tekniska Högskolan, Valhallavägen 79, Stockholm.

Universitetsservice US-AB, Stockholm 2005

A high order method for simulation of fluid flow in complex geometries

Erik Stålberg 2005
KTH Mechanics
SE-100 44 Stockholm, Sweden.

Abstract

A numerical high order difference method is developed for solution of the incompressible Navier–Stokes equations. The solution is determined on a staggered curvilinear grid in two dimensions and by a Fourier expansion in the third dimension. The description in curvilinear body-fitted coordinates is obtained by an orthogonal mapping of the equations to a rectangular grid where space derivatives are determined by compact fourth order approximations. The time derivative is discretized with a second order backward difference method in a semi-implicit scheme, where the nonlinear terms are linearly extrapolated with second order accuracy.

An approximate block factorization technique is used in an iterative scheme to solve the large linear system resulting from the discretization in each time step. The solver algorithm consists of a combination of outer and inner iterations. An outer iteration step involves the solution of two sub-systems, one for prediction of the velocities and one for solution of the pressure. No boundary conditions for the intermediate variables in the splitting are needed and second order time accurate pressure solutions can be obtained.

The method has experimentally been validated in earlier studies. Here it is validated for flow past a circular cylinder as an example of a physical test case and the fourth order method is shown to be efficient in terms of grid resolution. The method is applied to external flow past a parabolic body and internal flow in an asymmetric diffuser in order to investigate the performance in two different curvilinear geometries and to give directions for future development of the method. It is concluded that the novel formulation of boundary conditions need further investigation.

A new iterative solution method for prediction of velocities allows for larger time steps due to less restrictive convergence constraints.

Descriptors: Navier–Stokes equations, compact high order difference methods, approximate factorization, curvilinear staggered grids, spectral methods, boundary conditions.

Preface

This thesis considers the description and validation of an incompressible Navier-Stokes solution method. It is based on compact high order difference operators in two dimensions and a Fourier expansion in one dimension. This discretization allows for treatment of two dimensional curvilinear domains with a third periodic dimension. The thesis contains the following papers:

Paper 1. Brüger, A., Stålberg, E., Nilsson, J., Kress, W., Gustafsson, B., Lötstedt, P., Johansson, A. V. & Henningson, D. S. 2004 A hybrid high order method for incompressible flow in complex geometries /version 2. TRITA-MEK 2005:06.

Paper 2. Stålberg, E., Brüger, A., Lötstedt, P., Johansson, A. V. & Henningson, D. S. 2004 High order accurate solution of flow past a circular cylinder. *Accepted to J. Sci. Comput.*

Paper 3. Stålberg, E., Brandt, L. & Brüger, A. 2005 Applications of a high order method for fluid flows in complex geometries. *Internal report.*

Division of work between authors

Dan Henningson (DH) and Arne Johansson (AJ) formulated the goal and main strategy of a method designed for simulations of turbulent flow phenomena in complex geometries. The novel boundary conditions formulation come from Bertil Gustafsson (BG), Jonas Nilsson (JN) and Wendy Kress (WK). The formulation of an approximate factorization as a preconditioner in an iterative scheme is done by Per Lötstedt (PL). WK and PL are responsible for the stability analysis.

Large parts of the implementation was done by JN and Arnim Brüger (AB) and in the later stage by Erik Stålberg (ES).

The main writing of paper 1 was done by AB, ES and JN. Contributions regarding the stability analysis come from WK and PL. BG was responsible for the boundary conditions formulation. AJ and DH was responsible for the choice of numerical method.

DH and AJ formulated the idea of the study in paper 2. The necessary implementations for this study was done by ES and AB and the simulations was done by ES. The writing was done by ES, AB and PL.

The writing of paper 3 was done by ES with feedback from Luca Brandt (LB). Numerical experiments for flow past a parabolic body was done by ES and LB. The asymmetric diffuser flow numerical experiments was done by ES and the grid was generated with an numerical conformal mapping by AB and ES. Necessary implementations was done by ES.

Contents

Part 1

Summary

CHAPTER 1

Introduction

Research of the behaviour of liquids and gases is important from many different practical aspects such as design of aircraft, road vehicles and processes in the chemical industry. Studies in the area of fluid mechanics can also be motivated from a more fundamental point of view since there are fluid flow regimes where the underlying physics are not completely understood. Consider for example the complexity of the fluid motion connected to an aircraft under operative conditions. Aerodynamic forces will as a consequence of the design of the aircraft produce lift. Simultaneously, the vehicle will experience flow resistance. In order to achieve smaller fuel consumption, drag forces have to be reduced which in a longer run will decrease the economic and environmental costs. One way of reducing the drag on the aircraft is to postpone the transition from the well ordered laminar flow to the chaotic turbulent state in a thin layer close to the aircraft. At the same time the flow enters the transitional and turbulent regime the scientist enters the mystery of turbulence.

In general it is not possible to find analytical solutions to the governing equations. The scientists are left with experimental measurements and numerical methods and today these two methods live in a symbiosis. This thesis describes a numerical method for simulation of fluid flows. Focus is put on a method for turbulent flow in complex geometries. Turbulence involves a broad range of scales in both time and space including high-frequency fluctuations of random nature. The broadband phenomena in turbulence requires large resolution during long time integration to be accurately captured and put high demand on the available computer power.

One way to handle the inherent complexity of turbulence is to average the equations which means that models have to be incorporated to account the effects of turbulence. A more accurate and expensive way is to use models only for the fine-scale turbulent structures which means that the large energy-carrying scales are captured directly. This is referred to as large-eddy simulation, LES. The most demanding and accurate way to solve the governing equations is without any averaging as in the method described in this thesis. This is called direct numerical simulation, DNS.

The incompressible Navier–Stokes equations in dimensionless form are

$$\frac{\partial}{\partial t}\mathbf{u} + (\mathbf{u}\cdot\nabla)\,\mathbf{u} + \nabla p - \frac{1}{R}\nabla^2\mathbf{u} = 0\,, \tag{1.1}$$

$$\nabla\cdot\mathbf{u} = 0\,. \tag{1.2}$$

where $\mathbf{u} = (u\ v\ w)^T$ is the velocity vector and p the pressure. The Reynolds number R quantifies the ratio between inertial and viscous forces. The range of scales, and accordingly the computational cost, increases rapidly with increasing Reynolds number.

There are many different available solution methods for the Navier–Stokes equations. One category uses Fourier expansions and has shown to be a very efficient technique for turbulent simulations. Fourier discretization employed in all three space dimensions was used in an early study by Orszag & Patterson (1972) and this method is still used today but with much higher Reynolds numbers. However, this cannot be used in context with complex geometries. Spectral element methods used in for example Fischer *et al.* (2002) can be applied to complex geometries. It combines the geometric flexibility of finite elements with the efficiency of spectral methods. High order accurate finite differences of compact type allow for geometric flexibility and have almost spectral properties, see Lele (1992).

The standard indicator for accuracy of finite difference approximations is order of accuracy which constitutes a relation between mesh refinement and accuracy. A mesh refinement of a factor two using a second order accurate discretization would result in an accuracy improvement by a factor four and for a fourth order accurate scheme this factor would be 16.

Temporal and spatial accuracy verifications in a two dimensional formulation of the method presented in this thesis can be found in Nilsson *et al.* (2003) and Brüger *et al.* (2005) and in Brüger *et al.* (2004) a similar investigation is done in three dimensions. A stable formulation of boundary conditions is presented in Kress & Nilsson (2003) and in Brüger (2004) turbulent channel flow in a minimal unit is studied.

The method is here presented in three spatial dimensions. In paper 3 numerical experiments of the method is presented. It involves two cases with major configuration differences. The first is two dimensional flow past a parabolic body as an example of an external flow problem in a curvilinear geometry. The second application is three-dimensional flow in an asymmetric diffuser. A validation of the two dimensional formulation of the method is presented in paper 2 for laminar flow past a circular cylinder. It is a widely used test case for numerical algorithms since both curvilinear coordinates and complex physics are involved. For a certain range of Reynolds numbers the flow is globally unstable and alternating vortices are convected downstream from the cylinder. This is called the von Kármán street and Acheson (1990) tells the story from von Kármán (1954). The interest in the vortex shedding phenomenon stems from about 1911 when von Kármán was a graduate assistant in Ludwig Prandtl's

laboratory in Göttingen where Prandtl's doctoral candidate Karl Hiemenz experimentally investigated the pressure distribution on a circular cylinder. The flow in the channel oscillated violently and Prandtl told him: 'Obviously your cylinder is not circular'. After careful machining of the cylinder the flow continued to oscillate and Hiemenz was told that the channel was not symmetric. Hiemenz adjusted the channel and every morning von Kármán asked him, 'Herr Hiemenz, is the flow steady now?' he answered sadly, 'It always oscillates'.

CHAPTER 2

The numerical method

Finite difference schemes can be either explicit or implicit. An explicit scheme of arbitrary order of accuracy expresses the derivatives as an weighted sum of the nodal values. Implicit schemes, also known as compact schemes, express a weighted sum of the derivatives as a weighted sum of the nodal values.
Also the time discretization of the governing equations can be explicit or implicit. An explicit method would put a strong condition on the maximum time step due to stability constraints. An fully implicit method allows for large time steps but need solutions of nonlinear systems at each time level. Here, a semi-implicit method is used where the nonlinear terms are treated explicit.

2.1. Time discretization

For time discretization a backward differentiation formula (BDF) is used. BDF methods have good stability properties and a second order accurate method is used, BFD-2. Higher order time advancement would allow for larger time steps applied to wall bounded turbulent flow, see Kress & Lötstedt (2004). However, this would require more memory storage compared with lower order methods and for other flow regimes, such as transitional flow, the time step has to be smaller for good accuracy.

The time derivative in the momentum equations discretized with BDF-2 is

$$\frac{\partial \mathbf{u}^{n+1}}{\partial t} = \frac{3\mathbf{u}^{n+1} - 4\mathbf{u}^{n} + \mathbf{u}^{n-1}}{2\Delta t} + \quad \mathcal{O}(\Delta t^2), \tag{2.1}$$

where n is the discrete time level and Δt the time step. The explicitly treated nonlinear terms are extrapolated to time level $n+1$ from the two preceding time levels

$$\left[(\mathbf{u}\cdot\nabla)\,\mathbf{u}\right]^{n+1} = 2\,\left[(\mathbf{u}\cdot\nabla)\,\mathbf{u}\right]^{n} - \left[(\mathbf{u}\cdot\nabla)\,\mathbf{u}\right]^{n-1}. \tag{2.2}$$

The requirement of a divergence free velocity field is enforced implicit and the semi-implicit time discretized Navier–Stokes equations read

$$\frac{3}{2}\mathbf{u}^{n+1} + \Delta t\,\nabla p^{n+1} - \frac{\Delta t}{R}\nabla^2\mathbf{u}^{n+1} = 2\mathbf{u}^{n} - \frac{1}{2}\mathbf{u}^{n-1} + \Delta t\left[(\mathbf{u}\cdot\nabla)\,\mathbf{u}\right]^{n-1} - 2\,\Delta t\left[(\mathbf{u}\cdot\nabla)\,\mathbf{u}\right]^{n}, \tag{2.3a}$$

$$\nabla\cdot\mathbf{u}^{n+1} = 0\,. \tag{2.3b}$$

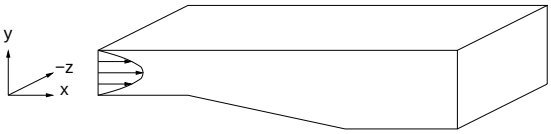

FIGURE 2.1. 3D physical space defined by a 2D curvilinear geometry.

2.2. Orthogonal coordinate transformation

Consider a physical space (x, y, z) defined to be curvilinear in two dimensions (x, y) including a periodic spanwise direction z without geometry dependence as in figure 2.1. By using a mapping from physical space to an equidistant Cartesian space (ξ, η, ζ) classical schemes can be used to discretize the equations in curvilinear coordinates. $\zeta = z$ since the method only is curvilinear in the (x, y) plane. In order to reduce the number of terms in the mapping an orthogonal transformation is used. For a general case where analytical mapping functions are unknown a numerical algorithm can be used. An example of grid generated by the method of Driscoll & Trefethen (2002) can be found in paper 3.

2.3. Space discretization on staggered grid

When solving the incompressible Navier–Stokes equations on a collocated grid, which means that the unknowns are discretized on the same grid points, it is a well known fact that parasitic fluctuations not corresponding to the physical solution occurs. A staggered grid as in figure 2.2 where the different variables are represented on alternating grid points avoid this oscillatory solutions without the use of filtering schemes or artificial viscosity.

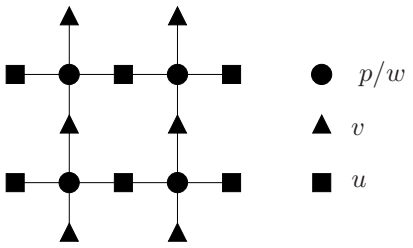

FIGURE 2.2. Staggered grid.

For space discretization on the curvilinear grid in two space dimensions fourth order accurate compact finite differences are employed. High order methods reduce the number of grid points per smallest length scale in comparison with low order methods for the same accuracy. This is of primary importance since the available memory is limited. Most differencing schemes do well for low frequency functions. However, the solution of nonlinear differential equations often contain several frequencies. Low order explicit schemes suffer from large

dispersion errors resulting in bad resolved high wavenumber waves. Compact schemes are known to have good dispersive properties, see Lele (1992), and the relation between the unknown derivative f' for a function f using a fourth order accurate discretization is

$$\frac{1}{6}f'_{i-1} + \frac{2}{3}f'_i + \frac{1}{6}f'_{i+1} = \frac{1}{2h}(f_{i+1} - f_{i-1}), \tag{2.4}$$

where h is the grid spacing. A thorough description of the compact difference operators for first and second derivatives on the staggered grid can be found in paper 1. In order to solve the systems resulting from the compact difference approximations closed systems of the form $\mathcal{P}f' = \mathcal{Q}f$ are needed. Since no boundary conditions for the derivatives f' are available one-sided stencils are used.

In the same manner as for the derivatives interpolations between staggered grid locations adopt sixth order accurate compact operators. For solutions of the tridiagonal systems generated by the compact operators, LU factors of the system matrices are used in forward and backward substitution steps.

Under the assumption that the spanwise length of physical space z_l is large enough to capture all relevant scales in the problem the spanwise dimension can be considered homogeneous (statistically independent of spatial position) and a discretization by expanding the unknowns in Fourier modes is used. For a variable f at grid point (x_j, y_k, z_l) the inverse Fourier transform in the spanwise direction is

$$f(x_j, y_k, z_l) = \sum_{m=-N_z/2}^{N_z/2-1} \hat{f}(x_j, y_k)\, e^{i\beta_m z_L l/N_z}. \tag{2.5}$$

N_z is the number of grid points in the spanwise direction and β_m the discrete wavenumbers. The resulting discrete system will be solved in Fourier transformed space. In order to avoid evaluation of convolution sums in Fourier transformed space the nonlinear terms are computed in physical space by first transforming the variables at time levels n and $n-1$ back to physical space. Then all necessary operations are performed in physical space and finally the explicitly treated terms at time level $n+1$ are transformed to Fourier space. Methods known as Fast Fourier Transforms (FFT) are used to transform the variables between Fourier and physical space, see Canuto *et al.* (1988).

2.4. Boundary conditions

A numerical method for a differential equation is said to be well posed if an unique solution to the problem exists which depends continuously on the initial and boundary data. This means that small changes in the initial and boundary data result in small changes in the solution. Gustafsson & Nilsson (2000) propose a well posed boundary condition formulation for the steady Stokes equations. Kress & Nilsson (2003) generalize the formulation to involve the linearized Navier–Stokes equations for staggered grids with one periodic

direction. On a Cartesian domain $\Omega = 0 \leq x \leq 1, 0 \leq y \leq 2\pi$ where the velocity components are prescribed on all boundaries, i.e. Dirichlet boundary conditions, the continuous formulation is

$$\begin{aligned} u(0,y) - \frac{1}{2\pi}\int_0^{2\pi} u(0,y)dy = g_0(y)\,, \qquad & u(1,y) = u_1(y)\,, \\ v(0,y) = v_0(y)\,, \qquad & v(1,y) = v_1(y)\,, \\ w(0,y) = w_0(y)\,, \qquad & w(1,y) = w_1(y)\,, \\ \int_0^{2\pi} u(0,y)dy + \int_0^{2\pi} p(0,y)dy = q_0\,, & \end{aligned} \tag{2.6}$$

where $\int_0^{2\pi} g_0(y)dy = 0$. q_0 is an arbitrary constant. An extension of the formulation to involve cases where derivatives of the velocity components, Neumann conditions, are prescribed can be found in paper 1. Small perturbations in the discrete version of (2.6) will still give a solvable system. The system matrix is non-singular and the equations can be solved to machine precision.

CHAPTER 3

Solution procedure

The numerical solution of equations (1.1) and (1.2) is not easy to obtain since the resulting discrete system is large and indefinite, see Perot (1993). In order to satisfy an incompressible velocity field it would be necessary to update a simultaneous solution for velocities and the pressure. By interpreting the role of the pressure in the momentum equations (1.1) as a projection operator (see Kim & Moin 1985) of an arbitrary velocity field onto a divergence–free counterpart, an operator splitting scheme can be defined. Projection or fractional step methods integrate the equations in a segregated manner by first solving a convection–diffusion equation to predict intermediate velocities, which are then projected onto the space of divergence–free velocity fields by solving a Poisson equation for the pressure.

3.1. Approximate factorization

From the proposed semi-implicit method it is evident that, in each time step, a system of linear equations has to be solved for the unknowns. After elimination of boundary values the Fourier transformed solution vector $\hat{\mathbf{U}}$ at time step $n+1$ satisfies

$$\underbrace{\begin{pmatrix} A & 0 & G \\ 0 & \tilde{A} & i\beta\Delta t I \\ D & i\beta\tilde{D} & -E \end{pmatrix}}_{M_R} \underbrace{\begin{pmatrix} \hat{\mathbf{u}}_1^{n+1} \\ \hat{w}^{n+1} \\ \hat{p}^{n+1} \end{pmatrix}}_{\hat{\mathbf{U}}^{n+1}} = \underbrace{\begin{pmatrix} \hat{\mathbf{b}}_1^* \\ \hat{b}_w^* \\ \hat{b}_2^* \end{pmatrix}}_{\hat{\mathbf{b}}}, \tag{3.1}$$

where $\hat{\mathbf{b}}$ consists of the explicitly treated nonlinear terms from time steps n and $n-1$ and boundary data. G and D are the discretized gradient and divergence operators in the (ξ, η) plane. I is the identity matrix and $\hat{\mathbf{u}}_1 = (\hat{u}, \hat{v})^T$. A and $\tilde{A}$ have the structure

$$A = \frac{3}{2}I + \frac{\Delta t}{R}\beta^2 I - \frac{\Delta t}{R}L\,, \tag{3.2}$$

where L approximates the Laplace operator in the (ξ, η) plane. E stems from the boundary condition formulation for the pressure and $E = 0$ for $\beta \neq 0$.
Due to the implicit difference operators M_R is not known explicitly and an approximated system matrix M^* with corresponding LU factors is constructed,

see Perot (1993)

$$\begin{aligned} M^* &= \begin{pmatrix} A & 0 & 0 \\ 0 & \tilde{A} & 0 \\ D & i\beta\tilde{D} & I \end{pmatrix} \begin{pmatrix} I & 0 & \gamma G \\ 0 & I & \gamma i\beta\Delta t I \\ 0 & 0 & Q \end{pmatrix} \\ &= \begin{pmatrix} A & 0 & \gamma AG \\ 0 & \tilde{A} & \gamma i\beta\Delta t\tilde{A} \\ D & i\beta\tilde{D} & -E \end{pmatrix}, \end{aligned} \tag{3.3}$$

where $Q = -E - \gamma\, DG + \gamma\beta^2\Delta t\, \tilde{D}$ and γ is set to

$$\gamma = \frac{1}{\frac{3}{2} + \frac{\Delta t}{R}\beta^2} \tag{3.4}$$

for BDF-2. In this approximate factorization no boundary conditions for the intermediate variables are needed since they are defined by the matrices in the semi-implicit discretization and the pressure is second order time accurate.
System (3.1) is solved in a fixed point iteration scheme with iteration index k using the approximate factorization of M_R. In this outer iteration a correction $\mathbf{z}^{(k)}$ is computed as

$$\hat{\mathbf{U}}^{(k+1)} = \hat{\mathbf{U}}^{(k)} + \mathbf{z}^{(k)} = \hat{\mathbf{U}}^{(k)} + {M^*}^{-1}\left(\hat{\mathbf{b}} - M_R\hat{\mathbf{U}}^{(k)}\right) = \hat{\mathbf{U}}^{(k)} + {M^*}^{-1}\mathbf{r}^{(k)} \tag{3.5}$$

where $\mathbf{r}^{(k)}$ is the residual from iteration k.
With $\mathbf{r}^{(k)} = (\mathbf{r}_1^{(k)}, r_w^{(k)}, r_2^{(k)})$ and $\mathbf{z}^{(k)} = (\mathbf{z}_1^{(k)}, z_w^{(k)}, z_2^{(k)})$ one outer iteration involves the following steps

1. Solve $A\mathbf{y}_1^{(k)} = \mathbf{r}_1^{(k)}$.
2. Solve $\tilde{A}y_w^{(k)} = r_w^{(k)}$.
3. $y_2^{(k)} = r_2^{(k)} - D\mathbf{y}_1^{(k)} - i\beta\tilde{D}y_w^{(k)}$.
4. Solve $Qz_2^{(k)} = y_2^{(k)}$.
5. $z_w^{(k)} = y_w^{(k)} - \gamma i\beta\Delta t z_2^{(k)}$
6. $\mathbf{z}_1^{(k)} = \mathbf{y}_1^{(k)} - \gamma G z_2^{(k)}$.
7. $\hat{\mathbf{U}}^{(k+1)} = \mathbf{z}^{(k)} + \hat{\mathbf{U}}^{(k)}$.
8. $\mathbf{r}^{(k+1)} = \hat{\mathbf{b}} - M_R\hat{\mathbf{U}}^{(k+1)}$.

If the residual $\mathbf{r}^{(k+1)}$ is within the specified convergence threshold, $\hat{\mathbf{U}}^{n+1} = \hat{\mathbf{U}}^{(k+1)}$, otherwise $k \leftarrow k+1$ and a new outer iteration is started at 1.
All sub-matrices in (3.3) are real and the equations are separately solved for the real and imaginary parts in order to avoid complex arithmetics. The dominating computational work in one outer iteration is to iteratively solve the linear subsystems in steps 1, 2 and 4. The systems in step 1 and 2 are advection-diffusion equations for u, v and w. In step 4 a system related to the Poisson equation is solved and the solution is then used to project the predictions made in 1 and 2 on a divergence–free space. From the well posed formulation of boundary conditions (2.6) Q is nonsingular and a solution $z_2^{(k)}$ in 4 is guaranteed, see Brüger *et al.* (2005). A description of the iterative solution techniques of the subsystems in 1, 2 and 4 is the objective in the following section.

3.2. Iterative technique for subsystems

The matrices A and $\tilde{A}$ have structures as in (3.2) and are diagonally dominant for large wavenumbers β. By neglecting the Laplace operator L in the (ξ, η) plane the inverse of the diagonal, γI, can be used as a preconditioner in fixed point iteration schemes for the A systems. In Kress & Lötstedt (2004) it is shown that the fixed point iterative solver with diagonal preconditioner will converge if $\Delta t/(Rh^2) < 3/16$ where h is the grid spacing. This is a severe limitation on the maximum allowable time step Δt particularly for cases with low Reynolds numbers and high spatial resolution. The stabilized bi-conjugate gradient (Bi-CGSTAB) iterative solver is less restrictive in terms of the maximum time step. In paper 1 a comparative study of the performance for the fixed point and Bi-CGSTAB iterative solvers for the A systems can be found and it is shown that Bi-CGSTAB is the more efficient choice. For detailed description of the Bi-CGSTAB routine, see van der Vorst (1992) and Greenbaum (1997).

The system $Qz_2 = y_2$ is similar to a Poisson equation but Q is not symmetric due to the formulation of boundary conditions. The Bi-CGSTAB routine is here used as iterative solver since it is suitable for non-symmetric system matrices.

As preconditioner to the Bi-CGSTAB routine incomplete LU factors (ILU) of the system matrices are used, see Meijerink & van der Vorst (1977). Since the system matrices A, $\tilde{A}$ and Q are not known explicitly the ILU factorizations is based on an explicit second order accurate discretization of those matrices.

CHAPTER 4

Summary of Papers

Paper 1

This paper is a complete description of the numerical method. Basic aspects of the modular implementation in the Fortran90 language are described. Rigorous descriptions of the compact difference operators on staggered grids and the boundary conditions formulation are presented. The approximate factorization technique with its associated block LU factorization is formulated and efficiency measures using two different iterative solution methods for prediction of velocities are shown.

Paper 2

The aim of this paper is to apply the high order method to flow around a circular cylinder, which is a test case well documented by both numerical and experimental data. This flow problem fits well as two dimensional validation experiment since it exhibits unsteady separation and is laminar up to a certain Reynolds number. At low Reynolds numbers ($3 \leq R = U_\infty \, d/\nu \leq 45$), two attached eddies appear behind the cylinder, where U_∞ is the free-stream velocity, d the diameter and ν the kinematic viscosity. These eddies become larger with increasing Reynolds numbers. For $R \geq 45$ the flow becomes globally unstable and vortex shedding occurs. We present numerical simulations of flow past a circular cylinder at Reynolds numbers in both the stable and the vortex shedding regime. The results agree very well with the data known from numerical and experimental studies in the literature.

Paper 3

The method is in this paper applied to two different flow configurations in order to investigate the performance and to give directions for future improvements. The first case is external laminar flow past a parabolic body. It poses problems in terms of in- and outflow boundary conditions as well as solution of flow with a stagnation point. The second case is three-dimensional flow in a plane asymmetric diffuser and the efficiency is challenged in terms of computer time. The asymmetric diffuser grid is generated with a numerical conformal mapping technique.

CHAPTER 5

Conclusions and outlook

An accurate discretization of the incompressible Navier–Stokes equations in primitive variables has been developed. The discretization combines a fourth order compact difference method on a two dimensional curvilinear staggered grid with a spectral approximation in the third dimension. The time discretization is semi-implicit and second order accurate.

An approximate factorization type preconditioner of the system matrix is used in an outer iterative scheme. The arising sub-systems for the velocities and the pressure are solved in sets of inner iterations.

A well posed description of boundary conditions for the incompressible Navier–Stokes equations is used.

A two dimensional version of the high order method is applied for flow past a circular cylinder. The validation includes both the steady and unsteady flow regime and match very well the data known from other numerical simulations and experiments. It is also shown that the results are achieved with much fewer grid points than those reported from studies performed with low order methods.

The present state of the code is an existing three-dimensional version. With the implementation of a Bi-CGSTAB iterative solution algorithm for prediction of velocities it is shown that time steps close to the stability limit can be used, if this is allowed by accuracy considerations. DNS of turbulent flow phenomena in geometries and for Reynolds numbers of todays interest need highly optimized and efficient codes. Focus should first be put on the efficiency on a single processor since the most important impediment to good parallel performance is a poor single node performance. A starting point in terms of performance monitoring is to use a profiling software in order to determine the actual behaviour of the code. Optimized arithmetic libraries such as BLAS (Basic Linear Algebra Subroutines) can be used to increase efficiency of often used matrix and vector operations.

The most time consuming part of the code is in the solution algorithm for the pressure. Multigrid methods are known to result in considerably speed-up for elliptic partial differential equations like this.

In order to perform large scale computations a parallelization standard has to be chosen. The methods of interest are OpenMP for shared memory and MPI for distributed memory machines. A natural strategy would be to parallelize

over the wave numbers from the Fourier discretization. Since the work load is wave number dependent this would result in an uneven load distribution. A block parallelization in the two dimensional domain would result in even work load but is more advanced since the implementation has to be on compact difference operator level.

The formulation of boundary conditions needs further investigations and modifications. It results in a well posed system of equations but artificial boundaries demanding more advanced and flexible treatment than the standard Dirichlet and Neumann conditions are not obvious how to handle. An open question is how the formulation interact with a buffer zone or similar technique used to prevent convective wave reflections or upstream propagation of information from the outflow boundaries.

Starting from the predictions that the fast performance increase of supercomputers will continue, the ultimate situation would be to know all advantages and disadvantages of different numerical methods and formulations. This is not the case and we need to be prepared to adjust our methods during the way to larger and more efficient simulations.

Acknowledgement

I would like to thank my main supervisor Professor Dan Henningson. His excellent knowledge and experience in fluid mechanics and numerical methods are greatly appreciated. He is always available for discussions and has provided a stimulating and fun research atmosphere.

My co-supervisor Professor Arne Johansson is acknowledged for his guidance and insight into flow physics.

I would like to express my deep gratitude to Dr Arnim Brüger for his patience when explaining the world of scientific computing. He has been a big source of inspiration, not only in terms of science.

Professor Per Lötstedt and Professor Bertil Gustafsson from Uppsala University are acknowledged for a very fruitful time of cooperation and for always sharing their expertise in numerical analysis. I would also like to thank Dr Jonas Nilsson and Dr Wendy Kress for their invaluable contributions and cooperation.

A big thank goes to Dr Luca Brandt for sharing his knowledge in fluid mechanics and for a very fun time of cooperation.

Finally, I would like to thank my colleagues Astrid, Carl-Gustav, Martin, Espen, Jérôme, Mattias, Ori and Stefan for the friendly environment. I really liked the German-English-Italian-Swedish-French-Norwegian (correct order) mix, even if I sometimes expressed a different opinion.

Bibliography

Acheson, D. J. 1990 *Elementary fluid Dynamics*. New York: Oxford University Press.

Brüger, A. 2004 A hybrid high order method for simulation of turbulent flow in complex geometries. PhD Thesis, Dept. of Mechanics, KTH, Stockholm, Sweden.

Brüger, A., Gustafsson, B., Lötstedt, P. & Nilsson, J. 2004 Splitting methods for high order solution of the incompressible Navier-Stokes equations in 3D. *To appear in Int. J. Numer. Meth. Fluids* .

Brüger, A., Gustafsson, B., Lötstedt, P. & Nilsson, J. 2005 High order accurate solution of the incompressible Navier-Stokes equations. *J. Comput. Phys.* **203**, 49–71.

Canuto, C., Hussaini, M. Y., Quarteroni, A. & Zang, T. A. 1988 *Spectral Methods in Fluid Dynamics*. New York: Springer.

Driscoll, T. A. & Trefethen, L. N. 2002 *Schwartz-Christoffel Mapping*. Cambridge Univ. Press.

Fischer, P. F., Kruse, G. W. & Loth, F. 2002 Spectral element methods for transitional flows in complex geometries. *J. Sci. Comput.* **17**, 81–98.

Greenbaum, A. 1997 *Iterative Methods for Solving Linear Systems*. Philadelphia: SIAM.

Gustafsson, B. & Nilsson, J. 2000 Boundary conditions and estimates for the steady Stokes equations on staggered grids. *J. Sci. Comput.* **15**, 29–54.

von Kármán, T. 1954 *Aerodynamics: selected topics in the light of their historical development*. New York: Cornell University Press.

Kim, J. & Moin, P. 1985 Application of a fractional-step method to incompressible Navier-Stokes equations. *J. Comput. Phys.* **59**, 308–323.

Kress, W. & Lötstedt, P. 2004 Time step restrictions using semi-implicit methods for the incompressible Navier-Stokes equations. *Tech. Rep.* TR 2004-030, available at http://www.it.uu.se/research/reports/. Dept of Information Technology, Uppsala University, Uppsala, Sweden.

Kress, W. & Nilsson, J. 2003 Boundary conditions and estimates for the linearized Navier-Stokes equations on a staggered grid. *Computers & Fluids* **32**, 1093–1112.

Lele, S. K. 1992 Compact finite difference schemes with spectral-like resolution. *J. Comput. Phys.* **103**, 16–42.

Meijerink, J. A. & van der Vorst, H. A. 1977 An iterative solution method for

linear systems of which the coefficient matrix is a symmetric m-matrix. *Math. Comp.* **31**, 148–162.

Nilsson, J., Gustafsson, B., Lötstedt, P. & Brüger, A. 2003 High order difference method on staggered, curvilinear grids for the incompressible Navier-Stokes equations. In *Second MIT Conference on Computational Fluid and Solid Mechanics 2003* (ed. K. J. Bathe), pp. 1057–1061. Elsevier.

Orszag, S. A. & Patterson, G. S. 1972 Numerical simulation of three-dimensional homogeneous isotropic turbulence. *Phys. Lett. Rev.* **28**, 76–79.

Perot, J. B. 1993 An analysis of the fractional step method. *J. Comput. Phys.* **108**, 51–58.

van der Vorst, H. A. 1992 Bi-cgstab: A fast and smoothly converging variant of bi-cg for solution of nonsymmetric systems. *J. Sci. Stat. Comput.* **13**, 631–644.

Part 2

Papers

1

Paper 1

A hybrid high order method for incompressible flow in complex geometries / version 2

By Arnim Brüger[1], Erik Stålberg[1], Jonas Nilsson[2], Wendy Kress[2], Bertil Gustafsson[2], Per Lötstedt[2], Arne V. Johansson[1] and Dan S. Henningson[1]

A numerical method is presented for solution of the incompressible Navier–Stokes equations in primitive variables. The method can be applied to three-dimensional, internal or external flows with one ore more periodic directions. Well posed formulations of the boundary conditions are used. The discretization is of hybrid type consisting of compact high order difference operators applied to a two dimensional curvilinear domain and a Fourier expansion in the periodic direction. Parasitic checkerboard oscillations in the pressure are suppressed by staggering the grid locations. Orthogonal transformations are used between computational and physical space. For stability reasons the local representation of velocities is chosen. The time derivative is discretized with the second order backward difference method in a semi-implicit scheme. An approximate factorization method is applied in order to solve the resulting large linear system of equations. It is based on a block LU factorization of an approximated system matrix. The solution procedure is formulated in an iterative scheme combining inner and outer iterations. The inner iterations consist of the solution of three subsystems, two for the velocities and one for the pressure.

---

[1]Department of Mechanics, KTH, SE-10044 Stockholm, Sweden
[2]Department of Information Technology, Scientific Computing, Uppsala University, P. O. Box 337, SE-75105 Uppsala, Sweden.

Preface

The present document is a revised version of the report about the hybrid high order incompressible Navier-Stokes solver developed in a cooperation project between the Department of Mechanics, KTH and the Department of Information Technology, Uppsala University. The documentation focuses on the numerical implementation in the Fortran90 language. The main differences in the text compared to the first version are the description of the solution technique for prediction of velocities and an extended description regarding the boundary conditions. The formulation of the underlying equations in orthogonal curvilinear coordinates has also been enhanced.

1. Introduction

Solution of the Navier–Stokes equations in turbulent flow applications requires accurate resolution of both the large and the small scale fluctuations. Any standard local discretization technique of second order accuracy can achieve the desired level of numerical error by increasing the number of discretization points, but only at the expense of efficiency loss. This is why still at this point of time it is solely the spectral methods that achieve reasonable Reynolds numbers in direct numerical simulation (DNS) of turbulent flows. The downside is that we lack good numerical experiments in complex geometries, which can not be described with global discretization techniques.

Comparably few approaches exist, where focus is put on both numerical accuracy and flexibility concerning the geometry. In the method described here, we aim at accuracy which is not significantly inferior to that of spectral methods and we offer to choose geometries that can be defined by two dimensions. Our hybrid numerical method is designed from the scratch and combines a high order compact discretization technique in the two dimensional curvilinear space with an expansion in Fourier modes in the third direction, which is assumed to be homogeneous.

The goal with this approach is neither to achieve higher Reynolds numbers nor to compete with the efficiency of a purely spectral method. We aim at simulations in body-fitted coordinates with flow exposed to an environment closer to realistic situations involving curvature and non-trivial boundaries.

The potential in higher order approaches to the Navier–Stokes equations is documented in a number of successful studies as in Gullbrand (2000), Visbal & Gaitonde (2002), Bhaganagar *et al.* (2002), Nagarajan *et al.* (2003), Piller & Stalio (2004).

Related work in iterative and fractional step solution techniques can be found in Brown *et al.* (2001), Greenbaum (1997), Kim & Moin (1985), Perot (1993), Strikwerda & Lee (1999), van Kan (1986).

Discussion of spectral methods can be found in Canuto *et al.* (1988), for description of high order discretizations see e.g. Lele (1992), Fornberg & Ghrist (1999).

A number of related investigations and numerical examples have been provided in context with the development of the method proposed here, see Brüger (2004), Brüger (2002), Brüger *et al.* (2002), Brüger *et al.* (2005), Brüger *et al.* (2004), Göran (2001), Gustafsson & Nilsson (2002), Gustafsson & Nilsson (2000), Gustafsson *et al.* (2003), Kress & Lötstedt (2004), Kress & Nilsson (2003), Nilsson (2000), Nilsson *et al.* (2003) and Stålberg *et al.* (2004).

2. Discretization method

2.1. *Governing equations*

The continuous three-dimensional problem to be solved is described by the incompressible, time-dependent Navier–Stokes equations

$$\frac{\partial \mathbf{u}}{\partial t} + \mathcal{N}(\mathbf{u}) + \mathcal{L}(\mathbf{u}, p) = 0, \tag{1}$$

and the condition

$$\nabla \cdot \mathbf{u} = 0, \tag{2}$$

which explicitly enforces a divergence free field of the components of the velocity vector $\mathbf{u} = (u, v, w)^T$. p is the scalar pressure field divided by density and $R = U_b l/\nu$ is the flow Reynolds number based on characteristic velocity and length scales U_b and l and kinematic viscosity ν. $\mathcal{N}$ are the nonlinear terms

$$\mathcal{N}(\mathbf{u}) = \mathbf{u} \cdot \nabla \mathbf{u}, \tag{3}$$

and $\mathcal{L}$ are the linear terms including the gradient of the pressure

$$\mathcal{L}(\mathbf{u}, p) = \nabla p - \frac{1}{R} \nabla^2 \mathbf{u}. \tag{4}$$

The physics of turbulent flow are entirely contained in these equations provided that all fluctuating scales are resolved in both time and space. The remaining error of a direct simulation is then within the accuracy of the numerical discretization and solution techniques.

2.2. *Boundary conditions*

The specification of boundary conditions for the incompressible Navier–Stokes equations should be handled with care as we learn from numerous discussions in the literature, see e.g. Peyret & Taylor (1983), Ferziger (1987), Rempfer (2003). The formulation used here is based on the ideas of Gustafsson & Nilsson (2000) where a non-singular formulation of boundary conditions was suggested for the steady Stokes equations. A direct generalization of this type to the linearized Navier–Stokes is found in Kress & Nilsson (2003). The problematic nature inhering the divergence form is demonstrated considering a two dimensional domain $\Omega = \{0 \le x \le 1, 0 \le y \le 2\pi\}$, which is periodic in y direction. The integration of the divergence condition (2) leads to the expression

$$\int_0^{2\,\pi} u(0, y) \mathrm{d}y = \int_0^{2\,\pi} u(1, y) \mathrm{d}y, \tag{5}$$

which is a restriction in the choice of the boundary data. When velocities conditions only were set along the boundaries, the system of equations resulting from the discretization would become singular.

Kress & Nilsson (2003) propose a formulation that removes the restriction and that allows small perturbations in the boundary data without causing instabilities in the solution. The idea behind is to include a condition on the pressure in order to obtain a well posed linear system of equations. By means of

an integral formulation of the boundary data, one degree of freedom is released and the pressure condition can be specified.

Two types of boundary conditions are considered when investigating internal flow problems as in e.g. channel flow. For certain problems, Dirichlet conditions can be specified at the walls and at both inlet and outlet. The continuous formulation is here

$$\begin{aligned} u(0,y) - \frac{1}{2\pi}\int_0^{2\pi} u(0,y)\mathrm{d}y = g_0(y)\,, \qquad & u(1,y) = u_1(y)\,, \\ v(0,y) = v_0(y)\,, \qquad & v(1,y) = v_1(y)\,, \\ w(0,y) = w_0(y)\,, \qquad & w(1,y) = w_1(y)\,, \\ \int_0^{2\pi} u(0,y)\mathrm{d}y + \int_0^{2\pi} p(0,y)\mathrm{d}y = q_0\,, & \end{aligned} \tag{6}$$

with $\int_0^{2\pi} g_0(y)\mathrm{d}y = 0$. The more common condition at the outlet is to prescribe derivatives of the velocity components, i.e. the Neumann condition. The continuous formulation is

$$\begin{aligned} u(0,y) = u_0(y)\,, \qquad & \frac{\partial u(1,y)}{\partial x} - \frac{1}{2\pi}\int_0^{2\pi}\frac{\partial u(1,y)}{\partial x}\mathrm{d}y = g_1(y)\,, \\ v(0,y) = v_0(y)\,, \qquad & \frac{\partial v(1,y)}{\partial x} = v_1(y)\,, \\ w(0,y) = w_0(y)\,, \qquad & \frac{\partial w(1,y)}{\partial x} = w_1(y), \\ & \int_0^{2\pi}\frac{\partial u(1,y)}{\partial x}\mathrm{d}y + \int_0^{2\pi} p(1,y)\mathrm{d}y = q_0\,. \end{aligned} \tag{7}$$

with $\int_0^{2\pi} g_1(y)\mathrm{d}y = 0$. q_0 in (6) and (7) is an arbitrary constant.
For a Neumann condition with zero derivative of the velocity components prescribed, $g_1 = v_1 = w_1 = 0$.

2.3. *Staggered grids*

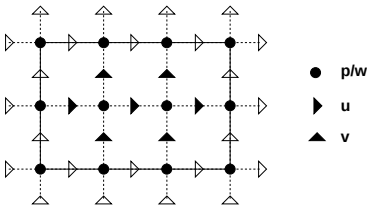

FIGURE 1. The staggered grid

Solution of incompressible flow may induce parasitic fluctuations in the pressure, which is not detected by the numerical scheme. This problem occurs

on standard grids and can be tackled by means of filter functions. A cleaner way to circumvent pressure oscillations is to stagger the variables. Figure 1 shows the arrangement of variables in the staggered grid. The number of pressure discretization points in the two dimensional domain, that may be mapped to the physical curvilinear space are N_x N_y in x, y dimension and the corresponding grid spacings are $\mathrm{d}x$, $\mathrm{d}y$. The pressure points are located in the center of the cells assigned to $(i\,\mathrm{d}x, j\,\mathrm{d}y)$ with $(i = 0 \ldots N_x - 1, j = 0 \ldots N_y - 1)$ and the velocity points are distributed around them at different positions. u and v are assigned to $(i\,\mathrm{d}x - 0.5\ \mathrm{d}x, j\,\mathrm{d}y)$ with $(i = 0 \ldots N_x, j = 0 \ldots N_y - 1)$ and $(i\,\mathrm{d}x, j\,\mathrm{d}y - 0.5\ \mathrm{d}y)$ with $(i = 0 \ldots N_x - 1, j = 0 \ldots N_y)$, respectively. The location of the w component coincides with the one for the pressure. It implies that the first momentum equation is solved on the u-points, the second on v and the third and the divergence condition is enforced on the p/w-positions.

2.4. *Time integration*

For the time integration, one has the choice of explicit, implicit, or semi-implicit schemes. For explicit schemes, the time step restriction is often of the form $\Delta t \leq c\,R\,\Delta x^2$, where c is a constant. Especially for locally refined meshes in space this can be a very severe limitation resulting in long computing times. Fully implicit schemes on the other hand require the solution of a nonlinear system of equations in each time-step. We use a semi-implicit scheme, where the nonlinear convection terms are treated in an explicit manner, and all other terms are treated implicitly.

We use a second order time discretization scheme in which the time derivative is approximated by a backward differentiation formula, BDF2

$$\frac{\partial \mathbf{u}^{n+1}}{\partial t} = \frac{3\mathbf{u}^{n+1} - 4\mathbf{u}^{n} + \mathbf{u}^{n-1}}{2\Delta t} + \ \mathcal{O}(\Delta t^2), \tag{8}$$

where n indicates the time level. The system of equations is treated implicitly except for the nonlinear terms. In order to achieve second order in time solutions the following linear extrapolation from the two preceding time steps is used

$$\mathcal{N}^{n+1} = 2\,\mathcal{N}^{n} - \ \mathcal{N}^{n-1} + \ \mathcal{O}(\Delta t^2) \tag{9}$$

The implicit-explicit time discretized scheme reads

$$\frac{3}{2}\mathbf{u}^{n+1} + \Delta t \mathcal{L}^{n+1} = 2\mathbf{u}^{n} - \frac{1}{2}\mathbf{u}^{n-1} - \Delta t\,\mathcal{N}^{n+1} \tag{10}$$

$$\nabla \cdot \mathbf{u}^{n+1} = 0. \tag{11}$$

Remark that the gradient of the pressure appears in $\mathcal{L}^{n+1}$ and thus is treated implicitly. The divergence condition (11) is enforced implicitly.

In Kress & Lötstedt (2004), a stability analysis for a class of semi-implicit integration schemes including the scheme described above is performed using Fourier analysis, see also Gustafsson *et al.* (2003).

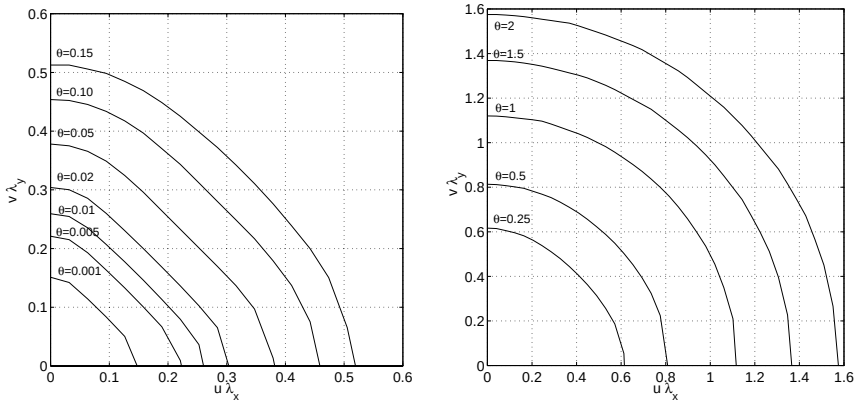

FIGURE 2. Stability domain for BDF-2 and different θ. Close up view for small θ (left) and for larger θ (right). $\lambda_x = \frac{\Delta t}{\Delta x}$, $\lambda_y = \frac{\Delta t}{\Delta y}$.

In figure 2.4 the boundary of the stability domain for the above scheme is shown for different values of $\theta = \frac{\Delta t}{R\,\Delta x^2}$. The time stepping scheme is stable if Δt is chosen such that $\left(u\frac{\Delta t}{\Delta x}, v\frac{\Delta t}{\Delta y}\right)$ lies inside the lines depicted in the figure.

It can be shown that the time step restriction due to stability can be relaxed considerably compared to explicit schemes. Furthermore, it is always possible to choose a sufficiently small Δt so that the integration is stable and then Δt will be proportional to $\Delta x^{2/3} R^{-1/3}$. For the case where $\Delta x << R^{-1/2}$, which includes the case of locally highly refined grids the stability condition can be further relaxed.

2.5. *Coordinate transformation*

A key requirement to our numerical method is the capability to handle complex geometries. We allow to define a particular geometry by two dimensions. The mapping from the equidistant (Cartesian) computational space to the curvilinear physical space is performed by an orthogonal transformation. The advantage is to reduce the number of terms to evaluate. It remains to find suitable orthogonal transformations for a particular geometries. Conformal mappings have this property and numerical algorithms exist to compute the desired grids, see for example Driscoll & Trefethen (2002).

Starting from the Navier–Stokes equations in Cartesian coordinates one applies an orthogonal transformation between computational (ξ, η, ζ) and physical space (x, y, z)

$$\begin{aligned} x &= x(\xi, \eta, \zeta), \\ y &= y(\xi, \eta, \zeta). \end{aligned} \tag{12}$$

Because of the two dimensionality of the geometries, we have $z = \zeta$.

At this point one has the choice between a global or a local formulation of velocity components. The global one is based on applying the chain rule to all space derivatives whereas the local one makes the velocities pointing in the coordinate directions ξ and η. Nilsson (2000) performed a stability analysis for the Stokes equations for local and global coordinate transformations and showed that the problem of spurious oscillations remained, when the global formulation was used. Applying the local transformation leads to a more complex structure of the Laplace operator, but the formulation of the convective terms is simpler than in the global approach. We obtain the transformed Navier–Stokes equations in the form

$$\frac{\partial \mathbf{u}}{\partial t} + \mathcal{N}^*(\mathbf{u}) + \mathcal{L}^*(\mathbf{u}, p) = 0, \tag{13}$$

$$\nabla \cdot \mathbf{u} = 0, \tag{14}$$

where $\mathcal{N}^*(\mathbf{u})$ are the transformed nonlinear terms and $\mathcal{L}^*(\mathbf{u}, p)$ the transformed linear terms. The divergence reads now

$$\nabla \cdot \mathbf{u} = \frac{1}{n_1 n_2}\left(\frac{\partial}{\partial \xi}(n_2 u) + \frac{\partial}{\partial \eta}(n_1 v) + n_1 n_2 \frac{\partial}{\partial z} w\right), \tag{15}$$

and the pressure gradient is

$$\nabla p = \left(\frac{1}{n_1}\frac{\partial p}{\partial \xi},\ \frac{1}{n_2}\frac{\partial p}{\partial \eta},\ \frac{\partial p}{\partial z}\right)^T, \tag{16}$$

including the scale factors of the orthogonal transformation

$$n_1 = \sqrt{x_\xi^2 + y_\xi^2}, \tag{17}$$

$$n_2 = \sqrt{x_\eta^2 + y_\eta^2}. \tag{18}$$

where subscript ξ and η denotes derivative.
In the orthogonal coordinate system $\boldsymbol{\xi} = (\xi, \eta, \zeta)$ we have

$$\mathcal{N}^* = \mathcal{N}^*(\mathbf{u}) = \sum_j u_j (\nabla u)_{ji} = \sum_j \frac{u_j}{n_j}\frac{\partial u_i}{\partial \xi_j} + \frac{u_i}{n_i}\frac{u_j}{n_j}\frac{\partial n_i}{\partial \xi_j} - \frac{u_j}{n_i}\frac{u_j}{n_j}\frac{\partial n_j}{\partial \xi_i}, \tag{19}$$

see Redžić (2001). More specific we have the components

$$\mathcal{N}_\xi^* = \frac{u}{n_1 n_2}(n_2 \frac{\partial}{\partial \xi} u + n_{1\eta}\ \mathcal{E}v) + \frac{\mathcal{E}v}{n_1 n_2}(n_1 \frac{\partial}{\partial \eta} u - n_{2\xi}\ \mathcal{E}v) + w\frac{\partial}{\partial \zeta} u, \tag{20}$$

$$\mathcal{N}_\eta^* = \frac{\mathcal{E}u}{n_1 n_2}(n_2 \frac{\partial}{\partial \xi} v - n_{1\eta}\ \mathcal{E}u) + \frac{v}{n_1 n_2}(n_1 \frac{\partial}{\partial \eta} v + n_{2\xi}\ \mathcal{E}u) + w\frac{\partial}{\partial \zeta} v, \tag{21}$$

$$\mathcal{N}_\zeta^* = \frac{\mathcal{E}u}{n_1}\frac{\partial}{\partial \xi} w + \frac{\mathcal{E}v}{n_2}\frac{\partial}{\partial \eta} w + w\frac{\partial}{\partial \zeta} w, \tag{22}$$

where $\mathcal{E}$ is understood as generic interpolation operator between the nodes of the staggered grid. The laplacian $\nabla^2\mathbf{u}$ can be written as

$$\begin{aligned}\nabla^2\mathbf{u} &= \sum_j(\nabla\cdot\nabla\mathbf{u})_{jji} = \sum_j\left\{\frac{1}{n_1n_2}\frac{\partial}{\partial\xi_j}\left[\frac{n_1n_2}{n_j^2}\frac{\partial u_i}{\partial\xi_j}\right]\right.\\ &+ \frac{n_iu_i}{n_1n_2}\frac{\partial}{\partial\xi_j}\left[\frac{n_1n_2}{n_i^2n_j^2}\frac{\partial n_i}{\partial\xi_j}\right] + \frac{2}{n_i^2n_j^2}\frac{\partial n_i}{\partial\xi_j}\frac{\partial}{\partial\xi_j}(n_ju_j) \qquad (23)\\ &- \left.\frac{2}{n_in_j^3}\frac{\partial n_j}{\partial\xi_i}\frac{\partial}{\partial\xi_j}(n_ju_j) + \frac{n_ju_j}{n_i}\frac{\partial}{\partial\xi_i}\left[\frac{1}{n_1n_2}\frac{\partial}{\partial\xi_j}\left(\frac{n_1n_2}{n_j^2}\right)\right]\right\}.\end{aligned}$$

For convenience we rewrite

$$\mathcal{L}^* = \nabla p - \frac{1}{n_1n_2R}\,\mathcal{B}^*, \qquad (24)$$

where the components of $\mathcal{B}^* = n_1n_2\nabla^2\mathbf{u}$ are

$$\begin{aligned}\mathcal{B}^*{}_\xi &= \frac{\partial}{\partial\xi}\left(\frac{n_2}{n_1}\frac{\partial}{\partial\xi}u\right) + \frac{\partial}{\partial\eta}\left(\frac{n_1}{n_2}\frac{\partial}{\partial\eta}u\right) - \frac{n_{1\,\eta}^2 + n_{2\,\xi}^2}{n_1n_2}u + 2\,\frac{n_{1\,\eta}}{n_1}\frac{\partial}{\partial\xi}\mathcal{E}v\\ &- 2\,\frac{n_{2\,\xi}}{n_2}\frac{\partial}{\partial\eta}\mathcal{E}v + \frac{\partial}{\partial\xi}\left(\frac{n_{1\,\eta}}{n_1} - \frac{n_{2\,\eta}}{n_2}\right)\mathcal{E}v + n_1n_2\frac{\partial^2}{\partial\zeta^2}u \qquad (25)\\ \mathcal{B}^*{}_\eta &= \frac{\partial}{\partial\xi}\left(\frac{n_2}{n_1}\frac{\partial}{\partial\xi}v\right) + \frac{\partial}{\partial\eta}\left(\frac{n_1}{n_2}\frac{\partial}{\partial\eta}v\right) - \frac{n_{1\,\eta}^2 + n_{2\,\xi}^2}{n_1n_2}v - 2\,\frac{n_{1\,\eta}}{n_1}\frac{\partial}{\partial\xi}\mathcal{E}u\\ &+ 2\,\frac{n_{2\,\xi}}{n_2}\frac{\partial}{\partial\eta}\mathcal{E}u - \frac{\partial}{\partial\xi}\left(\frac{n_{1\,\eta}}{n_1} - \frac{n_{2\,\eta}}{n_2}\right)\mathcal{E}u + n_1n_2\frac{\partial^2}{\partial\zeta^2}v \qquad (26)\\ \mathcal{B}^*{}_\zeta &= \frac{\partial}{\partial\xi}\left(\frac{n_2}{n_1}\right)\frac{\partial}{\partial\xi}w + \frac{\partial}{\partial\eta}\left(\frac{n_1}{n_2}\right)\frac{\partial}{\partial\eta}w\\ &+ \left(\frac{n_2}{n_1}\right)\frac{\partial^2}{\partial\xi^2}w + \left(\frac{n_1}{n_2}\right)\frac{\partial^2}{\partial\eta^2}w + n_1n_2\frac{\partial^2}{\partial\zeta^2}w \qquad (27)\end{aligned}$$

where

$$n_{1\xi} = \frac{x_\xi x_{\xi\xi} + y_\xi y_{\xi\xi}}{n_1}, \qquad (28)$$

$$n_{1\eta} = \frac{x_\xi x_{\xi\eta} + y_\xi y_{\xi\eta}}{n_1}, \qquad (29)$$

$$n_{2\xi} = \frac{x_\eta x_{\xi\eta} + y_\eta y_{\xi\eta}}{n_2}, \qquad (30)$$

$$n_{2\eta} = \frac{x_\eta x_{\eta\eta} + y_\eta y_{\eta\eta}}{n_2}, \qquad (31)$$

2.6. *Spanwise Fourier transform*

We start with the discretization of the spanwise dimension, since it has special properties. The geometries considered here are described in two dimensional space which allows to assume that the turbulence is homogeneous in the third spanwise direction. Provided that the computational box has a sufficient extension in spanwise direction periodicity may be assumed and the variables can be expanded in Fourier modes. The computational grid contains N_z evaluation points in physical space in the z-dimension at locations

$$z_j = \frac{2\pi j}{N_z} \qquad j = 1, 2, 3, \ldots, N_z. \tag{32}$$

The coefficients of the discrete one dimensional forward Fourier transform are then

$$\hat{u}(x,y)_k = \frac{1}{N_z}\sum_{j=1}^{N_z} u(x,y)\, e^{-ikz_j} \qquad -N_z/2 \le k \le (N_z/2-1). \tag{33}$$

The system of equations will be treated in Fourier transformed space except for the computation of the nonlinear convective terms $\mathcal{N}$. In that case we transform the necessary variables back to physical space and we can directly evaluate the nonlinearities. Such methods are often referred as pseudo-spectral type. For computation of the physical values we have the discrete one dimensional inverse Fourier transform in spanwise direction

$$u(x,y,z) = \sum_{k=-N_z/2}^{N_z/2-1} \hat{u}(x,y)\, e^{i\beta_k z_L j/N_z} \qquad j = 1, 2, 3, \ldots, N_z \tag{34}$$

where z_L is the spanwise box length and $\beta_k = 2\pi k/z_L$ are the corresponding wave numbers (the index k will be dropped for convenience).

2.6.1. *Fourier transformed Navier–Stokes equations*

We expand the set of equations (13) and (14) in spanwise Fourier modes and obtain the transformed nonlinear and linear terms $\hat{\mathcal{N}}^* = \hat{\mathcal{N}}^*(\hat{\mathbf{u}})$ and $\hat{\mathcal{B}}^* = \hat{\mathcal{B}}^*(\hat{\mathbf{u}})$

$$\hat{\mathcal{N}}^*_\xi = \frac{\hat{u}}{n_1 n_2}(n_2 \frac{\partial}{\partial\xi}\hat{u} + n_{1\eta}\,\mathcal{E}\hat{v}) + \frac{\mathcal{E}\hat{v}}{n_1 n_2}(n_1 \frac{\partial}{\partial\eta}\hat{u} - n_{2\xi}\,\mathcal{E}\hat{v}) + \hat{w} i\beta \hat{u}, \tag{35}$$

$$\hat{\mathcal{N}}^*_\eta = \frac{\mathcal{E}\hat{u}}{n_1 n_2}(n_2 \frac{\partial}{\partial\xi}\hat{v} - n_{1\eta}\,\mathcal{E}\hat{u}) + \frac{\hat{v}}{n_1 n_2}(n_1 \frac{\partial}{\partial\eta}\hat{v} + n_{2\xi}\,\mathcal{E}\hat{u}) + \hat{w} i\beta \hat{v}, \tag{36}$$

$$\hat{\mathcal{N}}^*_\zeta = \frac{\mathcal{E}\hat{u}}{n_1}\frac{\partial}{\partial\xi}\hat{w} + \frac{\mathcal{E}\hat{v}}{n_2}\frac{\partial}{\partial\eta}\hat{w} + \hat{w} i\beta \hat{w}, \tag{37}$$

$$\begin{aligned}
\hat{\mathcal{B}}^*_\xi &= \frac{\partial}{\partial\xi}\left(\frac{n_2}{n_1}\frac{\partial}{\partial\xi}\hat{u}\right)+\frac{\partial}{\partial\eta}\left(\frac{n_1}{n_2}\frac{\partial}{\partial\eta}\hat{u}\right)-\frac{n_{1\,\eta}^2+n_{2\,\xi}^2}{n_1n_2}\hat{u}+2\,\frac{n_{1\,\eta}}{n_1}\frac{\partial}{\partial\xi}\mathcal{E}\hat{v}\\
&- 2\,\frac{n_{2\,\xi}}{n_2}\frac{\partial}{\partial\eta}\mathcal{E}\hat{v}+\frac{\partial}{\partial\xi}\left(\frac{n_{1\,\eta}}{n_1}-\frac{n_{2\,\eta}}{n_2}\right)\mathcal{E}\hat{v}-n_1n_2\beta^2\hat{u} \qquad (38)\\
\hat{\mathcal{B}}^*_\eta &= \frac{\partial}{\partial\xi}\left(\frac{n_2}{n_1}\frac{\partial}{\partial\xi}\hat{v}\right)+\frac{\partial}{\partial\eta}\left(\frac{n_1}{n_2}\frac{\partial}{\partial\eta}\hat{v}\right)-\frac{n_{1\,\eta}^2+n_{2\,\xi}^2}{n_1n_2}\hat{v}-2\,\frac{n_{1\,\eta}}{n_1}\frac{\partial}{\partial\xi}\mathcal{E}\hat{u}\\
&+ 2\,\frac{n_{2\,\xi}}{n_2}\frac{\partial}{\partial\eta}\mathcal{E}\hat{u}-\frac{\partial}{\partial\xi}\left(\frac{n_{1\,\eta}}{n_1}-\frac{n_{2\,\eta}}{n_2}\right)\mathcal{E}\hat{u}-n_1n_2\beta^2\hat{v} \qquad (39)\\
\hat{\mathcal{B}}^*_\zeta &= \frac{\partial}{\partial\xi}\left(\frac{n_2}{n_1}\right)\frac{\partial}{\partial\xi}\hat{w}+\frac{\partial}{\partial\eta}\left(\frac{n_1}{n_2}\right)\frac{\partial}{\partial\eta}\hat{w}\\
&+ \left(\frac{n_2}{n_1}\right)\frac{\partial^2}{\partial\xi^2}\hat{w}+\left(\frac{n_1}{n_2}\right)\frac{\partial^2}{\partial\eta^2}\hat{w}-n_1n_2\beta^2\hat{w} \qquad (40)
\end{aligned}$$

The condition of divergence free velocity field reads

$$\nabla\cdot\hat{\mathbf{u}}=\frac{1}{n_1n_2}\left(\frac{\partial}{\partial\xi}(n_2\hat{u})+\frac{\partial}{\partial\eta}(n_1\hat{v})\right)+i\beta\hat{w}=0, \qquad (41)$$

and we have the Fourier transformed pressure gradient

$$\nabla\hat{p}=\left(\frac{1}{n_1}\frac{\partial\hat{p}}{\partial\xi},\ \frac{1}{n_2}\frac{\partial\hat{p}}{\partial\eta},\ i\beta\hat{p}\right)^T \qquad (42)$$

2.7. *High order discretization*

2.7.1. *Space discretization*

Padé operators are discussed in detail in Lele (1992). They are here used throughout for computation of space derivatives in planes that can be subjected to coordinate transformation. In any case, we need to solve systems of the type

$$\mathcal{P}f'=\mathcal{Q}f \qquad (43)$$

in order to determine the derivative f' of a function f. For a first, staggered derivative we have the operators $P_1\,,Q_1$ for solution pressure derivatives and $P_2\,,Q_2$ applied to velocities. $\tilde{P}\,,\tilde{Q}$ denote the regular (collocated) first derivatives and $R\,,S$ the regular second derivatives. Applied to the $x-$ direction the complete set of spatial fourth order operators reads

$P_1f'=Q_1f$:

$$P_1f' = \begin{cases} \frac{1}{12672}(24f'_{-\frac{1}{2}}+528f'_{\frac{1}{2}})\,, \\ \frac{1}{24}(f'_{i-\frac{1}{2}}+22f'_{i+\frac{1}{2}}+f'_{i+\frac{3}{2}})\,, \quad i=0,\ldots,N_x-1\,, \\ \frac{1}{12672}(528f'_{N_x-\frac{1}{2}}+24f'_{N_x+\frac{1}{2}})\,, \end{cases} \qquad (44)$$

$$Q_1f = \begin{cases} \frac{1}{12672\Delta x}(-577f_0+603f_1-27f_2+f_3)\,, \\ \frac{1}{\Delta x}(f_{i+1}-f_i)\,, \quad i=0,\ldots,N_x-1\,, \\ \frac{1}{12672\Delta x}(-f_{N_x-3}+27f_{N_x-2}-603f_{N_x-1}+577f_{N_x})\,. \end{cases}$$

$P_2 f' = Q_2 f$:

$$P_2 f' = \begin{cases} \frac{1}{24} f'_1 \,, \\ \frac{1}{24}(f'_{i-1} + 22 f'_i + f'_{i+1}) \,, \quad i = 1, \ldots, N_x - 1 \,, \\ \frac{1}{24} f'_{N_x} \,, \end{cases}$$

$$Q_2 f = \begin{cases} \frac{1}{576\Delta x}(f_{-\frac{1}{2}} - 27 f_{\frac{1}{2}} + 27 f_{\frac{3}{2}} - f_{\frac{5}{2}}) \,, \\ \frac{1}{\Delta x}(f_{i+\frac{1}{2}} - f_{i-\frac{1}{2}}) \,, \quad i = 1, \ldots, N_x - 1 \,, \\ \frac{1}{576\Delta x}(f_{N_x-\frac{5}{2}} - 27 f_{N_x-\frac{3}{2}} + 27 f_{N_x-\frac{1}{2}} - f_{N_x+\frac{1}{2}}) \,. \end{cases} \tag{45}$$

$\tilde{P} f' = \tilde{Q} f$:

$$\tilde{P} f' = \begin{cases} \frac{1}{108}(6 f'_0 + 18 f'_1) \,, \\ \frac{1}{6}(f'_{i-1} + 4 f'_i + f'_{i+1}) \,, \quad i = 1, \ldots, N_x - 1 \,, \\ \frac{1}{108}(18 f'_{N_x-1} + 6 f'_{N_x}) \,, \end{cases}$$

$$\tilde{Q} f = \begin{cases} \frac{1}{108\Delta x}(-17 f_0 + 9 f_1 + 9 f_2 - f_3) \,, \\ \frac{1}{2\Delta x}(f_{i+1} - f_{i-1}) \,, \quad i = 1, \ldots, N_x - 1 \,, \\ \frac{1}{108\Delta x}(f_{N_x-3} - 9 f_{N_x-2} - 9 f_{N_x-1} + 17 f_{N_x}) \,. \end{cases} \tag{46}$$

$R f'' = S f$:

$$R f'' = \begin{cases} \frac{1}{100}(f''_0 + 10 f''_1) \,, \\ \frac{1}{10}(f''_{i-1} + 10 f''_i + f''_{i+1}) \,, \quad i = 1, \ldots, N_x - 1 \,, \\ \frac{1}{100}(10 f''_{N_x-1} + f''_{N_x}) \,, \end{cases}$$

$$S f = \begin{cases} \frac{1}{1200\Delta x^2}(145 f_0 - 304 f_1 + 174 f_2 - 16 f_3 + f_4) \,, \\ \frac{6}{5\Delta x^2}(f_{i-1} - 2 f_i + f_{i+1}) \,, \quad i = 1, \ldots, N_x - 1 \,, \\ \frac{1}{1200\Delta x^2}(f_{N_x-4} - 16 f_{N_x-3} + \\ +174 f_{N_x-2} - 304 f_{N_x-1} + 145 f_{N_x}) \,. \end{cases} \tag{47}$$

These operators can simply be applied to the y-coordinate by changing Δx to Δy and replacing the number of grid points in x-direction N_x with N_y, the number of points in y-direction.

2.7.2. *Solution of tridiagonal systems*

As described above the implicit formulation of spatial derivatives results in a linear system of equations with a tridiagonal system matrix. The computation of spatial derivatives is one of the most often called routines at execution. Still the number of the various system matrices is limited and the corresponding operators are repeatedly applied. Hence, instead of saving the actual tridiagonal operators the LU factors of the system matrices are computed at initialization which just requires a forward and backward substitution step each time we solve a tridiagonal system.

2.7.3. *Padé interpolations*

Because of the staggered representation of grid data there is a need to interpolate between u, v and p positions. To maintain high accuracy the interpolations are constructed in compact form. Another advantage is that the interpolation operators can be made periodic in the same manner as the compact derivatives.

In the case of interpolations between u and v locations, the Padé interpolation has to be called twice, since it is applied in one dimension at a time. The velocities are first interpolated to p nodes and then to the final position. In order to reduce the error of interpolations to a level below the discretization errors sixth order compact interpolation operators are used.

The operators E_0,F_0 perform the interpolation from u positions to p positions in the x direction. The same operator can be used to interpolate in y direction from v to p and can simply be applied by substitution of N_x by N_y.

In the opposite direction we interpolate with the operators E_1 and F_1 from p to u locations. Also here we can apply this operator to the interpolation from p to v by changing the number of grid points and the direction. Because of the sixth order expansion of E_1 and F_1 also the computation of the point next to the boundary differs from the inner points. However the operators were derived in a way that makes the structure of E_1 similar to the ones in the previously defined left hand side operators. In this way, the same tridiagonal solution routine can be applied to the system.

$$E_0 p = F_0 u\text{:}$$

$$\begin{aligned}
E_0 p &= \begin{cases} \frac{1}{3}(3p_0 + 7p_1)\,, \\ \frac{1}{10}(3p_{i-1} + 10p_i + 3p_{i+1})\,, \quad i = 0, \ldots, N_x - 1\,, \\ \frac{1}{3}(7p_{N_x-1} + 3p_{N_x})\,, \end{cases} \\
F_0 u &= \begin{cases} \frac{1}{192}(+35u_{-\frac{1}{2}} + 420u_{\frac{1}{2}} + 210u_{\frac{3}{2}} - 28u_{\frac{5}{2}} + 3u_{\frac{7}{2}})\,, \\ \frac{1}{20}(u_{i-\frac{3}{2}} + 15u_{i-\frac{1}{2}} + 15u_{i+\frac{1}{2}} + u_{i+\frac{3}{2}})\,, \quad i = 0, \ldots, N_x - 1\,, \\ \frac{1}{192}(+3u_{N_x-\frac{7}{2}} - 28u_{N_x-\frac{5}{2}} + 210u_{N_x-\frac{3}{2}} + 420u_{N_x-\frac{1}{2}} + 35u_{N_x+\frac{1}{2}})\,. \end{cases}
\end{aligned} \tag{48}$$

$E_1 u = F_1 p$:

$$E_1 u = \begin{cases} u_{-\frac{1}{2}} + 9u_{\frac{1}{2}}\,, \\ \frac{1}{10}(3u_{-\frac{1}{2}} + 10u_{+\frac{1}{2}} + 3u_{+\frac{3}{2}})\,, \\ \frac{1}{10}(3u_{i-\frac{1}{2}} + 10u_{i+\frac{1}{2}} + 3u_{i+\frac{3}{2}})\,, \quad i = 1, \ldots, N_x - 1\,, \\ \frac{1}{10}(3u_{N_x-\frac{3}{2}} + 10u_{N_x-\frac{1}{2}} + 3u_{N_x+\frac{3}{2}})\,, \\ 9u_{N_x-\frac{1}{2}} + u_{N_x+\frac{1}{2}}\,, \end{cases}$$

$$F_1 p = \begin{cases} \frac{1}{64}(315p_0 + 105p_1 - 63p_2 + 9p_3 - 5p_4)\,, \\ \frac{1}{20}(21p_0 + 21p_2 - 15p_3 + 6p_4 - 1p_5)\,, \\ \frac{1}{20}(p_{i-1} + 15p_i + 15p_{i+1} + p_{i+2})\,, \quad i = 1, \ldots, N_x - 1\,, \\ \frac{1}{20}(p_{N_x-5} + 6p_{N_x-4} - 15p_{N_x-3} + 21p_{N_x-2} + 21p_{N_x})\,, \\ \frac{1}{64}(-5p_{N_x-4} + 9p_{N_x-3} - 63p_{N_x-2} + 105p_{N_x-1} + 315p_{N_x})\,. \end{cases} \tag{49}$$

2.7.4. *Periodic operators*

Periodic operators are useful for a number of flow problems that involve two periodic dimensions. Examples are external flows that involve O-grids as in Stålberg *et al.* (2004), or turbulent flow in a straight or curved channel, see Brüger (2004). From the previously defined non-periodic operators the periodic formulations can easily be obtained by just considering the prescription for inner locations. The major difference is to solve systems with periodic tridiagonal matrices. The corresponding solution algorithm differs from the one used for the non-periodic cases due to the extra elements in the lower left and upper right corners of the system matrix.

2.8. *Discrete form of the Navier–Stokes equations*

Now the tools are available for spatial discretization of the terms $\hat{\mathcal{N}}^*(\hat{\mathbf{u}})$ and $\hat{\mathcal{B}}^*(\hat{\mathbf{u}})$. The components of $\hat{\mathcal{N}}^* = \hat{\mathcal{N}}^*(\hat{\mathbf{u}})$ in discrete form are

$$\hat{\mathcal{N}}^*_\xi = \frac{\hat{u}}{n_1 n_2}(n_2 \tilde{P}_\xi^{-1} \tilde{Q}_\xi \hat{u} + n_{1\eta}\, \mathcal{E}\hat{v}) + \frac{\mathcal{E}\hat{v}}{n_1 n_2}(n_1 \tilde{P}_\eta^{-1} \tilde{Q}_\eta \hat{u} - n_{2\xi}\, \mathcal{E}\hat{v}) + \hat{w} i\beta \hat{u}, \tag{50}$$

$$\hat{\mathcal{N}}^*_\eta = \frac{\mathcal{E}\hat{u}}{n_1 n_2}(n_2 \tilde{P}_\xi^{-1} \tilde{Q}_\xi \hat{v} - n_{1\eta}\, \mathcal{E}\hat{u}) + \frac{\hat{v}}{n_1 n_2}(n_1 \tilde{P}_\eta^{-1} \tilde{Q}_\eta \hat{v} + n_{2\xi}\, \mathcal{E}\hat{u}) + \hat{w} i\beta \hat{v}, \tag{51}$$

$$\hat{\mathcal{N}}^*_\zeta = \frac{\mathcal{E}\hat{u}}{n_1} \tilde{P}_\xi^{-1} \tilde{Q}_\xi \hat{w} + \frac{\mathcal{E}\hat{v}}{n_2} \tilde{P}_\eta^{-1} \tilde{Q}_\eta \hat{w} + \hat{w} i\beta \hat{w}. \tag{52}$$

Note that also the scale factors n_1, n_2 and their derivatives are discretized with the same type of high order operators, not shown here for convenience. We also keep the generic operator $\mathcal{E}$ for the interpolation between staggered grid nodes.

The discrete components of $\hat{\mathcal{B}}^* = \hat{\mathcal{B}}^*(\hat{\mathbf{u}})$ are

$$\begin{aligned}
\hat{\mathcal{B}}^*_\xi &= \tilde{P}_\xi^{-1}\tilde{Q}_\xi\left(\frac{n_2}{n_1}\tilde{P}_\xi^{-1}\tilde{Q}_\xi\hat{u}\right) + \tilde{P}_\eta^{-1}\tilde{Q}_\eta\left(\frac{n_1}{n_2}\tilde{P}_\eta^{-1}\tilde{Q}_\eta\hat{u}\right) - \frac{n_{1\,\eta}^2 + n_{2\,\xi}^2}{n_1 n_2}\hat{u} \\
&+ 2\,\frac{n_{1\,\eta}}{n_1}\tilde{P}_\xi^{-1}\tilde{Q}_\xi\mathcal{E}\hat{v} - 2\,\frac{n_{2\,\xi}}{n_2}\tilde{P}_\eta^{-1}\tilde{Q}_\eta\mathcal{E}\hat{v} \\
&+ \tilde{P}_\xi^{-1}\tilde{Q}_\xi\left(\frac{n_{1\,\eta}}{n_1} - \frac{n_{2\,\eta}}{n_2}\right)\mathcal{E}\hat{v} - n_1 n_2\beta^2\hat{u}, \qquad (53) \\
\hat{\mathcal{B}}^*_\eta &= \tilde{P}_\xi^{-1}\tilde{Q}_\xi\left(\frac{n_2}{n_1}\tilde{P}_\xi^{-1}\tilde{Q}_\xi\hat{v}\right) + \tilde{P}_\eta^{-1}\tilde{Q}_\eta\left(\frac{n_1}{n_2}\tilde{P}_\eta^{-1}\tilde{Q}_\eta\hat{v}\right) - \frac{n_{1\,\eta}^2 + n_{2\,\xi}^2}{n_1 n_2}\hat{v} \\
&- 2\,\frac{n_{1\,\eta}}{n_1}\tilde{P}_\xi^{-1}\tilde{Q}_\xi\mathcal{E}\hat{u} + 2\,\frac{n_{2\,\xi}}{n_2}\tilde{P}_\eta^{-1}\tilde{Q}_\eta\mathcal{E}\hat{u} \\
&- \tilde{P}_\xi^{-1}\tilde{Q}_\xi\left(\frac{n_{1\,\eta}}{n_1} - \frac{n_{2\,\eta}}{n_2}\right)\mathcal{E}\hat{u} - n_1 n_2\beta^2\hat{v}, \qquad (54) \\
\hat{\mathcal{B}}^*_\zeta &= \tilde{P}_\xi^{-1}\tilde{Q}_\xi\left(\frac{n_2}{n_1}\right)\tilde{P}_\xi^{-1}\tilde{Q}_\xi\hat{w} + \tilde{P}_\eta^{-1}\tilde{Q}_\eta\left(\frac{n_1}{n_2}\right)\tilde{P}_\eta^{-1}\tilde{Q}_\eta\hat{w} \\
&+ \left(\frac{n_2}{n_1}\right)\tilde{R}_\xi^{-1}\tilde{S}_\xi\hat{w} + \left(\frac{n_1}{n_2}\right)\tilde{R}_\eta^{-1}\tilde{S}_\eta\hat{w} - n_1 n_2\beta^2\hat{w}. \qquad (55)
\end{aligned}$$

The high order operators used so far are of collocated type. In the divergence equation and in the pressure gradient we need to employ the staggered alternatives. The discrete divergence condition is

$$\hat{\mathcal{D}}^*(\hat{\mathbf{u}}) = \frac{1}{n_1 n_2}\left(P_{2\,\xi}^{-1}Q_{2\,\xi}(n_2\hat{u}) + P_{2\,\eta}^{-1}Q_{2\,\eta}(n_1\hat{v})\right) + i\beta\hat{w} = 0, \qquad (56)$$

and we have finally the discretized pressure gradient

$$\nabla\hat{p} = \left(\frac{1}{n_1}P_{1\,\xi}^{-1}Q_{1\,\xi}\,\hat{p},\ \frac{1}{n_2}P_{1\,\eta}^{-1}Q_{1\,\eta}\,\hat{p},\ i\beta\,\hat{p}\right)^T. \qquad (57)$$

2.9. *More about boundary conditions*

2.9.1. *Continuous formulation*

We need to specify more clearly how the boundary conditions depend on the spanwise wave number β. So far the formulation was introduced just for the two dimensional case. For simplicity, consider a Cartesian domain in Fourier space $\Omega = \{x_0 \le x \le x_1, y_0 \le y \le y_1,\ \beta\}$. For Dirichlet boundary conditions on all four boundaries in the (x, y) plane we have in the 3D method to differentiate the cases

$\underline{\beta = 0}$

$$\begin{aligned}
\hat{u}(x_0, y) - \frac{1}{L_y}\int_{y_0}^{y_1} \hat{u}(x_0, y)\mathrm{d}y = g_0(y)\,, \qquad & \hat{u}(x_1, y) = \hat{u}_1(y)\,, \\
\hat{v}(x_0, y) = \hat{v}_0(y)\,, \qquad & \hat{v}(x_1, y) = \hat{v}_1(y)\,, \\
\hat{w}(x_0, y) = \hat{w}_0(y)\,, \qquad & \hat{w}(x_1, y) = \hat{w}_1(y)\,, \\
\int_{y_0}^{y_1} \hat{u}(x_0, y)\mathrm{d}y + \int_{y_0}^{y_1} \hat{p}(x_0, y)\mathrm{d}y = q_0\,, & \\
\hat{u}(x, y_0) = \hat{u}^0(x)\,, \qquad & \hat{u}(x, y_1) = \hat{u}^1(x)\,, \\
\hat{v}(x, y_0) = \hat{v}^0(x)\,, \qquad & \hat{v}(x, y_1) = \hat{v}^1(x)\,, \\
\hat{w}(x, y_0) = \hat{w}^0(x)\,, \qquad & \hat{w}(x, y_1) = \hat{w}^1(x)\,,
\end{aligned} \tag{58}$$

$\underline{\beta \neq 0}$

$$\begin{aligned}
\hat{u}(x_0, y) = \hat{u}_0(y)\,, \qquad & \hat{u}(x_1, y) = \hat{u}_1(y)\,, \\
\hat{v}(x_0, y) = \hat{v}_0(y)\,, \qquad & \hat{v}(x_1, y) = \hat{v}_1(y)\,, \\
\hat{w}(x_0, y) = \hat{w}_0(y)\,, \qquad & \hat{w}(x_1, y) = \hat{w}_1(y)\,, \\
\hat{u}(x, y_0) = \hat{u}^0(x)\,, \qquad & \hat{u}(x, y_1) = \hat{u}^1(x)\,, \\
\hat{v}(x, y_0) = \hat{v}^0(x)\,, \qquad & \hat{v}(x, y_1) = \hat{v}^1(x)\,, \\
\hat{w}(x, y_0) = \hat{w}^0(x)\,, \qquad & \hat{w}(x, y_1) = \hat{w}^1(x)\,.
\end{aligned} \tag{59}$$

L_y is the length of the path of integration, q_0 is an arbitrary constant and $\int_{y_0}^{y_1} g_0(y)\mathrm{d}y = 0$. In a similar way the Neumann conditions are defined.

2.9.2. *Discrete formulation of boundary conditions*

The dashed line in figure 3 represents the physical boundary and it is obvious that boundary data on u and v needs to be interpolated to its staggered position.

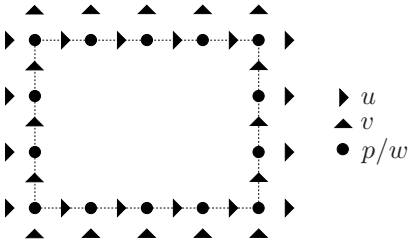

FIGURE 3. Outer point on staggered grid.

Let us introduce the discrete generic operators $\mathcal{I}_E$ for the interpolation of boundary data, $\mathcal{I}_B$ for the discretization of the streamwise derivative at the outflow and $\mathcal{I}_I$ for the numerical integration along boundaries. Then we can reformulate the boundary data ((58), (59)) for the discrete case. The Dirichlet conditions are

$\underline{\beta = 0}$

$$
\begin{aligned}
&\mathcal{I}_{E0}(\hat{u}_{jk}) - \frac{1}{L_\eta}\mathcal{I}_I(n_{2_{0k}}\mathcal{I}_{E0}(\hat{u}_{jk}), \Delta\eta) = g_0(\eta_k) \qquad \forall k \neq k_p\,,\\
&\mathcal{I}_I(n_{2_{0k}}\mathcal{I}_{E0}(\hat{u}_{jk}), \Delta\eta) + \mathcal{I}_I(n_{2_{0k}}\,\hat{p}_{0k}, \Delta\eta) = q_0 \qquad k = k_p\,,\\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \mathcal{I}_{E1}(\hat{u}_{jk}) = \hat{u}_1(\eta_k)\,,\\
&\qquad\qquad\qquad\qquad \hat{v}_{0k} = \hat{v}_0(\eta_k)\,, \qquad \hat{v}_{N_x k} = \hat{v}_1(\eta_k)\,,\\
&\qquad\qquad\qquad\qquad \hat{w}_{0k} = \hat{w}_0(\eta_k)\,, \qquad \hat{w}_{N_x k} = \hat{w}_1(\eta_k)\,,\\
&\qquad\qquad\qquad\qquad \hat{u}_{j0} = \hat{u}^0(\xi_j)\,, \qquad \hat{u}_{jN_y} = \hat{u}^1(\xi_j)\,,\\
&\qquad\qquad\qquad\qquad \mathcal{I}_{E0}(\hat{v}_{jk}) = \hat{v}^0(\xi_j)\,, \qquad \mathcal{I}_{E1}(\hat{v}_{jk}) = \hat{v}^1(\xi_j)\,,\\
&\qquad\qquad\qquad\qquad \hat{w}_{j0} = \hat{w}^0(\xi_j)\,, \qquad \hat{w}_{jN_y} = \hat{w}^1(\xi_j)
\end{aligned}
\tag{60}
$$

with $\mathcal{I}_I(g_{0k}, \Delta\eta) = 0$. and $L_\eta = \mathcal{I}_I(n_{2_{0k}}, \Delta\eta)$

$\underline{\beta \neq 0}$

$$
\begin{aligned}
&\mathcal{I}_{E0}(\hat{u}_{jk\beta}) = \hat{u}_0(\eta_k, \beta)\,, \qquad \mathcal{I}_{E1}(\hat{u}_{jk\beta}) = \hat{u}_1(\eta_k, \beta)\,,\\
&\qquad \hat{v}_{0k\beta} = \hat{v}_0(\eta_k, \beta)\,, \qquad \hat{v}_{N_x k\beta} = \hat{v}_1(\eta_k, \beta)\,,\\
&\qquad \hat{w}_{0k\beta} = \hat{w}_0(\eta_k, \beta)\,, \qquad \hat{w}_{N_x k\beta} = \hat{w}_1(\eta_k, \beta)\,,\\
&\qquad \hat{u}_{j0\beta} = \hat{u}^0(\xi_j, \beta)\,, \qquad \hat{u}_{jN_y\beta} = \hat{u}^1(\xi_j, \beta)\,,\\
&\mathcal{I}_{E0}(\hat{v}_{jk\beta}) = \hat{v}^0(\xi_j, \beta)\,, \qquad \mathcal{I}_{E1}(\hat{v}_{jk\beta}) = \hat{v}^1(\xi_j, \beta)\,,\\
&\qquad \hat{w}_{j0\beta} = \hat{w}^0(\xi_j, \beta)\,, \qquad \hat{w}_{jN_y\beta} = \hat{w}^1(\xi_j, \beta)
\end{aligned}
\tag{61}
$$

k_p is an arbitrary point at the inflow boundary, where the pressure condition is specified. For the Neumann outflow boundary condition we here assume that the derivatives are set to 0, i.e. $g_1 = v_1 = w_1 = 0$ in (7), and that all other boundaries in the (x, y) plane is Dirichlet boundaries

$\underline{\beta = 0}$

$$
\begin{aligned}
&\mathcal{I}_{E0}(\hat{u}_{jk}) = \hat{u}_0(\eta_k)\,,\\
&\qquad\qquad \forall k \neq k_p \qquad \mathcal{I}_{B1}(\hat{u}_{jk}) - \frac{1}{L_\eta}\mathcal{I}_I(n_{2_{N_x k}}\mathcal{I}_{B1}(\hat{u}_{jk}), \Delta\eta) = 0\,,\\
&\qquad\qquad k = k_p \qquad \mathcal{I}_I(n_{2_{N_x k}}\mathcal{I}_{B1}(\hat{u}_{jk}), \Delta\eta) + \mathcal{I}_I(n_{2_{N_x k}}\,\hat{p}_{N_x k}, \Delta\eta) = q_0\,,\\
&\qquad\qquad \hat{v}_{0k} = \hat{v}_0(\eta_k)\,, \qquad \mathcal{I}_{B1}(\hat{v}_{jk}) = 0\,,\\
&\qquad\qquad \hat{w}_{0k} = \hat{w}_0(\eta_k)\,, \qquad \mathcal{I}_{B1}(\hat{w}_{jk}) = 0\,,\\
&\qquad\qquad \hat{u}_{j0} = \hat{u}^0(\xi_j)\,, \qquad \hat{u}_{jN_y} = \hat{u}^1(\xi_j)\,,\\
&\qquad\qquad \mathcal{I}_{E0}(\hat{v}_{jk}) = \hat{v}^0(\xi_j)\,, \qquad \mathcal{I}_{E1}(\hat{v}_{jk}) = \hat{v}^1(\xi_j)\,,\\
&\qquad\qquad \hat{w}_{j0} = \hat{w}^0(\xi_j)\,, \qquad \hat{w}_{jN_y} = \hat{w}^1(\xi_j)
\end{aligned}
\tag{62}
$$

with $L_\eta = \mathcal{I}_I(n_{2_{N_x k}}, \Delta\eta)$

$\underline{\beta \neq 0}$

$$\begin{aligned}
\hat{u}_{0k\beta} &= \hat{u}_0(\eta_k, \beta)\,, & \mathcal{I}_{B1}(\hat{u}_{jk\beta}) &= 0\,,\\
\mathcal{I}_{E0}(\hat{v}_{jk\beta}) &= \hat{v}_0(\eta_k, \beta)\,, & \mathcal{I}_{B1}(\hat{v}_{jk\beta}) &= 0\,,\\
\hat{w}_{0k\beta} &= \hat{w}_0(\eta_k, \beta)\,, & \mathcal{I}_{B1}(\hat{w}_{jk\beta}) &= 0\,,\\
\hat{u}_{j0\beta} &= \hat{u}^0(\xi_j, \beta)\,, & \hat{u}_{jN_y\beta} &= \hat{u}^1(\xi_j, \beta)\,,\\
\mathcal{I}_{E0}(\hat{v}_{jk\beta}) &= \hat{v}^0(\xi_j, \beta)\,, & \mathcal{I}_{E1}(\hat{v}_{jk\beta}) &= \hat{v}^1(\xi_j, \beta)\,,\\
\hat{w}_{j0\beta} &= \hat{w}^0(\xi_j, \beta)\,, & \hat{w}_{jN_y\beta} &= \hat{w}^1(\xi_j, \beta)
\end{aligned} \tag{63}$$

$\mathcal{I}_B$ involves the coefficients to discretize the Neumann condition and k_p is an arbitrary point on the Neumann boundary. The operators $\mathcal{I}_E$ and $\mathcal{I}_I$ are discussed more thorough in section 3.5.4.

2.10. *Solution procedure*

We have, so far, presented the linear and the nonlinear terms $\hat{\mathcal{L}}$ and $\hat{\mathcal{N}}$ in Fourier space because of the spectral spanwise dimension in our discretization method. Assume, that we can provide the right hand side of our discretized linear system of equations in Fourier space in a specific time level, we can with the help of a suitable solver determine the unknowns. The question is however, how to effectively compute the right hand side, that includes the nonlinear terms $\hat{\mathcal{N}}$. Evaluating the convolution sums in spectral space is expensive since it takes $\mathcal{O}(N^2)$ operations (see, e.g. Canuto *et al.* (1988)). The more efficient way is to use Fast Fourier Transforms (FFT) to bring the variables back to physical space, where the multiplications can easily be performed, and finally transform forward to Fourier space in order to get the spectral right hand side. The operation counts can then be reduced to $\mathcal{O}(Nlog_2N)$ thanks to the efficiency of the FFT algorithm, see e.g. Canuto *et al.* (1988).

The hybrid high order discretization method results in a linear system to solve for each time step. Since we solve in Fourier space we get decoupled systems for each spanwise Fourier mode

$$M(\beta)\hat{U}^{n+1} = \hat{b}^{n+1}(\beta). \tag{64}$$

$\hat{U}^{n+1}$ is the solution vector containing the velocity components and the pressure. The right hand side $\hat{b}^{n+1}(\beta)$ depends on the solution of the preceding time levels only.

The basic steps to perform in the time integration loop are then

1. initialize $\hat{\mathbf{u}}^0$, $\hat{\mathbf{u}}^1$.
2. $n = 1$
3. inverse Fourier transform of $\hat{\mathbf{u}}^n$, $\hat{\mathbf{u}}^{n-1}$ and those spanwise, η and ξ derivatives of $\hat{\mathbf{u}}^n$ and $\hat{\mathbf{u}}^{n-1}$, which are needed to compute $\mathcal{N}$ at the discrete time levels n and $n-1$.
4. compute $\mathcal{N}^{n+1}$ in physical space
5. compute b^{n+1}
6. perform the Fourier transform to obtain $\hat{b}^{n+1}(\beta)$.

7. $\forall \beta$
   solve linear system in Fourier space: $M(\beta)\hat{U}^{n+1} = \hat{b}^{n+1}(\beta)$.
8. $n = n + 1$
9. goto 3.

The solution of the linear systems in step 7 is discussed in the following chapter.

3. Approximate factorization and iterative solution

3.1. *Algorithm*

Our discretization technique leads to a large linear system of equations with a constant system matrix that needs to be solved in Fourier space in each time step. We can write the discrete form of (13) (14) in the following way

$$\frac{3}{2}\hat{\mathbf{u}}^{n+1} + \Delta t \hat{\mathcal{L}}^*(\hat{\mathbf{u}}^{n+1}, \hat{p}^{n+1}) = \hat{\mathbf{b}}, \tag{65}$$

$$\hat{\mathcal{D}}^*(\hat{\mathbf{u}}^{n+1}) = 0, \tag{66}$$

where the right hand side $\hat{\mathbf{b}}$ consists of the nonlinear terms computed according to (9) and the contribution from the discrete time derivative (8), i.e. the solutions from the two preceding time levels. The components of $\hat{\mathcal{L}}^* = \hat{\mathcal{L}}^*(\hat{\mathbf{u}}, \hat{p})$ are

$$\hat{\mathcal{L}^*}_\xi = \frac{1}{n_1} P^{-1}_{1\,\xi} Q_{1\,\xi}\, \hat{p}^{n+1} - \frac{1}{n_1 n_2 R}\, \hat{\mathcal{B}^*}_\xi, \tag{67}$$

$$\hat{\mathcal{L}^*}_\eta = \frac{1}{n_2} P^{-1}_{1\,\eta} Q_{1\,\eta}\, \hat{p}^{n+1} - \frac{1}{n_1 n_2 R}\, \hat{\mathcal{B}^*}_\eta, \tag{68}$$

$$\hat{\mathcal{L}^*}_\zeta = i\beta\, \hat{p}^{n+1} - \frac{1}{n_1 n_2 R}\, \hat{\mathcal{B}^*}_\zeta. \tag{69}$$

$\hat{\mathcal{D}}^*$ is the discrete divergence of $\hat{\mathbf{u}}$. The right hand side vector $\hat{\mathbf{b}}$ is constructed according to (10), (11) including the solution of the two preceding time levels and computation of the discrete nonlinear terms $\hat{\mathcal{N}}$ at both time levels. The algorithm that computes the solution to (65), (66) is one of the key ingredients in the numerical method. The two main requirements to fulfil in order to allow large scale simulations are fast convergence and efficient memory allocation.

3.2. *Linear system of equations*

We rewrite the linear system in the following matrix vector representation

$$\begin{pmatrix} A & 0 & G \\ 0 & \tilde{A} & i\beta\Delta t I \\ D & i\beta I & 0 \end{pmatrix} \begin{pmatrix} \hat{\mathbf{u}}_1^{n+1} \\ \hat{w}^{n+1} \\ \hat{p}^{n+1} \end{pmatrix} = \begin{pmatrix} \hat{\mathbf{b}}_1 \\ \hat{b}_w \\ 0 \end{pmatrix}, \tag{70}$$

where $\hat{\mathbf{u}}_1 = (\hat{u}, \hat{v})^T$. We have

$$A = \frac{3}{2} I - \frac{\Delta t}{R n_1 n_2} (\hat{\mathcal{B}}_\xi + \hat{\mathcal{B}}_\eta). \tag{71}$$

$\tilde{A}$ is the corresponding operator resulting from the w equation and has similar properties. We find the estimate

$$A, \tilde{A} \sim \frac{3}{2} I + \frac{\Delta t}{R} \beta^2 \, I - \frac{\Delta t}{R} L. \tag{72}$$

D is the divergence operator and G represents the gradient on the pressure. L is a discrete Laplace operator corresponding to the 2D (ξ, η) case. The solution vector U consists of the first two velocity components in $\mathbf{u}_1$, the w component and the pressure p. The right hand side vector b includes the explicitly computed nonlinear terms with the solutions from the two preceding time levels. The system matrix M is of the size $((N_x + 1) \times N_y + (N_y + 1) \times N_x + 2N_x \times N_y)^2$ and is too large to be stored. It is not known explicitly due to the compact difference schemes. What is easier to provide are matrix - vector products which advises the use of iterative solution methods.

3.3. *Reduction of boundary data*

The system (70) includes not only the idealized operators A and $\tilde{A}$ but also the boundary data. In order to obtain operators with a *clean* structure the data is formally segregated in inner and boundary locations.

We rearrange the system (70) as follows

$$\underbrace{\begin{pmatrix} A_0 & 0 & A_1 & 0 & G \\ 0 & \tilde{A}_0 & 0 & \tilde{A}_1 & i\beta\Delta t I \\ A_2 & 0 & C & 0 & E \\ 0 & \tilde{A}_2 & 0 & \tilde{C} & 0 \\ D_0 & i\beta I & D_1 & i\beta I & 0 \end{pmatrix}}_{M} \begin{pmatrix} \hat{\mathbf{u}}_1^{n+1} \\ \hat{w}^{n+1} \\ \hat{\tilde{\mathbf{u}}}^{n+1} \\ \hat{\tilde{w}}^{n+1} \\ \hat{p}^{n+1} \end{pmatrix} = \begin{pmatrix} \hat{\mathbf{b}}_1 \\ \hat{b}_w \\ \hat{\tilde{\mathbf{b}}} \\ \hat{\tilde{b}}_w \\ 0 \end{pmatrix}, \tag{73}$$

where the data on inner grid locations is kept in $\mathbf{u_1}$ and w, whereas the boundary data is put in $\tilde{\mathbf{u}}$ and $\tilde{w}$. The matrices A and $\tilde{A}$ in the system (70) are now split into those parts that discretize the inner data (i.e. A_0 and $\tilde{A}_0$) and those operators that act on the boundary data (A_1 and $\tilde{A}_1$). The same principle is applied to the divergence operator. The advantage is that the operators A_0 and $\tilde{A}_0$ have undisturbed diagonally dominant structures.

In the present arrangement we can see the matrices A_2, $\tilde{A}_2$, C and $\tilde{C}$, that stem from the high order interpolation of the boundary data. Note that the actual value of a variable on the physical boundary has to be extrapolated to its position on the numerical boundary of the staggered grid. A_2 and $\tilde{A}_2$ contain the coefficients of the interpolation that belong to data values on inner grid locations, whereas C and $\tilde{C}$ consist of the coefficients of the interpolation on the outermost grid location.

The matrix E involves an implicit condition on the pressure and $E \neq 0$ is applied only for $\beta = 0$. If $E = 0$ would be chosen for $\beta = 0$ the pressure solution would become under-determined and would include an unknown additive constant. Moreover, it causes problems in the solver, since the system matrix

becomes singular. In a particular simulation any condition on the pressure could be chosen that leads to a non-singular system matrix of the discretized equations and the systems can be solved to machine precision. Note, that E is in our method specified directly from the formulation of boundary conditions (58). This coherence of solver and boundary conditions is explained in more detail in Brüger *et al.* (2005).

We now remove the boundary data from the solution vector by solving formally for $\tilde{\mathbf{u}}$

$$\hat{\tilde{\mathbf{u}}}^{n+1} = C^{-1}(\hat{\tilde{\mathbf{b}}} - A_2\hat{\mathbf{u}}_1^{n+1} - E\hat{p}^{n+1}) \tag{74}$$

and analogically for $\tilde{w}$

$$\hat{\tilde{w}}^{n+1} = \tilde{C}^{-1}(\hat{\tilde{b}}_w - \tilde{A}_2\hat{w}^{n+1}). \tag{75}$$

Equations (74), (75) are introduced in the remaining equations for $\mathbf{u}_1$, w and p. The reduced system reads now

$$\underbrace{\begin{pmatrix} A_0 - A_1C^{-1}A_2 & 0 & G - A_1C^{-1}E \\ 0 & \tilde{A}_0 - \tilde{A}_1\tilde{C}^{-1}\tilde{A}_2 & i\beta\Delta t I \\ D_0 - D_1C^{-1}A_2 & i\beta(I - \tilde{C}^{-1}\tilde{A}_2) & -D_1C^{-1}E \end{pmatrix}}_{M_R} \underbrace{\begin{pmatrix} \hat{\mathbf{u}}_1^{n+1} \\ \hat{w}^{n+1} \\ \hat{p}^{n+1} \end{pmatrix}}_{\hat{\mathbf{U}}^{n+1}} = \underbrace{\begin{pmatrix} \hat{\mathbf{b}}_1^* \\ \hat{b}_w^* \\ \hat{b}_2^* \end{pmatrix}}_{\hat{\mathbf{b}}}, \tag{76}$$

including the modified right hand sides

$$\hat{\mathbf{b}}_1^* = \hat{\mathbf{b}}_1 - \tilde{A}_1\tilde{C}^{-1}\hat{\tilde{\mathbf{b}}}, \tag{77}$$

$$\hat{b}_w^* = \hat{b}_w - A_1C^{-1}\hat{\tilde{b}}_w, \tag{78}$$

and

$$\hat{b}_2^* = -D_1C^{-1}\hat{\tilde{\mathbf{b}}} - i\beta\tilde{C}^{-1}\hat{\tilde{b}}_w. \tag{79}$$

In order to simplify the notation we introduce the following operators

$$\hat{A} = A_0 - A_1C^{-1}A_2, \tag{80}$$

$$\hat{D} = D_0 - D_1C^{-1}A_2, \tag{81}$$

$$\hat{G} = G - A_1C^{-1}E, \tag{82}$$

$$\hat{E} = D_1C^{-1}E, \tag{83}$$

$$\hat{\tilde{A}} = \tilde{A}_0 - \tilde{A}_1\tilde{C}^{-1}\tilde{A}_2, \tag{84}$$

and we can write the system as

$$\underbrace{\begin{pmatrix} \hat{A} & 0 & \hat{G} \\ 0 & \hat{\tilde{A}} & i\beta\Delta t I \\ \hat{D} & i\beta(I-\tilde{C}^{-1}\tilde{A}_2) & -\hat{E} \end{pmatrix}}_{M_R} \begin{pmatrix} \hat{\mathbf{u}}_1^{n+1} \\ \hat{w}^{n+1} \\ \hat{p}^{n+1} \end{pmatrix} = \begin{pmatrix} \hat{\mathbf{b}}_1^* \\ \hat{b}_w^* \\ \hat{b}_2^* \end{pmatrix}. \tag{85}$$

3.4. *Block LU factorization*

An approximate factorization of M_R in 85 is constructed in a manner similar to Perot (1993)

$$L = \begin{pmatrix} \hat{A} & 0 & 0 \\ 0 & \hat{\tilde{A}} & 0 \\ \hat{D} & i\beta(I-\tilde{C}^{-1}\tilde{A}_2) & I \end{pmatrix},$$
$$U = \begin{pmatrix} I & 0 & \gamma\hat{G} \\ 0 & I & \gamma i\beta\Delta t I \\ 0 & 0 & Q \end{pmatrix},$$

where

$$Q = -\hat{E} - \gamma\hat{D}\hat{G} + \gamma\beta^2\Delta t(I-\tilde{C}^{-1}\tilde{A}_2). \tag{86}$$

The approximated system matrix

$$M^* = LU = \begin{pmatrix} \hat{A} & 0 & \gamma\hat{A}\hat{G} \\ 0 & \hat{\tilde{A}} & \gamma i\beta\Delta t\hat{\tilde{A}} \\ \hat{D} & i\beta(I-\tilde{C}^{-1}\tilde{A}_2) & -\hat{E} \end{pmatrix}$$

has the following approximation error compared to the initial reduced system matrix

$$M^* - M_R = \begin{pmatrix} 0 & 0 & e1 \\ 0 & 0 & e2 \\ 0 & 0 & 0 \end{pmatrix}.$$

The errors in the approximation can be reduced by a reasonable choice of the factor γ. With (72) and since $\hat{G}$ is of order $\Delta t/h$, we get an estimate for the error $e1$

$$e1 = \gamma\hat{A}\hat{G} - \hat{G} = \hat{G}(\gamma\hat{A} - I). \tag{87}$$

If we choose

$$\gamma = \frac{1}{3/2 + \frac{\Delta t}{R}\beta^2}, \tag{88}$$

the error $e1$ reduces to

$$e1 \sim \mathcal{O}\left(\frac{\gamma\Delta t^2}{h^3\,R}\right). \tag{89}$$

An analysis of the error $e2$ leads to the estimation

$$e2 \sim \mathcal{O}\left(\frac{\gamma\beta\Delta t^2}{h^2\,R}\right). \tag{90}$$

3.5. *Iteration prescription*

We derive an iterative algorithm based on a fixed point iteration scheme with iteration index k for the reduced right hand side $\hat{\mathbf{b}}$ to solve

$$M_R\hat{\mathbf{U}} = \hat{\mathbf{b}}. \tag{91}$$

The solution vector $\hat{\mathbf{U}}^{k+1}$ is the sum

$$\hat{\mathbf{U}}^{k+1} = \mathbf{z}^k + \hat{\mathbf{U}}^k, \tag{92}$$

where $\hat{\mathbf{U}}^k$ is the known solution from the preceding iteration level and $\mathbf{z}^k$ is the contribution from the present iteration. Thus, we want to solve for $\mathbf{z}^k$ in

$$M_R\mathbf{z}^k = \hat{\mathbf{b}} - M_R\hat{\mathbf{U}}^k = \mathbf{r}^k, \tag{93}$$

where $\mathbf{r}^k$ is understood as the residual resulting from iteration k. An approximate solution is

$$\mathbf{z}^k = M^{*-1}\left(\hat{\mathbf{b}} - M_R\hat{\mathbf{U}}^k\right), \tag{94}$$

or written as fixed point iteration

$$\hat{\mathbf{U}}^{k+1} = \hat{\mathbf{U}}^k + M^{*-1}\left(\hat{\mathbf{b}} - M_R\hat{\mathbf{U}}^k\right). \tag{95}$$

With exact or approximate LU factors of the system matrix considerable speed up of the iterative solver can be achieved. The general iteration prescription is then

1. $\mathbf{r}^0 = \hat{\mathbf{b}} - M_R\mathbf{U}^0$
2. $L\mathbf{y}^k = \mathbf{r}^k$
3. $U\mathbf{z}^k = \mathbf{y}^k$
4. $\hat{\mathbf{U}}^{k+1} = \mathbf{z}^k + \hat{\mathbf{U}}^k$
5. $\mathbf{r}^{k+1} = \hat{\mathbf{b}} - M_R\hat{\mathbf{U}}^{k+1}$
6. if $||\mathbf{r}^{k+1}|| \leq \epsilon$ then $\hat{\mathbf{U}}^{n+1} = \hat{\mathbf{U}}^{k+1}$ else $k \leftarrow k+1$, *goto* 2,

where ϵ is the convergence criterion. Within one *outer iteration* we have to solve iteratively for the large systems in the steps 2 and 3, i.e. the forward and backward substitution using the LU factors of M^*.

3.5.1. *Forward and backward substitution*

The system to be solved in step 2 reads

$$\begin{pmatrix} \hat{A} & 0 & 0 \\ 0 & \hat{\tilde{A}} & 0 \\ \hat{D} & i\beta(I - \tilde{C}^{-1}\tilde{A}_2) & I \end{pmatrix} \begin{pmatrix} \mathbf{y}_1 \\ y_w \\ y_2 \end{pmatrix} = \begin{pmatrix} \mathbf{r}_1 \\ r_w \\ r_2 \end{pmatrix}. \tag{96}$$

The computational work to be done is the solution of the two A systems, which can be understood as a predictor step for the velocities u and v and for the w component. In the formal back substitution step 3 we solve the system

$$\begin{pmatrix} I & 0 & \gamma\hat{G} \\ 0 & I & \gamma i\beta\Delta t I \\ 0 & 0 & Q \end{pmatrix} \begin{pmatrix} \mathbf{x}_1 \\ x_w \\ x_2 \end{pmatrix} = \begin{pmatrix} \mathbf{y}_1 \\ y_w \\ y_2 \end{pmatrix}. \tag{97}$$

The over-all dominating work load is the solution of

$$Qx_2 = y_2. \tag{98}$$

Due to the structure of Q we denote this system further on as DG system. Step 3 finds a correction to the predicted velocities and enforces the zero divergence condition.

3.5.2. *A systems*

The first time consuming part of the solution process is solving iteratively for the two subsystems

$$(A_0 - A_1 C^{-1} A_2)\mathbf{y}_1 = \mathbf{r}_1, \tag{99}$$

and

$$(\tilde{A}_0 - \tilde{A}_1 \tilde{C}^{-1} \tilde{A}_2) y_w = r_w. \tag{100}$$

Both system matrices have a similar structure $\sim 3/2\, I + \Delta t \beta^2 / R\, I - \Delta t / R\, L$. L is the discrete Laplace operator in the corresponding 2D (ξ, η) case. Two different iterative solution routines are implemented. The first is a fixed point iteration scheme preconditioned with the inverse of the diagonal γI, see (88). However, the maximum time step may at certain conditions be limited by the fixed point iteration scheme of the A systems and not as expected by the CFL criterion. In Kress & Lötstedt (2004) the convergence of the fixed point iterative solver with diagonal preconditioner is studied and shown to converge if

$$\frac{\Delta t}{R\, h^2} < \frac{\alpha_0}{12\, \beta_i^0} \tag{101}$$

where α_0 and β_i^0 are coefficients in BDF methods and $\Delta t/(R\, h^2) < 3/16$ for BDF2. h is the grid spacing and for small Reynolds numbers with high spatial resolution the restriction (101) on Δt is too high.

The second implementation of iterative solver for the A systems is the Bi-CGSTAB routine, see van der Vorst (1992) and Greenbaum (1997), which is less restrictive than (101) and allows the use of time steps around the theoretical CFL criterion if this is accurate enough.

In order to speed up the convergence of the Bi-CGSTAB method one can provide the routine with the approximate LU factors of the system matrices. Since we do not have the explicit form of the system matrices in (99) and (100), the factorizations of A is based on an explicit 2nd order accurate representation of the system matrices. Then it is factorized once by an incomplete LU decomposition (ILU) and given to the solution routine. The ILU factorization can be based on the explicit 2nd order representation of $3/2\, I + \Delta t \beta^2 / R\, I - \Delta t / R\, L$ or the diagonal $3/2\, I + \Delta t \beta^2 / R\, I$, i.e. with the Laplace term excluded. See section 5.4.2 for practical details regarding the ILU factorization.

3.5.3. *DG system*

The first step in the backward substitution is the solution of system (98) which is

$$\left[-\gamma(D_0 - D_1C^{-1}A_2)G + \gamma\beta^2\Delta t(I - \tilde{C}^{-1}\tilde{A}_2) - D_1C^{-1}E\right]x_2 = y_2. \quad (102)$$

For the solution a Bi-CGSTAB method is used for three main reasons. The first is that the left hand side matrix is non-symmetric, the second is that one can not explicitly compute its transpose and the third is that the number of extra vectors for intermediate results are limited. Other iterative methods require the matrix to be symmetric and positive definite (the conjugate gradient method CG) or to have access to the transpose (the quasi-minimum residual method QMR). The GMRES method is more demanding with respect to work space for the iterations. A possible alternative is the transpose-free QMR algorithm.

The Bi-CGSTAB routine is preconditioned with an ILU decomposition of the *DG* matrix in (102) based on an explicit 2nd order accurate representation in a similar manner as for the A system. The 2nd order discretization has been shown to be spectrally equivalent to the fourth order method for a straight channel, see Brüger *et al.* (2005).

Detailed descriptions of the applied iterative solution routines can be found in Greenbaum (1997).

3.5.4. *Interpolation of boundary data*

We take a closer look at the matrices A_2 and C and their origin. Since the physical and numerical boundaries in the staggered grid do not generally coincide, boundary data need to be interpolated as figure 4 illustrates. A quantity

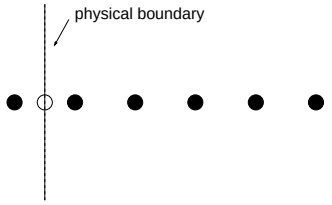

FIGURE 4. Numerical and physical boundary in the staggered grid

$\tilde{f}$ is in computational space interpolated by a suitable high order formula to the physical grid boundary as in

$$\tilde{f}_0 = \mathcal{I}_E(\tilde{f}_{-\frac{h}{2}}, \cdots, \tilde{f}_N) = b_1\tilde{f}_{-\frac{h}{2}} + b_2\tilde{f}_{\frac{h}{2}} + b_3\tilde{f}_{\frac{3h}{2}} + \cdots + b_N\tilde{f}_{\frac{(1+2(N-2))\,h}{2}}. \quad (103)$$

Figure 5 shows the outer points in the staggered grid. The dashed line represents the physical boundary. Further on, we denote the upper, lower, left and right wall as north (N), south (S), west (W) and east (E) position. We see that the interpolation of u boundary data has to be done at locations (W) and (E), whereas the v data has to be interpolated at the walls (S) and (N).

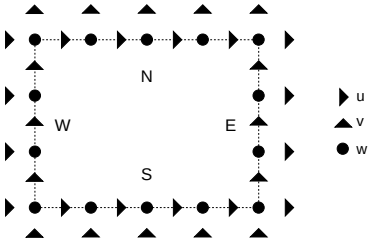

FIGURE 5. Outer points in staggered grid

3.5.5. *The matrix A_2*

Due to the splitting in inner and outer grid points the matrix A_2 appears as a sub-matrix of the system matrix. A_2 holds the interpolation coefficients of formula (103) that are operating on inner grid locations.

In the execution of the solver we need to provide $A_2 z$, where z is a vector defined on inner grid points. The result of $A_2 z$ is equal to zero for the points that coincide with the physical boundary except for those with boundary condition that is not of Dirichlet type. For those grid points that involve interpolation there are two cases to distinguish. The first is simple and this is when no boundary integral is applied. Then, we just apply the part of formula (103) that operates on inner grid locations, $(2 \dots N)$. The exception appears only for wave number $\beta = 0$ at the boundary, where an integral condition is set.

Consider the case that the $u - \int u$ condition is set at the inflow boundary. Then we have to compute the part of the interpolation operating on inner grid locations to obtain the first term in $u - \int u$. To compute the integral contribution we have again to perform the inner part of the interpolation, integrate it and subtract its result from the first term. Note, that in one point of the boundary where the integrals are applied we substitute the boundary condition with just the contribution from the velocity integral in order to end up in formulation (58).

3.5.6. *The matrix C and its inverse*

Another consequence of the grid point division in inner and outer points is that we have to apply the operation $C^{-1} z$ in the solver algorithm, where z is an arbitrary vector containing data on outer grid locations.

To handle the matrix C involves some effort in the case that the integral formulation of boundary conditions is chosen (58). Here, we discuss the structure for two dimensional channel flow, where the integral form is applied to the inflow data at boundary (W).

The general structure of C is shown in figure 6. The part of C that acts on data without integral coupling is simple. The off-diagonal elements come from the computation of the corner points. For this data we can explicitly compute

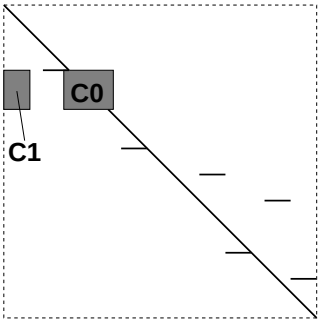

FIGURE 6. A sketch of the structure of the matrix C for the 2D channel flow case with an integral coupling of u points. $C0$ couples the outer points of u at (W) due to the boundary integral except for the corner points that are coupled through $C1$ with the boundaries (N) and (S).

the result of $C^{-1}z$. In channel flow this is the case with the v velocity data at (S) and (N) and to the u data at the outflow (E). Define in the following z_0 as a grid function $z_0 = z_0(\xi_i, \eta_j)$. At all those boundary points except for the corners we simply perform $b_1^{-1}z_0$. At the corner positions we get the relation

$$C^{-1}z_0(\xi_1, \eta_j) \mid_{corner} = b_1^{-1}\left(-\sum_{i=2}^{N} b_i z_0(\xi_i, \eta_j)\right) \tag{104}$$

$C0$ and $C1$ come from the integral coupling of boundary points (in this case at the inflow). The sub-matrix $C1$ is the part of the integral that is computed on the (N) and (S) boundary of the channel. This operation can explicitly be performed in contrast to the inversion of $C0$.

In order to provide a complete description of $C0$ we need to come back to the boundary conditions (58). Before we introduce the discrete formulation a fourth order numerical integration is needed. We derive the formula from the Euler-MacLaurin relation

$$\int_{y_0}^{y_M} \tilde{f}(y)dy = \Delta y\left(\frac{\tilde{f}_0}{2} + \tilde{f}_1 + \tilde{f}_2 + \cdots + \tilde{f}_{M-1} + \frac{\tilde{f}_M}{2}\right) - \frac{\Delta y^2}{12}(\tilde{f}'_M - \tilde{f}'_0) + O(\Delta y^4). \tag{105}$$

Using one-sided second-order stencils to approximate the derivatives in the above equation the numerical fourth order integration is obtained

$$\begin{aligned} \mathcal{I}_I(\tilde{f}, \Delta y) \quad &:= \quad \Delta y\left(\frac{3}{8}\tilde{f}_0 + \frac{2}{3}\tilde{f}_1 + \frac{23}{24}\tilde{f}_2 + \tilde{f}_3\right. \\ +\cdots+ \quad \tilde{f}_{M-3} \quad &+\frac{23}{24}\tilde{f}_{M-2} + \frac{2}{3}\tilde{f}_{M-1} + \left.\frac{3}{8}\tilde{f}_M\right). \end{aligned} \tag{106}$$

For convenience we write the integration formula as the sum

$$\mathcal{I}_I(\Delta y, \tilde{f}) := \sum_{k=1}^{M} a_k \tilde{f}_k \, \Delta y \tag{107}$$

We can now find the discrete formulation of boundary conditions for u at the inflow

$$\sum_{i=1}^{N} b_i u_{i,k} - \frac{1}{L_y} \sum_{k=1}^{M} \left(a_k n_2(k) \Delta y \sum_{i=1}^{N} b_i u_{i,k} \right) = w_0(y_k) \tag{108}$$

Note, that (108) is in one point substituted by another condition in order to set the pressure integral. The contribution of the left hand side in equation (108) that couples points on the boundary is entering the matrix $C0$. Since this matrix is time *in*dependent it is computed just once and immediately factorized in LU factors. Each time we have to invert C in the solution algorithm the result of $C0^{-1} z_0$ is conveniently obtained by forward and back substitution with the help of the LU factors of $C0$. In order to compute the factorization $C0$ is set up according to the fourth order integration formula (106).

The 3D case differs just slightly from the described 2D case. The boundary points of the w component coincide with the physical boundary. In the Dirichlet case there appear just additional identity elements come into the diagonal. Observe that the integral coupling is just used for the spanwise wave number $\beta = 0$. In all other cases C is inverted explicitly.

4. Implementation

4.1. *Program structure*

The typical flow problems to be investigated are of the type shown in figure 7. This is internal flow, where the streamwise x coordinate and the wall normal y direction together describe the curvilinear 2D domain. The flow has a homogeneous, periodic spanwise dimension z that allows to expand the unknowns in Fourier modes.

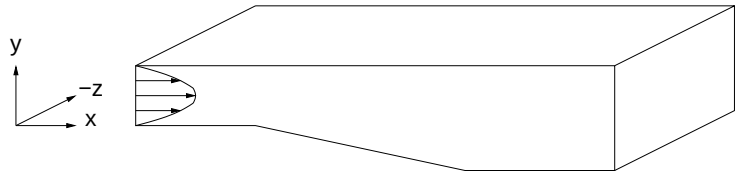

FIGURE 7. Flow configuration for internal 3D flow

Figure 8 shows a sketch of the general program structure. In each time integration step n we have to transform the solution of the preceding time levels back to physical space in order to conveniently evaluate the convolution sums in the nonlinear part of the Navier–Stokes equations.

At first the inverse Fast Fourier Transform (FFT) of the unknowns computed at time steps n and $n-1$ are performed. Additionally, the spanwise derivatives are needed to compute the nonlinear terms. Since the FFT has to access data in the spanwise direction we get the data in (xz) planes out of the main storage, then compute the FFT of the data and finally put back the (xz) plane to the main storage. This has to be repeated for a number of N_y (xz) planes.

In our hybrid discretization technique the only decoupled coordinate direction is the spanwise. Both the computation of the nonlinear and the linear terms requires the solution with the Padé operators in the curvilinear (xy) space. Note that in contrast to standard, explicit difference methods we couple all points in the (xy) domain through tridiagonal systems. Hence, we need to step through the 3D space by picking (xy) planes, computing the right hand side of the large linear system in time step n and putting it back to the main storage.

Before we are able to solve the implicit linear part of the equations, we have to call the forward FFT in order to transform the right hand side to Fourier space. The FFT requires again the treatment of (xz) planes. Finally we pick (xy) planes from the main storage in order to solve iteratively for the unknowns in time step n. Also here we have fully coupled (xy) planes.

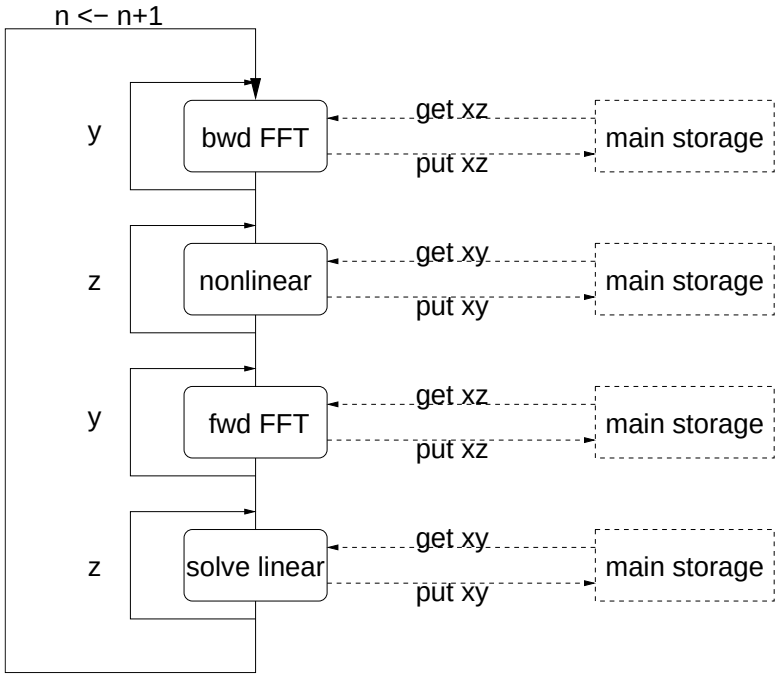

FIGURE 8. The general program structure in the time integration loop

The implementation is done in the Fortran90 language with a modular structure. Focus is put on achieving the degree of object orientedness permitted by Fortran90. To some extent we can generate black boxes, that hide information from the user by generic functions and derived data types. In the following the relevant modules and some of their more important derived data types and generic functions are presented. (INT) denotes integer data.

4.2. *Parameter module*

This module is used to read in and access input data except for those that are related to the solver routines. For simplicity the set of parameters is gathered in a single derived type PARAM which reduces the number of variables that have to be given to subroutines.

Float variables are represented in the self defined real kind (RK) format. If float values are specified, they must be tagged with the (RK) suffix.

Derived data types: PARAM

PARAM		
Contents	Type	Description
Nx, Ny, Zmax	INT	The number of grid points in x and y direction and the maximum spanwise wave number
x_min, x_max, y_min, y_max	RK	The geometrical bounds in ξ and η direction in computational space
dx, dy	RK	The grid spacings in ξ and η direction
k_max	INT	The maximum number of time steps
count	INT	Output to file after *count* time steps
dt	RK	Time step
Re	RK	The flow Reynolds number
gridnum	INT	Analytical or numerical conformal mapping
ϕ	RK	Turning angle of mean flow

Generic functions:

Generic function name	Description
`READ_PARAMS`	reads data from file *param.dat*

4.3. *Grid function module*

Self defined data types for grid functions are of great advantage in context with the staggered arrangement of variables in the computational grid. We define grid functions for three different types of grids (u-, v-, p- nodes) which have locally defined dimensions. In this way we achieve that index mismatches are widely eliminated. A convenient way of handling a full solution field in a (xy) plane is the use of the derived type GF_VECTOR which combines a U_MATRIX a V_MATRIX and two P_MATRIX types (one for w and one for p). (xz) planes are needed for computation of the FFTs and can be assigned to XZVP_MATRIX (v, w, p nodes) or XZU_MATRIX types (u nodes). For a specific variable we have the three-dimensional main storages U_MATRIX3D,

V_MATRIX3D and P_MATRIX3D (for w and p, respectively).

Note: Define for the following description of grid functions N as the number of *pressure* grid points in ξ direction and M as the number of *pressure* grid points in η direction, valid on a *non-periodic* grid. Nx, Ny are the *local* values in the specific grid function, dependent on type and periodicity.

Derived data types: U_MATRIX, V_MATRIX, P_MATRIX, GF_VECTOR, U_MATRIX3D, V_MATRIX3D, P_MATRIX3D, XZU_MATRIX, XZVP_MATRIX

U_MATRIX		
Contents	Type	Description
Nx, Ny	INT	Local number of grid points, non-periodic: Nx=N+1, Ny=M x-periodic: Nx=N-1, Ny=M, y-periodic: Nx=N+1, Ny=M-1
W	RK	2D array ($Nx \times Ny$) of grid function values

V_MATRIX		
Contents	Type	Description
Nx, Ny	INT	Local number of grid points, non-periodic Nx = N, Ny = M+1 x-periodic: Nx=N-1, Ny=M+1, y-periodic: Nx=N, Ny=M-1
W	RK	2D array ($Nx \times Ny$) of grid function values

P_MATRIX		
Contents	Type	Description
Nx, Ny	INT	Local number of grid points, non-periodic: Nx = N, Ny = M x-periodic: Nx=N-1, Ny=M, y-periodic: Nx=N, Ny=M-1
W	RK	2D array ($Nx \times Ny$) of grid function values

GF_VECTOR		
Contents	Type	Description
Nx, Ny	INT	Number of grid points, Nx = N, Ny = M
Dx, Dy	RK	Grid spacing in computational space
U	U_MATRIX	2D array on u-nodes
V	V_MATRIX	2D array on v-nodes
W	W_MATRIX	2D array on p-nodes
P	P_MATRIX	2D array on p-nodes

U_MATRIX3D		
Contents	Type	Description
Nx, Ny	INT	Number of grid points, Nx = N+1, Ny = M x-periodic: Nx=N-1, Ny=M, y-periodic: Nx=N+1, Ny=M-1
Nz	INT	Maximum spanwise wave number
W	RK	3D array of grid function values

V_MATRIX3D		
Contents	Type	Description
Nx, Ny	INT	Number of grid points, Nx = N, Ny = M+1 x-periodic: Nx=N-1, Ny=M+1, y-periodic: Nx=N, Ny=M-1
Nz	INT	Maximum spanwise wave number
W	RK	3D array of grid function values

P_MATRIX3D		
Contents	Type	Description
Nx, Ny	INT	Number of grid points, Nx = N, Ny = M x-periodic: Nx=N-1, Ny=M, y-periodic: Nx=N, Ny=M-1
Nz	INT	Maximum spanwise wave number
W	RK	3D array of grid function values

XZU_MATRIX		
Contents	Type	Description
Nx	INT	Number of grid points, Nx = N+1, x-periodic: Nx=N-1, y-periodic: Nx=N+1
Nz	INT	Maximum spanwise wave number
W	RK	2D array of grid function values

XZVP_MATRIX		
Contents	Type	Description
Nx	INT	Number of grid points, Nx = N, x-periodic: Nx=N-1, y-periodic: Nx=N
Nz	INT	Maximum spanwise wave number
W	RK	2D array of grid function values

Most of the generic functions listed here do work for all kinds of grid function types. The function `ADD_GF` adds e.g. grid functions of types U_MATRIX, V_MATRIX, P_MATRIX or GF_VECTOR dependent on which types are specified as arguments. The allocation of memory for derived types is performed in the generic `CREATE_GF` routines and memory can be released in `DESTROY_GF`. The listed get and put functions are used to access the main storage.

Generic functions:

Generic function name	Description
CREATE_GF	allocates memory for any type of grid function
DESTROY_GF	deallocates memory for any type of grid function
WRITE_GF	save grid function to ascii file
READ_GF	read grid function from ascii file
SAVE_GF	save grid function to binary file
LOAD_GF	read grid function from binary file
ASSIGN_GF	perform Y = a, Y = X or Y = a X
ADD_GF	compute Y = X + a Y
ADD_ACC_GF	compute Y = Y + a X
MULTIPLY_GF	compute Y = X * Z (point-wise multiplication)
NORM_GF	compute $l2$ norm of a grid function
DOTPROD_GF	compute dot product
INT_GF	interpolation between grid functions by splines or Lagrange
GET_XY	get xy plane from 3D main storage
PUT_XY	put xy plane in 3D main storage
GET_XZ	get xz plane from 3D main storage
PUT_XZ	put xz plane in 3D main storage

4.4. *Grid generation module*

The task of this module is to provide the grid data for a specific simulation. Since the geometries are described in two dimensions, it is not too memory consuming to save the data statically in a derived type GRID. GRID includes the physical node locations, the scale factors of the orthogonal transformation and the necessary derivatives of the scale factors on the staggered locations. For a specific geometry the grid data is initialized with the CREATE_GRID function. For grids generated with a numerical conformal mapping the staggered grid data is read from file and all necessary scale factors are computed with compact difference operators. For cases when analytical conformal mappings can be found the grids are constructed in the grid generation module according to a number of provided stretching functions in the ξ and η directions chosen at run time level from the parameter input file.

Derived data types: GRID

GRID		
Contents	Type	Description
UX, UY	U_MATRIX	Physical grid on u nodes
UN1, UN2	U_MATRIX	Scale factors on u nodes
UN1_ETA, UN2_XI, UN2_ETA	U_MATRIX	Derivatives of scale factors
VX, VY	V_MATRIX	Physical grid on v nodes
VN1, VN2	V_MATRIX	Scale factors on v nodes
VN1_ETA, VN2_XI, VN2_ETA	V_MATRIX	Derivatives of scale factors
PX, PY	P_MATRIX	Physical grid on p nodes
PN1, PN2	P_MATRIX	Scale factors on p nodes

Specific functions:

Function name	Description
`CREATE_GRID`	creates the grid data
`DESTROY_GRID`	releases grid data memory

4.5. *FFT module*

The FFT module does not include derived data types. Generic interfaces exist that compute the spanwise Fast Fourier Transform of a grid function in 3D space. Those are applied to either u, v or p grid functions and step (xz) plane wise through the 3D space. The three steps are: getting a (xz) plane, compute the forward or backward FFT and then to put back the (xz) plane to the main storage. This is what the generic functions `INVFFT` and `FWDFFT` are constructed for. For computation of the right hand side we need also the spanwise derivatives of grid functions. For this application a modified version of `INVFFT` is of use. `INVFFTD` adds a spectral differentiation step after the (xz) plane is picked from 3D space and before it the inverse Fourier transform is performed.

Note, that in physical space we separate the data in odd and even points (in the FFT direction), whereas in Fourier space, the data is explicitly decomposed in real and imaginary parts.

Generic functions:

Generic function name	Description
`INVFFT`	inverse Fast Fourier Transform of 3D field
`INVFFTD`	compute first spanwise derivative of 3D field and then perform inverse Fast Fourier Transform
`FWDFFT`	Fast Fourier Transform of 3D field

Specific functions:

Specific function name	Description
`VRFFTI`	initialisation of FFT routines
`VRFFTF`	forward FFT of 2D field in one dimension
`VRFFTB`	backward FFT of 2D field in one dimension

The routines `INVFFT`, `INVFFTD` and `FWDFFT` call the routines `VRFFTF` and VRFFTB in each xz plane for forward or inverse Fourier transformation, respectively. Those where used in e.g. Lundbladh *et al.* (1992). `VRFFTF` and `VRFFTB` can separately be used for e.g. transformation of boundary conditions.

4.6. *Compdiff module*

In this module the discretization coefficients and algorithms are provided for solution of the tridiagonal systems that are obtained from the implicit or compact formulation of space derivatives. Note that also the Padé form of interpolations ends in the same type of linear system. In general we have to solve a linear system

$$\mathcal{P}f' = \mathcal{Q}f, \tag{109}$$

where $\mathcal{P}$ is a tridiagonal matrix. Dependent on boundary conditions and periodicity the first and last row entries in $\mathcal{P}$ may differ. In the case of non-periodic boundary conditions and for a fully symmetric system matrix the derived type SYM_TRI_OP can be used. TRI_OP describes a general non-periodic, symmetric or non-symmetric tridiagonal matrix $\mathcal{P}$. The periodic counterpart to TRI_OP is the PER_TRI_OP type, where the matrix has entries in the lower left and the upper right corner. Dependent on the matrix type, different solution routines are chosen for the system (109). The right hand side matrix $\mathcal{Q}$ is saved in a simple sparse format and does in the same way depend on periodicity. DIFF_OP and PER_DIFF_OP are the operators that match TRI_OP and PER_TRI_OP. For computation of 6th order Padé interpolations, we use PER_DIFF_OP_6 and DIFF_OP_2.

Any kind of operator can be created or destroyed with the generic functions `CREATE_COMPDIFF` and `DESTROY_COMPDIFF`. Creating a high order operator includes computing the LU factorization of the matrix. The factors are saved in the diagonal and the sub- and super-diagonal for the general TRI_OP type and in the diagonal and sub-diagonal for the SYM_TRI_OP type. When periodic operators are used, the system matrix is divided in blocks according to figure 9.

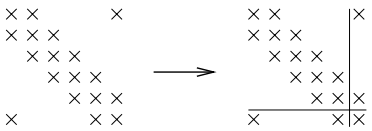

FIGURE 9. Periodic system matrix

The system can then be written in the form

$$\begin{pmatrix} A1 & A2 \\ A3 & A4 \end{pmatrix} \begin{pmatrix} x1 \\ x2 \end{pmatrix} = \begin{pmatrix} b1 \\ b2 \end{pmatrix}, \tag{110}$$

The algorithm rewrites the system as

$$\begin{aligned} x1 + A1^{-1}A2x2 &= A1^{-1}b1 \\ A3x1 + A4x2 &= b2 \end{aligned}$$

Solving the first equation for $x1$ in terms of $x2$ and putting it into the second equation leads to an explicit equation for $x2$.

Generally all the systems of type (109) are solved in two steps

$$b = \mathcal{Q}f \tag{111}$$

$$\text{solve:} \quad \mathcal{P}f' = b \tag{112}$$

Derived data types: SYM_TRI_OP, TRI_OP, PER_TRI_OP, DIFF_OP, DIFF_OP_2, PER_DIFF_OP, PER_DIFF_OP_6

SYM_TRI_OP		
Contents	Type	Description
TYPE	CHAR	Operator P in symmetric Padé system $Pf' = Qf$ dependent on order of derivative and staggered/nonstagg.
N	INT	Size of the tridiagonal linear system
D, E	RK	1D arrays, D: diagonal, E: super-diagonal of tridiag. system

TRI_OP		
Contents	Type	Description
TYPE	CHAR	Operator P in Padé system $Pf' = Qf$ dependent on order of derivative and staggered/nonstagg.
N	INT	Size of the tridiagonal linear system
DL, D, DU, DU2	RK	1D arrays, DL: sub-diagonal D: diagonal, DU: super-diag., DU2: second super-diagonal of tridiagonal system

PER_TRI_OP		
Contents	Type	Description
TYPE	CHAR	Operator P in periodic Padé system $Pf' = Qf$ dependent on order of derivative and staggered/nonstagg.
N	INT	Size of the periodic tridiagonal linear system
A1, A2, A3	RK	1D arrays, A1: sub-diagonal A2: diagonal, A3: super-diag.
WORK2, WORK3	RK	1D arrays, work space for split of system matrix
WORK4	RK	Work space, one matrix element

DIFF_OP		
Contents	Type	Description
A	RK	Operator Q in non-periodic Padé system $Pf' = Qf$ stored in 5×3 array
I0, I1, I	INT	Help 1D arrays of length 3

DIFF_OP_2		
Contents	Type	Description
A	RK	Operator E_2 in non-periodic Padé interpolation $E_1 f_i = E_2 f$ stored in 5×6 array
I0, I1, I	INT	Help 1D arrays of length 5

PER_DIFF_OP		
Contents	Type	Description
A	RK	Operator Q in periodic Padé system $Pf' = Qf$ stored in 1D array of length 3
I0, I1, I	INT	Help values

PER_DIFF_OP_6		
Contents	Type	Description
A	RK	Operator E_2 in periodic Padé interpolation $E_1 f_i = E_2 f$ stored in 1D array of length 5
I0, I1, I	INT	Help values

Both of the two steps (111, 112) are performed by one single generic routine `APPLY_COMPDIFF`. Step 1 is a sparse matrix vector multiplication with the matrix $\mathcal{Q}$ stored in arrays with either 3, 5 or 1 rows dependent on which type is needed. 1 row of coefficients is sufficient in the case of periodic operators. 3 rows are used in all $4th$ order operators. 5 rows are needed in the $6th$ order interpolations. Step 2 performs the the solution of the tridiagonal system by application of the LU factors of the system matrix.

Generic functions:

Generic function name	Description
`CREATE_COMPDIFF`	creates any derived operator type
`DESTROY_COMPDIFF`	releases memory of any derived operator type
`APPLY_COMPDIFF`	two specific types of functions are involved `MULTIPLY`: sparse matrix-vector multiplication $b = Bf$ in Padé system $Af' = Bf$ `SOLVE`: solution of the system $Af' = b$ by forward/bwd substitution using the LU factors

4.7. *PDE module*

The PDE module stands for discretization of the partial differential equations, that describe the physics. Here, we define two essential data types. M_MATRIX contains the compact difference operators that are necessary for computation of the linear part of (70), i.e. the matrix-vector multiplication $M\,x$, where x is an arbitrary solution vector.

The difference operators needed for construction of the right hand side b in (70) are available in the N_OPERATOR. The types of the specific difference operators in M_MATRIX and N_OPERATOR change when periodic directions are defined as listed in the tables.

Creator and destructor routines exist for M_MATRIX and N_OPERATOR. The M_MATRIX type is used in the routines `COMPUTE_D0_D1` for computation of the divergence in curvilinear 2D space, in `COMPUTE_G`, that determines the gradient of the pressure, in `COMPUTE_A0_A1_G` for computation of the Laplace term and the pressure gradient and in the generic function `COMPUTE_A0_A1` which computes the Laplace term for either $(u, v)^T$ or $(w)^T$.

The N_OPERATOR is applied in the routine `COMPUTE_NLIN`, which is an interface for computation of the right hand side by advancing (xy) plane-wise through the 3D domain. `COMPUTE_NLIN` calls the routine `COMPUTE_RHS` which computes the right hand side in a particular (xy) plane. `COMPUTE_RHS` itself is generic and depends on the choice of backward difference scheme for time integration (BDF-1 or BDF-2).

Furthermore, we have the generic interpolation routine `INT_PADE6`. It uses 6th order accurate Padé schemes to interpolate between any grid locations. For interpolation from u to v locations and vice versa two Padé solutions are called, since the interpolation works via p locations.

Derived data types: M_MATRIX, N_OPERATOR

M_MATRIX		
Contents	Type	Description
AN	INT	Number of coefficients for extrapolation of boundary data
A	RK	1D array, size AN. High order coefficients for extrapolation of boundary data
UN	INT	Number of coefficients for Neumann boundary condition, staggered
BC_OUT_U	RK	1D array, size UN, coefficients for Neumann boundary condition, staggered
VN	INT	Number of coefficients for Neumann boundary condition, collocated
BC_OUT_V	RK	1D array, size VN, coefficients for Neumann boundary condition, colloc.

non-periodic		
PUx, PVx	TRI_OP	1st derivative, collocated, u/v nodes, x dir.
PUy, PVy	TRI_OP	1st derivative, collocated, u/v nodes, y dir.
PPx, PPy	TRI_OP	1st derivative, collocated, w/p nodes, x/y dir.
RUx, RVx	TRI_OP	2nd derivative, collocated, u/v nodes, x dir.
RUy, RVy	TRI_OP	2nd derivative, collocated, u/v nodes, y dir.
GUx	TRI_OP	1st derivative, staggered, p nodes, x dir.
GVy	TRI_OP	1st derivative, staggered, p nodes, y dir.
GPx	TRI_OP	1st derivative, staggered, u nodes, x dir.
GPy	TRI_OP	1st derivative, staggered, v nodes, y dir.
UF0, VF0	TRI_OP	Interpolation, p/w nodes, from u/v nodes
UE0, VE0	TRI_OP	Interpolation, u/v nodes, from p/w nodes
QUx, QVx	DIFF_OP	1st derivative, collocated, u/v nodes, x dir.
QUy, QVy	DIFF_OP	1st derivative, collocated, u/v nodes, y dir.
QPx, QPy	DIFF_OP	1st derivative, collocated, (p/w) nodes, x/y dir.
SUx, SVx	DIFF_OP	2nd derivative, collocated, u/v nodes, x dir.
SUy, SVy	DIFF_OP	2nd derivative, collocated, u/v nodes, y dir.
HUx	DIFF_OP	1st derivative, staggered, u → p/w nodes, x dir.
HVy	DIFF_OP	1st derivative, staggered, v → p/w nodes, y dir.
HPx	DIFF_OP	1st derivative, staggered, p → u nodes, x dir.
HPy	DIFF_OP	1st derivative, staggered, p → v nodes, y dir.
UF1, VF1	DIFF_OP	Interpolation, u/v nodes, to p/w nodes
UE1, VE1	DIFF_OP_2	Interpolation, p/w nodes, to u/v nodes

ξ-periodic		
PUx, PVx	PER_TRI_OP	1st derivative, collocated, u/v nodes, x dir.
PUy, PVy	TRI_OP	1st derivative, collocated, u/v nodes, y dir.
PPx	PER_TRI_OP	1st derivative, collocated, w/p nodes, x dir.
PPy	TRI_OP	1st derivative, collocated, w/p nodes, y dir.
RUx, RVx	PER_TRI_OP	2nd derivative, collocated, u/v nodes, x dir.
RUy, RVy	TRI_OP	2nd derivative, collocated, u/v nodes, y dir.
GUx	PER_TRI_OP	1st derivative, staggered, p nodes, x dir.
GVy	TRI_OP	1st derivative, staggered, p nodes, y dir.
GPx	PER_TRI_OP	1st derivative, staggered, u nodes, x dir.
GPy	TRI_OP	1st derivative, staggered, v nodes, y dir.
UF0	PER_TRI_OP	Interpolation, p/w nodes, from u nodes
VF0	TRI_OP	Interpolation, p/w nodes, from v nodes
UE0	PER_TRI_OP	Interpolation, u nodes, from p/w nodes
VE0	TRI_OP	Interpolation, v nodes, from p/w nodes
QUx, QVx	PER_DIFF_OP	1st derivative, collocated, u/v nodes, x dir.
QUy, QVy	DIFF_OP	1st derivative, collocated, u/v nodes, y dir.
QPx	PER_DIFF_OP	1st derivative, collocated, p/w nodes, x dir.
QPy	DIFF_OP	1st derivative, collocated, p/w nodes, y dir.
SUx, SVx	PER_DIFF_OP	2nd derivative, collocated, u/v nodes, x dir.
SUy, SVy	DIFF_OP	2nd derivative, collocated, u/v nodes, y dir.
HUx	PER_DIFF_OP	1st derivative, staggered, u $\rightarrow$ p/w nodes, x dir.
HVy	DIFF_OP	1st derivative, staggered, v $\rightarrow$ p/w nodes, y dir.
HPx	PER_DIFF_OP	1st derivative, staggered, p $\rightarrow$ u nodes, x dir.
HPy	DIFF_OP	1st derivative, staggered, p $\rightarrow$ v nodes, y dir.
UF1	PER_DIFF_OP_6	Interpolation, u nodes, to p/w nodes
VF1	DIFF_OP	Interpolation, v nodes, to p/w nodes
UE1	PER_DIFF_OP_6	Interpolation, p/w nodes, to u nodes
VE1	DIFF_OP_2	Interpolation, p/w nodes, to v nodes
periodic η operators definied in a similar manner		

N_OPERATOR		
non-periodic		
Contents	Type	Description
PUx, PVx	TRI_OP	1st derivative, collocated, u/v nodes, x dir.
PUy, PVy	TRI_OP	1st derivative, collocated, u/v nodes, y dir.
PPx, PPy	TRI_OP	1st derivative, collocated, w/p nodes, x/y dir.
UF0, VF0	TRI_OP	Interpolation, p/w nodes, from u/v nodes
UE0, VE0	TRI_OP	Interpolation, u/v nodes, from p/w nodes
QUx, QVx	DIFF_OP	1st derivative, collocated, u/v nodes, x dir.
QUy, QVy	DIFF_OP	1st derivative, collocated, u/v nodes, y dir.
QPx, QPy	DIFF_OP	1st derivative, collocated, (p/w) nodes, x/y dir.
UF1, VF1	DIFF_OP	Interpolation, u/v nodes, to p/w nodes
UE1, VE1	DIFF_OP_2	Interpolation, p/w nodes, to u/v nodes
ξ and η periodic operators defined in similar manner as in M_MATRIX type		

Generic functions:

Generic function name	Description
INT_PADE6	6th order Padé interpolation between any staggered grid locations
COMPUTE_RHS	compute rhs of linear system (BDF-2 or BDF-1)
COMPUTE_A0_A1	compute the Laplace term including the diagonal term from the time derivative, works for (u, v) and (w) part of solution vector

Specific functions:

Specific function name	Description
CREATE_M_MATRIX	create high order operators in system matrix
DESTROY_M_MATRIX	release all high order operators in system matrix
CREATE_N_OPERATOR	create all necessary high order operators for rhs
DESTROY_N_OPERATOR	release all high order operators for right hand side
NUMINT	numerical integration routine
COMPUTE_D0_D1	compute the divergence of a velocity field (u,v)
COMPUTE_G	compute the pressure gradient
COMPUTE_A0_A1_G	compute the Laplace term including the diagonal term and the pressure gradient (u,v)
COMPUTE_NLIN	compute the rhs of the 3D system of equations: pick xy plane, compute rhs, put back xy plane
RHS_INT	replaces the boundary condition in the right hand side by the integral formulation for the $\beta = 0$ case
SET_BC	set boundary conditions in physical space
VEL_INIT	create initial velocity field

4.8. *Solve module*

This module contains the routines necessary to iteratively solve the linear system (65). For convenience, we define the type PRECOND that contains the time independent data used in the solvers.

In PRECOND we have the convergence tolerances and iteration limits for the outer iteration, the solution of the A systems and the DG systems. The criterion controls both the real and the imaginary solvers.

Further on, the data for the incomplete LU factorization of the DG system matrix is provided. The factorization of the DG matrix can be based on a 2nd or 4th order discretization of the Laplace operator and is saved as its LU factors. The approximation of the DG matrix depends on periodicity in ξ or η direction and of the spanwise wavenumber. The explicit setup of DG is performed in modified sparse row format (MSR), while the LU factors are returned in compressed sparse row format (CSR). CSR is also the format applied in the algorithm computing the forward backward substitution using the LU factorization. In the PRECOND type we also choose between fixed point iteration and Bi-CGSTAB solution routine for the A systems. LU factors of the A system matrices are based on either γI (cf. (88)) or explicit 2nd order approximation of (72) in a similar manner as for the DG matrix.

We also save the LU factorization of the C matrix (see 3.5.6), which is constant in time. C holds the coefficients of the boundary extrapolation and boundary conditions that act on the outer boundary locations. The PRECOND type is created and released by the routines `CREATE_PRECOND` and `DESTROY_PRECOND`.

The routines used in the solution algorithm are `COMPUTE_LIN` for (xy) plane wise solution of the linear system. `COMPUTE_LIN` calls `SOLVE3D` in a particular (xy) plane and its corresponding wavenumber. `SOLVE3D` itself performs the outer iteration algorithm, which calls the generic solver routine `SOLVE_A_SYS`. It is applied to the solution of both the (u, v) and the (w) velocity prediction. The second system to solve is the pressure-Poisson system (102) for which the routine `BICGSTAB3` is used. In the Bi-CGSTAB algorithm we need to solve systems of the type $DGx = b$ with the help of the incomplete LU factors of DG. The corresponding routine is `DG_ILU_SOLVE`.

Since the solution algorithm uses the form reduced by the boundary conditions, we need routines for inversion of C and application of A_2. Here, we provide routines for the matrix vector products $C^{-1}x$ and A_2x named `INV_C` and `COMPUTE_A2`.

Derived data types: PRECOND

PRECOND		
Contents	Type	Description
MAXIT	INT	Outer iteration: maximum number of iterations
TOL	RK	Outer iteration: convergence tolerance
TOL_TYPE	INT	Outer iteration: absolute/relative error
GAMMA	RK	Preconditioning factor (cf. (88))
A_MAXIT	INT	A systems: maximum number of iterations
A_TOL	RK	A systems: convergence tolerance
A_TOL_TYPE	INT	A systems: absolute/relative error
A_SOL_TYPE	INT	A systems: simple iteration or Bi-CGSTAB
A_ILU_TYPE	INT	ILU for A: choice of factorization algorithm
A_EPS	RK	ILU for A: drop threshold for 2nd order ILU
A_TYPE	INT	ILU for A: swith on and off 2nd order Laplace appr.
DG_MAXIT	INT	DG systems: maximum number of iterations
DG_TOL	RK	DG systems: convergence tolerance
DG_TOL_TYPE	INT	DG systems: absolute/relative error
DG_TYPE	INT	ILU for DG: switch 2nd or 4th order Laplace appr.
DG_NNZ	INT	ILU for DG: number of non-zero elements in DG appr.
DG_EPS	RK	ILU for DG: drop threshold in 4th order ILU
ILU_TYPE	INT	ILU for DG: choice of factorization algorithm
ALU, JLU	RK	ILU: matrix stored in MSR format
AU_ALU, AU_JLU	RK	ILU for Au: matrix stored in MSR format
AV_ALV, AV_JLV	RK	ILU for Av: matrix stored in MSR format
AW_ALW, AW_JLW	RK	ILU for Aw: matrix stored in MSR format
NNZ	INT	ILU: length of ALU
JU	INT	ILU: array holding the pointers to rows of U
IPERM	INT	ILU: permutation array
DROPTOL	RK	ILU: threshold for dropping small terms
LFIL	INT	ILU: fill in parameter for L and U
ALPHA	RK	ILU: diagonal compensation factor
PERMTOL	RK	ILU: permutation tolerance (0: never permute)
NC	INT	Size of boundary value matrix C
CLU	RK	LU factorization of C stored in 2D array
CPIV	INT	Pivoting array for LU factorzation of C

Generic functions:

Generic function name	Description
`SOLVE_B`	compute update in fixed point iteration (A system)
`CREATE_PRECOND`	create the preconditioner
`DESTROY_PRECOND`	release the preconditioner
`SOLVE_A_SYS`	solve A system with fixed point iteration
`A_ILU_SOLVE`	solve sysmtems $Ay = b$ in Bi-CGStab with ILU factors of A
`INV_C`	compute $C^{-1}x$
`NEG_BC`	change sign of boundary data
`COMPUTE_A2`	compute $A_2 x$

Specific functions:

Specific function name	Description
COMPUTE_LIN	solve entire linear system by stepping xy plane wise through the 3D space
SOLVE3D	solve linear system with outer iteration scheme in xy plane
BICGSTAB3	solve pressure Poisson system with Bi-CGStab
DG_ILU_SOLVE	solve system $DGy = b$ in Bi-CGStab with ILU factors of DG
COPY_BC	copy boundary data from a GF_VECTOR to another GF_VECTOR
ADD_UV_INTERIOR	the operation $Z = a\,X + b\,Y$ is performed for the interior velocity data (u,v) in a GF_VECTOR
ADD_W_INTERIOR	the operation $Z = a\,X + b\,Y$ is performed for the interior velocity data (w) in a P_MATRIX
COMPUTE_E	compute the integral pressure condition that replaces a velocity condition in one point

5. Operation

5.1. *Compilation*

Due to the modular structure of the implementation, the correct order in which the modules are compiled is of importance. A top level makefile exists which includes a preprocessor file (*'preprocessing.pde'*) and a makefile definitions file (*'makefile.defs'*). The definition file varies with the platform on which the code is compiled. A number of those is provided in the package. Especially the flags for compilation of the modules may vary on different systems.

The modules are to be compiled in the sequence

1. parameter_module
2. gf_module
3. fft_module
4. compdiff_module
5. grid_module
6. pde_module
7. solve_module
8. frhs_module
9. main

The module frhs_module is for inclusion of right hand side forcing terms and is not described here.

In Fortran90 the most convenient way to specify the precision of float parameters is in the code, which makes an explicit specification of double precision in a compiler flag unnecessary. Floats are in the implementation throughout declared with the Fortran90 KIND command. Note that all assigned values for parameters declared with KIND need to be marked with the corresponding KIND suffix. Otherwise the variable might be reset to standard single precision.

5.1.1. *Preprocessing*

In order to choose different source code portions the Fortran preprocessor is used. When changes in the preprocessor are carried out, the code has to be recompiled entirely. The preprocessor specifications are therefore very essential and have in general not to be changed under a set of simulations with a specific problem.

The parameters in the preprocessor are COMP_DIFF_44 for selection of compact operators of either 4th order in the complete domain or of 4th order accuracy in the interior and 3rd order at the boundaries, BC_TYPE for determining the inflow/outflow conditions and PER_TYPE for specification of periodicity.

Preprocessor flag	Value	Description
COMP_DIFF_44	0	interior: 4th order accurate, boundary: 3rd order
	1	interior: 4th order accurate, boundary: 4th order
BC_TYPE	0	inflow: Dirichlet, outflow: Dirichlet
	1	inflow: Dirichlet, outflow: Neumann
	2	inflow: Dirichlet, outflow: convective
PER_TYPE	0	ξ: non-periodic, η: non-periodic
	1	ξ: periodic, η: non-periodic
	2	ξ: non-periodic, η: periodic

5.2. *Complex numbers*

The main part of the code performs operations in Fourier space. Hence, it is necessary to store both real and imaginary parts of the transformed quantities. For performance reasons it is not advisable to use the complex data type provided by Fortran. Instead, complex numbers are stored in separate arrays for the real and imaginary parts.

Recall that the system matrices appearing in the solution routines are real. Therefore, we have duplicative calls of the A and DG systems to determine the real and the imaginary solution.

5.3. *To set boundary conditions*

Boundary conditions are set in physical space. Together with the right hand side they are transformed to Fourier space. There, the integrals are computed for the zero wavenumber as described in section 2.9. The integral should always be applied to the velocity component which is normal to the boundary. If one periodic direction is used the integral condition is set on a boundary normal to the periodic direction, see for example Stålberg *et al.* (2004).

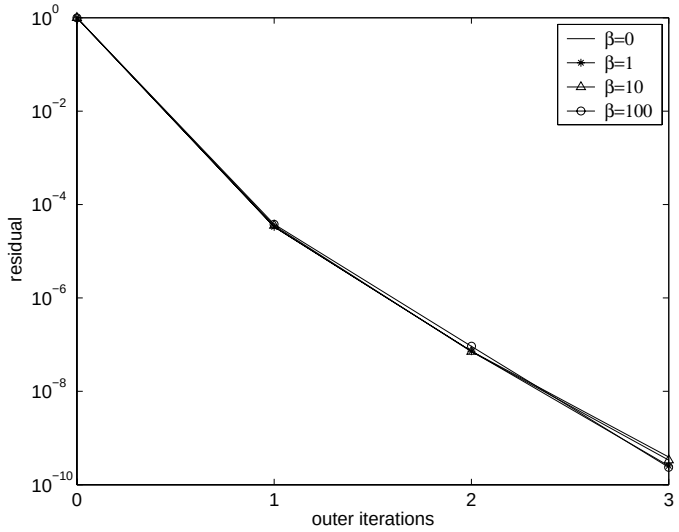

FIGURE 10. Convergence in outer iterations dependent on wavenumber, problem size 41×41

5.4. *Execution*

5.4.1. *Execution time*

The major time consumption per time step is due to the solution of the pressure-Poisson system. In addition, the work load is not independent of the wavenumber. It turns out that the zero wavenumber is the one demanding most work. All wavenumbers $\neq 0$ give a contribution to the diagonal of the DG matrix.

Figure 10 shows the result from a solver test with a random right hand side at $R = 500$ and problem size $N \times M = 41 \times 41$. The outer iterations converge uniformly.

The A systems need 5 iterations with fixed point iteration scheme at all wavenumbers and independently of the outer iteration. Figure 11 gives a clue about how much more work has to be done for the zero wavenumber.

The data shown in figure 11 are taken from the real solvers. The imaginary parts converge similarly.

5.4.2. *Incomplete LU factorization*

The DG system is solved with the Bi-CGSTAB routine, see Greenbaum (1997) and van der Vorst (1992), preconditioned with incomplete LU factors (ILU) based on an explicit second 2nd order accurate representation of the DG matrix.
For the A systems the user has to choose between fixed point iteration and an ILU preconditioned Bi-CGSTAB routine. The preconditioner for the fixed point iteration scheme is γI, see (88).

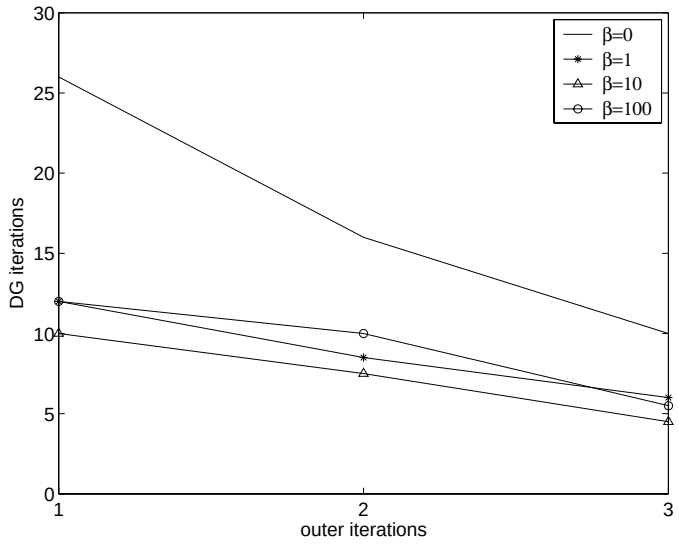

FIGURE 11. Convergence in pressure Poisson system dependent on wavenumber, problem size 41×41

The DG and A systems matrices depends on the square of the spanwise wavenumbers which enters the diagonal as an additive constant multiplied with a factor containing the time step. Hence, the systems becomes more diagonally dominant with increasing wavenumber.
The main parameters for ILU preconditioning of the A (99), (100) and DG systems (102) are

i the number of ILU decompositions
ii the drop tolerance for small terms.

For optimal solutions ILU decompositions should be provided for each wavenumber β. If this is possible depends primary on the spanwise resolution and on how much memory may be used. An alternative is to share ILU factorizations for some wavenumbers. For the smaller wavenumbers, the exact factorizations should preferably be used. Elements smaller than the drop tolerance are neglected in the ILU factors of the system matrices. A small tolerance results in a good approximation and will sustain fast convergence but at the cost of increased memory consumption.

5.4.3. *Iterative solvers for the A systems*

The time step at a certain simulation should be chosen so that the time integration is accurate enough, stable and quick convergence of iterative solvers is achieved. At certain conditions the maximum time step can be limited by the fixed point iteration scheme, see section 3.5.2. The Bi-CGSTAB routine is

less restrictive in terms of the maximum time step and allows the use of time steps close to the theoretical stability limit if this is considered to be accurate enough.

In order to approximate the maximum time step in a general physical domain with $\Delta x \neq \Delta y$ such that the time integration is stable the curves in figure 2.4 are approximated by

$$f(\theta) \quad = \quad \begin{cases} \frac{3}{2}\left(\frac{\theta}{2}\right)^{1/2}, & \theta < 2 \\ \frac{5}{6} + \frac{\theta}{3}, & \theta \geq 2\,. \end{cases} \tag{113}$$

However, the curves in figure 2.4 are computed on an equidistant grid and with

$$\theta = \frac{\Delta t}{R\left(\max(\Delta x, \Delta y)\right)^2} \tag{114}$$

the curves are favourable, i.e. the stability region is greater than in figure 2.4. According to the Fourier analysis in Kress & Lötstedt (2004) the time integration is stable if

$$\left(\frac{u\Delta t}{\Delta x}\right)^2 + \left(\frac{v\Delta t}{\Delta y}\right)^2 \leq f^2(\theta). \tag{115}$$

The maximum time step in each cell is then

$$(\Delta t_{\max})_{ij} \quad = \quad \begin{cases} \dfrac{9}{8\,R\,(\,\max[\,(\Delta x)_{ij}, (\Delta y)_{ij}\,]\,)^2\,c_{ij}^2}\,, & \theta < 2 \\ \dfrac{5}{6\,c_{ij} - \dfrac{2}{R\,(\,\max[\,(\Delta x)_{ij}, (\Delta y)_{ij}\,]\,)^2}}\,, & \theta \geq 2\,. \end{cases} \tag{116}$$

where

$$c_{ij}^2 = \left(\frac{u_{ij}}{(\Delta x)_{ij}}\right)^2 + \left(\frac{v_{ij}}{(\Delta y)_{ij}}\right)^2. \tag{117}$$

If the denominator in (116) for $\theta \geq 2$ is negative there is no bound on $(\Delta t_{\max})_{ij}$. The time integration is stable if $\Delta t < \min[(\Delta t_{\max})_{ij}]$. As a comparative study using the different iterative solvers for the A systems we consider straight and constricting channel flow, see figure 12.

For the straight channel the in- and outflow boundary conditions are Poiseuille profiles rotated $\pi/4$ around the y-axis in order to achieve nonzero spanwise velocity. The $N_x \times N_y \times N_z = 81 \times 41 \times 2$ grid is equidistant in the streamwise direction x with spacing $\Delta x = 6.25 \cdot 10^{-2}$ and in the wall normal direction a hyperbolic stretching function is applied with minimum spacing $\min[(\Delta y)_{ij}] = 1.70 \cdot 10^{-3}$.

A constricting channel geometry is used as an example of a curvilinear grid, see De A Mancera & Hunt (1997) and Brüger *et al.* (2002). The grid size is $N_x \times N_y \times N_z = 81 \times 81 \times 2$ and the minimum mesh spacing in the x and y directions are $\min[(\Delta x)_{ij}] = 3.69 \cdot 10^{-1}$ and $\min[(\Delta y)_{ij}] = 8.70 \cdot 10^{-3}$. In order to study the impact of different boundary conditions two cases are studied.

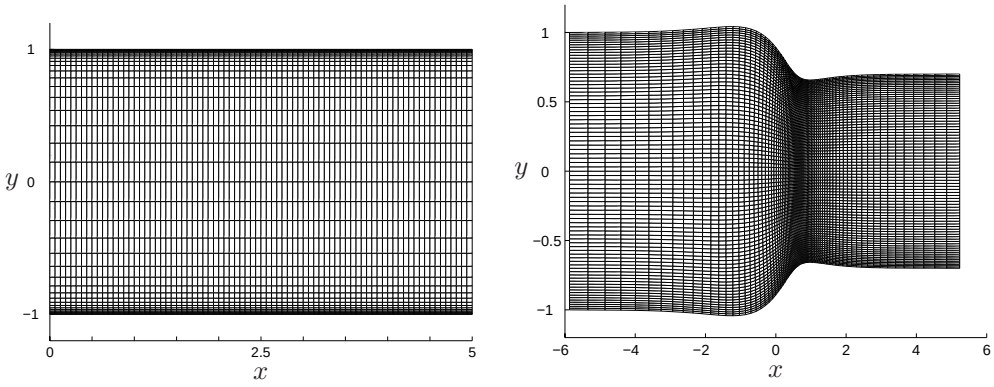

FIGURE 12. Geometries used for comparative study of different iterative solvers for the A systems.

The first uses Dirichlet Poiseuille profiles on both in- and outflow boundaries rotated $\pi/4$ around the y-axis. The second case has a Poiseuille profile with zero w component as inflow condition and a Neumann outflow condition, i.e. the streamwise derivatives of the velocity components are set to 0.

The Reynolds number is $R = 100$ in all numerical experiments and a simulation is considered as stable if the solvers converge to the specified tolerance under 2000 time steps without visible instabilities in the solution. The convergence tolerance is 10^{-9} in both inner (A and DG) and outer iterations. The initial velocity and pressure fields are zero in all numerical experiments. In addition to the maximum time step $\Delta t_{\max}$ we measure the number of iterations in the A systems (99) and (100) necessary to converge to the desired tolerance. The computational cost is expressed in wall clock time (wct) for the simulations normalized by wall clock time for the fixed point iteration alternative. The convergence of the DG system is here unaffected by the choice of iterative solver for the A systems. Furthermore, we use $\mathrm{wct}/\Delta t_{\max}$ as efficiency measure, where $\mathrm{wct}/\Delta t_{\max}$ for the fixed point iteration alternative is used as normalization.

The fixed point iteration scheme is preconditioned with γI, see (88). As preconditioning for the Bi-CGSTAB alternatives for the A systems we use two different alternatives, A and B. Alternative A is based on ILU of the explicit 2nd order representation of $3/2I + \Delta t\beta^2/RI$ and alternative B is for $3/2I + \Delta t\beta^2/RI - \Delta t/RL$. Table 1 shows the results for the straight channel case. The maximum time step based on the approximation (116) for the steady state solution is $\min[(\Delta t)_{ij}] = 1.9 \cdot 10^{-2}$. The fixed point iteration scheme imposes a severe limitation on $\Delta t_{\max}$ but the Bi-CGSTAB allow for time step close to the stability criterion (116). B is the most efficient alternative for the straight channel as a consequence of a better preconditioning. However, by

	Iterative solver		
	Fixed point iteration	Bi-CGSTAB A	Bi-CGSTAB B
$\Delta t_{\max}$	$9.0 \cdot 10^{-5}$	$2.5 \cdot 10^{-2}$	$2.5 \cdot 10^{-2}$
$\hat{A}\mathbf{y_1} = \mathbf{r_1}$ iterations	44	60	37
$\hat{\hat{A}}y_w = r_w$ iterations	5	65	27
Relative wct	1.0	1.4	1.2
Relative (wct/$\Delta t_{\max}$)	1.0	$4.9 \cdot 10^{-3}$	$4.5 \cdot 10^{-3}$

TABLE 1. Study of maximum time step and timing characteristics using different iterative solvers for the A systems. Straight channel with rotated Poiseuille profiles as in- and outflow boundary conditions.

	Iterative solver		
	Fixed point iteration	Bi-CGSTAB A	Bi-CGSTAB B
$\Delta t_{\max}$	$1.7 \cdot 10^{-3}$	$1.5 \cdot 10^{-2}$	$1.5 \cdot 10^{-2}$
$\hat{A}\mathbf{y_1} = \mathbf{r_1}$ iterations	43	14	29
$\hat{\hat{A}}y_w = r_w$ iterations	14	11	20
Relative wct	1.0	$9.3 \cdot 10^{-1}$	1.1
Relative (wct/$\Delta t_{\max}$)	1.0	$1.1 \cdot 10^{-1}$	$1.2 \cdot 10^{-1}$

TABLE 2. Study of maximum time step and timing characteristics using different iterative solvers for the A systems. Constricting channel with rotated Poiseuille profiles as in- and outflow boundary conditions.

slightly decreasing the time step in alternative A the number of iterations in the A systems will decrease resulting in lower wct.

For constricting channel we do not get the same increase in maximum time step using the Bi-CGSTAB routine as for the straight channel since the stability criterion (116) is closer to the stability limit for the fixed point iteration scheme, see table 2 and 3. The maximum time step based on (116) is $\Delta t_{\max} = 1.1 \cdot 10^{-2}$ for the case with tilted Poiseuille profiles as in- and outflow boundary conditions and $\Delta t_{\max} = 7.2 \cdot 10^{-3}$ for the Neumann outflow boundary condition case. In this curvilinear geometry the preconditioning applied in A result in the most efficient alternative.

	Iterative solver		
	Fixed point iteration	Bi-CGSTAB A	Bi-CGSTAB B
$\Delta t_{\max}$	$1.7 \cdot 10^{-3}$	$1.0 \cdot 10^{-2}$	$1.0 \cdot 10^{-2}$
$\hat{A}\mathbf{y_1} = \mathbf{r_1}$ iterations	46	11	24
$\hat{\hat{A}} y_w = r_w$ iterations	0	0	0
Relative wct	1.0	$9.2 \cdot 10^{-1}$	1.0
Relative (wct/$\Delta t_{\max}$)	1.0	$1.6 \cdot 10^{-1}$	$1.7 \cdot 10^{-1}$

TABLE 3. Study of maximum time step and timing characteristics using different iterative solvers for the A systems. Constricting channel with Poiseuille profile as inflow boundary condition and Neumann condition on outflow boundary.

References

Bhaganagar, K., Rempfer, D. & Lumley, J. 2002 Direct numerical simulation of spatial transition to turbulence using fourth-order vertical velocity second-order vertical vorticity formulation. *J. Comput. Phys.* **180**, 200–228.

Brown, D. L., Cortez, R. & Minion, M. L. 2001 Accurate projection methods for the incompressible Navier-Stokes equations. *J. Comput. Phys.* **168**, 464–499.

Brüger, A. 2002 Higher order methods suitable for direct numerical simulation of flows in complex geometries, Licentiate Thesis. *Tech. Rep.*. Dept. of Mechanics, KTH, Stockholm, Sweden.

Brüger, A. 2004 A hybrid high order method for simulation of turbulent flow in complex geometries. PhD Thesis, Dept. of Mechanics, KTH, Stockholm, Sweden.

Brüger, A., Gustafsson, B., Lötstedt, P. & Nilsson, J. 2005 High order accurate solution of the incompressible Navier-Stokes equations. *J. Comput. Phys.* **203**, 49–71.

Brüger, A., Gustafsson, B., Lötstedt, P. & Nilsson, J. 2004 Splitting methods for high order solution of the incompressible Navier-Stokes equations in 3D. *To appear in Int. J. Numer. Meth. Fluids* .

Brüger, A., Nilsson, J. & Kress, W. 2002 A compact higher order finite difference method for the incompressible Navier-Stokes equations. *J. Sci. Comput.* **17**, 551–560.

Canuto, C., Hussaini, M. Y., Quarteroni, A. & Zang, T. A. 1988 *Spectral Methods in Fluid Dynamics*. New York: Springer.

De A Mancera, P. F. & Hunt, R. 1997 Fourth order method for solving the Navier-Stokes equations in a constricting channel. *Int. J. Numer. Meth. Fluids* **25**, 1119–1135.

Driscoll, T. A. & Trefethen, L. N. 2002 *Schwartz-Christoffel Mapping*. Cambridge Univ. Press.

Ferziger, J. 1987 Simulation of incompressible fluid flows. *J. Comput. Phys.* **69**, 1–48.

Fornberg, B. & Ghrist, M. 1999 Spatial finite difference approximations for wave-type equations. *SIAM J. Numer. Anal.* **37**, 105–130.

Göran, A. 2001 Preconditioned iterative solution of the incompressible Navier-Stokes equations. Master's thesis, Dept. of Scientific Computing, Uppsala University, Uppsala.

Greenbaum, A. 1997 *Iterative Methods for Solving Linear Systems*. Philadelphia: SIAM.

Gullbrand, J. 2000 An evaluation of a conservative fourth order dns code in turbulent channel flow. *CTR, Annual Research Briefs* pp. 211–218.

Gustafsson, B., Lötstedt, P. & Göran, A. 2003 A fourth order difference method for the incompressible Navier-Stokes equations. In *Numerical Simulations of Incompressible Flows* (ed. M. M. Hafez), pp. 263–276. Singapore: World Scientific Publishing.

Gustafsson, B. & Nilsson, J. 2000 Boundary conditions and estimates for the steady Stokes equations on staggered grids. *J. Sci. Comput.* **15**, 29–54.

Gustafsson, B. & Nilsson, J. 2002 Fourth order methods for the Stokes and Navier-Stokes equations on staggered grids. In *Frontiers of Computational Fluid Dynamics – 2002* (ed. D. A. Caughey & M. M. Hafez), pp. 165–179. Singapore: World Scientific Publishing.

van Kan, J. 1986 A second-order accurate pressure correction method for viscous incompressible flow. *SIAM J. Sci. Stat. Comput.* **7**, 870–891.

Kim, J. & Moin, P. 1985 Application of a fractional-step method to incompressible Navier-Stokes equations. *J. Comput. Phys.* **59**, 308–323.

Kress, W. & Lötstedt, P. 2004 Time step restrictions using semi-implicit methods for the incompressible Navier-Stokes equations. *Tech. Rep.* TR 2004-030, available at http://www.it.uu.se/research/reports/. Dept of Information Technology, Uppsala University, Uppsala, Sweden.

Kress, W. & Nilsson, J. 2003 Boundary conditions and estimates for the linearized Navier-Stokes equations on a staggered grid. *Computers & Fluids* **32**, 1093–1112.

Lele, S. K. 1992 Compact finite difference schemes with spectral-like resolution. *J. Comput. Phys.* **103**, 16–42.

Lundbladh, A., Henningson, D. S. & Johansson, A. V. 1992 An efficient spectral integration method for the solution of the Navier–Stokes equation. *Tech. Rep.* TN 1992-28. FFA, Stockholm.

Nagarajan, S., Lele, S. K. & Ferziger, J. H. 2003 A robust high-order compact method for large eddy simulation. *J. Comput. Phys.* **191**, 392–419.

Nilsson, J. 2000 Initial-boundary-value problems for the Stokes and Navier-Stokes equations on staggered grids. PhD Thesis, Dept. of Scientific Computing, Uppsala University, Uppsala, Sweden.

Nilsson, J., Gustafsson, B., Lötstedt, P. & Brüger, A. 2003 High order difference method on staggered, curvilinear grids for the incompressible Navier-Stokes equations. In *Second MIT Conference on Computational Fluid and Solid Mechanics 2003* (ed. K. J. Bathe), pp. 1057–1061. Elsevier.

Perot, J. B. 1993 An analysis of the fractional step method. *J. Comput. Phys.* **108**, 51–58.

Peyret, R. & Taylor, T. D. 1983 *Computational Methods for Fluid Flow*. New York: Springer.

Piller, M. & Stalio, E. 2004 Finite-volume compact schemes on staggered grids. *J. Comput. Phys.* **197**, 299–340.

Redžić, D. V. 2001 The operator ∇ in orthogonal curvilinear coordinates. *European Journal of Physics* **22**, 595–599.

Rempfer, D. 2003 On boundary conditions for incompressible Navier-Stokes problems. *Manuscript* .

Strikwerda, J. C. & Lee, Y. S. 1999 The accuracy of the fractional step method. *SIAM J. Numer. Anal.* **37**, 37–47.

Ståalberg, E., Brüger, A., Lötstedt, P., Johansson, A. V. & Henningson, D. S. 2004 High order accurate solution of flow past a circular cylinder. *Accepted to J. Sci. Comput* .

Visbal, M. R. & Gaitonde, D. V. 2002 On the use of higher-order finite difference schemes on curvilinear and deforming meshes. *J. Comput. Phys.* **181**, 155–185.

van der Vorst, H. A. 1992 Bi-cgstab: A fast and smoothly converging variant of bi-cg for solution of nonsymmetric systems. *J. Sci. Stat. Comput.* **13**, 631–644.

Paper 2

2

High order accurate solution of flow past a circular cylinder

By Erik Stålberg[1], Arnim Brüger[1], Per Lötstedt[2], Arne V. Johansson[1] and Dan S. Henningson[1]

Accepted to Journal of Scientific Computing

A high order method is applied to time dependent incompressible flow around a circular cylinder geometry. The space discretization employs compact fourth order difference operators. In time we discretize with a second order semi-implicit scheme. A large linear system of equations is solved in each time step by a combination of outer and inner iterations. An approximate block factorization of the system matrix is used for preconditioning. Well posed boundary conditions are obtained by an integral formulation of boundary data including a condition on the pressure. Two-dimensional flow around a circular cylinder is studied for Reynolds numbers in the range $7 \leq R \leq 180$. The results agree very well with the data known from numerical and experimental studies in the literature.

1. Introduction

High order finite difference methods have become a powerful tool for accurate solution of the Navier–Stokes equations. In comparison with e.g. spectral methods the decisive advantage is that non-trivial geometries can be described by suitable coordinate transformations. Our aim is to validate such a high order difference method with incompressible, time dependent flow past a two-dimensional circular cylinder. For recent reviews of high order techniques see Brüger *et al.* (2005), Piller & Stalio (2004). In fluid dynamics most high order methods are developed for simulation of turbulent flow where accuracy and efficiency become particularly important. This is also the intention with the present method which has been extended to three dimensions by a Fourier expansion, see Brüger *et al.* (2004). In our case where the considered geometries of the boundaries are of not too complicated shape, an accurate and efficient method is to provide an orthogonal mapping from the physical to the Cartesian computational domain. In that way the number of terms to evaluate is reduced. The spatial derivatives are approximated by compact high order

[1]Department of Mechanics, KTH, SE-10044 Stockholm, Sweden
[2]Department of Information Technology, Division of Scientific Computing, Uppsala University, SE-75105 Uppsala, Sweden.

difference operators. Parasitic oscillations in the pressure are inhibited by a staggered grid arrangement of the unknowns. A semi-implicit second order accurate time integration method is chosen in order to avoid the implicit solution of non-linear equations. A well posed system of equations is achieved by a novel formulation of the boundary conditions where one condition on the velocities is substituted by a condition on the pressure. Considerable effort is devoted to the construction of the solution algorithm. A combination of outer and inner iterations is applied to solve the linear system of equations in each time step. The outer iteration is preconditioned by an approximate block LU factorization of the system matrix. The advantages over classical fractional step methods are that the solution procedure does not require extra intermediate boundary conditions and that second order time accurate pressure solutions can be obtained. In Brüger *et al.* (2005) the iterative method is described and tested for curvilinear geometries and the order of accuracy of the discretization is experimentally demonstrated. A description of the three-dimensional method is found in Brüger *et al.* (2004).

Flow around circular cylinders is a test case widely used to validate numerical methods. Numerous experimental, numerical and analytical studies exist for both laminar and turbulent flows, see Roshko (1954) and Braza *et al.* (1986).

2. Space and time discretization

We consider the equations that describe two-dimensional incompressible flow. Denote the velocity components as u, v and p as the pressure. The Reynolds number, $R = U_\infty\, d/\nu$, is obtained with the cylinder diameter d as length scale, the far field velocity U_∞ and kinematic viscosity ν. With $\mathbf{v} = (u\ v)^T$ we arrive at the non-dimensionalized incompressible Navier–Stokes equations

$$\mathbf{v}_t + N(\mathbf{v}) + L(\mathbf{v}, p) = 0, \qquad \nabla \cdot \mathbf{v} = 0 \tag{1}$$

with the nonlinear and linear terms

$$N(\mathbf{v}) = (\mathbf{v} \cdot \nabla)\mathbf{v}, \quad L(\mathbf{v}, p) = \nabla p - R^{-1}\Delta\mathbf{v}.$$

A subscript in (1) denotes differentiation. The equations (1) are discretized in time by extrapolation of the nonlinear terms $N(\mathbf{v})$ from the levels t^{n-1} and t^n to t^{n+1} and applying the second order backward differentiation formula (BDF-2) to $\mathbf{v}_t$ and $L(\mathbf{v}, p)$ at t^{n+1} to obtain

$$\frac{3}{2}\mathbf{v}^{n+1} + \Delta t L(\mathbf{v}, p)^{n+1} = 2\mathbf{v}^n - \frac{1}{2}\mathbf{v}^{n-1} - \Delta t(2N(\mathbf{v}^n) - N(\mathbf{v}^{n-1})). \tag{2}$$

In addition to (2), $\mathbf{v}^{n+1}$ satisfies the discrete form of the divergence equation in (1). The result is a linear system of equations to solve for $\mathbf{v}^{n+1}$ and p^{n+1} in every time step. The method allows mapping of the equations from physical space (x, y) into computational space (ξ, η) through an orthogonal transformation. The velocity components are defined locally in the coordinate directions

ξ and η, respectively. A staggered grid type is used for discretization of the transformed system of partial differential equations. With this formulation the structure of the equations is similar to the Cartesian case, especially for the gradient terms and the divergence, i.e.

$$\nabla p^T = \left(\frac{1}{n_1} p_\xi,\ \frac{1}{n_2} p_\eta \right), \quad \nabla \cdot \mathbf{v} = \frac{1}{n_1 n_2} \left(\frac{\partial}{\partial \xi} (n_2 u) + \frac{\partial}{\partial \eta} (n_1 v) \right),$$

where $n_1 = \sqrt{x_\xi^2 + y_\xi^2}$ and $n_2 = \sqrt{x_\eta^2 + y_\eta^2}$ are the scale factors of the transformation. However, the Laplace part becomes more complicated including also first order derivatives. Two different fourth order compact operators are defined, see Lele (1992), one for regular grids (3) and one for staggered grids (4). For a function f and grid spacing h, we have the formulas

$$\frac{1}{6} f'_{i-1} + \frac{2}{3} f'_i + \frac{1}{6} f'_{i+1} = \frac{1}{2h} (f_{i+1} - f_{i-1}), \tag{3}$$

$$\frac{1}{24} f'_{i-1} + \frac{11}{12} f'_i + \frac{1}{24} f'_{i+1} = \frac{1}{h} (f_{i+1/2} - f_{i-1/2}). \tag{4}$$

No boundary conditions are available for the derivatives, and there we use one-sided stencils, see Brüger *et al.* (2005). The grids used in the present simulations have one periodic coordinate direction where the corresponding solution routines are replaced by solvers with periodic tridiagonal matrices. For interpolation between any positions in the staggered grid compact schemes are used. For more details of the discretization method see Brüger *et al.* (2005) and Brüger *et al.* (2002).

3. Well posed boundary conditions

The difficulty in specifying boundary conditions for the incompressible Navier–Stokes equations in divergence form is discussed in Kress & Nilsson (2003). The analysis reveals that we over-determine the prescription of velocity data, while a condition on the pressure is lacking when solely velocities are specified as boundary conditions.

We apply the approach of Kress & Nilsson (2003) to the circular cylinder case. By an integral formulation of velocity data one degree of freedom is released that is replaced by a condition on the pressure. Due to the periodicity of the azimuthal coordinate direction, boundary conditions are set on the cylinder wall and on the farfield boundary only. In the computational domain $\Omega = \{(\xi, \eta) \in [\xi_0 \quad \xi_1] \times [\eta_0 \quad \eta_1]\}$ the continuous formulation of Dirichlet boundary conditions is

$$u(\xi,\eta_0)=u_0(\xi), \qquad v(\xi,\eta_0)-\frac{1}{l_c}\int_{\xi_0}^{\xi_1} v(\xi,\eta_0)d\xi = w_c(\xi), \tag{5}$$

$$u(\xi,\eta_1)=u_1(\xi), \qquad v(\xi,\eta_1)=v_1(\xi), \tag{6}$$

$$\int_{\xi_0}^{\xi_1} v(\xi,\eta_0)d\xi + \int_{\xi_0}^{\xi_1} p(\xi,\eta_0)d\xi = p_c\,, \tag{7}$$

where $\int_{\xi_0}^{\xi_1} w_c(\xi)d\xi = 0$, p_c is an arbitrary constant and $l_c = \int_{\xi_0}^{\xi_1} d\xi$. $\eta = \eta_0$ and $\eta = \eta_1$ correspond to boundaries on the cylinder wall and at the farfield of the physical grid. Note that in the implementation condition (7) substitutes condition (5) in one v point. Further discussion of similar types of boundary conditions is available in Brüger *et al.* (2004).

4. Solution algorithm

After space discretization of (1) and elimination of the boundary conditions Brüger *et al.* (2005) a linear system of the form $M\mathbf{U} = \mathbf{g}$ is to be solved in each time step, where

$$M = \begin{bmatrix} A & G \\ D & B \end{bmatrix}, \quad \mathbf{U} = \begin{bmatrix} \tilde{\mathbf{v}}^{n+1} \\ p^{n+1} \end{bmatrix}, \quad \mathbf{g} = \mathbf{g}(\tilde{\mathbf{v}}^n, \tilde{\mathbf{v}}^{n-1}) = \begin{bmatrix} \mathbf{g}_1 \\ g_2 \end{bmatrix}.$$

Modified approximations of the gradient and the divergence are G and D. The submatrix B is sparse and appears as a consequence of the implicit boundary formulation of the pressure and the reduction of boundary data. $\tilde{\mathbf{v}}$ in $\mathbf{U}$ represents the unknowns on inner grid locations only. A is block diagonal and with L as the discrete Laplacian one block is similar to $3/2I - R^{-1}\Delta t L$ in each coordinate direction. An approximate factorization Perot (1993) of M is here constructed as

$$\tilde{M} = \begin{bmatrix} A & 0 \\ D & I \end{bmatrix} \begin{bmatrix} I & \frac{2}{3}G \\ 0 & -(\frac{2}{3}DG - B) \end{bmatrix} = \begin{bmatrix} A & \frac{2}{3}AG \\ D & B \end{bmatrix}. \tag{8}$$

In what is denoted as *outer iteration* we solve for the new iterate $\mathbf{U}^{k+1}$

$$\mathbf{U}^{k+1} = \mathbf{U}^k + \mathbf{z} = \mathbf{U}^k + \tilde{M}^{-1}(\mathbf{g} - M\mathbf{U}^k). \tag{9}$$

$\mathbf{z} = (\mathbf{z}_1\, z_2)^T$ is the update determined in iteration k. One or two fixed point iterations usually suffice, except for the case of too small Reynolds numbers. Using the factorization of $\tilde{M}$ in (8), there are block forward and back substitution steps. In the forward step a system is solved for prediction of the velocities:

$$A\mathbf{y}_1 = \mathbf{g}_1\,, \quad y_2 = g_2 - D\mathbf{y}_1. \tag{10}$$

The back substitution step includes an equation for the pressure and a projection of the solution onto the divergence-free velocity field.

$$-\left(\frac{2}{3}DG - B\right) z_2 = y_2\,, \quad \mathbf{z}_1 = \mathbf{y}_1 - \frac{2}{3}Gy_2. \tag{11}$$

The first equation in (10) is solved by fixed point iteration and is preconditioned with an approximate inverse H of A. The simplest choice of H is $\frac{2}{3}I$. Since D and G are not known explicitly in (11), a suitable class of methods is of Krylov type, where only calculation of $\left(\frac{2}{3}DG - B\right)x$ for an arbitrary x is required for the method to be applicable. The Krylov method in our implementation is Bi-CGSTAB. For rapid convergence the pressure system in (11) is preconditioned by an incomplete LU-factorization (ILU). For computation of ILU the matrix DG needs to be provided explicitly. In the ILU factorization it is replaced by the corresponding second order approximation which is easy to compute explicitly. This is appropriate, because one can show that the second order discretization is spectrally equivalent to our fourth order method. For more details of the iterative method see Brüger *et al.* (2005).

5. Numerical experiments

With the above described method we simulate laminar flow past a circular cylinder geometry. The experiments are performed in the laminar regime where the assumption of two dimensional flow holds at a Reynolds number varying between $R = 7$ and $R = 180$.

For comparison we consider numerical and experimental studies from Dennis & Chang (1970), Park *et al.* (1998), Williamson & Brown (1998) and Williamson (1989). The numerical method in Dennis & Chang (1970) employs standard high order difference approximations and solves the steady Navier–Stokes equations. The unsteady simulation of Park *et al.* (1998) is run with a standard second order difference discretization and a fully implicit solver allowing for large time steps. The grid is of C-type with the outflow boundary only 20 diameters downstream of the cylinder. The grid resolution in Dennis & Chang (1970) is comparable to the present study. Approximately ten times as many grid points are used in Park *et al.* (1998).

The present study is split in two parts, one for $R < 45$ where the flow is steady and the second at $45 < R \leq 180$ where the flow is unstable due to the vortex shedding. The grids used are of O-type mapped by a polar coordinate transformation.

The no slip condition is set at $\eta = \eta_0$ leading to $u_0 = w_c = 0$ in (5) and the far field boundary is assumed to be far enough from the cylinder to justify the free stream condition. The local components $u_1(\xi)$ and $v_1(\xi)$ in (6) are therefore prescribed to generate a streamwise velocity component equal to one. In connection with Dirichlet far field boundary conditions as in the present study the buffer zone technique is common. Here, a radial grid stretching function is defined as in Zhang & Ko (1996) with rapidly increasing cell sizes in the outer region where the velocity approaches the constant free stream condition.

We consider in detail the pressure coefficient and dimensionless vorticity

$$C_p = \frac{p - p_\infty}{\frac{1}{2}\rho U_\infty^2}, \quad \omega = \frac{1}{n_1\, n_2}\left[\frac{\partial}{\partial \xi}\,(n_2\, v) - \frac{\partial}{\partial \eta}\,(n_1\, u)\right], \tag{12}$$

and the global drag and lift coefficients

$$C_D = \underbrace{\frac{1}{2}\int_0^{2\pi} C_{p\,w}\cos(\xi)\,d\xi}_{C_{D_p}} \underbrace{-\frac{2}{R}\int_0^{2\pi} \omega_{\,w}\sin(\xi)\,d\xi}_{C_{D_v}}, \tag{13}$$

$$C_L = \underbrace{-\frac{1}{2}\int_0^{2\pi} C_{p\,w}\sin(\xi)\,d\xi}_{C_{L_p}} \underbrace{-\frac{2}{R}\int_0^{2\pi} \omega_{\,w}\cos(\xi)\,d\xi}_{C_{L_v}}, \tag{14}$$

where the subscript w denotes the cylinder surface, indices p and v label the pressure and viscous contributions, respectively. These quantities are computed with fourth order accuracy.

5.1. *Steady flow regime*

Results from numerical simulations are here presented at small Reynolds numbers 7, 10, 20 and 40 where the flow is steady. For comparison, simulations from Dennis & Chang (1970) are considered. Our data come from simulations with the grid size 90×45 in the ξ-η plane. The outer boundary is located at 40 cylinder diameters away from the cylinder.

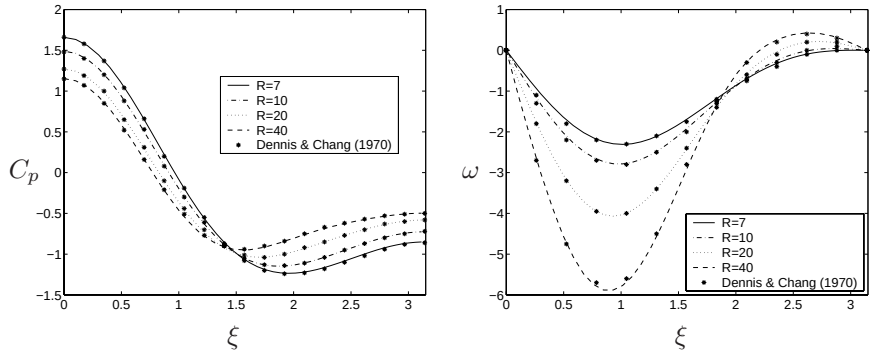

FIGURE 1. Left: The pressure coefficient on the cylinder surface. $\xi = 0$ is the front stagnation point. Right: The vorticity on the cylinder surface.

Table 1 summarises the quantities measured in our study and compares them with numerical simulations of Dennis & Chang (1970). The pressure and viscous drag coefficients, C_{D_p}, C_{D_v} defined in (13) are presented. Moreover, we show $C_p(\xi = \pi)$, the pressure coefficient at the rear of the cylinder. L/d is the separation bubble length, measured as the distance from the center of the

	Present work				Dennis & Chang (1970)			
R	C_{D_p}	C_{D_v}	$C_p(\xi=\pi)$	L/d	C_{D_p}	C_{D_v}	$C_p(\xi=\pi)$	L/d
7	1.854	1.554	-0.851	0.67	1.868	1.553	-0.870	0.60
10	1.589	1.251	-0.726	0.79	1.600	1.246	-0.742	0.77
20	1.229	0.823	-0.581	1.40	1.233	0.812	-0.589	1.44
40	0.994	0.536	-0.482	2.63	0.998	0.524	-0.509	2.85

TABLE 1. Comparison of results at $R = 7, 10, 20, 40$ with data from Dennis & Chang (1970)

cylinder to the point where the streamwise velocity is zero and normalized with the cylinder diameter. The distribution of the pressure coefficient and vorticity (12) on the cylinder surface as a function of the azimuthal coordinate is shown in Figure 1. C_p and ω are plotted for $R = 7, 10, 20, 40$. The data from our simulations match the study in Dennis & Chang (1970) very well over the range of investigated Reynolds numbers.

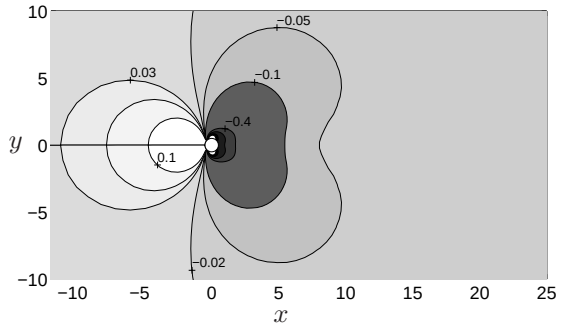

FIGURE 2. Simulation at $R = 40$. Isolines of the pressure coefficient C_p are plotted at levels between -0.7 and 0.1.

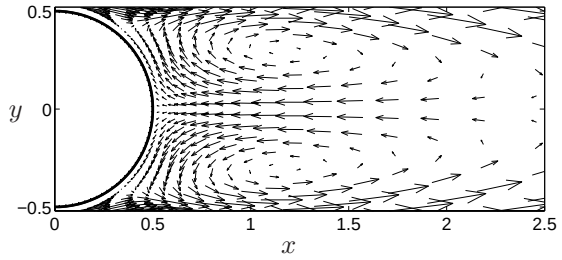

FIGURE 3. Velocity vectors in the wake region from simulation at $R = 40$.

In Figure 2 isolines of the pressure coefficient (12) are shown for $R = 40$ and the recirculation zone with the characteristic vortex pair at $R = 40$ can be seen in Figure 3. Dependence of the grid resolution is demonstrated for the $R = 40$ case. Three grid sizes are investigated, 72×32, 90×45 and 108×54. The smallest grid cell in the fine grid is $\Delta\xi \times \Delta\eta = 0.0228 \times 0.0276$ and in the coarse grid 0.0344×0.0491. The drag coefficient (13) is 1.546, 1.530 and 1.527 for the coarse, medium and fine grid respectively. Assuming fourth order accuracy, the extrapolated value when $h \to 0$ is 1.525. The separation bubble length, L/d, is 2.55, 2.63 and 2.66. We conclude that the variations are small, and in order to limit the computational cost the medium grid is chosen for our simulations. We have integrated the unsteady equations in time to obtain a stationary solution for flow past circular cylinder. Another approach for this low Reynolds number case is to solve steady equations directly as in Dennis & Chang (1970). Our data are in good agreement with that study.

5.2. *Vortex shedding regime*

In this section results are shown from simulations with unsteady flow at the Reynolds numbers $60, 100, 120, 160$ and 180. At those conditions alternating vortices are convected downstream from the cylinder, i.e. the well-known von Kármán street. This phenomenon has been studied extensively with numerical and experimental methods in e.g. Park *et al.* (1998), Williamson (1989) and Williamson & Brown (1998). The characteristic frequency present in the shedding is commonly expressed in the dimensionless Strouhal number $St = f\, d/U_\infty$, where f is the measured frequency. Williamson & Brown (1998) formulated a functional relationship between the Strouhal and the Reynolds number by expanding data from two-dimensional computations and experiments in powers of $1/\sqrt{R}$,

$$St = 0.2731 - \frac{1.1129}{\sqrt{R}} + \frac{0.4821}{R}. \tag{15}$$

The present numerical simulations are carried out without imposing any perturbation on the flow field. The flow is globally unstable, and even small numerical errors are disturbing sufficiently to trigger the vortex shedding. $R \approx 180$ marks the upper end of the laminar regime. The grid used in the unsteady regime is 130×90 with the farfield boundary located 80 diameters away from the cylinder.

The Strouhal number determined from numerical experiments versus Reynolds number is shown to the left in Figure 4 for Reynolds numbers up to 180. Comparisons are included with the experimental data of Williamson (1989) and the relation (15) from Williamson & Brown (1998). The instantaneous vorticity field at $R = 180$ can be seen to the right in Figure 4. Table 2 shows the results from our simulation in comparison with computational studies reported in Park *et al.* (1998). C_D and C_{D_p} are mean values and $\hat{\cdot}$ denotes the amplitude of a quantity. The data obtained with our fourth order method are in very good

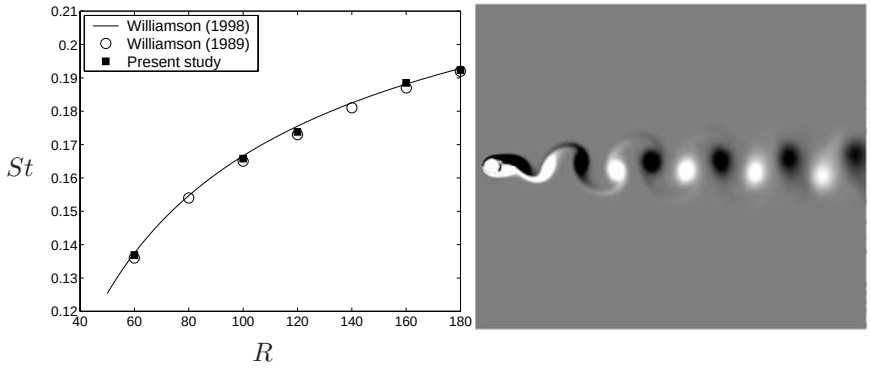

FIGURE 4. Left: ■: present study; —: $St = 0.2731 - \frac{1.1129}{\sqrt{R}} + \frac{0.4821}{R}$ Williamson & Brown (1998); ∘: experiments Williamson (1989). Right: Snapshot of vorticity field at $R = 180$.

agreement with the second order accurate results reported in Park *et al.* (1998) but with less than one tenth as many grid points.

	Present work				Park *et al.* (1998)			
R	60	100	120	160	60	100	120	160
C_D	1.39	1.32	1.31	1.31	1.39	1.33	1.32	1.32
C_{D_p}	0.954	0.972	0.981	1.02	0.96	0.99	1.01	1.04
$\hat{C}_D \cdot 10^3$	1.31	9.00	14.2	29.3	1.4	9.1	15.2	29.3
$\hat{C}_L$	0.134	0.330	0.401	0.568	0.1344	0.3321	0.4103	0.5501

TABLE 2. Results from computations at $R = 60, 100, 120, 160$

6. Conclusions

A high order difference method comprising novel boundary conditions and solution techniques has been applied to incompressible flow past a circular cylinder. The goal with this study is to validate the numerical method with a complex physical flow problem. Numerical experiments have been performed for laminar steady and unsteady flow in two dimensions. The results show very good agreement with data from earlier experiments and numerical simulations in the literature and our fourth order method allows us to use much fewer grid points for comparable accuracy.

Acknowledgements

The authors wish to thank Jonas Nilsson from the Department of Petroleum Engineering, Stanford University, and Bertil Gustafsson from the Department

of Information Technology, Uppsala University, who have contributed with many valuable ideas.

References

Braza, M., Chassaing, P. & Minh, H. 1986 Numerical study and physical analysis of the pressure and velocity fields in the near wake of a circular cylinder. *J. Fluid Mech.* **165**, 79–130.

Brüger, A., Gustafsson, B., Lötstedt, P. & Nilsson, J. 2005 High order accurate solution of the incompressible Navier-Stokes equations. *J. Comput. Phys.* **203**, 49–71.

Brüger, A., Gustafsson, B., Lötstedt, P. & Nilsson, J. 2004 Splitting methods for high order solution of the incompressible Navier-Stokes equations in 3D. *To appear in Int. J. Numer. Meth. Fluids* .

Brüger, A., Nilsson, J. & Kress, W. 2002 A compact higher order finite difference method for the incompressible Navier-Stokes equations. *J. Sci. Comput.* **17**, 551–560.

Dennis, S. C. R. & Chang, G. 1970 Numerical solution for steady flow past a circular cylinder at Reynolds numbers up to 100. *J. Fluid Mech.* **42**, 471–489.

Kress, W. & Nilsson, J. 2003 Boundary conditions and estimates for the linearized Navier-Stokes equations on a staggered grid. *Computers & Fluids* **32**, 1093–1112.

Lele, S. K. 1992 Compact finite difference schemes with spectral-like resolution. *J. Comput. Phys.* **103**, 16–42.

Park, J., Kwon, K. & Choi, H. 1998 Numerical solutions of flow past a circular cylinder at Reynolds numbers up to 160. *KSME Int. J.* **12**, 1200–1205.

Perot, J. B. 1993 An analysis of the fractional step method. *J. Comput. Phys.* **108**, 51–58.

Piller, M. & Stalio, E. 2004 Finite-volume compact schemes on staggered grids. *J. Comput. Phys.* **197**, 299–340.

Roshko, A. 1954 On the development of turbulent wakes from vortex streets. *NACA Rep.* **1191**, 1–23.

Williamson, C. H. K. 1989 Oblique and parallel modes of vortex shedding in the wake of a circular cylinder at low Reynolds number. *J. Fluid Mech.* **206**, 579–627.

Williamson, C. H. K. & Brown, G. L. 1998 A series in $1/\sqrt{Re}$ to represent the Strouhal–Reynolds number relationship of the cylinder wake. *J. Fluids Struct.* **12**, 1073–1085.

Zhang, H. L. & Ko, N. W. M. 1996 Numerical analysis of incompressible flow over smooth and grooved circular cylinders. *Computers & Fluids* **25**, 263–281.

Paper 3

3

Applications of a high order method for fluid flows in complex geometries

By Erik Stålberg, Luca Brandt and Arnim Brüger

Department of Mechanics, KTH, SE-10044 Stockholm, Sweden

A hybrid high order method for the solution of the incompressible Navier–Stokes equations in curvilinear geometries is presented in connection with numerical experiments. The discretization combines a fourth order compact difference method on a staggered, curvilinear grid with a Fourier expansion in the third dimension. The time discretization is semi-implicit and second order accurate. Numerical experiments for external flow past a parabolic body and internal flow in a plane asymmetric diffuser are presented. Issues regarding boundary conditions and grid generation for the two configurations are considered.

1. Introduction

A high order accurate pseudo-spectral method for simulation of incompressible fluid flow in complex geometries is here presented in context with numerical experiments. The geometries considered here are allowed to have curvature in two dimensions where compact finite difference operators are used for approximation of space derivatives. For three-dimensional flow problems the third dimension is discretized using a Fourier expansion of the variables.

Compact finite difference operators are known to have near spectral properties, see Lele (1992), and allow at the same time for geometric flexibility in comparison with standard spectral methods. The interest in using high order methods stems from the fact that the same error level can be achieved with less number of grid points, and thus less memory usage, in comparison with low order accurate methods. Other approaches for simulation of fluid flow in complex geometries using high order discretizations can be found in, for example, Visbal & Gaitonde (2002) and Fischer *et al.* (2002).

Two distinct problems are addressed in this study. The receptivity to external perturbations of the boundary layer forming on aerodynamic bodies and the solution of flow in a plane asymmetric diffuser. To solve the first type of problems one needs to compute the laminar flow around a body immersed in a uniform free-stream, i.e. an external flow. This is usually a two dimensional problem and it poses problems in terms of in- and outflow (non-reflective) boundary conditions as well as solution of flow with a stagnation point. In

the case of turbulent diffuser flow, conversely, the problem is three-dimensional and the efficiency of the code is challenged in terms of computer time.

Theoretical studies of the receptivity problem have been followed by direct numerical simulations only in the last decade. Buter & Reed (1994) studied receptivity to free-stream vorticity for flow over a semi-infinite flat plate with an elliptic leading edge using a vorticity-stream function formulation. Haddad & Corke (1998) simulated receptivity to free-stream sound on parabolic bodies also in vorticity-stream function variables. Collis & Lele (1999) used a compressible formulation in primitive variables to study surface roughness receptivity on a swept leading edge at low Mach numbers. The main advantage of using a primitive variables formulation is that three dimensional problems can be considered.

The first large eddy simulation of flow in asymmetric diffuser was performed by Kaltenbach *et al.* (1999). Ohta *et al.* (2003) performed a direct numerical simulation of this flow configuration by a fourth order explicit finite difference method in primitive variables on a collocated grid arrangement.

The aim of this study is not to produce data to gain insight into the physics of the flow problems considered here. We wish to investigate the present performance of this specific method for two completely different flow cases and to give directions for future improvements of the method.

2. Numerical method

Let $\mathbf{v}$ be the velocity vector, p the pressure and R the Reynolds number. By defining the nonlinear and linear terms

$$N(\mathbf{v}) = (\mathbf{v}\cdot\nabla)\mathbf{v}\,, \quad L(\mathbf{v},p) = \nabla p - Re^{-1}\Delta\mathbf{v}\,, \tag{1}$$

the Navier–Stokes equations in primitive variables and dimensionless form are

$$\mathbf{v}_t + N(\mathbf{v}) + L(\mathbf{v},p) = 0\,, \tag{2}$$

$$\nabla\cdot\mathbf{v} = 0\,. \tag{3}$$

A subscript in equation (2) denotes differentiation. The equations (2) and (3) are discretized in time by extrapolating $N(\mathbf{v})$ from time t^{n-1} and t^n to t^{n+1} and applying the second order backward differentiation formula (BDF-2) to $\mathbf{v}_t$ and $L(\mathbf{v},p)$ at t^{n+1} to obtain

$$\frac{3}{2}\mathbf{v}^{n+1} + \Delta t L(\mathbf{v}^{n+1},p^{n+1}) = 2\mathbf{v}^n - \frac{1}{2}\mathbf{v}^{n-1} - \Delta t(2N(\mathbf{v}^n) - N(\mathbf{v}^{n-1}))\,. \tag{4}$$

In addition to (4), $\mathbf{v}^{n+1}$ satisfies the discretization of the divergence equation (3). The result is a linear system of equations to solve for $\mathbf{v}^{n+1}$ and p^{n+1} in every time step.

2.1. *Space discretization and iterative solver in two dimensions*

In two space dimensions, the equations are mapped from computational space (ξ,η) into physical space (x,y) by an orthogonal transformation

$$x = x(\xi,\eta)\,,\ y = y(\xi,\eta)\,. \tag{5}$$

The transformed system of partial differential equations is discretized on a staggered grid with grid size ($\Delta\xi$, $\Delta\eta$). The local velocity components $(\tilde{u}, \tilde{v})$ in the (ξ, η) plane are assigned at the locations $(\xi_{i-\frac{1}{2}\Delta\xi}, \eta_j)$ and $(\xi_i, \eta_{j-\frac{1}{2}\Delta\eta})$, respectively, and the pressure values are assigned at a grid point (ξ_i, η_j). The pressure gradient ∇p and divergence $\nabla \cdot \mathbf{v}$ in (2) and (3) have the structures

$$\nabla p = \begin{bmatrix} n_1^{-1} p_\xi, \\ n_2^{-1} p_\eta \end{bmatrix}, \quad \nabla \cdot \mathbf{v} = \frac{1}{n_1 n_2} \left(\frac{\partial}{\partial \xi} (n_2 \tilde{u}) + \frac{\partial}{\partial \eta} (n_1 \tilde{v}) \right), \tag{6}$$

where $n_1 = \sqrt{x_\xi^2 + y_\xi^2}$ and $n_2 = \sqrt{x_\eta^2 + y_\eta^2}$ are the scale factors of the transformation (5). For space discretization in the (ξ, η) plane two different fourth order accurate compact operators are defined, one for regular grids, see equation (7), and one for staggered grids, equation (8), see Fornberg & Ghrist (1999) and Lele (1992). For a function f and grid spacing $\Delta\xi$, we have the formulas

$$\frac{1}{6} f'_{i-1} + \frac{2}{3} f'_i + \frac{1}{6} f'_{i+1} = \frac{1}{2\Delta\xi} (f_{i+1} - f_{i-1}), \tag{7}$$

$$\frac{1}{24} f'_{i-1} + \frac{11}{12} f'_i + \frac{1}{24} f'_{i+1} = \frac{1}{\Delta\xi} (f_{i+1/2} - f_{i-1/2}). \tag{8}$$

In order to achieve closed systems of form $Pf' = Qf$ for regular grids and $Rf' = Sf$ for staggered grids, one-sided stencils are used near boundaries, see Brüger *et al.* (2005).

In the same manner as for the fourth order compact difference operators, sixth order accurate compact interpolation operators are used for interpolation between grid points on the staggered grid.

For boundary conditions, a formulation similar to the approach proposed in Gustafsson & Nilsson (2000) for the steady Stokes equations is used. This formulation has been generalized for the linearized Navier–Stokes equations on staggered grids by Kress & Nilsson (2003). The formulation cover cases where Dirichlet conditions for the velocities are prescribed as well as cases with Neumann conditions. For the numerical experiments considered in this study, Neumann boundary conditions for the velocity components are prescribed on the outflow boundaries. In a computational domain $\Omega = \{(\xi, \eta) \in [\xi_0 \quad \xi_1] \times [\eta_0 \quad \eta_1]\}$ and if all boundaries except from the outflow boundary $\xi = \xi_1$ are considered as Dirichlet boundaries for the velocity components the formulation is

$$\tilde{u}(\xi_0, \eta, t) = \tilde{u}_{\text{west}}(\eta, t), \tag{9}$$

$$\frac{\partial \tilde{u}(\xi_1, \eta, t)}{\partial \xi} - \frac{1}{L_\eta} \int_{\eta_0}^{\eta_1} n_2(\xi_1, \eta) \frac{\partial \tilde{u}(\xi_1, \eta, t)}{\partial \xi} \mathrm{d}\eta = 0, \tag{10}$$

$$\tilde{v}(\xi_0, \eta, t) = \tilde{v}_{\text{west}}(\eta, t), \qquad \frac{\partial \tilde{v}(\xi_1, \eta, t)}{\partial \xi} = 0, \tag{11}$$

$$\int_{\eta_0}^{\eta_1} n_2(\xi_1,\eta)\frac{\partial\tilde{u}(\xi_1,\eta,t)}{\partial\xi}\mathrm{d}\eta + \int_{\eta_0}^{\eta_1} n_2(\xi_1,\eta)\, p(\xi_1,\eta,t)\mathrm{d}\eta = q_0\,, \tag{12}$$

$$\tilde{u}(\xi,\eta_0,t) = \tilde{u}_{\mathrm{south}}(\xi,t)\,, \qquad \tilde{u}(\xi,\eta_1,t) = \tilde{u}_{\mathrm{north}}(\xi,t)\,, \tag{13}$$

$$\tilde{v}(\xi,\eta_0,t) = \tilde{v}_{\mathrm{south}}(\xi,t)\,, \qquad \tilde{v}(\xi,\eta_1,t) = \tilde{v}_{\mathrm{north}}(\xi,t)\,, \tag{14}$$

where $L_\eta = \int_{\eta_0}^{\eta_1} n_2(\xi_1,\eta)\mathrm{d}\eta$ and q_0 is an arbitrary constant. Note that in the discrete case (12) substitutes condition (10) in one grid point. For extension of the boundary condition formulation to three dimensional problems, see the following section.

From (4) it is evident that a linear system of the form $M\mathbf{U} = \mathbf{b}$ has to be solved in each time step. In order to get a system matrix M consisting of operators with clean structure, the unknowns are segregated in inner and boundary data. The boundary data are then eliminated and we have

$$M = \begin{bmatrix} A & G \\ D & -E \end{bmatrix}, \quad \mathbf{U} = \begin{bmatrix} \mathbf{v}^{n+1} \\ p^{n+1} \end{bmatrix}, \quad \mathbf{b} = \mathbf{b}(\mathbf{v}^n, \mathbf{v}^{n-1}) = \begin{bmatrix} \mathbf{b}_1 \\ b_2 \end{bmatrix}. \tag{15}$$

A has the structure

$$A = \frac{3}{2}I - R^{-1}\Delta t L \tag{16}$$

where L is the discrete Laplacian. G and D represents the discrete gradient and divergence operators and E stems from the boundary condition formulation for the pressure.

An approximate factorization of M is constructed similar to Perot (1993)

$$M^* = \begin{bmatrix} A & 0 \\ D & I \end{bmatrix}\begin{bmatrix} I & \frac{2}{3}G \\ 0 & -\frac{2}{3}DG - E \end{bmatrix} = \begin{bmatrix} A & \frac{2}{3}AG \\ D & -E \end{bmatrix}. \tag{17}$$

In an outer fixed point iteration a correction $\delta\mathbf{U}$ is computed as

$$\mathbf{U}^{(k+1)} = \mathbf{U}^{(k)} + \delta\mathbf{U}^{(k)} = \mathbf{U}^{(k)} + {M^*}^{-1}\left(\mathbf{b} - M\mathbf{U}^{(k)}\right) = \mathbf{U}^{(k)} + {M^*}^{-1}\mathbf{r}^{(k)} \tag{18}$$

where $\mathbf{r}^{(k)}$ is the residual from iteration k. If the residual is within the specified convergence threshold $\mathbf{U}^{n+1} = \mathbf{U}^{(k+1)}$. As a result of the approximate factorization two sub-systems are iteratively solved in each outer iteration, one advection-diffusion system for prediction of velocities and one system related to a Poisson equation for the pressure. The system matrices A and $-\frac{2}{3}DG - E$ are not known explicit due to the compact difference approximations and the iterative Krylov method Bi-CGSTAB (see van der Vorst 1992) only requiring computation of matrix-vector products is used. Incomplete LU factors (ILU) of the system matrices are used as preconditioning, see Meijerink & van der Vorst (1977). The ILU factorizations are based on an explicit second order accurate discretization of the matrices.

2.2. *Extension of the method to three-dimensional problems*

For three-dimensional flow problems the variables are expanded in Fourier modes in the periodic spanwise direction z. The inverse spanwise Fourier transform of a variable f is

$$f(x,y,z) = \sum_{m=-N_z/2}^{N_z/2-1} \hat{f}(x,y)\; e^{i\beta_m z_L l/N_z} \qquad l = 1,2,3,...,N_z. \tag{19}$$

z_L is the spanwise length of the domain, $\beta_m = 2\pi m/z_L$ the spanwise wavenumbers and N_z the number of grid points in the spanwise direction.

The spanwise velocity component w is assigned at the same points as the pressure, (ξ_i, η_j, z_l). In the same manner as in (15) the linear system in three dimensions is

$$\underbrace{\begin{pmatrix} A & 0 & G \\ 0 & \tilde{A} & i\beta\Delta t I \\ D & i\beta\tilde{D} & -E \end{pmatrix}}_{M_{3D}} \begin{pmatrix} \hat{\mathbf{v}}^{n+1} \\ \hat{w}^{n+1} \\ \hat{p}^{n+1} \end{pmatrix} = \underbrace{\begin{pmatrix} \hat{\mathbf{b}}_1 \\ \hat{b}_w \\ \hat{b}_2 \end{pmatrix}}_{\hat{\mathbf{b}}_{3D}}. \tag{20}$$

with the approximate factorization

$$\begin{aligned} M^*_{3D} &= \begin{pmatrix} A & 0 & 0 \\ 0 & \tilde{A} & 0 \\ D & i\beta\tilde{D} & I \end{pmatrix} \begin{pmatrix} I & 0 & \gamma G \\ 0 & I & \gamma i\beta\Delta t I \\ 0 & 0 & Q \end{pmatrix} \\ &= \begin{pmatrix} A & 0 & \gamma AG \\ 0 & \tilde{A} & \gamma i\beta\Delta t\tilde{A} \\ D & i\beta\tilde{D} & -E \end{pmatrix}, \end{aligned} \tag{21}$$

where $Q = -E - \gamma\, DG + \gamma\beta^2\Delta t\, \tilde{D}$ and γ is set to

$$\gamma = \frac{1}{\frac{3}{2} + \frac{\Delta t}{R}\beta^2} \tag{22}$$

for time integration with BDF-2.

The system is solved in Fourier transformed space. The right hand side $\hat{\mathbf{b}}_{3D}$ consists of the explicitly treated non-linear terms in (1) and in order to avoid expensive convolution sums in Fourier transformed space a pseudo-spectral approach is followed. The variables at time levels n and $n-1$ are transformed to physical space where the non-linear terms and the right hand side $\mathbf{b}_{3D}$ at time level $n+1$ are constructed and $\mathbf{b}_{3D}$ is finally transformed to Fourier transformed space. Fast Fourier Transforms (FFT) are used to transform the variables between Fourier and physical space, see Canuto *et al.* (1988).

The boundary condition formulation for the zero wavenumber, $\beta = 0$, will have the same structure as in the two dimensional case. By again assuming that all boundaries in a (ξ, η) plane are considered as Dirichlet boundaries except from the outflow boundary, $\xi = \xi_1$, which is considered as a Neumann boundary for the velocity components, the formulation in three dimensions is

$\underline{\beta = 0}$

$$\tilde{u}(\xi_0, \eta, z, t) = \tilde{u}_{\text{west}}(\eta, z, t)\,, \tag{23}$$

$$\frac{\partial \tilde{u}(\xi_1, \eta, z, t)}{\partial \xi} - \frac{1}{L_\eta} \int_{\eta_0}^{\eta_1} n_2(\xi_1, \eta) \frac{\partial \tilde{u}(\xi_1, \eta, z, t)}{\partial \xi} \mathrm{d}\eta = 0\,, \tag{24}$$

$$\tilde{v}(\xi_0, \eta, z, t) = \tilde{v}_{\text{west}}(\eta, z, t)\,, \qquad \frac{\partial \tilde{v}(\xi_1, \eta, z, t)}{\partial \xi} = 0\,, \tag{25}$$

$$w(\xi_0, \eta, z, t) = w_{\text{west}}(\eta, z, t)\,, \qquad \frac{\partial w(\xi_1, \eta, z, t)}{\partial \xi} = 0\,, \tag{26}$$

$$\int_{\eta_0}^{\eta_1} n_2(\xi_1, \eta) \frac{\partial \tilde{u}(\xi_1, \eta, z, t)}{\partial \xi} \mathrm{d}\eta + \int_{\eta_0}^{\eta_1} n_2(\xi_1, \eta)\, p(\xi_1, \eta, z, t) \mathrm{d}\eta = q_0\,, \tag{27}$$

$$\tilde{u}(\xi, \eta_0, z, t) = \tilde{u}_{\text{south}}(\xi, z, t)\,, \qquad \tilde{u}(\xi, \eta_1, z, t) = \tilde{u}_{\text{north}}(\xi, z, t)\,, \tag{28}$$

$$\tilde{v}(\xi, \eta_0, z, t) = \tilde{v}_{\text{south}}(\xi, z, t)\,, \qquad \tilde{v}(\xi, \eta_1, z, t) = \tilde{v}_{\text{north}}(\xi, z, t)\,, \tag{29}$$

$$w(\xi, \eta_0, z, t) = w_{\text{south}}(\xi, z, t)\,, \qquad w(\xi, \eta_1, z, t) = w_{\text{north}}(\xi, z, t)\,. \tag{30}$$

$\underline{\beta \neq 0}$

$$\begin{aligned}
&\tilde{u}(\xi_0, \eta, z, t) = \tilde{u}_{\text{west}}(\eta, z, t)\,, && \frac{\partial \tilde{u}(\xi_1, \eta, z, t)}{\partial \xi} = 0\,, \\
&\tilde{v}(\xi_0, \eta, z, t) = \tilde{v}_{\text{west}}(\eta, z, t)\,, && \frac{\partial \tilde{v}(\xi_1, \eta, z, t)}{\partial \xi} = 0\,, \\
&w(\xi_0, \eta, z, t) = w_{\text{west}}(\eta, z, t)\,, && \frac{\partial w(\xi_1, \eta, z, t)}{\partial \xi} = 0\,, \\
&\tilde{u}(\xi, \eta_0, z, t) = \tilde{u}_{\text{south}}(\xi, z, t)\,, && \tilde{u}(\xi, \eta_1, z, t) = \tilde{u}_{\text{north}}(\xi, z, t)\,, \\
&\tilde{v}(\xi, \eta_0, z, t) = \tilde{v}_{\text{south}}(\xi, z, t)\,, && \tilde{v}(\xi, \eta_1, z, t) = \tilde{v}_{\text{north}}(\xi, z, t)\,, \\
&w(\xi, \eta_0, z, t) = w_{\text{south}}(\xi, z, t)\,, && w(\xi, \eta_1, z, t) = w_{\text{north}}(\xi, z, t)\,.
\end{aligned} \tag{31}$$

Again, (27) substitutes (24) in one grid point. In (20) $E = 0$ for $\beta \neq 0$ since the implicit integral condition on the pressure (27) only is applied for $\beta = 0$. If $E = 0 \quad \forall \beta$, Q in (21) would be singular, see Brüger *et al.* (2005). The well posed formulation of boundary conditions resulting in $E \neq 0$ removes this singularity.

3. Laminar flow past parabolic body

Flow past parabolic bodies is an interesting test case since the flow proceeds from stagnation point flow at the nose to Blasius flow far downstream without separation, see Erturk *et al.* (2004). van Dyke (1962) pointed out that there is no singularity at the nose as in the case of a semi-infinite flat plate. Flow past parabolic bodies is used in receptivity studies, i.e. the study of the mechanism by which disturbances in the free stream or on the body (surface roughness) enter the boundary layer and excite unstable modes. The surface of a parabolic body is continuous and infinitely differentiable which inhibit additional sources

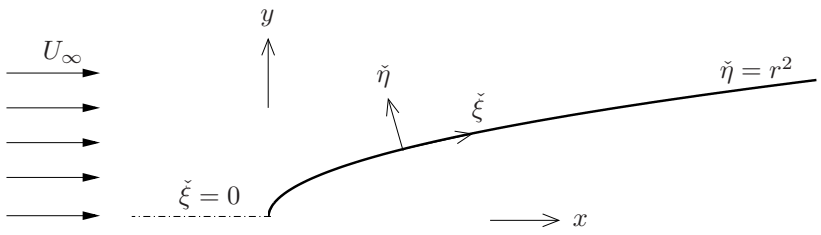

FIGURE 1. Schematic configuration of steady flow past parabolic body.

of receptivity in comparison with smooth leading edge attached to flat plate geometries as in Buter & Reed (1994).

A conformal map exist between the parabolic coordinates and a rectangular domain which makes this geometry suitable for our method.

3.1. *Configuration*

Consider the two dimensional flow past a parabolic body as in figure 1. The parabolic coordinates are $(\check{\xi}, \check{\eta})$ and the body is located at $\check{\eta} = r^2$, where r is the nose radius of curvature. The domain is limited to the upper half of the flow field owing to the symmetry about the x-axis. The conformal map between the physical and parabolic coordinates is

$$x = \frac{1}{2}\left(r^4 + \check{\xi}^2 - \check{\eta}^2\right), \qquad y = \check{\xi}\check{\eta}. \tag{32}$$

The parabolic coordinates $(\check{\xi}, \check{\eta})$ are transformed to computational space (ξ, η) using hyperbolic stretching functions designed to cluster grid points near the leading edge and in the boundary layer in a manner similar to Collis (1997). The stretching function in the ξ direction is

$$\check{\xi} = \frac{\check{\xi}_{max}}{\Xi}\left\{c_m \xi + \log\left(\frac{\cosh\left[b\left(\xi - \xi_c\right)\right]}{\cosh\left[b\left(\xi + \xi_c\right)\right]}\right)\right\}, \tag{33}$$

where $\check{\xi}_{max}$ is the maximum $\check{\xi}$ location and ξ_c the location of stretching transition (between fine and coarse grid) in computational space. b is a stretching parameter.

$$\Xi = c_m + \log\left(\frac{\cosh\left[b\left(1 - \xi_c\right)\right]}{\cosh\left[b\left(1 + \xi_c\right)\right]}\right) \tag{34}$$

and

$$c_m = \frac{2b\tanh(b\,\xi_c) + \dfrac{(N_\xi - 1)\Delta\check{\xi}_{min}}{\check{\xi}_{max}}\log\left(\dfrac{\cosh\left[b\left(1 - \xi_c\right]\right.}{\cosh\left[b\left(1 + \xi_c\right)\right]}\right)}{1 - \dfrac{(N_\xi - 1)\Delta\check{\xi}_{min}}{\check{\xi}_{max}}} \tag{35}$$

with N_ξ as the number of grid points in the ξ direction and $\Delta\check{\xi}_{min}$ as the minimum desired grid spacing in $\check{\xi}$. The stretching in the η direction is similar

but with $\check{\eta} - r^2$ instead of $\check{\xi}$ and $\check{\eta}_{max} - r^2$ instead of $\check{\xi}_{max}$. A typical mesh is shown in figure 2.

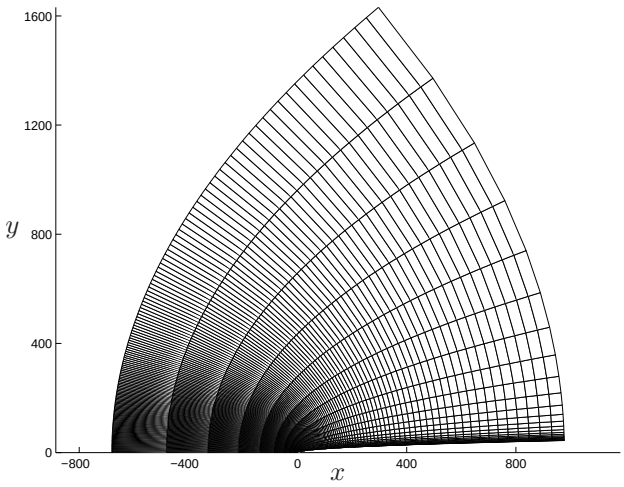

FIGURE 2. A typical mesh used in flow past parabolic body simulations. Each second grid line is shown.

The boundary condition formulation (9)–(14) is used with the modification

$$\frac{\partial \tilde{v}(\xi_0, \eta, t)}{\partial \xi} = 0 \tag{36}$$

in (11) in order to account for the symmetry condition at $\xi_0 = \check{\xi}_0 = 0$. $\tilde{u}_{\text{west}} = 0$ for symmetry and $\tilde{u}_{\text{south}} = \tilde{v}_{\text{south}} = 0$ from the no slip condition at the body. The stream function for potential flow past a parabolic body is

$$\Psi = \check{\xi}\left(\check{\eta} - r^2\right), \tag{37}$$

see Davis (1972). The inflow velocity components are therefore based on potential flow theory and are found to be

$$\tilde{u}_{\text{north}}(\check{\xi}) = \frac{\check{\xi}}{\sqrt{\check{\xi}^2 + \check{\eta}_1^2}}, \qquad \tilde{v}_{\text{north}}(\check{\xi}) = \frac{r^2 - \check{\eta}_1}{\sqrt{\check{\xi}^2 + \check{\eta}_1^2}}. \tag{38}$$

3.2. *Results*

Steady flow over an infinitely long parabolic body is considered in Davis (1972). The infinite conformal space is mapped to a finite domain and a parabolic approximation is used in vorticity-stream function formulation. Davis presents the results in the variable

$$\omega_D \equiv -\frac{2x+1}{\sqrt{2xR}}\omega_w \tag{39}$$

where ω_w is the wall vorticity. The streamwise coordinate direction is scaled as

$$\xi_D \equiv \sqrt{2xR} \tag{40}$$

The results by Haddad & Corke (1998) showed excellent agreement with Davis (1972) using a $N_\xi \times N_\eta = 500 \times 36$ grid. Both results will be considered here as exact solutions. The Reynolds number is $R = 1000$ based on the nose radius of curvature $r = 1$ and the free-stream velocity $U_\infty = 1$ in all computations presented here. The initial conditions are zero velocity and pressure fields in all computations. The integration of the flow to a steady state solution can be interpreted as a instantaneously starting parabolic body. Thus, the flow field should have physical relevance also during the transient phase to the steady state solution.

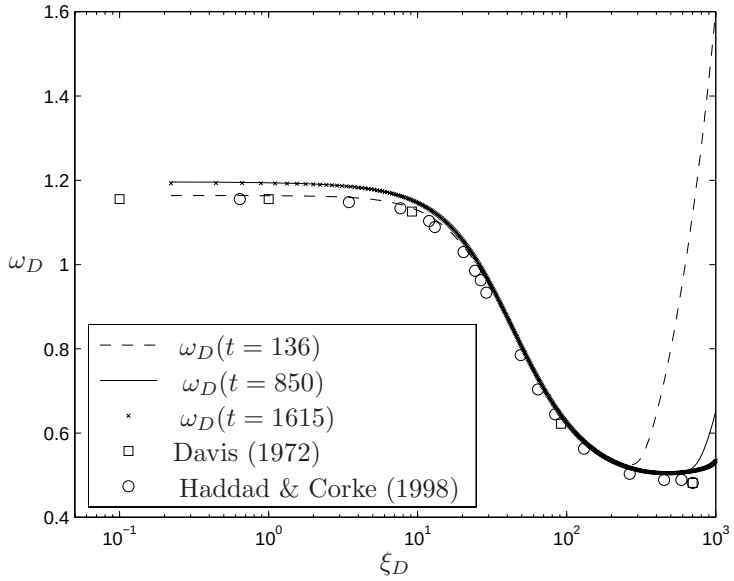

FIGURE 3. Scaled wall vorticity, ω_D at different times in comparison with results from Davis (1972) and Haddad & Corke (1998). The grid is 490×40.

Figure 3 shows the result from a $N_\xi \times N_\eta = 490 \times 40$ simulation with the trailing edge of the body located at $x = 2000$. $(\Delta\check{\xi}_{min}, \Delta\check{\eta}_{min}) = (0.007, 0.003)$, $b = 4.0$ and $\xi_c = 0.80$. The symmetry line is extended to the upstream position $x = -150$. The solution at relative early times, here shown for $t = 136$, reproduce the results reported in Davis (1972) and Haddad & Corke (1998) at the leading part of the body. There is an over-prediction of scaled wall vorticity (ω_D) at the trailing edge. At later times, $t = 850$ and $t = 1615$, the boundary layer is forming also at positions further downstream and ω_D at the trailing edge converges towards the steady state results known from the

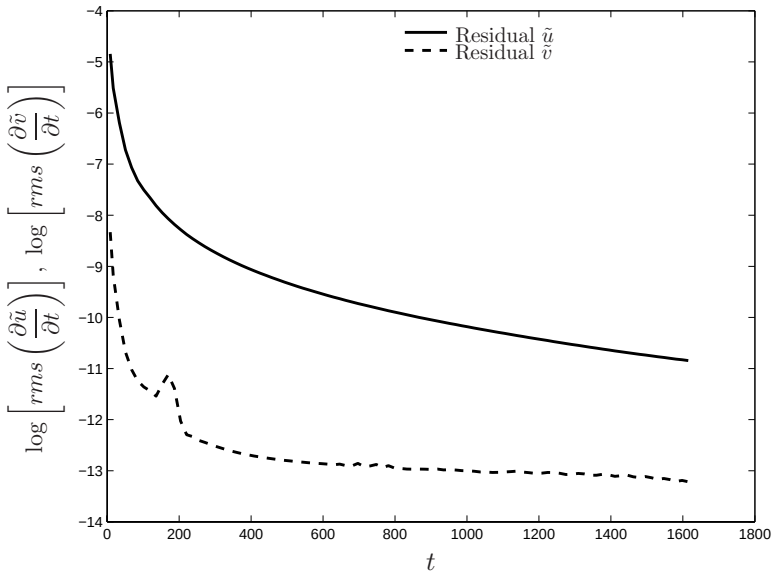

FIGURE 4. Convergence history for steady flow past parabolic body. The grid is 490×40.

literature. However, there is a long time effect in the flow which makes ω_D to increase at the leading edge. This could be a consequence of information propagation from the artificial boundaries in the finite truncated domain.

In figure 4 the convergence history of the 490×40 simulation can be seen. It shows the logarithm of the root-mean-square time derivative of the local velocity components. For a function f it is calculated as

$$\log\left[rms\left(\frac{\partial f}{\partial t}\right)\right] = \log\left[\sum_{i,j}\left(\frac{\Delta f_{i,j}}{\Delta t}\right)^2 / N\right], \tag{41}$$

where N is the number of grid points. Isocontours of the pressure can be seen in figure 5.

In figure 6 the inflow boundary is moved further away from the body in order to investigate the effect of the inflow potential flow. The grid is 300×70 and the inflow boundary has its leftmost point at $x = -800$ and the trailing edge is located at $x = 1000$. The minimum mesh spacing in the $(\check{\xi}, \check{\eta})$ space is $(\Delta\check{\xi}_{min}, \Delta\check{\eta}_{min}) = (0.0033, 0.004)$, the stretching parameter $b = 3.0$ and the location of stretching transition is $\xi_c = 1.0$. The long time effect with increasing ω_D at the leading edge is still present and in fact even worse in this configuration.

The location of the trailing edge of the body is of primary importance and figure 7 uses the trailing edge position $x = 10000$ on a coarse 150×40 grid. The inflow boundary is located at $x = -500$, $(\Delta\check{\xi}_{min}, \Delta\check{\eta}_{min}) = (0.05, 0.007)$,

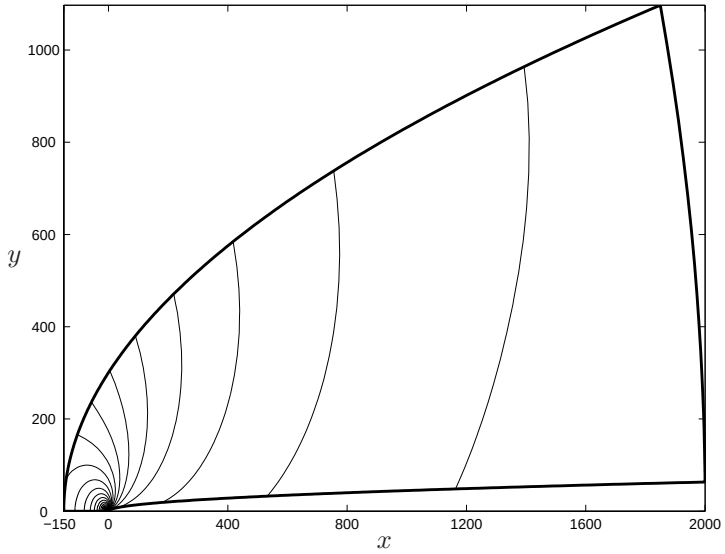

FIGURE 5. Pressure contours for flow past parabolic body. The grid is 490×40 and the contour level spacing is 0.0108.

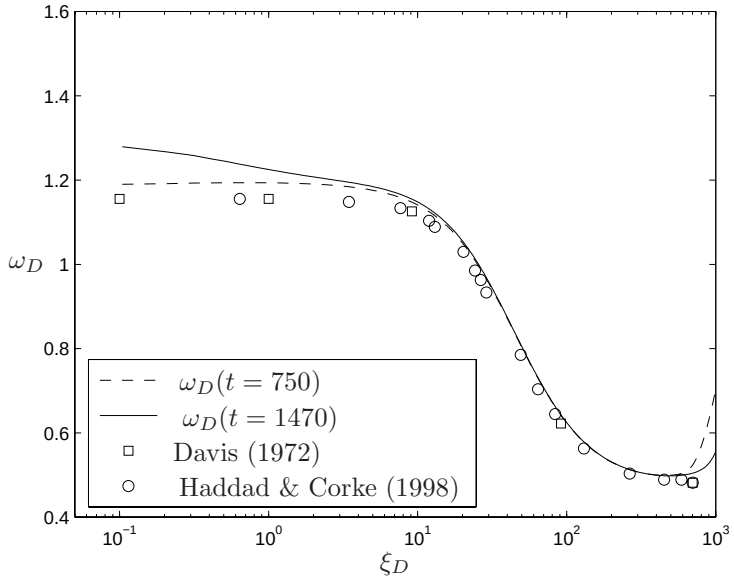

FIGURE 6. Scaled wall vorticity, ω_D at different times in comparison with results from Davis (1972) and Haddad & Corke (1998). The grid is 300×70.

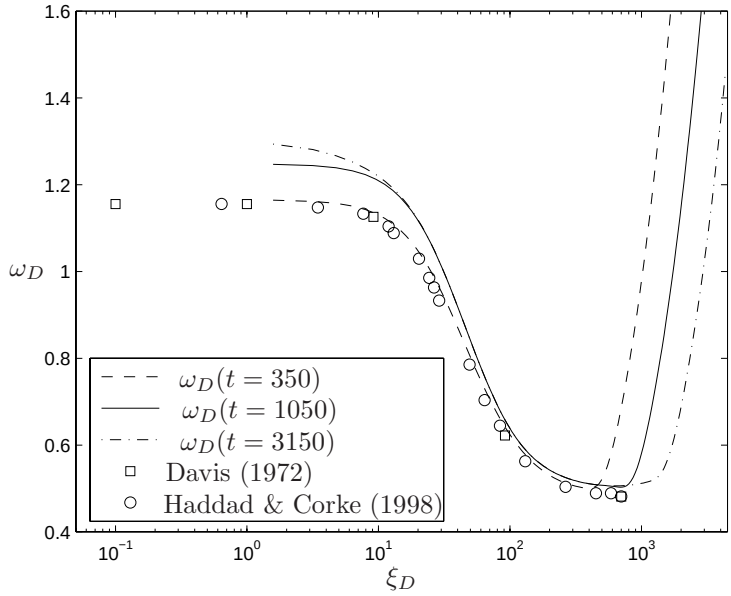

FIGURE 7. Scaled wall vorticity, ω_D at different times in comparison with results from Davis (1972) and Haddad & Corke (1998). The grid is 150×40.

$b = 3.0$ and $\xi_c = 1.0$. As can be seen in figure 7 the long time effect at the leading edge is present also for larger downstream domains.

3.3. *Discussion*

The 490×40 case shown in figure 3 is comparatively closest to the reference cases among the results presented here. It is also the case with smallest desired grid spacing in $\check{\eta}$. Smaller values of $\Delta\check{\eta}$ has been tested (on smaller domains in order to limit the computational costs) without further improvment with respect to the reference cases. In addition, the 490×40 case is the simulation with largest number of grid points in the ξ direction. This indicates that long domains together with relatively high resolution in the ξ direction is important to achieve results close to the reference cases. The increase in ω_D close to the leading edge occurs later in time for domains with large distances to the in- and outflow boundaries. It is therefore suggested that the increase is an effect from both the in- and outflow boundary conditions.

The choice of outflow boundary conditions is not trivial since the viscously dominated boundary layer intersects the boundary. Instead of using Neumann boundary conditions on the velocity components on the outflow boundary the following suggestion for convective outflow conditions could be used. Define $N_{\xi_{\text{mod}}}$ and $N_{\eta_{\text{mod}}}$ to be the parts of the nonlinear terms in (1) that are convected by $\tilde{u}$. The convective boundary condition then reads

$$\frac{\partial \tilde{u}(\xi_1,\eta,t)}{\partial t} + N_{\xi_{\mathrm{mod}}}(\xi_1,\eta,t) = 0, \tag{42}$$

$$\frac{\partial \tilde{v}(\xi_1,\eta,t)}{\partial t} + N_{\eta_{\mathrm{mod}}}(\xi_1,\eta,t) = 0\,. \tag{43}$$

The modified nonlinear terms are extrapolated from time levels n and $n-1$ to level $n+1$ in the same manner as in (4) and the time derivatives are approximated using BDF-2. The velocity components at the outflow boundary are then prescribed as

$$\tilde{u}^{n+1} = \frac{4}{3}\tilde{u}^n - \frac{1}{3}\tilde{u}^{n-1} + \frac{2}{3}\Delta t\left(N^{n-1}_{\xi_{\mathrm{mod}}} - 2N^{n}_{\xi_{\mathrm{mod}}}\right), \tag{44}$$

$$\tilde{v}^{n+1} = \frac{4}{3}\tilde{v}^n - \frac{1}{3}\tilde{v}^{n-1} + \frac{2}{3}\Delta t\left(N^{n-1}_{\eta_{\mathrm{mod}}} - 2N^{n}_{\eta_{\mathrm{mod}}}\right). \tag{45}$$

A buffer zone technique can be used to prevent reflections from the outflow boundary to carry signals back to the domain, see the discussion in the next section.

Regarding the steady state solution further convergence acceleration can be achieved with local time stepping, i.e. the time step is a function of (x, y, t). An estimate of how to compute the local time step can be found in Brüger *et al.* (2005).

In order to solve flow past parabolic bodies with angles of attack both the lower and upper half of the body has to be considered, i.e. no symmetry condition should be used. Then it may be necessary to move the integrals in the boundary condition formulation to either the body or the inflow boundary. The reason for this can be explained by considering the case with zero angle of attack (and no symmetry line). This flow should still retain symmetry but there is an obvious symmetry break in the formulation if one of the outflow boundaries is used as the boundary where the pressure is specified in one grid point. An external flow configuration where the integral formulation is used on the obstacle can be found in Stålberg *et al.* (2004).

4. Flow in asymmetric diffuser

The flow in a plane asymmetric diffuser is interesting since diffuser flows appear in many applications. Several experimental and numerical studies exist, see for example Gullman–Strand *et al.* (2004), Ohta *et al.* (2003), Kaltenbach *et al.* (1999) and Obi *et al.* (1993). A DNS of turbulent flow in an asymmetric diffuser geometry using Reynolds numbers of interest is computational expensive and need highly optimized codes running for long times on high performance computers. In this study a low Reynolds number is used on a single workstation in order to examine the performance of the present method.

4.1. *Configuration*

The configuration for flow in a plane asymmetric diffuser is shown in figure 8. The flow is going from left to right and exhibits an adverse pressure gradient.

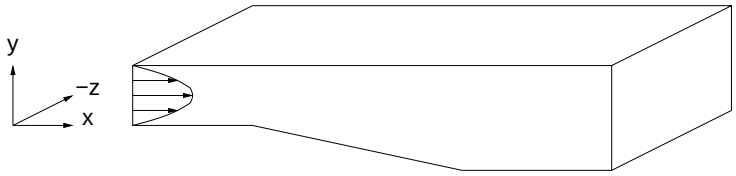

FIGURE 8. The plane asymmetric diffuser configuration.

The conformal map between physical and computational space for this geometry is not known and a numerical conformal mapping is applied. It is based on numerical computation of the Schwarz-Christoffel transformation and a MATLAB toolbox for this mapping is used, see Driscoll & Trefethen (2002). Before applying the conformal map the computational space is stretched in the ξ and η directions in order to cluster cells at the channel inflow part and in the boundary layers. In the ξ direction, a modification of a stretching used in Zhang & Ko (1996) is applied,

$$\check{\xi} = \xi_{\min} + (\xi_{\max} - \xi_{\min}) \left\{ \kappa\, \xi + (1-\kappa) \left[1 - \frac{\tanh\left(\tau\,(1-\Lambda)\right)}{\tanh\left(\tau\right)} \right] \right\}, \qquad (46)$$

with

$$\Lambda = \frac{\xi - \xi_{\min}}{\xi_{\max} - \xi_{\min}}\,. \qquad (47)$$

$\xi_{\min}$ and $\xi_{\max}$ are the minimum and maximum ξ locations. κ controls the stretching position in ξ and τ the rate of stretching. $\kappa = 0.50$ and $\tau = 4.0$ in this study. In the η direction a hyperbolic-tangent function is applied, see Gullbrand (2000),

$$\check{\eta} = \frac{1}{2} \left\{ 1 - \frac{\tanh\left[c \left(1 - 2\dfrac{\eta}{\eta_{\max}} \right) \right]}{\tanh\left(c\right)} \right\}, \qquad (48)$$

where $\eta_{\max}$ is the maximum η location and c a stretching parameter. $c = 2.1$ in this study.

The resulting orthogonal grid, which is invariant in the spanwise direction, is shown in figure 9 for each second grid line. Remark that three grids has to be constructed in order to achieve a staggered grid arrangement. The opening angle is $8.5\,^{\circ}$, the grid size $(151 \times 41 \times 32)$ in the x, y and z directions and the geometry has a polygonal structure. The spanwise box length is here chosen to be $z_L = \pi$.

The boundary conditions are set as in (23)–(31) with $\tilde{u}_{\text{south}} = \tilde{v}_{\text{south}} = w_{\text{south}} = \tilde{u}_{\text{north}} = \tilde{v}_{\text{north}} = w_{\text{north}} = 0$ from the no slip condition. For the inflow boundary conditions, two different cases are considered. The first is a Poiseuille profile and will, for low enough Reynolds numbers, result in a laminar flow field without spanwise variation. In order to achieve a three-dimensional

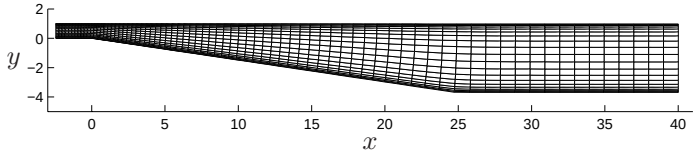

FIGURE 9. The asymmetric diffuser grid generated with numerical conformal mapping. Each second grid line is plotted. The actual grid size is $(N_x \times N_y \times N_z) = (151 \times 41 \times 32)$.

flow field, the Poiseuille profile is perturbed by two oblique Tollmien-Schlichting waves at the inflow. The Dirichlet boundary conditions on "west" in (23)–(31) are then set to

$$\tilde{u}_{\rm west}(\eta, z, t) = \tilde{u}_{\rm po}(\eta) + \epsilon\, \tilde{u}_{\rm ots}(\eta, z, t)\,, \tag{49}$$

$$\tilde{v}_{\rm west}(\eta, z, t) = \epsilon\, \tilde{v}_{\rm ots}(\eta, z, t)\,, \tag{50}$$

$$w_{\rm west}(\eta, z, t) = \epsilon\, w_{\rm ots}(\eta, z, t)\,, \tag{51}$$

$$\tag{52}$$

where $\tilde{u}_{\rm po}$ is the Poiseuille profile and index "ots" label two oblique Tollmien-Schlichting waves. $\epsilon \ll 1$ and the addition of two oblique Tollmien-Schlichting waves at $\xi = \xi_0$, here shown for the $\tilde{u}$ component, is

$$\tilde{u}_{\rm ots}(\eta, z, t) = \text{REAL}\left\{ \breve{u}(\eta) e^{i(\beta z - \omega t)} + \breve{u}(\eta) e^{i(-\beta z - \omega t)} \right\} = \tag{53}$$

$$= 2\cos(\beta z)\left[\, \breve{u}_R(\eta)\cos(\omega t) + \breve{u}_I(\eta)\sin(\omega t) \,\right]\,, \tag{54}$$

where $\breve{u}$ is the amplitude, β the spanwise wavenumber and ω the frequency of the waves. Indices R and I label the real and imaginary parts of a function.

4.2. *Results*

Results from simulations with a Poiseuille inflow profile are shown in figures 10 and 11. Isocontours of instantaneous velocities and pressure are shown in figure 10. Unsteadiness induced by laminar flow separation occurs on both the upper and lower walls and large vortical structures are convected through the domain. The Reynolds number is $R = 550$ based on the velocity scale $U = 1$ and the half channel height $h = 0.5$. Figure 11 shows isocontours of mean streamwise velocity. The mean is taken over a time period $\Delta t = 89$. The white region indicates flow in negative x direction. A separation bubble appears quite far downstream on the inclined wall and relatively surprisingly, a large separated region is also present on the upper straight wall.

Results from a simulation with a Poiseuille profile perturbed by two Tollmien-Schlivhting waves can be seen in figure 12. The frequency of the waves is $\omega = 0.4$ with spanwise wavenumber $\beta = 2$ and the Reynolds number is $R = 225$.

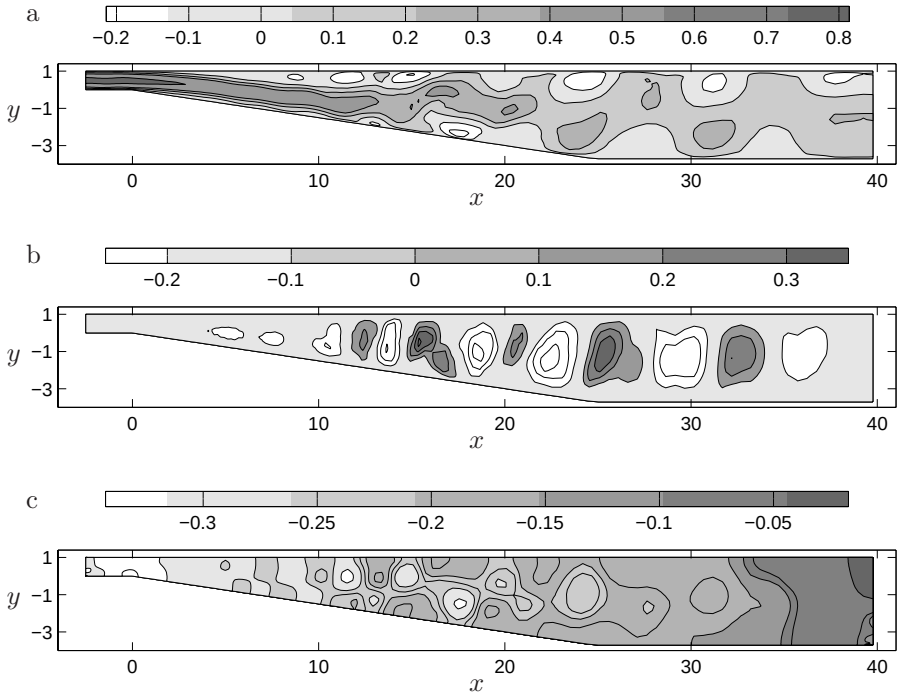

FIGURE 10. Asymmetric diffuser flow with Poiseuille inflow profile. a: Isocontours of instantaneous streamwise (x) velocity. b: Isocontours of instantaneous wall normal (y) velocity. c: Isocontours of instantaneous pressure. The Reynolds number is $R = 550$ based on $U = 1.0$ and the half channel height $h = 0.5$.

The isosurfaces in figure 12 represents perturbations from the spanwise mean for each velocity component. Three dimensional perturbation structures are convected through the domain and are slightly damped due to the low Reynolds number.

4.3. *Discussion*

In this study the velocity components are prescribed by Neumann conditions on the outflow boundary. In Kaltenbach *et al.* (1999) and Ohta *et al.* (2003) convective conditions of the form

$$\frac{\partial u_i}{\partial t} + U_c \frac{\partial u_i}{\partial x} = 0 \tag{55}$$

are used. U_c is the streamwise velocity integrated across the outflow plane. An alternative to convective boundary conditions is to use non-reflective boundary conditions such as buffer zone or mixed Neumann/viscous-sponge techniques,

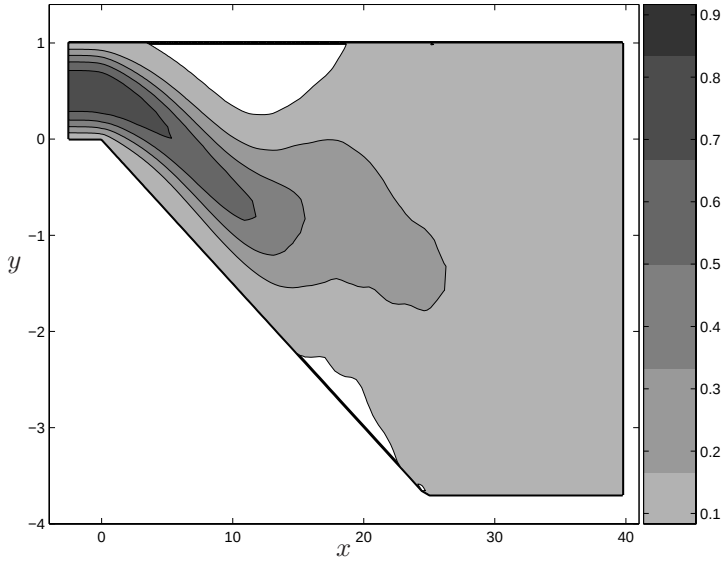

FIGURE 11. Asymmetric diffuser flow with Poiseuille inflow profile. Contours of mean streamwise (x) velocity. The white area represents flow in the negative x-direction. Remark that the axis scaling are not equal. The Reynolds number is $R = 550$ based on $U = 1.0$ and the half channel height $h = 0.5$.

see Karniadakis & Triantafyllou (1992). A method similar to the technique of Street & Macaraeg (1989) can eventually be used as buffer zone close to the outflow boundary. This technique allows large vortical disturbances to convect out of the domain without reflections, see Mittal & Balachandar (1996). It relies on the fact that the source of reflections from the outflow boundary stems from the elliptic nature of the equations. By smoothly attenuating the ξ elliptic terms in the momentum equations (2) and the source term in the Poisson like system for the pressure (with system matrix Q, equation 21) to zero in the buffer region, the ellipticity in the ξ direction can be removed. However, it is not clear how this technique will work together with the boundary conditions formulation (23)–(31).

For turbulent simulations the inflow boundary conditions should preferably be extracted from a fully developed turbulent simulation in a straight channel. The wide range of space and time scales in a turbulent asymmetric diffuser simulation requires long integration times with high resolution in order to obtain converged statistics and therefore put high demands on both the efficiency of the code and the performance of the computer.

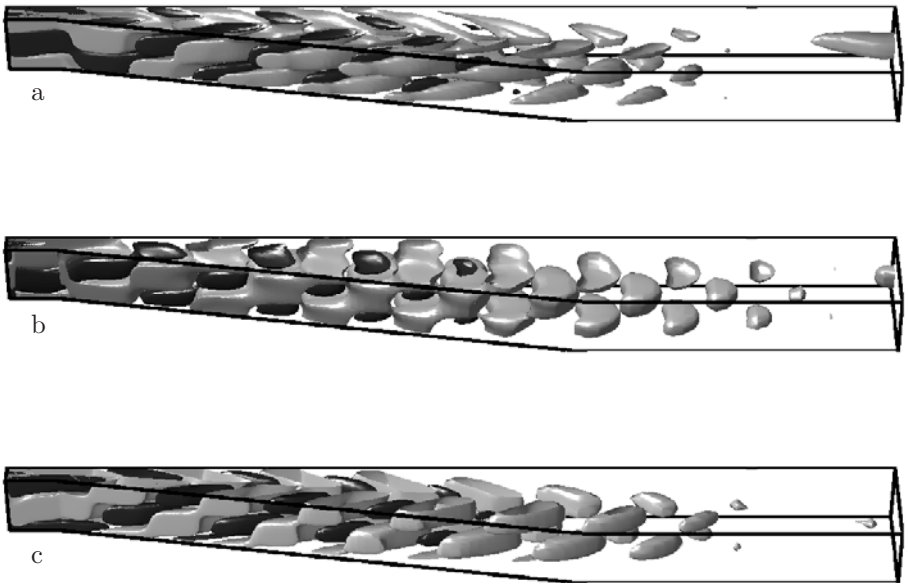

FIGURE 12. Asymmetric diffuser flow. Poiseuille profile perturbed by two oblique Tollmien-Schlichting waves is used as inflow boundary conditions. a: Isosurfaces of instantaneous streamwise (x) perturbations. Dark isosurfaces represents $-4.5 \cdot 10^{-3}$ and light $5.0 \cdot 10^{-4}$. b: Isosurfaces of instantaneous wall normal (y) perturbations. Dark isosurfaces represents $-2.0 \cdot 10^{-3}$ and light $1.0 \cdot 10^{-4}$. c: Isosurfaces of instantaneous spanwise perturbations. Dark isosurfaces represents $-4.0 \cdot 10^{-3}$ and light $3.5 \cdot 10^{-4}$. The Reynolds number is $R = 225$ based on $U = 1.0$ and the half channel height $h = 0.5$.

5. Conclusions

A high order method consisting of compact high order difference operators in a two dimensional curvilinear domain and Fourier series in the third homogeneous direction has been applied to flow past a parabolic body and in a plane asymmetric diffuser. A well posed formulation of the boundary conditions for the incompressible Navier–Stokes equations in primitive variables is used.

Steady state solutions for flow past a parabolic body on different sizes of the domain and with different resolutions are compared with similar numerical experiments known from the literature. Our solution diverge from the reference cases and it is suggested that this is an effect of the artificial boundary conditions.

An orthogonal asymmetric diffuser grid is generated with a numerical conformal mapping. A two dimensional solution is obtained with a Poiseuille

inflow profile and unsteady laminar separation occurs. In order to obtain three dimensional effects the inflow profile is slightly perturbed and as a result three-dimensional vortical structures are convected through the domain.

Suggestions regarding future improvements of the method are done in connection with the two numerical experiments.

Acknowledgements

The authors wish to thank Professor Arne Johansson and Professor Dan Henningson for many relevant advises during the numerical experiments. Professor Per Lötstedt and Professor Bertil Gustafsson from Department of Information Technology, Uppsala University and Doctor Jonas Nilsson from Department of Petroleum Engineering, Stanford University are acknowledged for their ideas regarding time step restrictions and boundary conditions.

References

Brüger, A. 2004 A hybrid high order method for simulation of turbulent flow in complex geometries. PhD Thesis, Dept. of Mechanics, KTH, Stockholm, Sweden.

Brüger, A., Gustafsson, B., Lötstedt, P. & Nilsson, J. 2005 High order accurate solution of the incompressible Navier-Stokes equations. *J. Comput. Phys.* **203**, 49–71.

Brüger, A., Gustafsson, B., Lötstedt, P. & Nilsson, J. 2004 Splitting methods for high order solution of the incompressible Navier-Stokes equations in 3D. *To appear in Int. J. Numer. Meth. Fluids* .

Brüger, A., Nilsson, J. & Kress, W. 2002 A compact higher order finite difference method for the incompressible Navier-Stokes equations. *J. Sci. Comput.* **17**, 551–560.

Brüger, A., Stålberg, E., Nilsson, J., Kress, W., Gustafsson, B., Lötstedt, P., Johansson, A. V. & Henningson, D. S. 2005 A hybrid high order method for incompressible flow in complex geometries / version 2. *Tech. Rep.* KTH/MEK/TR–05/06–SE. KTH Department of Mechanics, Stockholm.

Buter, T. A., & Reed, H. L. 1994 Boundary layer receptivity to free-stream vorticity. *Phys. Fluids.* **6**, 3368–3379.

Canuto, C., Hussaini, M. Y., Quarteroni, A. & Zang, T. A. 1988 *Spectral Methods in Fluid Dynamics*. New York: Springer.

Collis, S. S. 1997. A computational investigation of receptivity in high–speed flow near a swept leading–edge. PhD Thesis, Dept. of Mechanical Engineering, Stanford University.

Collis, S. S., & Lele, S. K. 1999 Receptivity to surface roughness near a swept leading edge. *J. Fluid Mech.* **380**, 141–168.

Davis, R. T. 1972 Numerical solution of the Navier–Stokes equations for symmetric laminar incompressible flow past a parabola. *J. Fluid Mech.* **51**, 417–433.

Driscoll, T. A. & Trefethen, L. N. 2002 *Schwartz-Christoffel Mapping*. Cambridge Univ. Press.

Erturk, E., Haddad, O. M. & Corke, T. C. 2004 Laminar incompressible flow past parabolic bodies at angles of attack. *AIAA J.* **42**, 2254–2265.

Fischer, P. F., Kruse, G. W. & Loth, F. 2002 Spectral element methods for transitional flows in complex geometries. *J. Sci. Comput.* **17**, 81–98.

Fornberg, B. & Ghrist, M. 1999 Spatial finite difference approximations for wave-type equations. *SIAM J. Numer. Anal.* **37**, 105–130.

Gullbrand, J. 2000 An evaluation of a conservative fourth order dns code in turbulent channel flow. *CTR, Annual Research Briefs* pp. 211–218.

Gullman–Strand, J., Törnblom, O., Lindgren, B., Amberg, G. & Johansson, A. V. 2004 Numerical and experimental study of separated flow in a plane asymmetric diffuser. *Int. J. Heat Fluid Flow* **25**, 451–460.

Gustafsson, B. & Nilsson, J. 2000 Boundary conditions and estimates for the steady Stokes equations on staggered grids. *J. Sci. Comput.* **15**, 29–54.

Haddad, O. M., & Corke, T. C. 1998 Boundary layer receptivity to free–stream sound on parabolic bodies. *J. Fluid Mech.* **368**, 1–26.

Karniadakis, G. EM. & Triantafyllou, G. S. 1992 Three-dimensional dynamics

and transition to turbulence in the wake of bluff objects. *J. Fluid Mech.* **238**, 1–30.

Kress, W. & Nilsson, J. 2003 Boundary conditions and estimates for the linearized Navier-Stokes equations on a staggered grid. *Computers & Fluids* **32**, 1093–1112.

Kaltenbach, H.-J., Fatica, M., Mittal, R., Lund, T. S. & Moin, P. 1999 Study of flow in a planar asymmetric diffuser using large-eddy simulation. *J. Fluid Mech.* **390**, 151–185.

Lele, S. K. 1992 Compact finite difference schemes with spectral-like resolution. *J. Comput. Phys.* **103**, 16–42.

Meijerink, J. A. & van der Vorst, H. A. 1977 An iterative solution method for linear systems of which the coefficient matrix is a symmetric m-matrix. *Math. Comp.* **31**, 148–162.

Mittal, R. & Balachandar, S. 1996 Direct numerical simulation of flow past elliptic cylinders. *J. Comput. Phys.* **124**, 351–367.

Obi, S., Ohimuzi, H., Aoki, K. & Masuda, S. 1993 Turbulent separation control in a plane asymmetric diffuser by periodic perturbatins. *Turbulence, Heat and Mass Transfer 4* **10**, 441–448.

Ohta, T., Kajishima, T., Fujii, S. & Nakagawa, S. 2003 Direct numerical simulation of turbulent separating flow in an asymmetric plane diffuser. In *Engineering Turbulence Modeling and Experiments 2*. Elsevier.

Perot, J. B. 1993 An analysis of the fractional step method. *J. Comput. Phys.* **108**, 51–58.

Street, C. L. & Macaraeg, M. 1989 Spectral multi-domain for large-scale fluid dynamic simulations. *Appl. Numer. Math.* **6**, 123–139.

Stålberg, E., Brüger, A., Lötstedt, P., Johansson, A. V. & Henningson, D. S. 2004 High order accurate solution of flow past a circular cylinder. *Accepted to J. Sci. Comput* .

van der Vorst, H. A. 1992 Bi-cgstab: A fast and smoothly converging variant of bi-cg for solution of nonsymmetric systems. *J. Sci. Stat. Comput.* **13**, 631–644.

van Dyke, M. 1962 Higher approximations in boundary–layer theory. *J. Fluid Mech.* **14**, 161–177.

Visbal, M. R. & Gaitonde, D. V. 2002 On the use of higher-order finite difference schemes on curvilinear and deforming meshes. *J. Comput. Phys.* **181**, 155–185.

Zhang, H. L. & Ko, N. W. M. 1996 Numerical analysis of incompressible flow over smooth and grooved circular cylinders. *Computers & Fluids* **25**, 263–281.