Chapter 1

Introduction

1.1 Computational Fluid Dynamics

The flow of fluids plays an important role in our lives. For instance the airflow around a car, airplane or building, the flow of gas, water or oil through pipelines, the flow of water in rivers and oceans and the flow in the atmosphere. To optimize the form of a car, building or airplane, to know the influence of a dam in a river on the water flow or to predict the weather, it is important to have a good insight in the flow patterns.

There are several ways to obtain information about these flows. The first way is through experiments. One can obtain data from measurements on real life situations or from measurements on scale models. In the latter case one can think of experiments in wind tunnels. A big disadvantage of such experiments is that they are very expensive. The second way is to obtain theoretical information about flows. From physical conservation laws differential equations describing the flow can be derived. Only in very simplified cases these equations can be solved exactly. Nowadays, as a consequence of the increase in computer power and the improvement of numerical algorithms these equations can be solved numerically. The field in which these numerical simulations are performed is called computational fluid dynamics (CFD). With CFD a large variety of simulations can be performed. The design of a car or aircraft can be easily changed in a computer simulation. Furthermore, the features of the fluid, like density or viscosity, can be changed easily.

The first step in CFD is the mathematical modeling of the flow. The motion of an incompressible fluid can be described by the time-dependent incompressible Navier-Stokes equations

$$\begin{aligned} &\nabla.\vec{u} = 0, \\ &\frac{\partial \vec{u}}{\partial t} + (\vec{u}\ .\nabla)\vec{u} - \nu\Delta\vec{u} + \nabla p = 0, \end{aligned}$$

with $\vec{u}$ the velocity, p the pressure and ν the viscosity of the fluid.

In general it is not possible to solve the Navier-Stokes equations exactly. Therefore, the second step in CFD is to transform the mathematical model into a numerical model. The equations will be discretized in both time and space. A grid covering the domain

is generated and the spatial derivatives of the Navier-Stokes equations are discretized on this grid.

If one is interested in steady-state solutions one can choose between solving the time-independent and the time-dependent equations. The obvious advantage of solving the time-independent equations is that no time steps are needed. When solving the time-dependent equations either an explicit or an implicit time-integration method can be used. For computing steady states an implicit method is preferable; such a method allows larger time steps than an explicit method.

A popular approach to solve the equations is with a pressure-correction method. In this approach the continuity equation is decoupled from the momentum equation. For the stationary equations this method is as follows. Assume an estimate for the pressure $p^{(k)}$ is known. Then the estimate for the velocity field $\vec{u}^{\,(k)}$ can be solved from

$$(\vec{u}^{\,(k)}.\nabla)\vec{u}^{\,(k)} - \nu\Delta\vec{u}^{\,(k)} = -\nabla p^{(k)}.$$

The resulting velocity field is in general not divergence free. Therefore, a pressure correction δp and a corresponding velocity correction $\delta\vec{u}$ are introduced. These corrections have to satisfy $((\vec{u}^{\,(k)} + \delta\vec{u}\,).\nabla)(\vec{u}^{\,(k)} + \delta\vec{u}\,) - \nu\Delta(\vec{u}^{\,(k)} + \delta\vec{u}\,) = -\nabla(p^{(k)} + \delta p)$. From these equations $\delta\vec{u}$ is solved approximately: $\delta\vec{u} = -\nabla\delta p$. Demanding $\nabla.(\vec{u}^{\,(k)} + \delta\vec{u}\,) = 0$, leads to

$$\nabla.\nabla\delta p = -\nabla.\vec{u}^{\,(k)}.$$

The new estimate for the pressure is given by

$$p^{(k+1)} = p^{(k)} + \delta p.$$

The disadvantage of a pressure-correction method is that although steady solutions are computed, time-like stepping is needed. For implicit time-integration a pressure-correction method can be applied as well.

An alternative approach is to solve the continuity and momentum equations in a coupled way. The discretized equations are put together in a very large system. To solve such large systems fast linear solvers are needed. The coupled approach has become feasible due to the increase of computer power and the development of fast algorithms for solving linear systems.

1.2 Continuation methods

A large number of physical problems can be described by partial differential equations which depend on one ore more physical parameters. In fluid dynamics one can think of the influence of the viscosity of the fluid on the flow patterns, or the influence of the temperature difference between the bottom and top of a layer of fluid on its flow. From a theoretical and practical point of view it is of interest to know all solutions and their dependence on the parameter [41]. Of special interest are the stability of a solution (which determines the physical relevance of this solution) and the values of the parameter for which new solutions appear (bifurcation points).

The numerical technique to compute branches of solutions is called continuation; assume that the solution is known up to a certain parameter value, then these solutions are used to start the computation of the solution at the next parameter value. Continuation methods are widely applied to problems with a small number of degrees of freedom (of order 10). Only recently they are applied to large systems (of order 10^5), for instance occurring from the discretization of PDEs describing fluid flows in two or three dimensions. The bottleneck in these computations are the linear solvers and eigenvalue solvers. The increase in computer power and the improvement of numerical algorithms enable us to deal with these large systems.

After semi-discretization an autonomous time-dependent system of nonlinear partial differential equations can be written as

$$B\frac{d}{dt}u(t) = f(u(t), \lambda),$$

with $u(t) \in R^n$ the solution vector, $\lambda \in R$ the physical parameter, $B \in R^{n\times n}$ the matrix representing the time-dependency, and f a nonlinear mapping from $R^n \times R \to R^n$.

Stationary solutions

A method to trace a branch of stationary solutions from the time-independent system $f(u, \lambda) = 0$ will be described. The first step in solving this system is to determine a parametrization γ for the solution branches; $(u(\gamma), \lambda(\gamma))$. One way is to parametrize with the physical parameter. A better way is to use a pseudo-arclength parametrization [28]. An additional equation is needed to establish the parametrization: $n(u, \lambda, \gamma) = 0$. The system that has to be solved is

$$\begin{aligned} f(u, \lambda) &= 0, \\ n(u, \lambda, \gamma) &= 0. \end{aligned}$$

Assume for a certain parameter value the solution at the branch is known. Then, the parameter value is increased, and the solution for the new parameter value is computed with a Euler predictor/Newton corrector method [14]. First a prediction for the new solution is made with Euler's method. Then, the new solution is computed with Newton's method in which the prediction is used as a starting vector. To determine the stability of the solutions and the position of the bifurcation points a generalized eigenvalue problem has to be solved.

Periodic solutions

After a Hopf bifurcation a stationary solution will become periodic. We will describe how to compute these periodic solutions and their stability [41, 31]. For simplicity the system parameter λ is used to parametrize the branch of periodic solutions (avoiding the additional parametrization equation) and the matrix B describing the time-dependency is taken equal to the identity matrix.

A periodic solution and its period T can be obtained from solving

$$\frac{d}{dt}u(t) = f(u(t), \lambda),$$
$$u(T) = u(0).$$

This system can be solved with a shooting method. Start with a guess v for $u(0)$, the periodic solution at $t = 0$, and its period T. A flow $\varphi(v, t, \lambda)$, which depends on the initial condition v, is determined. This flow is described by

$$\frac{d}{dt}\varphi(v, t, \lambda) = f(\varphi(v, t, \lambda), \lambda),$$
$$\varphi(v, 0, \lambda) = v.$$

The flow of a periodic solution has to satisfy

$$\varphi(v, T, \lambda) - v = 0. \tag{1.1}$$

This equation can be solved for v and T by Newton's method. The Jacobian is given by

$$J = (\frac{\partial}{\partial v}\varphi - I, \frac{\partial}{\partial T}\varphi).$$

The matrix $\frac{\partial}{\partial v}\varphi$ is called the monodromy matrix. The eigenvalues of this matrix (the Floquet multipliers) determine the stability of the periodic solution. The monodromy matrix is a dense matrix and expensive to compute.

In order to avoid the computation of the monodromy matrix, the periodic solutions have to be computed in a different way. A periodic solution can be computed by using time integration as a fixed-point process (Picard iteration). The flow over a period T is computed by time-integration. The iterates of the fixed-point process are the successive approximations of $u(0)$; $v_{i+1} = \varphi(v_i, T, \lambda)$. The period T is not known in advance, and has to be determined during the process. For unstable periodic solutions the fixed-point process will not convergence, and for stable periodic solutions it may converge very slowly. To accelerate the convergence or stabilize the process the recursive projection method from Shroff and Keller [42] can be used.

Recursive projection method

The recursive projection method (RPM) [42] can be used to stabilize or accelerate a fixed-point iteration

$$x_{i+1} = g(x_i, \lambda).$$

One can think of a steady-state computation by means of time integration or of detecting a periodic solution (see equation (1.1)). The RPM uses the fact that although the system is of high order the dynamics are only of low order; slow convergence or divergence of the fixed-point process is caused by only a few eigenvalues approaching or leaving the unit disk.

The space is split in two subspaces, the subspace $\mathcal{P}$ belonging to the slowly converging modes and its orthogonal complement $\mathcal{Q} = \mathcal{P}^{\perp}$, by the projections P and $Q = I - P$. The iteration is rewritten as

$$\begin{aligned} p_{i+1} &= p_i + C_1(Pg(x_i, \lambda) - p_i), \\ q_{i+1} &= q_i + C_2(Qg(x_i, \lambda) - q_i), \\ x_{i+1} &= p_{i+1} + q_{i+1}. \end{aligned}$$

In the RPM on the unstable subspace $\mathcal{P}$ Newton's method is applied, i.e. C_1 is minus the inverse of the Jacobian of $Pg(x_i, \lambda) - p_i$. On the stable subspace $\mathcal{Q}$ the fixed-point iteration is continued, i.e. C_2 is the identity matrix.

The Newton process is only applied to the small unstable subspace. For the computation of a periodic solution this means that only the monodromy matrix restricted to this subspace is needed. The stability of the periodic solutions can be obtained from the eigenvalues of the restricted monodromy matrix.

To obtain the projectors P and Q, an orthonormal basis for the unstable subspace $\mathcal{P}$ is needed. In a continuation method such a basis can be constructed easily from the iterates q_i [42]. If in a continuation step a slowly converging mode occurs in the fixed-point process applied to the stable subspace, the basis is expanded. To keep the basis accurate it is adapted in each continuation step.

Lust and Roose [31, 32] developed the Newton-Picard method, a generalization of the RPM. The derivation of the Newton-Picard method is elegant. Starting from a classical shooting method, the above splitting in a stable and unstable subspace is incorporated. Then, a variety of approximations are considered in order to make the computation feasible. In this way several methods result, one of which is the RPM.

1.3 Linear solvers and preconditioning

In fluid flow computations efficient linear solvers are needed. In this section we will consider methods to solve linear systems

$$Ax = b,$$

with the matrix A large and sparse. We will consider two types of solvers: the direct solver and the iterative solver.

With a direct method a complete LU-factorization of the matrix A is computed. By means of Gauss elimination a lower triangular matrix L and an upper triangular matrix U can be constructed such that $A = LU$. Then, the system $Ax = b$ can be solved in two steps: a forward solve $Ly = b$, followed by a backward solve $Ux = y$. Linear systems originating from the discretization of differential equations are sparse. A drawback of the direct method is that although the matrix A is sparse, the matrices L and U are not, due to fill-in during the factorization. Therefore, the storage demands for the matrices L and U are very high. Furthermore, for matrices of large dimension the construction of the LU-factorization is very time consuming.

An alternative is to use iterative methods, e.g. CG, Bi-CGSTAB or GMRES(m), to solve the linear system. The convergence of these methods strongly depends on the

spectrum of the matrix A. The convergence can be improved by transforming the linear system into an equivalent system with the same solution but a better location of its spectrum. A matrix P which performs such a transformation is called a preconditioner. The iterative method will be applied to the preconditioned system

$$P^{-1}Ax = P^{-1}b.$$

The construction and application of the preconditioner will cause additional costs. The improvement of the convergence rate must be such that it makes up for these additional costs. Therefore, the preconditioner P has to satisfy the following properties:

- it is a good approximation of A, i.e. $P^{-1}A$ is close to the identity matrix,
- it is easy to compute, and the system $Py = c$ for given c is easier to solve than the original system,
- the storage demand for P is moderate.

An important class of preconditioners is formed by the incomplete LU factorizations. These preconditioners are obtained by discarding part of the fill-in occurring during the factorization of A. An incomplete LU factorization of A is given by

$$A = LU + R,$$

with the matrix R small in some sense.

1.4 History of ILU factorizations

We will give a short historical review of the development of the ILU factorizations. The roots of these factorizations lie in the 1960's [10, 36, 37], where they were applied to specific problems. In a more general form they were introduced for M-matrices in 1977 [33]. The most important aspects of the ILU factorizations are the dropping of the fill-in and the ordering of the unknowns. The strategy for the dropping and ordering has improved greatly over the years.

The classical approach for dropping is the drop-by-position strategy; the fill-in is allowed only at a prescribed set of positions. A common strategy is to allow only fill-in at the positions of the non-zero elements of A. The index set S of allowable fill-in positions for this case is given by

$$S = \{\ (i,j) \mid a_{ij} \neq 0\ \}.$$

A very efficient implementation of this approach is introduced by Eisenstat [18]. A modification can be applied to the factorization which improves the convergence rate [17, 22]. In these modified ILU (MILU) factorizations the dropped fill-in is added to the diagonal. With this modification the factorization becomes exact for a constant vector. The drop-by-position strategy can be improved by enlarging the set S of allowable fill-in positions, e.g. by allowing more non-zero bands in the L and U matrices. In general it is difficult to determine where to allow fill-in. Therefore, a dropping strategy based on the matrix is preferable.

Basically, two approaches of matrix-dependent dropping can be seen. The first matrix-dependent approach is to use the concept of level of fill [22, 34], resulting in the ILU(k) factorizations. The levels of fill are defined recursively. The positions at which the matrix A has non-zeros belong to level of fill 0. Assume the positions with level of fill $k-1$ are known. Then, the positions of the fills caused by elements at positions of level of fill $k-1$ belong to level of fill k. Allowing fill at positions of at most level of fill k results in an ILU(k) factorization. Usually the level of fill is kept low, the idea behind this is that the higher levels of fill are less important than the lower levels. The second matrix-dependent approach is to use the drop-by-size strategy. Fill-in is dropped if its absolute value is below a certain tolerance. This tolerance can be chosen in different ways. The simplest way is to choose a fixed tolerance independently of the matrix. More advanced ways are to choose the tolerance relative to size of the elements in the original matrix [13] or size of the elements in the factorization [5].

The ordering of the unknowns has a strong influence on the amount of fill-in of the (M)ILU factorizations. The ordering strategies have undergone the same development as the dropping strategies; from a fixed prescribed ordering to an ordering which is matrix dependent and made during the factorization. The repeated red-black ordering is an ordering which is made in advance and does not depend on the matrix. The reverse Cuthill McKee, the Minimum Degree, and the nested dissection ordering are orderings which are made before the factorization is constructed but do depend on the matrix. The effect of various orderings on the preconditioned conjugate gradient method has been studied in [16].

In the matrix renumbering ILU (MRILU) factorization [8] and the multilevel ILU (MLILU) factorization [6] the ordering is determined during the factorization. Moreover, the dropping is based on the magnitude of the fill. This results in very effective preconditioners.

1.5 Outline

In this thesis we will be concerned with the above described techniques for solving fluid flow problems. To solve the Navier-Stokes equations we use the coupled approach. Therefore, efficient preconditioners are needed to solve the systems iteratively. We will consider two types of preconditioners: (M)ILU factorizations with respect to a repeated red-black ordering and matrix renumbering ILU (MRILU) factorizations. The latter preconditioner has been used in two types of continuation methods. The pseudo-arclength continuation method combined with a predictor-corrector method has been applied to the Rayleigh-Bénard problem. A bifurcation analysis of the lid-driven cavity problem has been performed with the Newton-Picard method. The outline of this thesis is as follows.

In Chapter 2 we consider block (M)ILU preconditioners with respect to a repeated red-black ordering (RRB). In this type of preconditioner both the ordering and the positions at which fill-in is dropped are prescribed. The RRB preconditioners are applied in various test cases. The influence of the number of levels and the treatment of the last level on the convergence behaviour is studied.

The performance of the RRB preconditioners is explained theoretically in Chapter 3.

Bounds for the condition number of the preconditioned system are derived. With these bounds the stagnation of the convergence of the iterative method preconditioned with the RRB factorizations in some test cases are explained.

The matrix renumbering ILU (MRILU) factorization is introduced in Chapter 4. The ordering of the unknowns and the positions in which the fill-in is dropped are determined during the factorization based on the sparsity pattern and the size of the coefficients. The MRILU preconditioner is tested on systems occurring from different types of discretization of the Navier-Stokes equations.

In Chapter 5 we use a pseudo-arclength continuation method in which a predictor-corrector method is used to compute the solution branches. To determine the stability of the solutions and the position of the bifurcation points a generalized eigenvalue problem is solved with the Jacobi-Davidson QZ method. As preconditioner the MRILU factorization is used. As test case a bifurcation analysis of the Rayleigh-Bénard problem, described by the Boussinesq equations, is performed. The bifurcation behaviour of the Rayleigh-Bénard problem has already been studied extensively. Therefore, this problem is very well suited to test our numerical techniques.

In Chapter 6 we will perform a bifurcation analysis of the lid-driven cavity problem. We make use of the Newton-Picard method preconditioned with the MRILU factorization. The lid-driven cavity problem is a well known test case for numerical methods for the Navier-Stokes equations. Nevertheless, little is known about the bifurcation behaviour of this problem. The first bifurcation is a Hopf bifurcation at which the stable stationary solutions becomes unstable and a periodic solution appears. In [12] a bifurcation analysis of the two-dimensional lid-driven cavity is performed using a low-dimensional system stemming from a projection of the Navier-Stokes equations on basis functions obtained from a proper orthogonal decomposition. We will repeat this analysis for the full Navier-Stokes equations.

Finally in Chapter 7 an overview of the results is given.