IT Licentiate theses
2002-007

Semi-Toeplitz Preconditioning for Linearized Boundary Layer Problems

SAMUEL SUNDBERG

UPPSALA UNIVERSITY
Department of Information Technology

UPPSALA
UNIVERSITET

Semi-Toeplitz Preconditioning for Linearized Boundary Layer Problems

BY
SAMUEL SUNDBERG

December 2002

DEPARTMENT OF INFORMATION TECHNOLOGY
SCIENTIFIC COMPUTING
UPPSALA UNIVERSITY
UPPSALA
SWEDEN

Dissertation for the degree of Licentiate of Philosophy in Numerical Analysis
at Uppsala University 2002

Semi-Toeplitz Preconditioning for Linearized Boundary Layer Problems

Samuel Sundberg
Samuel.Sundberg@tdb.uu.se

Department of Information Technology
Scientific Computing
Uppsala University
Box 337
SE-751 05 Uppsala
Sweden

http://www.it.uu.se/

ISSN 1404-5117
Printed by Eklundshofs Grafiska AB

Abstract

We have defined and analyzed a semi-Toeplitz preconditioner for time-dependent and steady-state convection-diffusion problems. Analytic expressions for the eigenvalues of the preconditioned systems are obtained. An asymptotic analysis shows that the eigenvalue spectrum of the time-dependent problem is reduced to two eigenvalues when the number of grid points go to infinity. The numerical experiments sustain the results of the theoretical analysis, and the preconditioner exhibits a robust behavior for stretched grids.

A semi-Toeplitz preconditioner for the linearized Navier–Stokes equations for compressible flow is proposed and tested. The preconditioner is applied to the linear system of equations to be solved in each time step of an implicit method. The equations are solved with flat plate boundary conditions and are linearized around the Blasius solution. The grids are stretched in the normal direction to the plate and the quotient between the time step and the space step is varied. The preconditioner works well in all tested cases and outperforms the method without preconditioning both in number of iterations and execution time.

Keywords: Iterative solution, preconditioning, finite difference methods, boundary layer flows.

Contents

Chapter 1

Summary

In order to properly present my research on semi-Toeplitz preconditioning a small context orientation is needed. The field of scientific computing is connected to several other fields in science such as mathematics, computer science, physics, chemistry and the biosciences. Therefore we need to understand the connection to these fields to properly understand the role of numerical methods.

Many different science fields use to some extent mathematical modeling to describe essential features of the objects studied in just a few equations. These equations are elegant since they contain an implicit description of complex interactions in a dense format. They suffer however from the drawback that they generally lack a solution that can be expressed in an explicit formula. In order to be able to predict the behavior of an actual object we need to find ways to solve these equations approximatively. This is done using numerical methods.

Numerical methods have a long and rich history even before the advent of the computer age. It was however the invention of the electronic computer that made these methods interesting for a wider audience. Today scientists use numerical simulation as a research tool as important as theory and experiments. We find computational methods as an engineering tool in the automobile industry to simulate crash tests, they are used in the aeronautical industry to design aero-planes that emits less noise and there exist many other applications as well.

1.1 Numerical context

The research presented here is concerned with iterative methods to solve linear systems of equations. These systems typically arise in scientific computing when we discretize Partial Differential Equations (PDEs) using finite differences or some other discretization method like finite elements or finite volumes. There are of course other applications like circuit theory and signal processing where iterative methods are used to solve systems of equations, but here we have restricted ourselves to the study of iterative methods in a PDE setting.

The use of preconditioning to enhance the performance of iterative solvers in [2, 19] was the turning point that made modern iterative methods widely used in numerical computations. To find an efficient preconditioner for a certain problem might even be more important than to choose the right iterative method. Here we study a preconditioning strategy that is based on knowledge of the origin of the system [25, 24].

In the reports that comprise this licentiate thesis we study the solution of equations that are derived from boundary layer problems in fluid dynamics. The ultimate goal is to make it feasible to use semi-Toeplitz preconditioning in Navier–Stokes solvers, and therefore we make some effort to ensure that the model problems we solve exhibit the same essential features.

The rest of this overview presents some material on iterative methods, preconditioning of these in general and some material on semi-Toeplitz preconditioners.

1.2 Solving linear systems of equations

There are several methods to solve a linear system of equations. The most well-known method is Gaussian Elimination (GE), which is a direct method. For a linear system of equations

$$Ax = b, \tag{1.1}$$

where A is an n-by-n nonsingular matrix, GE requires $2n^3/3$ arithmetic operations as well as storage of n^2 entries in the general case. For the large sparse matrices that often appear in scientific computing this is very inefficient, and we need to use the fact that only a few of the entries in the matrix are nonzero. This is difficult with GE, and other direct methods like frontal, multifrontal and supernodal methods have been developed to handle sparse matrices [4, 5]. Despite large improvements in this area there are still many situations where iterative methods are the only feasible choice, e.g. for discretizations of three-dimensional PDEs.

1.2.1 Krylov subspaces

The idea behind most iterative methods that are in use today is to find the solution in the subspace spanned by successive multiplications with A. Here we use b as a starting vector, but there are cases when a different starting vector is used. We thus find $x_1 \in \text{span}\{b\}$ and then compute the matrix-vector product Ab to find $x_2 \in \text{span}\{b, Ab\}$. At step k in this process we find the approximate solution as

$$x_k \in \text{span}\{b, Ab, \ldots, A^{k-1}b\}. \tag{1.2}$$

This subspace is usually called the *Krylov subspace* for the matrix A using the initial vector b at step k. We use the notation $\mathcal{K}_k(A, b)$ to denote this subspace.

To find the best approximative solution in a given Krylov subspace is not a trivial task. First of all there are several ways to define what criterion an optimal solution should fulfill. Some methods minimize the norm of the residual, $\|b - Ax_k\|$, while other methods find a residual that is orthogonal to the subspace. Secondly there is a choice how much iteration overhead[1] we will allow.

1.2.2 Methods for Hermitian matrices

For real symmetric and complex Hermitian matrices the task is not so difficult, however, as there exist methods with a fixed amount of overhead that generate the optimal solution in some sense. The most well-known method in this class is the *Conjugate Gradient* (CG) method [14]. This is a method that minimizes the A-norm[2] of the error over the subspace. CG requires that the matrix is positive definite in order to be able to compute its three-term recurrence, which is based on LU-decomposition of the Lanczos matrix. For general Hermitian matrices the *Minimum Residual* (MINRES) method [21] generates the approximation which minimizes the residual in the Krylov subspace. The coefficients for the MINRES recurrence are computed by means of Givens rotations.

For CG and MINRES we have sharp estimates of how good approximations these methods yield [7]. The residual r_k at step k generated by

[1] The extra work done in each iteration besides the matrix-vector multiply.

[2] $\|\cdot\|_A$ denotes the A-norm, $\|v\|_A = \sqrt{\langle v, Av \rangle}$

MINRES fulfills

$$\|r_k\|_2/\|r_0\|_2 \leq \min_{p_k} \max_{i=1,\dots,n} |p_k(\lambda_i)|, \tag{1.3}$$

where λ_i is an eigenvalue of A and p_k is any kth-degree polynomial that fulfills $p_k(0) = 1$. A similar error bound exists for CG, where we have

$$\|e_k\|_A/\|e_0\|_A \leq \min_{p_k} \max_{i=1,\dots,n} |p_k(\lambda_i)|, \tag{1.4}$$

see e.g. [8]. From these estimates we conclude that for Hermitian matrices the convergence is entirely determined by the spectrum of the eigenvalues for A.

1.2.3 Methods for non-Hermitian matrices

The situation is much more troublesome when we consider iterative methods for non-Hermitian matrices. Here we do not have any general method that finds the optimal solution with a fixed amount of iteration overhead, as shown by Faber and Manteuffel in [6]. Either we emphasize low iteration overhead and settle for a non-optimal solution, or we aim at finding the optimal solution and have to endure a linearly growing iteration overhead.

We first take a closer look at the *Generalized Minimum Residual*(GMRES) method [22]. This method is based on the Arnoldi orthogonalization method and finds the optimal solution in the Krylov subspace by minimizing the residual. We must however save all previous search directions for the orthogonalization procedure. Therefore we get a linearly growing iteration overhead which prohibits the use of the method when a large number of iterations is needed. To avoid too large overhead the method is sometimes restarted after some fixed number of iterations and is then called GMRES(l), but then the solution is no longer optimal.

There is a number of other approaches that try to find a near-optimal solution using only finite recurrences. Among these are BiCG, CGS, QMR, BiCGSTAB, and variants and hybrids of these. Which method to use is problem-dependent as shown in [20].

Unlike the situation for the Hermitian problems there is no sharp upper bound on the residual in general. There is an estimate of the residual reduction for GMRES given by

$$\|r_k\|_2/\|r_0\|_2 \leq \kappa(V) \min_{p_k(0)=1} \max_{i=1,\dots,n} |p_k(\lambda_i)|. \tag{1.5}$$

Here $\kappa(V) = \|V\| \cdot \|V^{-1}\|$ is the condition number of the eigenvector matrix

V of A and p_k is the k-th residual polynomial for the method. The estimate (1.5) is, however, sharp only for a normal[3] matrix [9].

For anyone interested in knowing more about iterative methods and issues concerning their application we refer to the introductory book [8].

1.3 Preconditioning

When using iterative methods for solving linear systems (1.1), sometimes multiplication by a *preconditioner* M is used to reduce the number of iterations to convergence. We instead solve e.g.

$$M^{-1}Ax = M^{-1}b, \tag{1.6}$$

which in this case is a left preconditioned system. The ultimate goal of preconditioning is that the runtime for the solver should be reduced. This places some limitations on the preconditioner we use. These limitations can be quantified by analyzing the overhead introduced by the preconditioner compared to the reduction in the number of iterations.

We let t_I denote the average time to complete one iteration of the unpreconditioned solver. For the preconditioner we assume that the construction cost is given by t_C, the cost for solving the preconditioner-system each iteration is αt_I and the number of iterations is reduced by a factor $r > 1$. If the unpreconditioned solver converges in k steps, the computing time becomes $T_U = kt_I$. The time for the preconditioned solver is then

$$T_P = \frac{k}{r}(1+\alpha)t_I + t_C.$$

We now get

$$\frac{T_P}{T_U} = \frac{1+\alpha}{r} + \frac{t_C}{T_U}. \tag{1.8}$$

From (1.8) it is obvious that the requirements for a good preconditioner are that

a.) t_C is small, i e the construction of M is cheap.

b.) α is small, meaning the solution of the preconditioner-system $Mx = y$ is cheap compared to the solution of $Ax = b$.

c.) r is large, i e the convergence for the preconditioned system is fast.

[3]a diagonalizable matrix with a complete set of orthonormal eigenvectors.

As an application of equation (1.8) we use the values for the finest grid from the tables in [24] to derive a lower requirement for the value of r in order to profit from using the semi-Toeplitz preconditioner. For T_P/T_U to be less than 1 in (1.8) we need

$$r > \frac{1+\alpha}{1-t_C/T_U}.$$

Use $\alpha = 0.31$, $t_C = 0.091 \cdot 9.2$, and $T_U = 52$ from [24] which gives us the requirement $r > 1.33$. We see that the costs for the preconditioner are small and even with a small reduction in iterations it pays off to use semi-Toeplitz preconditioning.

1.3.1 Application of the preconditioner

There are several ways to introduce preconditioning in an iterative method. We distinguish between left preconditioning, right preconditioning, and two-sided preconditioning. The way the preconditioner is applied does not change the eigenvalue spectrum of the preconditioned system (although it might affect the conditioning of the eigenvector-matrix), but nevertheless there are situations where one form of application is preferred.

Left preconditioning is introduced by applying the iterative method to the system $M^{-1}Ax = M^{-1}b$. We should note that left preconditioning alters the residual which might have an impact on stopping criteria for MINRES and GMRES.

Right preconditioning uses the system $AM^{-1}y = b$ where $x = M^{-1}y$. This approach has the benefit that the right-hand side is not affected by the preconditioning, which might be of use when constructing general application software. For iterative methods with stopping criteria based on the error, $A^{-1}b - x_k$, we have to make sure that the iterative process is not interrupted too early.

Two-sided preconditioning might be applied when the preconditioner can be written in the form $M = M_1M_2$, as e.g. when using incomplete Cholesky factorizations for preconditioning. We then solve the system $M_1^{-1}AM_2^{-1}y = M_1^{-1}b$ where $x = M_2^{-1}y$.

To find a suitable preconditioner is not a trivial task. Although we know what a preconditioner should do there are no general guidelines on how to find a good one. The only concept we have is the rather vague idea that the preconditioner should somehow approximate the inverse of A as we know that for the system $A^{-1}A = I$ convergence is reached in one iteration. In the next section we mention some approaches that have been used.

1.3.2 Some important classes of preconditioners

Different preconditioners can roughly be divided into three different categories. These are preconditioners for certain classes of matrices, for broad classes of underlying problems, or for a specific matrix or problem.

Preconditioners for general classes of matrices

Examples of preconditioners for general classes of matrices are *regular splittings*, among which Jacobi, Gauss–Seidel, and successive overrelaxation (SOR) are the most well-known. In [8] Greenbaum provides a thorough analysis of these methods. For ill-conditioned matrices these preconditioners do not perform well however.

A more successful approach is the use of incomplete decompositions that first was proposed by Varga in [26] and made popular by Meijerink and van der Vorst in [19]. Among these methods are e.g. incomplete Cholesky (IC), modified incomplete Cholesky (MIC) [10], and incomplete LU-decomposition (ILU) which exists in numerous variants. This type of preconditioners is widely used in production codes and works well for symmetric matrices. For unsymmetric problems however the performance is unpredictable.

Another approach that has been used is the *sparse approximate inverse* (SPAI) [3] preconditioning technique, that constructs a sparse inverse to A by solving the least-squares problem $\min_M \|I - AM\|_F$[4]. This technique yields good convergence properties for the preconditioned system but since it is rather expensive to form the preconditioner it is mostly used when we have several right-hand sides.

Preconditioners for broad classes of underlying problems

Although not perceived that way originally, the *multigrid* method can be viewed as a preconditioner for systems of linear equations originating from PDEs. As such it is very efficient for systems that arise from discretization of elliptic PDE. It can be shown that we achieve grid-independent convergence for multigrid on these problems [8].

[4] $\|\cdot\|_F$ denotes the Frobenius norm of a matrix, $\|A\|_F \equiv \sqrt{\Sigma_{i,j} a_{i,j}^2}$.

Another approach in this class of preconditioners for PDE problems is *domain decomposition* methods [11]. Here we have two different methodologies. One that uses overlapping domains, e.g. multiplicative and additive Schwarz methods. The other methodology uses non-overlapping domains and the methods are called substructuring methods. The Schwarz methods were originally developed to prove existence of a solution for elliptic PDEs. Today domain decomposition methods are mainly used for parallelization and convergence acceleration. It can be shown that the number of iterations is independent of the mesh size under certain assumptions [23].

There exists a large number of specialized preconditioners for different applications. In many cases these specialized preconditioners outperform more general approaches in their domain of interest. In the sequel we describe a suitable preconditioner for boundary layer flow problems.

1.4 Semi-Toeplitz preconditioning

Much effort in recent years has been invested in the research of preconditioners based on fast transforms. In the review paper [1] R. Chan and Ng comprises much of the research in this area. An important contribution is the use of fast transform-based preconditioners for hyperbolic problems introduced by e.g. Holmgren and Otto [16, 17, 18] with their semicirculant preconditioning strategy.

In her thesis L. Hemmingsson [13] uses semi-Toeplitz preconditioners for first order PDEs that yield good convergence behavior. In this thesis we apply the same approach to boundary layer problems.

To explain the idea of the preconditioner we consider Toeplitz matrices T of the type

$$T = \alpha I_{m_1} + \beta R_{m_1}, \tag{1.10}$$

where $\alpha, \beta \in \mathbb{R}$, I_{m_1} is the identity matrix of size $m_1 \times m_1$ and R_{m_1} is defined by

$$R_{m_1} = \begin{pmatrix} 0 & 1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & 1 \\ & & -1 & 0 \end{pmatrix}. \tag{1.11}$$

In the matrix R_{m_1} above, all elements outside the diagonals are zero. A semi-Toeplitz matrix is constructed by

$$M = \begin{pmatrix} T_{1,1} & T_{1,2} & & \\ T_{2,1} & \ddots & \ddots & \\ & \ddots & \ddots & T_{m_2-1,m_2} \\ & & T_{m_2,m_2-1} & T_{m_2,m_2} \end{pmatrix}, \qquad (1.12)$$

where $T_{i,j}$ are all of the form defined in (1.10). The parameters α and β are computed from averages over a base matrix that is derived by discretizing the differential equations with different boundary conditions.

The preconditioner system, $Mx = y$, is solved using a Fast Modified Sine Transform (FMST) and the solution of small tridiagonal systems and the work is of order $\mathcal{O}(m_1 m_2 \log(m_1 m_2))$. This is done in three steps:

1. Perform fast Fourier transforms of vectors $y \in \mathbb{C}^{\frac{m_1+1}{2}}$.
2. Solve $\frac{m_1+1}{2}$ tridiagonal systems of order m_2.
3. Perform fast inverse Fourier transforms of vectors $y \in \mathbb{C}^{\frac{m_1+1}{2}}$.

For further details concerning FMST and the preconditioner solve we refer to [12].

1.4.1 Analysis of a semi-Toeplitz preconditioner for a convection-diffusion problem

In [25] we study a scalar model problem that mimics the behavior of the Navier–Stokes equations in a boundary layer close to a solid surface,

$$\frac{\partial u}{\partial t} - \nu \frac{\partial^2 u}{\partial x_2^2} + \frac{\partial u}{\partial x_1} + v \frac{\partial u}{\partial x_2} = 0, \qquad 0 \le x_1, x_2 \le 1. \qquad (1.13)$$

We have Dirichlet boundary conditions at all boundaries except at $x_1 = 1$ where we have a Neumann condition. The viscosity parameter is ν, and u and v are the velocity components in the x_1- and x_2-directions in (1.13) and v is constant. This equation is solved both as a time-dependent problem and as a stationary problem with $\partial u/\partial t = 0$. We discretize this equation on a uniform grid with m_d points in the x_d-direction. For the spatial discretization we use centered second order approximations and in time we use the trapezoidal rule.

For this setting we define a semi-Toeplitz preconditioner as explained above. The paper comprises a theoretical analysis of the spectrum of the preconditioned systems and numerical experiments. In the theoretical analysis we find closed formulas for the eigenvalues of the preconditioned systems. It turns out that we only get m_2 eigenvalues that are different from 1. This is of great importance when using GMRES, since the convergence of GMRES is dependent on the number of distinct eigenvalues. This result holds for both the time-dependent problem and the stationary problem.

Furthermore we look at the asymptotic spectrum when we let the number of grid points m_1 and m_2 go to infinity. For the time-dependent problem we find that in the limit we only get two eigenvalues. The first eigenvalue is 1 and the second is given by

$$\mu_\infty = 4 - \frac{8}{\kappa_1} + \frac{12}{\kappa_1^2} + \mathcal{O}\left(\frac{1}{\kappa_1^3}\right). \tag{1.14}$$

Here $\kappa_1 = \Delta t/\Delta x_1$. For the stationary problem the eigenvalues are 1 or located in an interval on the real axis.

The convergence of iterative methods for non-Hermitian problems is however not completely determined by the spectrum of the preconditioned system, but also depends on the condition number of the eigenvector-matrix for the system, see (1.5). For both problems we can estimate the condition number reduction that occurs when the grid is refined.

These theoretical results are verified by numerical experiments using GMRES(6). When we perform a grid refinement study for the time-dependent problem we find that the number of iterations needed for convergence for the preconditioned problem is very small compared to the unpreconditioned problem. The iterations also decrease as the grid is refined and for the finest grids we reach convergence in only two iterations.

For the stationary problem the situation is slightly less favorable since here we obtain a small increase in iterations as the grid is refined. Compared to the unpreconditioned solver the gain is still substantial. For the unpreconditioned solver convergence is not reached for the finest grid.

Although not covered by the analysis, we also introduce a grid with a stretching in the x_2-direction to study the behavior of the preconditioner for this case. We find that the preconditioner exhibits essentially the same convergence behavior even for a large stretching factor. This is an important feature when we solve boundary layer problems.

1.4.2 Solving the linearized Navier–Stokes equations using semi-Toeplitz preconditioning

In [24] we solve the linearized Navier–Stokes equations using semi-Toeplitz preconditioning. These equations describe the flow over a flat plate and are derived by linearizing and symmetrizing the Navier–Stokes equations around the Blasius solution $\hat{U} = \begin{pmatrix} \hat{u} & \hat{v} & \hat{\rho} \end{pmatrix}^T$, yielding

$$\frac{\partial U}{\partial t}+A_1\frac{\partial U}{\partial x_1}+A_2\frac{\partial U}{\partial x_2}-B_1\frac{\partial^2 U}{\partial x_1^2}-B_2\frac{\partial^2 U}{\partial x_2^2}-B_3\frac{\partial^2 U}{\partial x_1 \partial x_2} = F(x_1, x_2, t), \quad (1.15)$$

where $U = \begin{pmatrix} u & v & \rho \end{pmatrix}^T$ and F depends on the Blasius solution. The computational domain is chosen to be $1 \leq x_1 \leq 2$, and $0 \leq x_2 \leq 1$ with the plate at $x_2 = 0$. The coefficients A_1, A_2, B_1, B_2, and B_3 are (3×3)-matrices of the form

$$A_1 = \begin{pmatrix} \hat{u} & 0 & 1 \\ 0 & \hat{u} & 0 \\ 1 & 0 & \hat{u} \end{pmatrix}, A_2 = \begin{pmatrix} \hat{v} & 0 & 0 \\ 0 & \hat{v} & 1 \\ 0 & 1 & \hat{v} \end{pmatrix},$$

$$B_1 = \nu \begin{pmatrix} \frac{4}{3} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, B_2 = \nu \begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{4}{3} & 0 \\ 0 & 0 & 0 \end{pmatrix}, B_3 = \nu \begin{pmatrix} 0 & \frac{1}{3} & 0 \\ \frac{1}{3} & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix},$$

where ν is the kinematic viscosity. Note that $\hat{u}$ and $\hat{v}$ in A_1 and A_2 depend on x_1 and x_2. In [15] Holmgren et. al. solve (1.15) as a steady state problem using explicit time marching with semicirculant preconditioning.

We discretize (1.15) on a grid with stretching in the x_2-direction. The spatial derivatives are approximated using centered second order accurate stencils and in time we employ the trapezoidal rule. This gives us a system of equations which we solve using GMRES(6). Semi-Toeplitz preconditioning is utilized to obtain a faster and more robust iterative solver.

The numerical results confirm that the preconditioned solver exhibits a similar convergence behavior as in [25]. In Fig. (1.1) a comparison between unpreconditioned GMRES(6) and preconditioned GMRES(6) with respect to iteration count and runtime for one time step is shown. The preconditioned GMRES(6) shows a considerable reduction in iterations and the runtime for the preconditioned method is much smaller, even though it includes the extra work needed to form the preconditioner.

We study the preconditioner in experiments with different stretching factors for the grid. It appears that the preconditioner performs well and

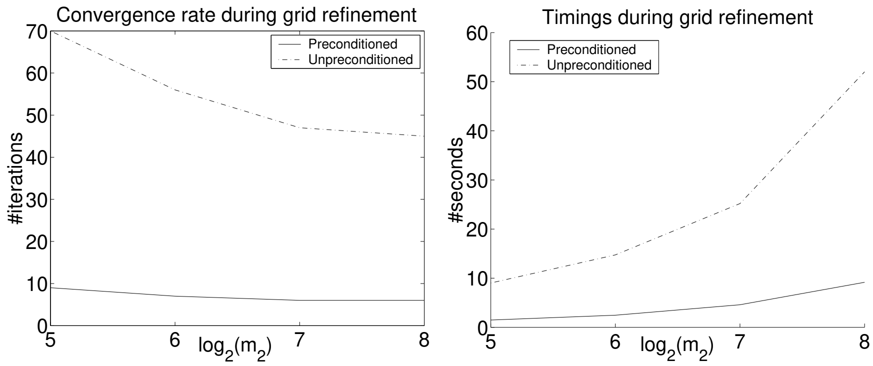

Figure 1.1: A comparison between preconditioned and unpreconditioned GMRES(6) regarding iteration count (left) and timings (right) on a uniform grid. We vary the number of grid points in the x_2-direction and keep the number of grid points in the x_1-direction constant. $\Delta t/\Delta x_2 = 10$.

actually we observe a decay in iterations as we increase the stretching, which is not the case for the unpreconditioned solver. We also compare the results for different time steps. Here we see an increase in iterations when we use a larger time step, but we still gain computing time to reach a time τ by increasing the time step.

Some experimentation with different bases for the preconditioner is also done. We compare using only the first order terms as basis with using both first and second order terms as basis for the preconditioner. The latter basis is also used in combination with terms arising from the artifical dissipation. This is done to remedy the slower convergence that arises when a large amount of artifical dissipation is used in the original problem.

1.5 Conclusions

In this thesis we have presented a preconditioner based on semi-Toeplitz approximations. We analyze this approach for a scalar model problem and find analytic expressions for the eigenvalues. The analysis shows that important properties of the preconditioned system are substantially improved. These theoretical results are sustained by numerical experiments which show that the convergence behavior of GMRES(6) is significantly enhanced for this model problem.

Furthermore we apply the same approach to the linearized Navier–Stokes equations for flow over a flat plate. We find that for the preconditioned solver we get a substantial reduction in both the number of iterations and in total runtime, and we also find that the preconditioned solver is more robust in a variety of situations.

We conclude that the use of semi-Toeplitz approximations is a successful approach for the problems considered and should be viable for use in more general settings.

Acknowledgments

I am very grateful for the invaluable support from my supervisors Docent Lina von Sydow and Professor Per Lötstedt. Without their support the completion of this thesis would not have been accomplished.

I thank Docent Sverker Holmgren, Docent Kurt Otto and Docent Lina von Sydow for the kindness of giving me access to computer programs they have written.

This research project has been supported by the Swedish Research Council for Engineering Sciences and the Swedish Foundation for Strategic Research under contract No 312-97-771.

Bibliography

[1] R.H. Chan and M.K. Ng. Conjugate gradient methods for Toeplitz systems. *SIAM Rev.*, 38:427–482, 1996.

[2] P. Concus, G.H. Golub, and D.P. O'Leary. A generalized conjugate gradient method for the numerical solution of elliptic partial differential equations. In J.R. Bunch and D.J. Rose, editors, *Sparse Matrix Computations*, pages 181–194. Academic Press, New York, 1976.

[3] J.D.F. Cosgrove, J.C. Diaz, and A. Griewank. Approximate inverse preconditionings for sparse linear system. *Int. J. Computer Mathematics*, 44:91–110, 1992.

[4] I.S. Duff. Sparse numerical linear algebra: direct methods and preconditioning. In I.S. Duff and G.A. Watson, editors, *The State of the Art in Numerical Analysis*, pages 27–62. Oxford University Press, 1997.

[5] I.S. Duff, M. Erisman, and J.K. Reid. *Direct Methods for Sparse Matrices.* Oxford University Press, Oxford, England, 1986.

[6] V. Faber and T. Manteuffel. Necessary and sufficient conditions for the existence of a conjugate gradient method. *SIAM J. Numer. Anal.*, 21:352–362, 1984.

[7] A. Greenbaum. Comparison of splittings used with the conjugate gradient algorithm. *Numer. Math.*, 33:181–194, 1979.

[8] A. Greenbaum. *Iterative Methods for Solving Linear Systems.* SIAM, Philadelphia, PA, 1997.

[9] A. Greenbaum and L. Gurwitz. Max-min properties of matrix factor norms. *SIAM J. Sci. Comput.*, 15:348–358, 1994.

[10] I. Gustafsson. A class of 1st order factorization methods. *BIT*, 18:142–156, 1978.

[11] W. Hackbusch. *Iterative Solution of Large Sparse Systems of Equations.* Springer-Verlag, Berlin, New York, 1994.

[12] L. Hemmingsson. A fast modified sine transform for solving block-tridiagonal systems with Toeplitz blocks. *Numer. Algorithms*, 7:375–389, 1994.

[13] L. Hemmingsson. *Domain Decomposition Methods and Fast Solvers for First-order PDEs.* Ph.D. thesis, Dept. of Scientific Computing, Uppsala Univ., Uppsala, Sweden, 1995.

[14] M.R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *J. Res. Natl. Bur. Stand.*, 49:409–436, 1954.

[15] S. Holmgren, H. Brandén, and E. Sterner. Convergence acceleration for the linearized Navier–Stokes equations using semicirculant approximations. *SIAM J. Sci. Comput.*, 21:1524–1550, 2000.

[16] S. Holmgren and K. Otto. Semicirculant preconditioners for first-order partial differential equations. *SIAM J. Sci. Comput.*, 15:385–407, 1994.

[17] S. Holmgren and K. Otto. A framework for polynomial preconditioners based on fast transforms I: Theory. *BIT*, 38:544–559, 1998.

[18] S. Holmgren and K. Otto. A framework for polynomial preconditioners based on fast transforms II: PDE applications. *BIT*, 38:721–736, 1998.

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

[20] N.M. Nachtigal, S. Reddy, and L.N. Trefethen. How fast are nonsymmetric matrix iterations? *SIAM J. Matrix Anal. Appl.*, 13:778–795, 1992.

[21] C.C. Paige and M.A. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM J. Numer. Anal.*, 12:617–629, 1975.

[22] Y. Saad and M.H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Statist. Comput.*, 7:856–869, 1986.

[23] B. Smith, P. Bjorstad, and W. Gropp. *Domain Decomposition. Parallel Multilevel Methods for Elliptic Partial Differential Equations.* Cambridge University Press, London UK, 1996.

[24] S. Sundberg. Solving the linearized Navier–Stokes equations using semi-Toeplitz preconditioning. Technical report, Dept. of Information Technology, Uppsala Univ., Uppsala, Sweden, 2002.

[25] S. Sundberg and L. von Sydow. Analysis of a semi-Toeplitz preconditioner for a convection-diffusion problem. Technical Report 2002-014, Dept. of Information Technology, Uppsala Univ., Uppsala, Sweden, 2002.

[26] R.S. Varga. Factorization and normalized iterative methods. In R.E. Langer, editor, *Boundary Problems in Differential Equations*, pages 121–142, 1960.

Recent licentiate theses from the Department of Information Technology

2001-010 Johan Edlund: *A Parallel, Iterative Method of Moments and Physical Optics Hybrid Solver for Arbitrary Surfaces*

2001-011 Pär Samuelsson: *Modelling and control of activated sludge processes with nitrogen removal*

2001-012 Per Åhgren: *Teleconferencing, System Identification and Array Processing*

2001-013 Alexandre David: *Practical Verification of Real-time Systems*

2001-014 Abraham Zemui: *Fourth Order Symmetric Finite Difference Schemes for the Wave Equation*

2001-015 Stefan Söderberg: *A Parallel Block-Based PDE Solver with Space-Time Adaptivity*

2001-016 Johan Furunäs Åkesson: *Interprocess Communication Utilising Special Purpose Hardware*

2002-001 Eva Mossberg: *Higher Order Finite Difference Methods for Wave Propagation Problems*

2002-002 Anders Berglund: *On the Understanding of Computer Network Protocols*

2002-003 Emad Abd-Elrady: *Harmonic Signal Modeling Based on the Wiener Model Structure*

2002-004 Martin Nilsson: *Iterative Solution of Maxwell's Equations in Frequency Domain*

2002-005 Kaushik Mahata: *Identification of Dynamic Errors-in-Variables Models*

2002-006 Kalyani Munasinghe: *On Using Mobile Agents for Load Balancing in High Performance Computing*

2002-007 Samuel Sundberg: *Semi-Toeplitz Preconditioning for Linearized Boundary Layer Problems*

UPPSALA
UNIVERSITET

Eklundshofs Grafiska AB