Aspects of The Recursive Projection Method applied to Flow Calculations

JOAKIM MÖLLER

Doctoral Thesis
Stockholm, Sweden 2005
Akademisk avhandling som med tillstånd av Kungl Tekniska högskolan framlägges till offentlig gransknings för avläggande av teknologie doktorsexamen torsdagen den 20 januari 2005 kl. 10.15 i Seminarierum L1, Kungl. Tekniska Högskolan, Drottning Kristinas väg 30, Stockholm.

© Joakim Möller, Januari 2005

Tryck: Universitetservice US AB
Abstract

In this thesis, we have investigated the Recursive Projection Method, RPM, as an accelerator for computations of both steady and unsteady flows, and as a stabilizer in a bifurcation analysis.

The criterion of basis extraction is discussed. It can be interpreted as a tolerance for the accuracy of the eigenspace spanned by the identified basis, alternatively it can be viewed as a criterion when the approximative Krylov sequence becomes numerically rank deficient.

Steady state calculations were performed on two different turbulent test-cases; a 2D supersonic nozzle flow with the Spalart-Allmaras 1-equation model and a 2D sub-sonic airfoil simulation using the $k-\epsilon$ model. RPM accelerated the test-cases with a factor between 2 and 5.

In multi-scale problems, it is often of interest to model the macro-scale behavior, still retaining the essential features of the full systems. The “coarse time stepper” is a heuristic approach for circumventing the analytical derivation of models. The system studied here is a linear lattice of non-linear reaction sites coupled by diffusion. After reformulation of the time-evolution equation as a fixed-point scheme, RPM coupled with arc-length continuation is used to calculate the bifurcation diagrams of the effective (but analytically unavailable) equation.

Within the framework of dual time-stepping, a common approach in unsteady CFD-simulation, RPM is used to accelerate the convergence. Two test-cases were investigated; the von Karman vortex-street behind a cylinder at $Re = 100$, and the periodic shock oscillation of a symmetric airfoil at $M_{\infty} = 0.76$ with a Reynolds number $Re = 11 \times 10^6$.

It was believed that once a basis had been identified, it could be retained for several steps. The simulations usually showed that the basis could only be retained for one step.

The need for updating the basis motivates the use of Krylov methods. The most common method is the (Block-) Arnoldi algorithm. As the iteration proceeds, Krylov methods become increasingly expensive and restart is required. Two different restart algorithms were tested. The first is that of Lehoucq and Maschhoff, which uses a shifted QR iteration, the second is a block extension of the single-vector Arnoldi method due to Stewart. A flexible hybrid algorithm is derived combining the best features of the two.
Preface

This thesis consists of five papers and a summary of the work presented. The five papers are:


JM contributed to the implementation and performed the RPM simulations, the analysis in section 3.1., and (to a lesser degree) contributed to the write-up.


SG developed the EDGE-part of the code and generated the grids, JM developed the RPM-part. The simulations and the write-up are joint work.


The work was divided between the authors like for paper III.

Acknowledgments

I would sincerely like to thank Jesper Oppelstrup, my supervisor; for his important hints and suggestions along the way. Without him, this thesis would not have been what it is today. Axel Ruhe, for inspiring me to investigate Krylov-methods. Mihai Dorobantu, for the introduction to RPM and for arranging the visit at UTRC, and Ioannis Kevrekidis for the introduction to the Coarse Time Steppers and letting me come and work with him at Princeton in sep. 2001. Richard Lehoucq, who always gracefully has answered my questions over the years and for providing the QR-algorithm used in paper V. I would also like to thank all my co-authors Stefan Götz, Olof Runborg, Kurt Lust, and Panayotis Kevrikidis. A special thank you to Gunilla Kreiss and Michael Hanke, for all the discussions, and the staff of PDC for all help and support. Finally a big thank you to all my colleagues who make NADA the pleasant place it is.

Financial support for the research is gratefully acknowledged from NADAPSCI (Parallel and Scientific Computing Institute), KTH, Princeton University, New Jersey, and UTRC (United Technology Research Center), Hartford, Connecticut.
Contents

1 Introduction 1

2 The Recursive Projection Method Applied to Steady-state CFD Calculations; Paper I 5

3 Equation-free, effective computation for discrete Systems: a time stepper based approach; Paper II 9

4 Recursive Projection Method for Efficient Unsteady CFD Simulations; Paper III & Paper IV 11

5 New Implementations of the Implicitly Restarted Block Arnoldi Method; Paper V 15

6 Conclusions 19

7 Outlook and Future Work. 21

Bibliography 23
Chapter 1

Introduction

Many applications require us to compute steady state solutions of partial differential equations (PDEs). The solution of the large non-linear system of algebraic equations that results from the spatial discretization is often obtained by integrating the corresponding system of ordinary differential equations (ODEs) in time until the time derivatives become sufficiently small. Assuming that this system has a steady-state $y^*$ and a fixed point iteration scheme of the type

$$y^{n+1} = F(y^n),$$

for the temporal evolution, the ultimate convergence of the iteration is determined by the dominant eigenvalues of the Jacobian $J = F_y$ evaluated at $y^*$. If all eigenvalues lie strictly within the unit circle, the scheme is asymptotically convergent in a neighborhood of $y^*$ and the linear asymptotic convergence factor is the modulus of the largest eigenvalue of $J$.

The Recursive Projection Method (RPM), as presented by Shroff and Keller in 1993, is an iterative procedure which can accelerate location of fixed points and stabilize unstable numerical procedures. It only requires the iterates obtained in the fixed point-iteration. It adaptively seeks to identify the subspace corresponding to large eigenvalues of $J$, hence directions of slow or unstable time-evolution in phase space. This is accomplished by monitoring the successive residuals $\Delta y^n = F(y^n) - y^n$ forming an approximate Krylov subspace. A projection $\mathcal{P}$, constructed from the identified subspace, is used to split $y^n$ into $p^n = \mathcal{P}y^n$ and $q^n = Qy^n$, where $Q = I - \mathcal{P}$. In the $p$ slow directions spanned by $\mathcal{P}$, the fixed point iteration is replaced by (approximate) Newton iteration, whereas the fixed point iteration is used in the remaining directions. The overall algorithm takes the form

$$y^{n+1} = QF(y^n) + \left(\mathcal{P}J\mathcal{P} - I\right)^{-1}\mathcal{P}(F(y^n) - y^n)$$

For the method to be efficient, the dimension of the slow/unstable subspace should be small, with a gap to the rest of the spectrum. The nature of many
transport PDEs encountered in engineering modeling (the action of viscosity, heat conduction, diffusion, and the resulting spectra) do indeed show a separation of time-scales, which translates into a gap in the spectrum of $J$ at the steady state.

The Jacobian is never formed explicitly since its actions can be approximated via finite-differences, making RPM a “matrix-free” method. The cost per step to advance the algorithm is one function evaluation and the solution of an order $p$ linear system of equations. RPM still retains the simplicity of the fixed point iteration, in the sense that no more information is needed than just the iteration function. It can be applied as a ‘black box’ wrap-around to any code which defines an iteration function $F$.

The applications of RPM can be divided into two main groups: as a tool for bifurcation analysis, and as a convergence accelerator.

The first group is by far the largest. As long as the Jacobian $(J - I)$ is non-singular, RPM as described above can be applied. At a turning point, the Jacobian becomes singular. By incorporating pseudo-arclength continuation, as proposed by Shroff and Keller [34], the RPM process can proceed past the turning point. Lust et al. [27] extended the idea of RPM to a hierarchy of different related methods by allowing more than one function evaluation per step and used the algorithms for bifurcation analysis of periodic solutions.

Since then, RPM and variants have been applied to a wide variety of problems. The different sections below contain a more complete list of references. In later years, much research has been conducted in the field of equation-free homogenized, or averaged, models and their properties. The method known as Coarse time stepping was introduced in [39]. Paper II is an early contribution to this fast growing field of literature.

The other important use is RPM as convergence accelerator for computational fluid dynamics computations. The applications can be further classified as steady- or unsteady- calculations. For steady flows, there are a few precursors to RPM. Bergman [3], attempted acceleration techniques for the transonic small perturbation equation. He assumed there was one real or a pair of complex conjugated dominant eigenvalues, and devised an extrapolation formula to eliminate the associated error. The formula has large coefficients and, not being a convergent iteration, it can only be applied a few times. The success of the scheme was limited.

Rizzi, Eriksson [31], applied a splitting technique for computing steady transonic Eulerian flow. An explicit smoothing algorithm first reduces the short wavelength errors. Long wavelength components of the error are eliminated by projection of the solution onto a low-dimensional space where Newton linearization can be employed. The acceleration obtained was quite small.

Dorobantu et al. [11] computed steady quasi-1D inviscid nozzle flows with good results. The work was used as a basis for [30], where RPM was successfully used to accelerate 2D compressible viscous flow calculations. The work in paper I is an extension of these results.

The time-step that can be used in an explicit CFD-scheme is severely restricted by stability. The interesting physical time scale can be several orders of magnitude
larger, which makes implicit time-stepping attractive. On the other hand, implicit schemes require the solution of large systems of non-linear equations in each step, usually by a pseudo-time marching scheme. The dual time-stepping approach due to Jameson, [13], can be accelerated by RPM. Paper III describes RPM used to accelerate time-accurate simulation of the vortex shedding in laminar flow around a cylinder at Reynolds-number 100. The idea was extended to a 2D Explicit Algebraic Reynolds Stress Model (EARSM) in paper IV for self-induced shock oscillations over an airfoil at Mach 0.76 and a chord Reynolds number of 11 million.

Since the time-steps are small enough to resolve important time scales, changes in the spectra of the linearized problems from step to step should be small. One believes that the basis for the slow subspace \( \mathcal{P} \) would be sufficiently accurate for a number of successive time-steps. Numerical experiments suggest that this number is very limited. Almost every step needs to either "restart" or apply some basis update technique. This motivates the study of matrix-free eigensolvers, e.g. Krylov methods.

The most well-known Krylov method for computing a few eigenvalues of a non-symmetric matrix is based on the Arnoldi iteration. From a unitary vector \( v_1 \) it constructs a Krylov subspace, spanned at step \( m \) by the orthogonal basis \( V_m \). The Arnoldi decomposition is

\[
AV_m = V_m H_m + f_m e_m^T.
\] (1.2)

where \( H_m \) is an upper Hessenberg matrix, i.e., \( h_{ij} = 0 \) for \( i > j + 1 \). As the iteration proceeds, \( V_m \) grows and with it the computational costs and memory requirements, necessitating some restarting technique. Sorensen [36] proposed the very elegant 'Implicit Restart' which produces a compressed Arnoldi-decomposition based on the starting vector \( v_1 \) where the unwanted directions present in \( V_m \) have been damped. Stewart [37] showed that the equivalent compressed decomposition is obtainable by applications of elementary unitary matrices.

Block Arnoldi is a generalization of (1.2). A block Krylov subspace is constructed from \( b \) orthogonal starting vectors forming the block-matrix \( V_b = [v_1, \ldots, v_b] \). A block Arnoldi decomposition

\[
AV_{mb} = V_{mb} H_{mb} + F_m E_{mb}^T.
\] (1.3)

results where \( H_{mb} \) is a banded upper Hessenberg matrix, i.e., a Hessenberg matrix with an additional \( b - 1 \) sub-diagonals. Block methods handle clustered and/or multiple eigenvalues more reliably than single vector methods do. Matrix-vector products are replaced by matrix-matrix multiplications, which is favorable for high performance computer architectures.

Lehoucq and Maschhoff [24] extended the implicit restarting techniques to Block Arnoldi. Paper V extends the Stewart restarting to block methods; for block-size \( b > 1 \) the two approaches are not equivalent.

The extremal eigenvalues and associated eigenvectors are the first to converge in the Arnoldi iteration. As this happens, the eigenvalues should be deflated, the
desired ones locked, and the unwanted ones purged, to speed convergence of other parts of the spectrum. This procedure is done differently in Lehoucq’s and Stewart’s schemes.

Numerical experiments suggest that the Lehoucq block Arnoldi iteration is much more efficient than Stewart’s algorithm. We constructed a hybrid version of the two algorithms. Using the Lehoucq shift strategy to gain good convergence, and strategies similar to Stewart’s to simplify deflation of converged eigenvalues, we arrive at a very flexible new version of the Implicitly Restarted Block Arnoldi method.

In general, for fixed dimension of the Krylov space, single-vector methods converge faster than block-methods, because the matrix-polynomial is of higher degree. This suggests that single vector methods are the favorable tool also when basis updates should performed in RPM. Most of the test examples confirm this conjecture, but a few cases were found where block methods were competitive, indicating that further investigation is needed to close the issue.
In many applications, it is of interest to compute steady state solutions to partial differential equations. This can be accomplished by solving the non-linear system of algebraic equations that results from the spatial discretization, e.g. by Newton’s Method. For large scale problems this becomes prohibitively expensive, since each step requires the formation and “inverting” of a large matrix.

Another approach is to integrate the partial differential equation (PDE) in time until the time derivatives become sufficiently small. The spatial discretization produces a system of ordinary differential equations (ODEs)

\[
\frac{dy}{dt} = f(y), \quad y \in \mathbb{R}^N.
\] (2.1)

Note that many models, e.g. the incompressible Navier–Stokes equations, do not give a system of ODEs, but rather a differential-algebraic system. In this case, it is necessary to first eliminate the algebraic constraints and transform the system to an ODE system. From (2.1) we derive a fixed point iteration scheme of the type

\[
y^{n+1} = F(y^n),
\] (2.2)

Even if initially the associated residual \(y^n - F(y^n)\) drops rapidly, the ultimate convergence to a fixed point \(y^*\), determined by the most dominant eigenvalue \((\lambda_1)\) of the Jacobian \(J = F_y\), will be slow if the modulus of this eigenvalue is close to one:

\[
||y^{n+1} - y^*|| \approx |\lambda_1| ||y^n - y^*||
\]

for large \(n\).
The Recursive Projection Method can be viewed as a compromise between the
two approaches discussed above. Via an orthogonal decomposition, obtained by the
projection $\mathcal{P}$, the solution is split in two parts,

$$
y^n = \mathcal{P}y^n + (I - \mathcal{P})y^n = \mathcal{P}y^n + \mathcal{Q}y^n = p^n + q^n
$$

$$
\mathcal{P} = V_p V_p^T
$$

$$
\mathcal{Q} = V_q V_q^T = (I - V_p V_p^T)
$$

$p^n$ is the $\mathcal{P}$ projection of $y^n$ onto the maximal invariant subspace associated with
the $p$ most dominant eigenvalues of $J$, where $p \ll N$. The space associated with $\mathcal{Q}$
is the orthogonal complement, which is not invariant for non-symmetric Jacobians.
Application of these projections to (2.2) splits the algorithm into two parts. The $\mathcal{Q}$-
part will now have better convergence properties since the influence of the dominant
eigenvalues has been eliminated, but the slow modes are still present in the $\mathcal{P}$-part.
To fix this, the $\mathcal{P}$-equation is replaced by an implicit relation,

$$
p^{n+1} = \mathcal{P}F(q^n + p^{n+1}).
$$

and applying one Newton step to (2.3), we arrive at RPM:

$$
q^{n+1} = F(y^n) - V_p \left( V_q^H F(y^n) \right),
$$

$$
p^{n+1} = p^n - V_p (V_q^H J^n V_p - I)^{-1} V_q^H \left( F(y^n) - y^n \right)
$$

$$
y^{n+1} = q^{n+1} + p^{n+1}
$$

Sufficiently close to the fixed point, the Newton part converges quadratically. Since
$p \ll N$, the cost of solving the linear system of equations is small.

The Jacobian $J^n$ occurring in (2.4) is never formed explicitly since it only occurs
as a matrix-matrix product. The directional derivatives needed can be found by
finite difference approximation. Usually it is sufficient to use a first order scheme,
since the error in general is smaller than the error associated with the approximate
projections.

According to Householder’s theorem [33], for any square matrix $A$, there is a
vector norm $\| \cdot \|$ such that $\| A \| \leq \rho(A) + \epsilon = \bar{\rho}$, where $\epsilon > 0$ is arbitrarily small.
Let $A = \mathcal{Q}J\mathcal{Q}$. Then, for sufficiently small $\| y^0 - y^* \|$, the scheme converges,
asymptotically with rate

$$
\frac{\| y^{n+1} - y^* \|}{\| y^k - y^* \|} < \bar{\rho}
$$

for sufficiently large $n$.

The basis $V_p$ is constructed as follows. A matrix $K_k$ with fixed $k$, is formed
from the successive $q$-iterates,
\[
K_k = \left[ \Delta q^{n-k+1}, \ldots, \Delta q^n \right], \\
\Delta q^n = q^{n+1} - q^n
\]

The \(K_k\) form an approximate \(k_s\)-dimensional Krylov space for the projected Jacobian \(QJQ\). Via a QR-factorization of \(K_k\), a basis is extracted. If the diagonal elements of \(R\) satisfy

\[
\left| \frac{r_{jj}}{r_{j+1,j+1}} \right| > k_a,
\]

the first \(i_r\) columns out of the \(k_s\) are added to the basis. \(k_a\) is called the Krylov acceptance ratio. The method periodically checks the \(k_a\)-criteria and thus decides how large and when a basis should be formed. For a linear problem, it can be shown that the obtained basis (in exact arithmetic) is equivalent to the one obtained from Arnoldi decomposition based on the normalized residuals \(\Delta q^{n-k+1}\). Furthermore, \(\frac{1}{k_a}\) is an upper bound for the eigenspace residual

\[
\|QJz - \theta z\| \leq \frac{r_{i+1,i}}{r_{i,i}},
\]

where the eigenvector \(z\) is spanned by the new basis vectors.

\(k_a\) can also be interpreted as a threshold for when \(K_k\) should be considered rank deficient, hence signaling when an approximate invariant subspace of dimension \(i_r\) is present. This motivates the use of QR-factorization with column pivoting to better determine the rank. The connection to the eigenspace residual is severed by pivoting, however.

The Newton part of the algorithm requires \(V_p\) to be computed with sufficient accuracy. Assume that the approximative basis \(V_p\) satisfies

\[
V_pR_p = \tilde{X}_p + \mathcal{E}
\]

where \(\tilde{X}_p\) is the matrix consisting of the \(p\) first eigenvectors and \(\mathcal{E}\) is the error. If

\[
\|R_p^{-1}\|\|\mathcal{E}\tilde{\Lambda}_p - J\mathcal{E}\| < \min\left| 1 - \tilde{\lambda}_i \right|, \quad i = 1, \ldots, p
\]

then the Newton algorithm behaves well. \(\tilde{\Lambda}_p\) is a diagonal matrix containing the first \(p\) eigenvalues \(\tilde{\lambda}_i\) of the Jacobian \(J\), see [30].

In CFD applications it is important to take into account the magnitudes of the different components of the state-vector \(y = [\rho, \rho u, \rho v, \rho w, \rho E]_T\) where \(\rho\) is the density, \(u, v\) and \(w\) are the velocity components and \(E\) is the total energy. The components may differ in magnitude by several decades, depending on the choice of units, or scaling, of the variables. If the state-vector is poorly scaled, the projections become easily perturbed which can affect the performance of RPM. [30] shows how the robustness of RPM is improved by proper scaling.
In many models, some components, e.g. density, pressure, and turbulent kinetic energy, must stay positive. This can be guaranteed for explicit time-stepping methods, as long as the time-step is sufficiently small, and the inherent dynamics of the model preserves positivity. For implicit methods with Newton linearizations and long time-steps, there is no guarantee. Proper scaling of variables improves robustness in this respect also.

RPM was applied as a wrap-around to the CFD-solver NSMB [44]. NSMB was transformed to a subroutine which, given a state-vector $y^n$, updates boundary conditions, turbulent source-terms, etc., and returns the state after $n_s$ iterations, i.e $F(y^n)$. RPM was implemented into MATLAB and linked to NSMB via a MEX interface.

Two test-cases were investigated: A 2D supersonic nozzle flow with the Spalart-Allmaras turbulence model, and the subsonic flow around a 2D airfoil, using the $k-\epsilon$ turbulence model. RPM substantially accelerated the convergence, consistent with the simulations in [30]. The extracted basis stayed small, in both cases of the size $p = 26$, compared to the size of the state-vector 40704 (nozzle) and 57344 (airfoil), $p \ll N$ indeed.

Preceding the original Shroff and Keller RPM paper, the idea of splitting the iteration can be found in a series of papers by Jarausch and Mackens [16], [17], [18]. They refer to the method as a 'Condensed Newton' method, but restrict the treatment to symmetric problems. Later Jarausch considered the unsymmetric case, [15], accomplishing the splitting via oblique projections based on singular subspaces. As stated earlier, for RPM to be effective, the subspace associated with the Newton part should be small. Davidson [10] considered a preconditioned RPM requiring knowledge of the system Jacobian in order to be effective. By allowing several function evaluations per iteration, Lust [26] and Lust et al. [27] obtained a hierarchy of different RPM related methods, known as 'Newton-Picard-Gauss-Seidel' NPGS.

There are several applications of RPM to CFD as a tool for bifurcation analysis. Keller and von Sosen [20] considered the incompressible Navier–Stokes equations, a differential-algebraic system of higher index than one, thus requiring special care. Tiesinga, [40] used NPGS to analyze the incompressible driven cavity problem, and Love analyzed Kolmogorov and Taylor-Couette vortical flows, [25].

One of the first attempts to use RPM as an accelerator can be found in [7] and [6], where Burrage et al. applied RPM to linear systems of equations. RPM as an accelerator has also been exploited by Scascighini and Troxler [1] in inverse design for 2D and axi-symmetric inviscid flows. Campobasso and Giles, [8] and [9] used RPM to stabilize a linear flow-solver.

The first successful case known to the author of RPM applied as a pure accelerator of a non-linear problem is due to Dorobantu et al, [11], who computed steady quasi-1D inviscid nozzle flows. The experience drawn from their work was used as a basis to accelerate 2D-Compressible Navier–Stokes calculations, [30].
Many interesting physical systems consist of a lattice of discrete interacting units, such as atoms in a crystal, neurons or other cells in tissue, or electronic functional units on a chip. We often like to model their macro-scale behavior by effective continuum evolution equations, which, averaging over the micro-scales, still retain the essential features of the discrete systems. As an example, the wave equation describes acoustics but averages over motions of individual atoms.

Recently, the “coarse time stepper” approach [39] has been developed as an alternative to actual derivation of such effective equations. It uses suitably initialized short time integrations of the detailed discrete model and extrapolates to macro-scale time steps. It is plausible that the degrees of freedom in constructing a coarse time-stepper may be used to better approximate the qualitative features of the discrete system than does an effective continuum model produced by analytical averaging. However, a continuum effective evolution equation has the advantage of allowing mathematical manipulation and use of a well-developed set of tools for studies of the model behavior in phase space. The current challenge, addressed in a rapidly growing number of papers, e.g. [39, 32, 21] is to carry out such studies without formulation of an effective continuum model, using only a coarse time stepper and matrix-free analysis techniques.

The system studied in this paper is a linear lattice of non-linear reaction sites coupled by diffusion over the inter-unit distance $\Delta x$. The lattice supports traveling reaction fronts, with speeds related to $\Delta x$. In particular, if $\Delta x$ is larger than a critical value, there are no traveling waves: the motion stops (“front pinning”), corresponding to a bifurcation in phase space. Note that the obvious reaction-diffusion continuum model does not have such a bifurcation, only a wave speed increasing with diffusional coupling strength. The effective computational process
of the coarse time stepper smears the bifurcation into a continuum transition, an acceptable and possibly optimal way to represent the discrete bifurcation.

The paper presents a phase-plane analysis of a coarse time stepper continuum model for a particular linear reacting lattice. It explains the construction of the coarse time stepper, and the use of RPM with arc-length continuation to trace the bifurcation diagram for the so defined effective (but analytically unavailable) equation. Initially, determination of the front propagation speed is reformulated as a fixed-point problem, amenable to RPM. Likewise, the “lifting” operator from a continuum profile to a set of discrete solutions, taking into account the translational invariance of the continuum model, is defined to satisfy the necessary algebraic relations with the grid sampling operator.

The relation of the type of bifurcation possessed by the discrete model, a Saddle-Node Infinite period, [22], to the smooth transition of the effective model is discussed. In particular, the continuum is operationally defined by averaging a number of shifted copies of the front profile sampled on the grid, and the effect of the number of copies is investigated.

As already noted in the introduction, Ch. 1, the dominant application of RPM is in the field of bifurcation and continuation. Here we find the initial paper by Shroff & Keller [34], where the method is introduced. The algorithm was further developed by Lust [26] and Lust et al., [27], who defined two classes of related schemes, the Newton Picard Gauss Seidel (NPGS), and Newton Picard Jacobi (NPJ) schemes. The method is used to compute stable and unstable orbits in Love [25], and Luzyanina, Engelborghs, Lust and Roose, [28], used NPGS to compute periodic solutions of delay-differential equations. Schwetlick et al. proposed an adaptive block elimination method based on RPM for solving the linear systems of Newton iteration of non-linear parameter dependent equations. Javanosky and Liberda [14] consider a projected RPM method, applying the RPM-technique directly to the right hand side of the semi-discrete form \( \dot{u} = G(u, \lambda) \) of the partial differential equation. Restricting the treatment to symmetric operators, they use a Cayley transform to improve convergence of the basis updates. Their method requires explicit knowledge of the Jacobian \( G_u \).

Noorden et al. apply variants of the Newton-Picard scheme in [42], [43], and [41] to computational chemistry applications. For RPM applied to bifurcation studies of CFD-related topics, see Ch. 2.
Chapter 4

Recursive Projection Method for Efficient Unsteady CFD Simulations; Paper III & Paper IV

Most fluid flows are inherently unsteady and require time-dependent simulations. Unsteady computations are in general very CPU-intensive, so algorithmic savings of computer time are necessary to pave the way for more wide-spread adoption of unsteady CFD in design and analysis. Papers III and IV introduce RPM for unsteady flows and demonstrate worthwhile savings.

A commonly used approach to solving the time-accurate Navier-Stokes equations is dual time stepping [13], where a residual equation is solved iteratively to reach a steady-state in the inner loop of every physical (outer) time step. Another approach is to solve the residual equation using Newton-Krylov methods, see e.g. [23], [29] and [5].

The starting point for dual time-stepping is the semi-discrete form:

$$\frac{dy}{dt} + R(y) = 0$$  \hspace{1cm} (4.1)

where $y$ is the state vector of conserved variables, and $R$ is the "residual", the flux imbalance of the various conserved quantities in the computational cells. A second-order accurate implicit backward difference formula (BDF) for the time derivative in Eq. (4.1) yields a fully discrete approximation:

$$\frac{3}{2\Delta t}y^{n+1} - \frac{4}{2\Delta t}y^n + \frac{1}{2\Delta t}y^{n-1} + R(y^{n+1}) = 0$$  \hspace{1cm} (4.2)

where $n - 1$, $n$ and $n + 1$ indicate the previous, present and future time level, respectively.

Because of the presence of $R(y^{n+1})$, Eq. (4.2) is a non-linear system of coupled equations, which has to be solved iteratively at each time step. The desirable
stability properties of the BDF(2) scheme - it is both A- and L-stable - enable much larger time-steps than for explicit methods.

Recently some attention has been given to *implicit* Runge-Kutta schemes, [4] and [19]. These methods require solution of multiple (or larger) non-linear systems, but enjoy higher temporal accuracy and easier time-step adaption.

The idea behind dual time stepping is to transform Eq. (4.2) into a steady-state problem by adding a set of pseudo-time derivatives to the left-hand side,

$$\frac{dw}{d\tau} + \left[ \frac{3w - 4y^n + y^{n-1}}{2\Delta t} + R(w) \right] = 0$$ (4.3)

and to advance $w$ in pseudo-time $\tau$ by some suitable method until convergence. On convergence, the pseudo-temporal terms vanish, and we take

$$u^{n+1} = \lim_{\tau \rightarrow \infty} (w(\tau))$$

To reduce the number of inner iterations, Hsu and Jameson [12] proposed an implicit-explicit hybrid scheme using dual time-stepping to correct the solution obtained from an ADI-formulation of the equations.

In our work, the idea is to reduce the number of inner iterations by applying RPM to (4.3). For the overall RPM procedure to be efficient, basis identification must be fast but still sufficiently accurate for RPM to behave well.

Assume that the solution at time step $n$ has been computed and there is an approximate basis $V$ for the slow subspace. At the next step $n+1$, there is a choice:

1. **Restart:** Before start, delete the old basis $V$.

2. **Save the basis:** Take $V$ to approximate the new basis.

   RPM continuously seeks to add vectors to the basis, which grows and eventually must be either updated (without growth) or discarded (and started anew).

   In the numerical experiments, in most cases the restart procedure was used, see also Ch. 1. A mix of the two approaches was also tested, with only slight increase in efficiency.

   The papers present the method and its implementation into the compressible flow solver EDGE for unstructured grids of arbitrary elements, jointly developed by the Swedish Defence Research Agency (FOI) and KTH. RPM sends a vector $y$ into the flow solver, which returns $F(y)$. A Matlab implementation of RPM was linked to EDGE through an interface, using the FFA Matlab Toolbox [38], by reading and writing files, see Fig.(4.1)
Figure 4.1: The link between the flow-solver and RPM.
In paper III, the implementation is evaluated by computing the periodic self-excited viscous flow over a cylinder for $Re = 100$. The computations could be accelerated by a factor of 2.

Paper IV studies a periodic self-excited turbulent flow around an 18% thick circular-arc airfoil in free flow at $M_{\infty} = 0.76$, $\alpha = 0^\circ$ and $Re = 11 \times 10^6$. The demanding nature of this flow problem was illustrated by Wang [45].

These conditions give a periodic, $180^\circ$ out-of-phase motion of the shocks over the upper and lower surface of the airfoil. The unsteadiness is driven by the interaction between the shocks, the boundary layer, and the vortex shedding in the wake. The Reynolds-averaged (RANS) Navier–Stokes equations were coupled with the EARSM turbulence model on a 2D hybrid grid with 12,000 nodes generated by ICEM CFD.

RPM cut the compute time by about a factor of 2.5.

It was believed that once a basis had been identified it could be retained for several steps, but this was not the case. The basis should be kept for at most one or two steps, hence restart was often the preferable strategy. In steps where the basis indeed could be re-used, RPM accelerated the convergence by as much as 5 times. This indicates that basis updating is a worthwhile issue to pursue.
Chapter 5

New Implementations of the Implicitly Restarted Block Arnoldi Method; Paper V

There is a large variety of applications where large-scale eigenvalue problems appear. Typically one is only interested in computing a small number $k$ of the $N$ eigenvalues. The problems of interest here are too large for direct diagonalization methods, like the QR-scheme. Krylov methods, on the other hand, are well suited for this purpose, needing only the action of the matrix on vector, and not the matrix itself. Via a starting block, a Krylov method generates an orthonormal basis $V_{mb} \in \mathbb{C}^{N \times mb}$, $V_{mb}^H V_{mb} = I_{mb}$ which spans the Block Krylov subspace

$$\mathcal{K}_{mb}(A; V_1) \equiv \text{span}\{V_1, AV_1, \ldots, A^{m-1}V_1\},$$

where $V_1 \in \mathbb{C}^{N \times b}$, $V_1^H V_1 = I_b$, is an orthonormal matrix. The Block Arnoldi decomposition is given by

$$AV_{mb} = V_{mb}H_{mb} + F_mE_{mb}^T,$$  \hspace{1cm} (5.2)

where in the case of $b = 1$, the classical single vector Arnoldi decomposition is obtained. The Arnoldi algorithm yields a banded upper Hessenberg matrix $H_{mb}$, which can be viewed as a restriction of $A$ to the Krylov subspace, [2].

With increasing $m$, storage and computational requirements increase and restart becomes necessary. For the single vector case [36], later generalized by Lehoucq to Block Arnoldi [24], Sorensen proposed an implicit restarting of the Arnoldi-method. A shifted QR-iteration on the Hessenberg matrix produces an improved starting block $V_1^+$. It is equivalent to an (implicit) application of a polynomial $\Phi(A) = (A - \mu_1 I) \cdots (A - \mu_{mb-p} I)$ on the starting matrix, where $m = p - r$, and $r$ is the number of remaining blocks of vectors after restart. When shifts $\mu$ are chosen among the unwanted eigenvalues of $H_{mb}$, these very eigenvalues will be de-emphasized after
restart. For $b > 1$ only a subset of the unwanted eigenvalues can be used as shifts, rendering a whole family of possible restarted Arnoldi decompositions.

Stewart proposed a different restart method for single-vector Arnoldi. Using that the Arnoldi decomposition is uniquely determined by the starting vector, he showed that the same compressed decomposition can be obtained via row-wise applications of Householder-transforms. For block-sizes larger than one, the two approaches are not equivalent.

When an eigenvalue has converged, it should be "deflated", i.e. take no further part in the iteration. There are two cases: if it belongs to the wanted set, it should be locked, else it should be purged from the decomposition. Let the number of eigenvalues to be deflated be $l = l_l + l_p$, where $l_l$ is the number to be locked, and $l_p$ the number to be purged. In the case of Stewart’s algorithm, (5.2) is first transformed to Schur form,

$$AU_{mb} = U_{mb}T_{mb} + F_{mb}E_{mb}^TW_{mb},$$
$$U_{mb} = V_{mb}W_{mb},$$

(5.3)

Eigenvalues to be locked are moved to the top, whereas the ones to be purged are moved to the bottom. Since the restart is accomplished by discarding the $mb-rb-l_l$ columns, the $l_p$ directions are suppressed from the decomposition. By successive application of row-wise Householder transforms, the decomposition is returned to Hessenberg form and the Arnoldi process can be restarted.

We showed how the same deflation procedure can be used together with the Lehoucq implicitly restarted Block Arnoldi iteration. Initially the process is the same as the one described above. First transforming (5.2) to (5.3), the $l_l$ converged eigenvalues are moved to the top, and the $l_p$ to the bottom of the Schur matrix. Then the decomposition is returned to Hessenberg form via Householder transformations, giving an Arnoldi decomposition of order $mb - l_p$.

After restart, we wish to have an Arnoldi decomposition of order $rb + l_l$, where the active part is $rb$. If $p$ shifts are to be applied, the Arnoldi decomposition must be extended before it can be restarted, see Fig.(5.1). The decomposition can then be restarted via shifted QR. If there is an odd number of complex eigenvalues among the shifts, one extra shift should be applied. In the case $r = 1$, this corresponds to a "full" restart, that is upon restarting the Arnoldi iteration again we only have the new starting block $V_1$.

The two different versions of Implicitly Restarted Arnoldi were tested on several matrices from the Harwell-Boeing collection. Numerical experiments showed that restart based on shifted QR showed better convergence than the restarted Block Arnoldi based on Householder transforms. Results obtained for the shifted QR restart were similar to those of Lehoucq and Maschhoff in [24].

Measuring efficiency in the number matrix-vector products, almost all experiments suggested that the single-vector Arnoldi was more effective than block Arnoldi.
Figure 5.1: The Arnoldi decomposition with $l_i$ locked eigenvectors after extending the order $mb - l_p$ decomposition with $l$ vectors.
Chapter 6

Conclusions

The thesis investigates the Recursive Projection method applied to both CFD and Coarse Time Stepper models. It is shown how RPM can accelerate the solutions of steady-state applications. RPM reduced the compute time by a factor between 2 and 5. The importance of scaling the state-vector has been discussed. Scaling reduced the skewness of the projections and improved robustness of RPM.

By use of dual-time stepping, the same techniques could be used in unsteady simulations successfully. Convergence in each time step was reached 2-3 times faster than without RPM. The hope that a basis could be retained for several steps was thwarted by the results of numerical experiments. The basis could at most be re-used once. In the steps when the basis indeed could be re-used, large savings of function evaluations were observed. This motivates the further study of different basis updating techniques.

We investigated the performance of Block Arnoldi methods, and derived a new flexible implementation of the restart procedure. It was found that in general it is preferable to use the single vector method, but some applications suggested that block methods can compete, so further work is necessary to close the issue.
Chapter 7

Outlook and Future Work.

RPM has been demonstrated as an easily applicable technique for stabilization, convergence acceleration, and root following for problems which can be cast as fixed-point iterations. A number of questions have been raised in the course of the development reported here. The most important concerns effective methods for subspace identification. For unsteady flow calculations, the state changes only little in each time-step, yet the basis identified in one step was, unexpectedly, not often useful in the next. Various techniques have been tried to update the basis, ranging from formulation and approximate solution of a Riccati matrix equation for the best basis update,[35], via simplified version with the corresponding Sylvester equation to more heuristic low-rank update attempts. Since the update, when it is useful, really saves work, further investigation along these lines is indicated, for instance to construct a method that automatically decides when a basis cab be re-used, when it should be updated, and when RPM should be restarted.

As the next point, the scaling of variables has been shown to be of importance. The scaling should be chosen to make all state vector components be of the same order of magnitude. Another goal is to decrease, by scaling, the spectral condition number of the Jacobian to improve the basis sensitivity to perturbations. The matter needs to be more analyzed in order to understand how the scaling affects the overall performance of the algorithm.

There is no guarantee that RPM preserves positivity variables. A positivity scheme that damps the RPM correction to ensure that e.g. $k, \epsilon > 0$ can be a solution. This scheme must be constructed such that it is consistent with the projections, is far as the author know such a scheme does not exists.

We have showed how RPM has successfully accelerated both steady and unsteady 1D and 2D CFD-applications in small test-problems. The next step is to apply RPM to a 3D-real world application problem.

The natural choice of updating the basis is the Block Arnoldi method, though there are are other methods such as Jacobi-Davidson and two-sided Lanczos. Numerical experiments indicate that block Krylov methods can not compete with
single-vector algorithm if efficiency is measured in number of matrix-vector products. On the other hand, the convergence rate of the method depends crucially on the choice of shifts, and further increase in efficiency is certainly to be expected if a better theory can be derived. Furthermore, since block methods enables the use of BLAS-3, measuring the efficiency in wall clock time, the efficiency picture might be totally different when performing parallel eigenvalue computations.
Bibliography


