University of Bath

PHD

Iterative methods for augmented linear systems

Benbow, Steven James

Award date:
1997

Awarding institution:
University of Bath

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal ?

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Download date: 22. May. 2019

Iterative Methods For Augmented Linear Systems

submitted by

Steven James Benbow

for the degree of Ph.D.

of the

University of Bath

1997

COPYRIGHT

Attention is drawn to the fact that copyright of this thesis rests with its author. This copy of the thesis has been supplied on the condition that anyone who consults it is understood to recognise that its copyright rests with its author and that no quotation from the thesis and no information derived from it may be published without the prior written consent of the author.

This thesis may be made available for consultation within the University Library and may be photocopied or lent to other libraries for the purposes of consultation.

Signature of Author .... S Benbow ....

Steven James Benbow

UMI Number: U601489

All rights reserved

INFORMATION TO ALL USERS
The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript
and there are missing pages, these will be noted. Also, if material had to be removed,
a note will indicate the deletion.

UMI U601489
Published by ProQuest LLC 2013. Copyright in the Dissertation held by the Author.
Microform Edition © ProQuest LLC.
All rights reserved. This work is protected against
unauthorized copying under Title 17, United States Code.

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

UNIVERSITY OF BATH

SCHOOL OF MATHEMATICAL SCIENCES

Iterative Methods For Augmented Linear Systems

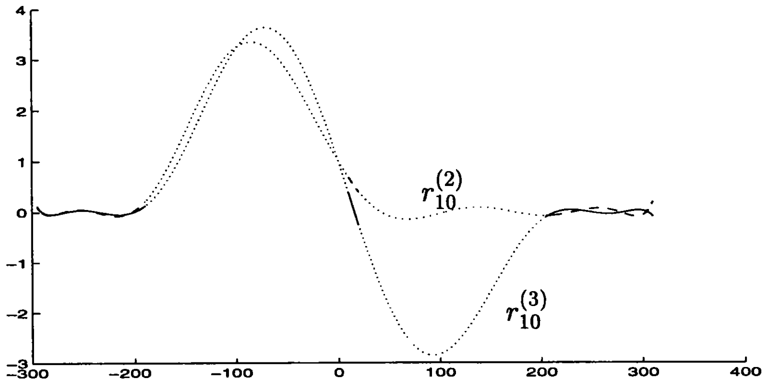
submitted by

Steven James Benbow

for the degree of

Ph.D.

of the University of Bath, 1997



I would like to thank the following people who, in their own way, have all contributed to this thesis and life in general over the past few years :

- Alastair Spence, my supervisor, for all his help and suggestions, and for having the confidence in me to let me explore my own ideas. We never did get round to having a game of tennis, but perhaps that's just as well ...
- Adrian Hill, a constant source of mathematical gems and an excellent friend since my early MSc days, for his many helpful discussions, many more pints, and for being the only person to come dressed as Harry Palmer.
- Andrew Cliffe, my industrial supervisor at AEA Technology, for finding time to explain some of the real-life difficulties in groundwater flow modelling that I had completely missed.
- Andy Wathen, for his help and pointers to a great deal of relevant literature, his interest in my work, and for inviting me to speak at Oxford.
- My officemates and friends
  - Chris Brooking, for endless computing help in the early days, teaching me to ski, and the odd Bass and Madras - as he puts it.
  - Michelle Orme, for putting up with Chris and myself, and always having a smile on her face. How she did it I'll never know.
  - Rob Laister, for his superb guidance in all things non-mathematical.
  - Steve, Andy, Stuart, Richard and Jon, for their patience when I hogged the computers for weeks on end.
  - Mat, for being an excellent friend and top laugh, ever since those days in Eastwood.

Lastly, but mostly of all :

- My parents, for all their help and encouragement since before my undergrad. days and right up to the present. Without their support and guidance, none of this would have been possible.
- Emma, for keeping me sane during these last few months of writing up, for looking after me and stopping me from scratching during my unfortunate spell of chickenpox, and for saying yes.

Contents

1 **Introduction** 1

1.1 Darcy's groundwater flow model 2

1.2 Iterative solution methods for symmetric linear systems 3

1.3 Chapter descriptions 5

2 **Polynomial based methods based on new eigenvalue information** 13

2.1 The classical Chebyshev semi-iterative method 14

2.2 A more general Chebyshev method 17

2.2.1 Optimal polynomials over two intervals which are symmetric about the origin 20

2.3 Distribution of eigenvalues of augmented systems 23

2.4 Least-squares characterisation of residual polynomials 28

2.4.1 Orthogonal polynomials on one interval 31

2.5 Generating orthonormal polynomials over three intervals 34

2.5.1 Generalisation of Saad's Stieltjes procedure to the generation of orthonormal polynomials over three intervals 35

2.5.2 An orthonormal polynomial basis for P_k' 39

2.5.3 Calculation of the modified moments φ_i 41

2.6 Iterative solution of augmented systems via least-squares residual polynomials 42

2.6.1 A minimisation property 45

2.7 Convergence of the generalised Legendre method 46

2.8 Numerical results 49

2.9 Inexact eigenvalue bounds and least-squares polynomial preconditioners for augmented systems 57

2.9.1 Gershgorin bounds on three intervals 59
2.9.2 Polynomial preconditioning results 62
2.10 Summary 67

3 A generalised least squares approach to solving augmented systems 69

3.1 Introduction 69
3.2 The generalised least-squares connection 71
3.3 Golub-Kahan bidiagonalisation and LSQR 71
3.3.1 The Lanczos process 72
3.3.2 Golub-Kahan bidiagonalisation and LSQR 74
3.3.3 The LSQR algorithm 78
3.4 Solution of generalised least-squares problems 82
3.4.1 A bidiagonalisation procedure with a different inner-product . . 83
3.4.2 Extension of the bidiagonalisation procedure to an iterative solution method 86
3.4.3 Preconditioning 87
3.5 Comparing LSQR(A^{-1}) with preconditioned Krylov methods 90
3.6 Numerical experiments and implementation 94
3.7 The case $A = D$ 97
3.8 Summary 102

4 Vavasis type finite elements, LSQR(D^{-1}) and preconditioning 104

4.1 Finite element methods for elliptic problems 106
4.2 Mixed elements based on unmixed approximations 114
4.3 Higher order Vavasis type elements 120
4.4 Preconditioning LSQR(D^{-1}) 121
4.5 Diagonal scaling and additive Schwarz preconditioners 123
4.5.1 Diagonal scaling 123
4.5.2 Additive Schwarz preconditioning 123
4.5.3 Spectral dependence on k 125
4.6 Incomplete LU preconditioners 129
4.7 Numerical results and a note on incomplete preconditioners 135
4.8 Summary 144

5 Mixed finite elements for the groundwater flow equations 147

5.1 Abstract mixed variational formulation of the groundwater flow equations 148

5.2 Abstract mixed finite element approximation 153

5.3 Raviart-Thomas mixed finite elements 155

5.4 Diagonal scaling of scaled mass matrices 157

5.5 Computation of Raviart-Thomas basis functions 160

5.5.1 Large aspect ratios and ill conditioning 163

5.6 Preconditioning of LSQR(A^{-1}) applied to mixed discretisations of the groundwater flow equations 167

5.7 Reducing the number of A^{-1} operations with a good initial guess 171

5.7.1 Numerical results 173

5.8 Summary 177

Appendix A : Further three-interval results 178

A.1 Unsteady Stokes operators 178

A.2 Simple scaling 179

A.3 Conclusion 183

Appendix B : The ΠCG algorithm 184

B.1 Preconditioning augmented systems by projection matrices 185

B.2 Nullspace methods 189

B.3 The action of $\mathcal{B}$ 191

B.4 The ΠCG algorithm 192

B.5 Convergence of the ΠCG algorithm 198

B.6 Generalisation of ΠCG and a connection with LSQR(D^{-1}) 200

B.7 Numerical results 200

B.8 Conclusions 201

Bibliography 203

Chapter 1

Introduction

Augmented linear systems of the form

$$\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix} \tag{1.1}$$

arise frequently in the numerical solution of problems in applied mathematics. For the purpose of this thesis, the matrices $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$ (typically $n \geq m$) are assumed to be large and sparse, and further it is assumed that the matrix A is symmetric positive-definite and B is of full column rank. Throughout, the notation

$$\mathcal{A} = \begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix}, \qquad \mathcal{A}_{\mathrm{LS}} = \begin{bmatrix} I & B \\ B^T & 0 \end{bmatrix},$$

may be used. Systems of the type (1.1) occur in many applications, for example in optimisation, discretisation of electrical networks and computational fluid dynamics. For a review of how these systems can arise in practise, and properties of the matrices A and B in each case see [16].

One way in which a system of the form (1.1) can occur in optimisation is in the constrained minimisation

$$\text{minimise } \frac{1}{2}x^TAx - x^Tb \text{ subject to } B^Tx = 0,$$

the variables y in (1.1) being the Lagrange multipliers in the associated saddle point problem. In optimisation applications the matrix A is often diagonal, most obviously in the context of weighted least-squares problems, and also when solving linear programming problems by interior point methods [43] where systems of the form (1.1) arise at each stage of an outer iteration, with the diagonal matrix A changing at each step and becoming increasingly badly scaled.

In discretisations of electrical network problems, the matrix A in (1.1) is typically diagonal and contains non-zero entries corresponding to resistances at nodes in the system. The matrix B in this case is a connectivity matrix which describes the geometry of the system, y is a vector of current and x a vector of potential values at the nodes, and b is a vector of external potentials applied to the system. For more details see for example [76].

Two classic examples from computational fluid dynamics in which systems of the form (1.1) can arise are Stokes' equations for incompressible flow,

$$-\mu\Delta u + \nabla p = f, \quad \nabla \cdot u = 0,$$

and Darcy's law for incompressible groundwater flow in a saturated porous medium,

$$\mu k^{-1} u + \nabla p = 0, \quad \nabla \cdot u = 0,$$

both in 2 or $3D$ together with some appropriate boundary conditions. Here the variables x of (1.1) represent the discretised velocity u, and y represent the discretised pressure p. The constant μ is the fluid viscosity and k is a permeability function. The equations are usually discretised with a mixed finite element or finite difference method, and A is, in general, not a diagonal matrix.

Although the solution methods for (1.1) will be applicable to any of the physical problems mentioned above, in this thesis the main emphasis is to construct solution methods for the case that (1.1) represents discretisations of groundwater flow problems, and so this problem will be described in more detail.

1.1 Darcy's groundwater flow model

Steady state groundwater flow of an incompressible fluid in a saturated porous medium is modelled in terms of the pressure p and Darcy velocity (or specific discharge) u. The Darcy velocity is the volume rate of flow per unit area. These two fluid properties are related by Darcy's law, and the incompressibility constraint is enforced with the continuity equation,

$$\begin{aligned} \mu k^{-1} u + \nabla p &= 0, \\ \nabla \cdot u &= 0, \end{aligned} \tag{1.2}$$

together with a mixture of Dirichlet and Neumann boundary conditions. Again, μ is the (constant) fluid viscosity and k is a permeability function. In an isotropic medium, k is taken to be a piecewise constant function, and in an anisotropic medium where the permeability is dependent on direction as well as position, k is represented by a tensor. For more details see [80]. For the purpose of this thesis all experiments are performed in isotropic media, since some of the preconditioning results presented in Chapter 4 are only applicable for this type of domain, although the iterative methods that are constructed will apply equally to discretisations of anisotropic media.

The Darcy velocity can be eliminated from (1.2) to form the pressure equation,

$$\nabla \cdot (k \nabla p) = 0, \tag{1.3}$$

which, when discretised with a finite element method forms the stiffness equations

$$K\widetilde{y} = \widetilde{b},$$

where K is a scaled stiffness matrix. The vector $\widetilde{y}$ represents the discretised pressure and will not usually be equal to y in (1.1), although it is possible to form a system of form (1.1) which corresponds to a discretisation of (1.3) and has the property that $y = \widetilde{y}$, as will be seen in Chapter 4.

1.2 Iterative solution methods for symmetric linear systems

All of the solution methods which are described in this thesis are iterative, rather than direct methods for which there is already a considerable amount of literature available. Iterative linear solvers became popular when computer speeds and storage became sufficiently great that partial differential equations could be solved to a useful level of accuracy. Although the existing direct methods could be adapted to these new problems, realistic discretisation sizes resulted in systems that became so large that the storage requirements and solution time of the direct methods made them impractical. Luckily the types of matrices that arose, although very large, were also typically very sparse so that matrix-vector product operations could be performed cheaply, and often had nice properties such as being diagonally dominant or M-matrices, for which extensive theory was already known. Iterative methods that involved the coefficient matrix only in terms of matrix-vector operations, or also involved simple solves with components of the coefficient matrix (e.g. its diagonal or lower triangular part) quickly became popular. Although the conjugate gradient method of Hestenes and Stiefel [36] and Lanczos' algorithm [46] were introduced at an early stage, their initial interpretation as exact projection methods led to them falling out of favour since their finite precision behaviour did not reproduce the qualitative behaviour that was expected of them. Instead the successive overrelaxation methods (SOR) and their variants [77] became the standard tools for tackling these problems. With the later interpretation of the conjugate gradient and Lanczos methods as iterative methods by Reid [62], the tide began to turn as these methods began to be adopted for large symmetric positive-definite problems. As preconditioners became better understood the conjugate gradient methods became even more efficient. The solution of arbitrary symmetric systems, not purely positive-definite systems, became practical with the development of the SYMMLQ and MINRES algorithms of Paige and Saunders [56]. Since the introduction of these methods, few new algorithms for the solution of symmetric linear systems were proposed, one exception being the LSQR algorithm, also due to Paige and Saunders. This algorithm is mainly interpreted as a least squares solver for overdetermined systems, but can be interpreted variously as a solver for arbitrary square linear systems, and a solver for symmetric linear systems with a specific form of coefficient matrix, which will be particularly applicable to this thesis.

Where possible, the approach taken in this thesis has been to attempt to develop new iterative methods for (1.1) which exploit the structure of the coefficient matrix, rather than to seek preconditioners for (1.1) (that exploit structure) that lead existing methods of solution to perform well. There is a great deal of literature on the subject of preconditioning for general matrices, see [10] and the references therein, and preconditioners for specific augmented systems of the form (1.1) have also received attention, although this is mostly restricted to the case that (1.1) represents a discretisation of a Stokes operator, see for example [85, 72, 64]. The assumption that (1.1) arises in a groundwater flow application, and the conditions that this imposes on the matrices A and B (especially A), allows a little more flexibility when making assumptions on the matrices A and B than might be allowable (or realistic) in the case that (1.1) arises from a Stokes problem.

1.3 Chapter descriptions

As has been mentioned, the main goal of this thesis is to attempt to solve the matrix problems arising in groundwater flow applications in an efficient way which exploits the structure and properties of the coefficient matrix. The first part of the thesis, Chapter 2, approaches the problem by considering the spectrum of the coefficient matrix. In Theorem 2.3.1, a new eigenvalue result for the coefficient matrix is presented, which is a refinement of a theorem of Rusten and Winther in [64], and defines three intervals on the real line which contain the eigenvalues of $\mathcal{A}$. Theorem 2.3.2 goes on to describe how many eigenvalues can be expected in each of the three intervals. In an effort to explore whether this new eigenvalue information can be used in a practical way in an iterative solver, algorithm LS(3), for solving linear systems with coefficient matrix $\mathcal{A}$ is developed. LS(3) is an extension of the ideas that Saad described in [66], where an iterative solver for the solution of symmetric systems with spectrum contained in two intervals is derived. This is achieved firstly by devising a scheme to generate orthogonal polynomials over three intervals, and then using the orthogonal polynomials to calculate the least squares polynomials of increasing order on the three intervals. The LS(3) algorithm takes these least squares polynomials to be its residual polynomials at each step, and hence minimises the residual in a relevant norm as the iteration number increases. As opposed to the use of Chebyshev inner product in [66], the LS(3) algorithm makes use of Legendre inner product (the L^2 inner product on the three intervals), since experience indicates that this provides a better uniform minimisation property on the whole of the spectrum, rather than simply minimising the maximum value of the residual polynomial on the spectrum. Since LS(3) uses orthogonal polynomials, as opposed to orthogonal vectors as in most Krylov methods, the inner products in the algorithm are inner products of polynomials and are extremely cheap to perform.

They can be expressed as a sum of k scalars, where k is the degree of the lowest degree polynomial in the inner product, rather than resorting to numerical integration techniques. These inner products are much cheaper than the corresponding vector inner products in the standard Krylov methods for large indefinite systems, and hence the LS(3) algorithm has a much shorter iteration time than the Krylov methods CGNR and SYMMLQ with which it is compared. The fact that there are no vector inner products makes the use of algorithms which use polynomial inner products attractive for parallel machines where vector inner products are well known to cause bottlenecks in computation. The overall convergence of LS(3) is seen to be be similar to CGNR in terms of error/residual reduction per iteration, but the shorter iteration time makes LS(3) the fastest converging of the algorithms considered. The work in the chapter goes on to explain how inexact eigenvalue information can be used in the LS(3) algorithm and a condition on eigenvalue bounds for the matrices A and B which describes a three interval bound on the eigenvalues of $\mathcal{A}$ is given in Lemma 2.9.1. The final numerical results in Chapter 2 are performed in an effort to gauge how useful the residual polynomial iterations can be as a preconditioning step in the standard Krylov methods. It is seen that the SYMMLQ algorithm with a low degree LS(3) preconditioner is much faster than any of the unpreconditioned algorithms. Hence the conclusion of Chapter 2 is that the new eigenvalue information can be used effectively when solving systems whose eigenvalues are contained in three intervals.

A rule of thumb for coefficient matrices of the type $\mathcal{A}$ to have its eigenvalues contained in the three intervals described in Theorem 2.3.1, is that the eigenvalues of A must be a lot smaller than the singular values of B. This is not a typical feature of groundwater flow problems and so in order for LS(3) to be effectively applied to such problems, the matrix $\mathcal{A}$ would need to first be preconditioned in a way which scales down the eigenvalues of A. Whether any practical problems satisfy the eigenvalue criterion in Theorem 2.3.1 is unknown. One possibility is that of a discretisation of a Stokes flow operator for a low viscosity fluid, since here the matrix A is a discretisation of $\nu\Delta$ (where ν is the fluid viscosity), although no numerical experiments on such problems have been performed in this thesis.

In Chapter 3, an iterative method for solving systems of the form (1.1) is described which is more suitable for discretisations of groundwater flow problems than the method of Chapter 2 which, as is mentioned above, would require preconditioning if it were to be applied. Again the aim is to discover a solution method which exploits the structure of the problem. It is demonstrated that the system (1.1) is equivalent to a generalised least squares problem in the A^{-1} norm, in the sense that the solution of the generalised least squares problem is equal to the y component of the solution of (1.1), the x component being simple to be recover once y is known. Hence a method of solution which is based on a generalised least squares reformulation of the problem is devised. The first step taken is to examine standard least squares solvers in the Euclidean norm, and in particular the LSQR method of Paige and Saunders [58]. The LSQR method is seen to be a stable implementation of the conjugate gradient algorithm for the indefinite system (1.1) which solves subproblems involving small least squares problems (this approach does not require that the coefficient matrix is positive definite as is the case for the standard conjugate gradient method). Where the conjugate gradient method is based on a Lanczos process on the coefficient matrix, which reduces the symmetric matrix to tridiagonal form, the LSQR method is based on the Golub-Kahan bidiagonalisation process which reduces a (usually overdetermined) matrix to bidiagonal form. The Golub-Kahan process is equivalent to a Lanczos process on the matrix $\mathcal{A}_{\text{LS}}$ but has the novel feature that, for a suitable starting vector, it requires half the number of matrix-vector multiplications that a standard Lanczos process would, and hence can perform approximately two Lanczos steps for the price of one. It will be seen that standard iterative methods for the coefficient matrix which ignore this fact have a redundant step at every second Lanczos iteration. By first supposing that the Cholesky decomposition of the positive definite, symmetric matrix A is available, so that the A in the (1,1) block of $\mathcal{A}$ can be eliminated, algorithms LSQR(A^{-1}) and LSQR(A) for the solution of the generalised least squares problem are devised. It will be seen that these algorithms can be performed using A^{-1} operations, as opposed to backsolves with the Cholesky factors of A, by altering the inner product with respect to which one of the sets of orthonormal vectors in the Golub-Kahan bidiagonalisation process is orthonormal. Both algorithms can be viewed as stable implementations of the conjugate gradient algorithm on the Schur complement equations. The algorithms are compared with another iterative method for the solution of (1.1) which also assumes that A^{-1} operations are possible, specifically the MINRES algorithm applied to (1.1) with a block diagonal preconditioner containing A in the (1,1) block. It has been shown by Fischer *et. al.* [22] that if MINRES is applied in this way, there is a redundancy in every second step of the algorithm of the type mentioned above. Since the LSQR(A^{-1}) (and LSQR(A)) algorithm inherits the property of being able to perform two Lanczos steps at the cost of approximately one, the redundancies in the MINRES approach are stepped over by LSQR(A^{-1}) , so that the LSQR(A^{-1}) approach requires approximately half the amount of work. A similar trick to the change of inner product above, on the second set of orthonormal vectors generated by the Golub-Kahan process, allows preconditioning of LSQR (and LSQR(A^{-1})) to be viewed in a different light from that in [58]. Paige and Saunders originally only considered right preconditioners in their least squares problems. With this second change of inner product, central preconditioners for the Schur complement equations (normal equations in the case of LSQR) can be applied, in a similar way that central preconditioners are applied in the conjugate gradient algorithm. This allows more exotic types of preconditioners to be used. For example in the LSQR algorithm, preconditioners for the symmetric positive definite matrix B^TB become available, rather than only right preconditioners for the overdetermined matrix B. Since the operator defined in the Schur complement equations is a discretisation of a scaled Laplace operator for the case that (1.1) represents a discretisation of (1.2), many standard preconditioning strategies such as additive Schwarz [12, 34, 33] and incomplete factorisations [51, 48] become available in the LSQR(A^{-1}) algorithm. If only right preconditioning were possible it would be necessary to form the Cholesky factorisation of the preconditioners mentioned above in order to apply them to the problem. The toolbox of preconditioners for the matrix B (or $A^{-\frac{1}{2}}B$) in the groundwater flow applications is not so diverse as that for $B^TA^{-1}B$, and so the new preconditioning approach is a far more practical one. The work in this chapter is summarised in the preprint [5].

Since the LSQR(A^{-1}) method developed in Chapter 3 requires a solve with A at every iteration, it is essential that these solves can be performed quickly and easily. In Chapter 4 a finite element method for scaled Laplacian problems of the form (1.3) due to Vavasis [79] is described. This finite element method can be considered to be a 'semi-mixed' finite element method for the groundwater flow equations since it provides an approximation to both the pressure and velocity components of the solution. The linear system arising in this method has the form (1.1) with the matrix A diagonal, and hence is ideally suited to the LSQR(A^{-1}) algorithm, as the solves with A in this case are trivial. This finite element method is not a truly mixed finite element method since there is no choice available for the velocity space, it must be taken to be the space spanned by the derivatives of the pressure basis functions (hence the term 'semi-mixed'), and hence the velocity approximation is one order lower than the pressure approximation. This is in contrast to the usual mixed finite element approach where accurate approximations of velocity are more desirable. Vavasis' finite element method is equivalent to the usual finite element method for the scaled Laplace equation, which is typically solved with the conjugate gradient algorithm, in the sense that it produces the same pressure approximation. The stiffness matrix system that arises in the standard approach is actually the Schur complement equations of Vavasis' discretisation and hence is more poorly conditioned. Experiments on groundwater flow problems show the Vavasis and LSQR(A^{-1}) discretisation and solution approach to be more stable than the usual finite element and conjugate gradient approach, for regions in which the permeability function varies by many orders of magnitude.

The Vavasis-discretised system is taken as a template in which to test preconditioners, H, in the preconditioned version of LSQR(A^{-1}) , called LSQR(A^{-1}, H^{-1}) , in the last half of Chapter 4. Two main types of preconditioners are considered, incomplete factorisations and additive Schwarz. Incomplete factorisations have been a popular choice of preconditioner for sparse systems since the early work of Stone [75] and Meijerink and van der Vorst [51]. Watts [86] and Kuiper [45] were early proponents of the use of incomplete factorisation preconditioners for the stiffness matrix systems arising in the pressure equation and found that the performance of the standard algorithms could be greatly improved with their use. A second type of preconditioner which is often used for the scaled Laplacian type problems are domain decomposition preconditioners and, in particular, additive Schwarz preconditioners. These preconditioners take advantage of knowledge of the geometry of the domain Ω and are effectively a sum of restrictions of the operator to subsets of the domain which are easily solved. Recently Graham and Hagger [34, 33] have produced some elegant results describing the spectrum of the additive Schwarz preconditioned stiffness matrix system in domains with highly varying permeabilities. They have shown that although the preconditioned systems tend to be badly conditioned (which leads to the assumption that iterative methods will perform badly) in fact only a few small eigenvalues are usually present so that iterative methods tend to perform rather better than anticipated. Graham and Hagger's results are made possible by the fact that the eigenvalues of the additive Schwarz preconditioned-scaled

Laplacian system can be bounded by the eigenvalues of the diagonally preconditioned system. In [48], Manteuffel showed that a similar result can be found which relates the eigenvalues of suitable incomplete Cholesky preconditioned systems to the eigenvalues of related diagonally preconditioned systems for any symmetric positive definite problem. This theorem is reviewed in §4.7 and its application to scaled Laplacian type systems is described in Theorem 4.7.2.

Traditional mixed finite element discretisations of the groundwater flow equations are treated in Chapter 5. Great care must be taken so that the chosen discrete velocity and pressure spaces give rise to stable discretisations. One of the most widely used (stable) mixed finite elements for groundwater flow type problems is the Raviart-Thomas mixed finite element [61]. These elements are designed so that the velocity approximation satisfies a continuity condition across element boundaries and their construction is described in §5.5. For a mixed finite element discretisation, the matrix A will no longer be diagonal, as was the case for the Vavasis elements in Chapter 4, and hence for the LSQR(A^{-1}) method to be an appropriate method of solution it must be the case that the A^{-1}operations at each step can be performed efficiently. The matrix A is a mass matrix of velocity basis functions which are scaled by the permeability function k, and two factors hinder iterative solution methods for such systems. The first is that the matrix A can be extremely badly conditioned because of the bad scaling in the permeability function, the values of which can vary by many orders of magnitude over the domain. The second factor is that, due to the nature of the typical domains upon which groundwater flow is modelled (generally regions that are wide and long but of much smaller depth), finite element cells with large aspect ratios will be present in the discretisation. This can again lead to mass matrices which are very badly conditioned. The first point above can be annulled completely however, due to a theorem of Wathen [82], all the effects of the permeability function can be removed by preconditioning A by its diagonal. The second point is not so easy to address, however experiments with the Raviart-Thomas elements are given in §5.5.1 which tend to suggest that the diagonally preconditioned Raviart-Thomas mass matrices are not so badly conditioned as might be expected form the theory. The preconditioners developed for the Vavasis finite elements are applied to the mixed finite element discretised problems in §5.6. In an effort to reduce the number of A^{-1} operations, which is the most time-consuming part of the LSQR(A^{-1}) iteration, a method is described in §5.7 whereby first a system with

A replaced by its diagonal is solved (an easy task for LSQR(A^{-1})) and then a system with $\mathcal{A}$ as coefficient matrix is solved for the correction which is required to update the first solution to a solution of the mixed problem. Numerical results for the MAC finite element / finite difference scheme [26] (another standard element for groundwater flow problems) indicate that this is an appropriate method for large systems.

The two appendices A and B contain work relating to the main chapters in the thesis, and other ideas that have yet to be fully explored. Appendix A contains further results based on the eigenvalue result Theorem 2.3.1, in particular the effect of scaling the matrix A to optimise the condition of $\mathcal{A}$ is considered, and its effects on the three intervals is seen. In Appendix B, another algorithm, called ΠCG , is described which is another method of solving (1.1) which exploits the structure of the system. It can be seen that the velocity component of the discretised solution lies in the nullspace of B^T (which is simply the discrete interpretation of the fact that the velocity component of (1.2) is divergence free). Methods which make use of this property are called nullspace methods. It is usual in nullspace methods to have to compute a basis for the nullspace of B^T, a very expensive task in terms of time and storage when (1.1) is large. The ΠCG algorithm avoids the need to compute such a basis by instead solving a system with B as coefficient matrix at each step of an outer iteration. The outer iteration is of conjugate gradient type, which cannot be directly applied to the matrix $\mathcal{A}$ since it is indefinite. However the solves with the matrix B force the iteration vectors to lie in a subspace of the domain of $\mathcal{A}$ with respect to which the matrix $\mathcal{A}$ is positive definite. The ΠCG algorithm can be viewed as an analogue of LSQR(A^{-1}) in the sense that LSQR(A^{-1}) solves a (generalised) least squares problem in B at the expense of several solves with A, whereas the reverse is true of ΠCG . For groundwater flow problems the solves with the mass matrix A are easy compared to solves on the least squares problem, and so LSQR(A^{-1}) is the more competitive of the two algorithms. In the case of a discretisation of a Stokes flow however, it may be the case that ΠCG is the more competitive since then A represents a large Laplacian problem, and the solves with B are equivalent to a smaller Laplacian problem, and so it may be reasonable to perform a number of solves with least squares systems in B in order to solve the larger problem with A as coefficient matrix. Since both of the solves with A and B are Laplacian type problems, ΠCG can be viewed as a multilevel solver for the Stokes problem. The ΠCG algorithm needs further exploration before it can be presented fully and is hence confined to Appendix B.

Chapter 2

Polynomial based methods based on new eigenvalue information

In this chapter, a method which assumes *a priori* information about the distribution of the eigenvalues of the symmetric indefinite coefficient matrix in the equation

$$Mz = f,$$

is explored. The search for such a method is motivated by the eigenvalue result in §2.3. It is shown that three intervals on the real line can be identified which contain all of the eigenvalues of $\mathcal{A}$ where

$$\mathcal{A} = \begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix}. \tag{2.1}$$

Here $A \in \mathbb{R}^{n\times n}$ is symmetric positive-definite and $B \in \mathbb{R}^{n\times m}$ is of full column rank (so that in particular $n \geq m$). $\mathcal{A}$ is symmetric and indefinite hence its eigenvalues are contained in two intervals on the real line, one entirely negative and one entirely positive, and bounds for these intervals are known (see [64]). It is shown that the positive interval given by the standard bounds can contain a redundant subinterval which is devoid of eigenvalues of $\mathcal{A}$, thus a more descriptive representation of the eigenvalue distribution of $\mathcal{A}$ would be the union one negative and two positive intervals.

It would seem appropriate to examine methods which can take advantage of this eigenvalue structure. Before attempting this some background is given on iterative methods which use eigenvalue information to achieve convergence. In §§2.1 and 2.2 an introduction to the classical Chebyshev semi-iterative method and a generalisation are given, in order that similarities in the approach of this method with that of the generalised-Legendre approach given can be seen. The new eigenvalue results are presented in §2.3 and in §2.4 a residual reduction property which can easily be enforced over the three eigenvalue intervals is described, together with some elementary results concerning orthogonal polynomials on one interval. In §2.5 the task of generating orthogonal polynomials over three intervals is approached and in §2.6 an iterative solver based on this theory is introduced. §2.7 contains results pertaining to the convergence of the solver and in §2.8 some numerical results are presented. Finally in §2.9 the running assumption that the three eigenvalue intervals can be found exactly is loosened, and conditions such that Gershgorin eigenvalue information for the matrices A and B defines three intervals which bound the three eigenvalue intervals are found. Results based on inexact eigenvalue intervals and polynomial preconditioning are given which show that methods based on three inexact eigenvalue intervals are still effective.

2.1 The classical Chebyshev semi-iterative method

Suppose that it is required to solve the symmetric linear system,

$$Mz = f, \tag{2.2}$$

where $M \in \mathbb{R}^{N \times N}$. Consider the *splitting* $M = F - G$ of the matrix M where F is nonsingular, and an iteration of the form

$$Fy_{k+1} = Gy_k + f. \tag{2.3}$$

Clearly, if y_l is a stationary point of this iteration for some value l, then $My_l = f$ so that $z = y_l$ is the solution of (2.2). As may be suspected, arbitrary splittings of the matrix M will not necessarily lead to converging iterations of the form (2.3). A necessary and sufficient condition for the splitting to lead to a converging method for all choices of y_0 is that $\rho(F^{-1}G) < 1$ where ρ denotes the spectral radius. This can be seen as follows. Defining the error at the kth step by $\varepsilon_k = y_k - z$, and using the fact that z solves (2.2), it can be seen that $F\varepsilon_k = G\varepsilon_{k-1}$ and so

$$\varepsilon_k = \left(F^{-1}G\right)^k \varepsilon_0. \tag{2.4}$$

Hence the error will tend to zero provided that $(F^{-1}G)^k \longrightarrow 0$. Varga [77] defines such matrices $F^{-1}G$ to be *convergent* and shows that a necessary and sufficient condition that $(F^{-1}G)^k \longrightarrow 0$ is that $\rho(F^{-1}G) < 1$.

Thus the iteration (2.3) will converge to the solution of (2.2) provided that $\rho(F^{-1}G) < 1$, and the smaller the spectral radius of $F^{-1}G$, the faster the asymptotic convergence will be. Of course it is also assumed that the system (2.3) is easily solvable for z_{k+1}, for example, when the matrix F is diagonal or triangular. Taking $F = D := \text{diag}(M)$ realises the Jacobi iteration (see [77]),

$$Dz_{k+1} = -(L+U)z_k + f,$$

where L and U are the below and above diagonal part of M. Then strict diagonal dominance of M is enough to ensure that $\rho(D^{-1}(L+U)) < 1$ (see [77, Theorem 3.4]).

It may be the case that the convergence of the iterative method (2.3) can be accelerated by forming, at the $k+1^{\text{th}}$ step , a new solution estimate z_{k+1} which is a linear combination of the solution estimates $y_0, \ldots, y_{k+1}$ rather than simply taking the approximation of the solution of (2.2) to be the new solution estimate. Then

$$z_{k+1} = \sum_{i=0}^{k+1} \mu_i^{(k+1)} y_i, \tag{2.5}$$

for some $\{\mu_i^{(k+1)}\}_{i=0}^{k+1}$. The method defined by (2.3) and (2.5) is called a *semi-iterative method*, since it comprises an iterative step (2.3) and an algebraic step (2.5). A constraint on $\{\mu_i^{(k+1)}\}_{i=0}^{k+1}$ can be found upon noticing that if the initial estimate $y_0 = z$, then the error recurrence (2.4) implies that $y_k = z \ \ \forall k \geq 0$. Then as z_{k+1} is designed to be an improvement on y_{k+1}, $z_k = z \ \ \forall k \geq 0$. This observation leads to the constraint

$$\sum_{i=0}^{k+1} \mu_i^{(k+1)} = 1. \tag{2.6}$$

Defining the error in terms of the new solution estimate z_{k+1} by $e_{k+1} = z_{k+1} - z$, it can be seen that $e_{k+1} = \sum_{i=0}^{k+1} \mu_i^{(k+1)} \varepsilon_i$, therefore

$$e_{k+1} = \left(\sum_{i=0}^{k+1} \mu_i^{(k+1)} (F^{-1}G)^i \right) . \varepsilon_0,$$

and so

$$||e_{k+1}|| \leq \left|\left|p_{k+1}(F^{-1}G)\right|\right| \, ||\varepsilon_0||, \tag{2.7}$$

where $p_{k+1}(t) = \sum_{i=0}^{k+1} \mu_i^{(k+1)} t^i$. Notice that (2.6) implies that $p_{k+1}(1) = 1$. If it is assumed that $F^{-1}G$ is symmetric (or normal - see [29]) then $\left|\left|p_{k+1}(F^{-1}G)\right|\right|_2 = \rho(p_{k+1}(F^{-1}G)) = \max_{1 \leq i \leq N} |p_{k+1}(\lambda_i)|$ where $\{\lambda_i\}_{i=1}^N$ is the set of eigenvalues of $F^{-1}G$. Therefore in order to ensure that the quantities $||e_{k+1}||$ reduce quickly it would be wise to choose p_{k+1} to be the solution of the *minimax* problem

$$\min_{p \in P_{k+1}, p(1)=1} \left(\max_{1 \leq i \leq N} |p(\lambda_i)| \right),$$

where P_{k+1} denotes the set of polynomials of degree less than or equal to $k+1$. It is not realistic to expect that all of the eigenvalues $\lambda_1, \ldots, \lambda_N$ will be known. However since $F^{-1}G$ is assumed to be convergent it is certainly known that $\exists \lambda_{\min}, \lambda_{\max}$ such that $-1 < \lambda_{\min} \leq \lambda_i \leq \lambda_{\max} < 1 \; \forall i = 1, \ldots, N$, and it is not quite so unrealistic to expect to be able to bound the values $\lambda_{\min}$ and $\lambda_{\max}$ below and above respectively (for example by an application of the Gershgorin theorem - see [77, Theorem 1.5]). The interval $[\lambda_{\min} \, , \, \lambda_{\max}]$ is said to be an *inclusion set* (superset) for the set of eigenvalues of $F^{-1}G$. If Ω is defined by $\Omega = [\lambda_{\min} \, , \, \lambda_{\max}]$ then the above minimax problem can be relaxed to

$$\min_{p \in P_{k+1}, p(1)=1} \left(\max_{t \in \Omega} |p(t)| \right).$$

Since the constraint point $1 \notin \Omega$ the solution of this problem is simply the degree $k+1$ *Chebyshev polynomial* on Ω (see [63]), scaled so as to satisfy the interpolatory constraint, i.e. if $T_{k+1}(t, \Omega)$ denotes the $k+1^{\text{th}}$ Chebyshev polynomial on Ω then

$$p_{k+1}(t) = \frac{T_{k+1}(t, \Omega)}{T_{k+1}(1, \Omega)},$$

see for example [20].

Hence the solution estimate $z_{k+1} = \sum_{i=0}^{k+1} \mu_i^{(k+1)} y_i$ is found by setting $\mu_i^{(k+1)}$ to be the coefficient of t^i in the $k+1^{\text{th}}$ scaled Chebyshev polynomial on Ω. The resulting method is known as the *Chebyshev semi-iterative method.* It is not necessary to perform this whole summation at each step, it can be avoided by exploiting the three term recurrence property of the Chebyshev polynomials. For a detailed analysis of this and other related iterative methods see [30, 31].

2.2 A more general Chebyshev method

The classical Chebyshev semi-iterative method has the attractive property that it reduces the norm of the error at each iteration. However it is let down by the fact that general splittings of the matrix M are not known to be convergent, except in specific cases such as Jacobi iteration for strictly diagonally dominant systems. Another, more general, Chebyshev method is to apply the iteration so as to reduce the residual at each step. This approach entirely bypasses the necessity for splitting M at the cost of losing the error reducing property for the weaker residual reduction property. Consider now the semi-iterative method

$$z_k = z_{k-1} + d_k, \tag{2.8}$$

for (2.2), where again z_k is an approximation to the solution of (2.2) and the update direction d_k is a linear combination of the previous residuals $r_i = f - Mz_i$, $i = 0, \dots, k-1$, i.e.

$$d_k = \sum_{i=0}^{k-1} \psi_i^{(k)} r_i.$$

Then the residual at the kth step can be seen to satisfy

$$r_k = r_{k-1} - \sum_{i=0}^{k-1} \psi_i^{(k)} M r_i, \tag{2.9}$$

so that $r_k = p_k(M) r_0$ for some $p_k \in P_k$. Equation (2.9) implies that $p_k(t) = p_{k-1}(t) - t \sum_{i=0}^{(k-1)} \psi_i^{(k)} p_i(t)$ and since $p_0(0) = 1$, the residual polynomials are subject to the restriction

$$p_k(0) = 1 \ \forall k. \tag{2.10}$$

An obvious choice then for $p_k, \ \ k \geq N$ is

$$p_k(t) = \frac{C_N(t)}{C_N(0)},$$

where $C_N(t) = \det(tI - M)$ is the characteristic polynomial for M. Then the Cayley-Hamilton theorem ([49, Theorem 3.28.2]) would imply that $r_k = 0, \ \ k \geq N$. This would of course require *a priori* knowledge of all of the eigenvalues of M which again, for all but very small systems is an unrealistic assumption, and further it would be hoped that a sufficient level of accuracy is achieved in far fewer than N iterations. A more realistic assumption arises if optimal polynomials are considered.

Definition 2.2.1 *The polynomial of degree k which solves*

$$\min_{p \in P_k, \ p(0)=1} \left(\max_{t \in \Omega} |p(t)| \right), \tag{2.11}$$

where Ω is a compact subset of $\mathbb{R}$, is called the optimal polynomial of degree k on Ω.

The norm of the residual at the kth step in (2.9) is bounded by

$$||r_k|| \leq ||p_k(M)|| \, ||r_0||,$$

(c.f. (2.7)) and so by the same reasoning as in §2.1, if Ω is an inclusion set for the eigenvalues of M it would be wise to set p_k to be the optimal polynomial of degree k on Ω. If M were symmetric positive-definite, Ω can be taken to be the interval $[\lambda_1, \lambda_N]$ where $0 < \lambda_1 \leq \ldots \leq \lambda_N$ are the eigenvalues of M. Notice that the constraint point (0) lies outside of this interval. The solution of (2.11) in this case can again be written in terms of the scaled Chebyshev polynomials on Ω,

$$p_k(t) = \frac{T_k(t, \Omega)}{T_k(0, \Omega)},$$

and a simple iterative process can be derived, using the three-term recurrence of the Chebyshev polynomials, which updates solution approximations without explicitly forming the action of the residual polynomials on M.

In the case that M is a symmetric indefinite matrix, the inclusion set Ω for the eigenvalues of M must be taken to be the union of two intervals on the real line, one entirely negative and the other entirely positive, so as to exclude the possibility that $0 \in \Omega$. Denoting the negative and positive intervals by $\mathcal{I}^-$ and $\mathcal{I}^+$ respectively, $\Omega = \mathcal{I}^- \cup \mathcal{I}^+$ and the minimax problem (2.11) becomes

$$\min_{p \in P_k,\ p(0)=1} \left(\max_{t \in \mathcal{I}^- \cup \mathcal{I}^+} |p(t)| \right). \tag{2.12}$$

It is not immediately possible to characterise the solution of (2.12) in terms of scaled Chebyshev polynomials, unless the domain Ω is of a specific form, see [20]. Such a domain Ω will be referred to as a *conforming domain*, and not all choices of $\mathcal{I}^-$ and $\mathcal{I}^+$ will give rise to conforming domains Ω. Lebedev [47] treated the case when $\mathcal{I}^-$ and $\mathcal{I}^+$ are located symmetrically about the origin, in this case Ω is a conforming domain, and showed that the optimal polynomials of even degree on Ω are of the form

$$p_{2k}(t) = T_k(q_2(t), \Omega')/T_k(q_2(0), \Omega'),$$

with $q_2 \in P_2$, and where $q_2 : \mathcal{I}^- \longrightarrow \Omega'$ and $q_2 : \mathcal{I}^+ \longrightarrow \Omega'$ are both bijections onto the interval Ω'. This result is described in §2.2.1 since it is useful for proving convergence results for the generalised Legendre method presented in §2.6, and the basic ideas presented generalise to provide convergence bounds on standard iterative methods for symmetric definite and indefinite systems like CG, CGNR, and MINRES. The general two interval case is treated in [20], and conditions on the upper bound on the interval $\mathcal{I}^-$ and lower bound on the interval $\mathcal{I}^+$ are found which characterise conforming domains Ω. Then the residual polynomial is again a scaled Chebyshev polynomial on Ω. In the case that Ω does not satisfy this property it can be embedded in a larger set $\overline{\Omega} = \overline{\mathcal{I}}^- \cup \overline{\mathcal{I}}^+$ where the lower bound on $\overline{\mathcal{I}}^-$ is the same as that for $\mathcal{I}^-$ and the upper bound on $\overline{\mathcal{I}}^+$ is the same as that for $\mathcal{I}^+$. A scaled Chebyshev polynomial on $\overline{\Omega}$ can then be used to define the residual polynomial on Ω. These considerations will not be taken further, since the results given in theorems 2.3.1 and 2.3.2 of §2.3 will motivate the search for an iterative method which minimises the residual polynomial over three eigenvalue intervals. Finding the optimal polynomial in this case is more difficult, this problem necessitates an alternative characterisation of polynomials which are small in some sense and which still provides a residual reduction property.

2.2.1 Optimal polynomials over two intervals which are symmetric about the origin

Lebedev [47] considered the case when $\Omega = \mathcal{I}^- \cup \mathcal{I}^+ = [-b,-a] \cup [a,b]$ where $b > a > 0$. Suppose there exists a degree 2 monic polynomial $q_2 \in P_2$ with the property that

$$\begin{aligned} q_2(-b) &= q_2(b) = & M, \\ q_2(-a) &= q_2(a) = & m, \end{aligned}$$

where $m, M \in \mathbb{R}$, $m < M$, and q_2 is monotonic on each of $\mathcal{I}^-$ and $\mathcal{I}^+$. i.e. q_2 maps each of the intervals which comprise Ω monotonically onto the interval $[m, M]$. Obviously the choice $q_2(t) = t^2$ will suffice here, the technical approach given here is taken since it is easy to extend to the case when Ω is comprised of more than two intervals (not necessarily of the same diameter and not necessarily symmetric about the origin). If such a q_2 exists,

$$q_2(t) - \frac{1}{2}(m + M)$$

is a monic polynomial on Ω and deviates least from zero among monic polynomials of degree 2 since it attains its maximum absolute value on Ω ($\frac{1}{2}(M-m)$) on the alternating set of (three) points $-b, -a$, and b (see for example [63]). Now consider the degree $2j$ polynomial

$$p_{2j}(t) = s_j(q_2(t)),$$

where s_j is the optimal polynomial of degree j on $[m, M]$ and $T_j(t) = \cos(j \arccos(t))$. Obviously

$$s_j(z) = T_j\left(\frac{2z - (m+M)}{m - M}\right) / T_j(z_0),$$

where $z_0 = -(m+M)/(m-M)$. With this choice of q_2 and s_j the following lemma holds.

Lemma 2.2.2 (Lebedev [47]) *The optimal polynomial of degree* $2j$ *on* Ω *is* p_{2j}, *and*

$$\|p_{2j}\|_\infty = |T_j(z_0)|^{-1}. \tag{2.13}$$

Proof See [47] □

It will be useful to uniformly bound p_{2j} on Ω with respect to j, which by (2.13) is equivalent to uniformly bounding $|T_j(z_0)|^{-1}$. Recalling that $q_2(t) = t^2$ for the symmetric interval case, it can be seen that

$$z_0 = 1 + 2\frac{1}{\kappa^2 - 1},$$

where $\kappa = b/a > 1$, so that $z_0 > 1$. Now suppose that

$$T_2(t_0) = z_0.$$

Then, using the Chebyshev identity $T_{2k}(t) = 2T_k(t)^2 - 1$, and the fact that $T_1(t) = t$,

$$t_0^2 = \frac{1}{2}(1 + z_0) = \left(1 - \frac{1}{\kappa^2}\right)^{-1},$$

so that $t_0^2 > 1$. Now using the semigroup property of the Chebyshev polynomials [63, p. 45],

$$T_j(z_0) = T_j(T_2(t_0)) = T_{2j}(t_0).$$

Using induction and the fact that $T_0(t) = 1$, $T_1(t) = t$ it can be shown that

$$T_{2j}(t) = \frac{1}{2}\left\{\left[t + (t^2 - 1)^{\frac{1}{2}}\right]^{2j} + \left[t - (t^2 - 1)^{\frac{1}{2}}\right]^{2j}\right\},$$

and as $(t_0 \pm (t_0^2 - 1)^{\frac{1}{2}})^2 = (\kappa \pm 1)/(\kappa \mp 1)$,

$$\begin{aligned} |T_{2j}(t_0)|^{-1} &= 2\left\{\left(\tfrac{\kappa+1}{\kappa-1}\right)^j + \left(\tfrac{\kappa-1}{\kappa+1}\right)^j\right\}^{-1}, \\ &< 2\left(\tfrac{\kappa-1}{\kappa+1}\right)^j, \end{aligned}$$

for $a \neq b$, since the first term is the dominant one. (In the case that $a = b$ the problem reduces to finding the optimal polynomial over a set of two points, which is trivial, even in the unsymmetric case!). Hence the following lemma has been proved.

Lemma 2.2.3 *The optimal polynomial* p_{2j} *on* Ω *satisfies*

$$\max_{t\in\Omega} |p_{2j}(t)| \leq 2\left(\frac{\kappa-1}{\kappa+1}\right)^j,$$

where $\kappa = b/a$.

The derivation of this result is similar to that for convergence of the conjugate gradient algorithm, see [29, Theorem 10.2.5]. Comparing Lemma 2.2.3 with this bound on convergence of the conjugate gradient algorithm it can be seen that $2j$ steps of an algorithm based upon the optimal polynomial on two intervals above will yield a similar reduction to that given by j steps of the conjugate gradient algorithm on a system with spectrum contained in $[a^2, b^2]$, that is the conjugate gradient method applied to the normal equations (CGNR) for a system with a coefficient matrix whose spectrum is contained on Ω. This sounds unfavourable but since CGNR requires two matrix-vector operations per iteration, Lemma 2.2.3 indicates that the number of matrix-vector operations required by each method to achieve a specified error / residual reduction tolerance can be predicted to be similar. It should also be noted that by embedding any two intervals about the origin inside two symmetrically placed intervals about the origin, bounds on the convergence of the MINRES and SYMMLQ algorithms applied to symmetric indefinite systems can be obtained. As is usually the case with results of this type, the bounds presented tend to be rather pessimistic and provide only a rough guide to the convergence of the underlying iterative method.

The above approach to constructing the optimal polynomial over two intervals will generalise to the case when the two intervals are not of the same size or symmetrically placed about the origin. The difficulty in this case however is that the monic polynomial q_2 is not guaranteed to exist, it will only exist for conforming domains Ω. Conditions which specify domains which satisfy this property are given in [20]. The case of k intervals spaced about the origin is also covered by the above ideas. In this case a degree k polynomial q_k must be found which is a bijection from each of the k intervals to an interval $[m, M]$. The optimal polynomials are then of the form $p_{jk}(t) = s_j(q_k(t))$.

Existence of these q_k polynomials is a more difficult question.

2.3 Distribution of eigenvalues of augmented systems

In this section the eigenvalues of matrices of the form

$$\mathcal{A} = \begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix},$$

are investigated. Here $A \in \mathbb{R}^{n \times n}$ is symmetric and positive-definite and $B \in \mathbb{R}^{n \times m}$ $(n \geq m)$ is of full column rank so as to ensure that $\mathcal{A}$ is nonsingular. $\mathcal{A}$ is clearly symmetric, but is not positive definite as can be seen by taking $x = \alpha By$, where $\alpha < 0$ and $|\alpha|$ is small, in the identity

$$\begin{bmatrix} x^T & y^T \end{bmatrix}^T \begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = x^T Ax + 2x^T By.$$

Hence $\mathcal{A}$ is indefinite and so its spectrum is contained in two intervals in the real line, one interval being entirely negative, the other entirely positive. It is well known [64] that if the eigenvalues of A are denoted $0 < \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$ and the singular values of B are denoted σ_i, $i = 1, \ldots, m$ where $0 < {\sigma_1}^2 \leq \sigma_2^2 \leq \ldots \leq {\sigma_m}^2$ then $\lambda(\mathcal{A}) \subset \mathcal{I}^- \cup \mathcal{I}^+$ where the intervals $\mathcal{I}^-$ and $\mathcal{I}^+$ are defined by

$$\begin{aligned} \mathcal{I}^- &= \left[\frac{\lambda_1}{2} - \sqrt{\frac{\lambda_1^2}{4} + {\sigma_m}^2}, \frac{\lambda_n}{2} - \sqrt{\frac{\lambda_n^2}{4} + {\sigma_1}^2}\right], \\ \mathcal{I}^+ &= \left[\lambda_1, \frac{\lambda_n}{2} + \sqrt{\frac{\lambda_n^2}{4} + {\sigma_m}^2}\right]. \end{aligned}$$

Notice that $\mathcal{I}^-$ is entirely negative and $\mathcal{I}^+$ is entirely positive.

In the special case that $A = I$ even more can be said about the eigenvalues of $\mathcal{A}$, indeed it is easy to show that

$$\lambda\left(\mathcal{A}_{A=I}\right) = \begin{cases} \frac{1}{2} \pm \sqrt{\frac{1}{4} + \sigma_i^2} & i = 1, \ldots, m \\ 1 & n - m \text{ times.} \end{cases}$$

The interval corresponding to $\mathcal{I}^+$ in the case that $A = I$ can be seen to contain the subinterval $(1, \frac{1}{2} + \sqrt{\frac{1}{4} + \sigma_1{}^2})$ which, by the explicit representation of $\lambda\left(\mathcal{A}_{A=I}\right)$ above, is known to contain no eigenvalues. Hence the representation $\mathcal{I}^- \cup \mathcal{I}^+_{A=I}$ where

$$\mathcal{I}^+_{A=I} = \{1\} \cup \left[\frac{1}{2} + \sqrt{\frac{1}{4} + \sigma_1{}^2} \;,\; \frac{1}{2} + \sqrt{\frac{1}{4} + \sigma_m{}^2}\right],$$

gives a better description of the eigenvalue distribution of $\mathcal{A}_{A=I}$ than $\mathcal{I}^- \cup \mathcal{I}^+$.

The following theorem shows that a more descriptive representation of the eigenvalue distribution of $\mathcal{A}$ is possible in the general case $A \neq I$. The positive eigenvalues of A are shown to be contained in the union of two intervals.

Theorem 2.3.1 *If $A \in \mathbb{R}^{n \times n}$ is symmetric, positive-definite and $B \in \mathbb{R}^{n \times m}$ is of full column rank then $\lambda(\mathcal{A}) \subset \mathcal{I}^- \cup \mathcal{I}^+_1 \cup \mathcal{I}^+_2$ where*

$$\begin{aligned}
\mathcal{I}^- &= \left[\frac{\lambda_1}{2} - \sqrt{\frac{\lambda_1{}^2}{4} + \sigma_m{}^2} \;,\; \frac{\lambda_n}{2} - \sqrt{\frac{\lambda_n{}^2}{4} + \sigma_1{}^2}\right], \\
\mathcal{I}^+_1 &= [\lambda_1 \;,\; \lambda_n], \\
\mathcal{I}^+_2 &= \left[\frac{\lambda_1}{2} + \sqrt{\frac{\lambda_1{}^2}{4} + \sigma_1{}^2} \;,\; \frac{\lambda_n}{2} + \sqrt{\frac{\lambda_n{}^2}{4} + \sigma_m{}^2}\right].
\end{aligned}$$

Here $0 < \lambda_1 \leq \ldots \leq \lambda_n$ are the eigenvalues of A and σ_i, $i = 1, \ldots, m$ are the singular values of B with $0 < \sigma_1{}^2 \leq \ldots \leq \sigma_m{}^2$.

Proof If λ is an eigenvalue of $\mathcal{A}$ with associated eigenvector $\begin{bmatrix} x^T & y^T \end{bmatrix}^T$ then

$$\begin{aligned}
Ax + By &= \lambda x && (2.14) \\
B^T x &= \lambda y && (2.15)
\end{aligned}$$

Since A is positive-definite and B is of full column rank, $\mathcal{A}$ is nonsingular and so $\lambda \neq 0$. Clearly if $y = 0$, $\lambda \in \lambda(A)$ and hence $\lambda \in \mathcal{I}^+_1$. Now assume $y \neq 0$. First the bound on negative eigenvalues of $\mathcal{A}$ will be found.

Suppose $\lambda < 0$. Defining $\varphi = -\lambda$, $A + \varphi I$ is positive definite and invertible, hence (2.15) can be used to eliminate x from (2.14) to form

$$B^T \left(A + \varphi I\right)^{-1} By = \varphi y. \qquad (2.16)$$

Defining $z = By$, $z^T(A+\varphi I)^{-1}z = \varphi\,||y||^2$ by (2.16). Since the spectrum of $A+\varphi I$ is given by $\lambda(A+\varphi I) = \{\lambda_i + \varphi \mid \lambda_i \in \lambda(\mathcal{A})\}$,

$$\frac{1}{\lambda_n+\varphi}\,||z||^2 \le z^T(A+\varphi I)^{-1}z \le \frac{1}{\lambda_1+\varphi}\,||z||^2.$$

Hence, using the fact that ${\sigma_1}^2 \le \frac{||z||^2}{||y||^2} \le {\sigma_m}^2$, the inequality

$$\frac{1}{\lambda_n+\varphi}{\sigma_1}^2 \le \varphi \le \frac{1}{\lambda_1+\varphi}{\sigma_m}^2 \tag{2.17}$$

is obtained. Solving (2.17) for $\varphi > 0$,

$$-\frac{\lambda_n}{2}+\sqrt{\frac{{\lambda_n}^2}{4}+{\sigma_1}^2} \le \varphi \le -\frac{\lambda_1}{2}+\sqrt{\frac{{\lambda_1}^2}{4}+{\sigma_m}^2},$$

and hence $\lambda \in \mathcal{I}^- \ \forall\lambda \in \lambda(\mathcal{A})$ with $\lambda < 0$.

The result that the positive eigenvalues of $\mathcal{A}$ are contained in $\mathcal{I}_1^+ \cup \mathcal{I}_2^+$ is now shown, first by demonstrating that all the positive eigenvalues of $\mathcal{A}$ lie in the interval $\mathcal{I}^+ = \left[\lambda_1\ ,\ \frac{\lambda_n}{2}+\sqrt{\frac{{\lambda_n}^2}{4}+{\sigma_m}^2}\right]$, and then that if the interval $\mathcal{I}_v = \left(\lambda_n\ ,\ \frac{\lambda_1}{2}+\sqrt{\frac{{\lambda_1}^2}{4}+{\sigma_1}^2}\right)$ is well-defined, $\mathcal{I}_v \cap \lambda(\mathcal{A}) = \emptyset$.

Suppose $0 < \lambda < \lambda_1$. Then $(A-\lambda I)^{-1}$ exists and with z defined above,

$$z^T(A-\lambda I)^{-1}z = -\lambda\,||y||^2. \tag{2.18}$$

A contradiction to (2.18) is obtained upon observing that $(A-\lambda I)^{-1}$ is positive-definite for $0 < \lambda < \lambda_1$. Hence all positive eigenvalues of $\mathcal{A}$ satisfy $\lambda \ge \lambda_1$.

Now suppose that $\lambda > \lambda_n$. Then (2.18) holds without contradiction since $(A-\lambda I)^{-1}$ is negative-definite and the inequality

$$\frac{1}{\lambda-\lambda_1}{\sigma_1}^2 \le \lambda \le \frac{1}{\lambda-\lambda_n}{\sigma_m}^2 \tag{2.19}$$

in λ can be obtained in the same way that (2.17) was obtained for φ. The rightmost inequality in (2.19) together with the fact that $\lambda \ge \lambda_1$ implies that $\lambda \in \mathcal{I}^+ \ \forall\lambda \in \lambda(A)$ with $\lambda > 0$, whilst the leftmost inequality yields the result that

$$\text{if } \lambda > \lambda_n \text{ then } \lambda \ge \frac{\lambda_1}{2}+\sqrt{\frac{{\lambda_1}^2}{4}+{\sigma_1}^2},$$

i.e. that the interval $\mathcal{I}_v$ (if well-defined) can contain no eigenvalues of $\mathcal{A}$. Noticing that $\mathcal{I}^+ \setminus \mathcal{I}_v = \mathcal{I}_1^+ \cup \mathcal{I}_2^+$ completes the proof. □

The next result gives an indication of how many eigenvalues of $\mathcal{A}$ can be expected to lie in each of the intervals found in Theorem 2.3.1.

Theorem 2.3.2 *With the same notation as Theorem 2.3.1, if $\mathcal{I}_1^+ \cap \mathcal{I}_2^+ = \emptyset$ then*

$$\begin{aligned} &\exists\ m \text{ eigenvalues of } \mathcal{A} \text{ in } \mathcal{I}^-, \\ &\exists\ n-m \text{ eigenvalues of } \mathcal{A} \text{ in } \mathcal{I}_1^+, \\ &\exists\ m \text{ eigenvalues of } \mathcal{A} \text{ in } \mathcal{I}_2^+. \end{aligned}$$

Further if $\mathcal{I}_1^+ \cap \mathcal{I}_2^+ \neq \emptyset$ then $\exists m$ eigenvalues of $\mathcal{A}$ in $\mathcal{I}^-$ and $\exists$ at least $n-m$ eigenvalues of $\mathcal{A}$ in $\mathcal{I}_1^+$, with the remaining eigenvalues being contained in $\mathcal{I}_2^+ \setminus \mathcal{I}_1^+$.

Proof Notice that

$$\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} = \begin{bmatrix} A & 0 \\ B^T & I \end{bmatrix} \begin{bmatrix} A^{-1} & 0 \\ 0 & -B^TA^{-1}B \end{bmatrix} \begin{bmatrix} A & B \\ 0 & I \end{bmatrix}.$$

Since A^{-1} is positive definite and $-B^TA^{-1}B$ is negative definite, Sylvester's law of inertia (see for example [29]) states that $\mathcal{A}$ has exactly m negative and n positive eigenvalues. Since $\mathcal{I}^-$ contains all of the negative eigenvalues (and no others) $\exists m$ eigenvalues of $\mathcal{A}$ in $\mathcal{I}^-$. A similar approach will yield the rest of the eigenvalue information.

Consider $\mathcal{S}(\mu) = \mathcal{A} - \mu\mathcal{I}$ where $\mu > 0$. The eigenvalues of $\mathcal{S}(\mu)$ are simply those of $\mathcal{A}$ shifted negatively by μ. A similar decomposition of $\mathcal{S}(\mu)$ to that of $\mathcal{A}$ can be found if $A - \mu I$ is nonsingular, namely

$$\mathcal{S}(\mu) = \begin{bmatrix} A-\mu I & 0 \\ B^T & I \end{bmatrix} \begin{bmatrix} (A-\mu I)^{-1} & 0 \\ 0 & -B^T(A-\mu I)^{-1}B - \mu I \end{bmatrix} \begin{bmatrix} A-\mu I & B \\ 0 & I \end{bmatrix}.$$

Suppose $\mu > \lambda_n$, then $(A-\mu I)^{-1}$ exists and is negative definite. Again Sylvester's law of inertia states that $\mathcal{S}(\mu)$ will have at least n negative eigenvalues, and will have exactly n negative eigenvalues if the lower-right block of the central matrix in the decomposition is positive definite. Since $\mu > \lambda_n$, $-B^T(A-\mu I)^{-1}B$ is positive definite, the eigenvalues of $-B^T(A-\mu I)^{-1}B$ being bounded below by $-\lambda_{\max}(A-\mu I)^{-1}\lambda_{\min}(B^TB)$. Hence

$$\frac{1}{\lambda_1-\mu}\left(\mu^2-\lambda_1\mu-\sigma_1{}^2\right)$$

is a lower bound on the eigenvalues of the lower-right block of the central matrix in the decomposition. Since $\mu > \lambda_n \geq \lambda_1$, this lower bound is positive whenever $\mu^2-\lambda_1\mu-\sigma_1{}^2<0$, specifically when

$$\mu < \frac{\lambda_1}{2}+\sqrt{\frac{\lambda_1}{4}+\sigma_1{}^2}.$$

Hence provided $\exists\mu$ such that $\lambda_n < \mu < \frac{\lambda_1}{2}+\sqrt{\frac{\lambda_1}{4}+\sigma_1{}^2}$ the lower-right block will be positive-definite and $\mathcal{S}(\mu)$ will possess n negative and m positive eigenvalues. This condition on μ is equivalent to insisting that the intervals $\mathcal{I}_1^+$ and $\mathcal{I}_2^+$ be disjoint. Hence whenever $\mathcal{I}_1^+\cap\mathcal{I}_2^+=\emptyset$, $\mathcal{I}_2^+$ contains m eigenvalues and so $\mathcal{I}_1^+$ must contain the remaining $n-m$ positive eigenvalues of $\mathcal{A}$.

If $\mathcal{I}_1^+\cap\mathcal{I}_2^+\neq\emptyset$ then the lower-right block will be indefinite with, say, m^* positive eigenvalues. Appealing to Sylvester's law of inertia once more, $\mathcal{I}_2^+\setminus\mathcal{I}_1^+$ will contain $m^*<m$ positive eigenvalues of $\mathcal{A}$ with the remaining $n-m^*>n-m$ eigenvalues being contained in $\mathcal{I}_1^+$. □

The results of theorems 2.3.1 and 2.3.2 are demonstrated in Figure 2.3. Here the matrix $\mathcal{A}$ is that in a MAC finite element discretisation (see §5.7.1) of a groundwater flow problem for an incompressible fluid, so that the matrix A is a mass matrix and B is a discretisation of *grad*. Here $n=32$ and $m=16$ and a careful count shows that each of the eigenvalue intervals $\mathcal{I}^-$, $\mathcal{I}_1^+$, $\mathcal{I}_2^+$ contains 16 eigenvalues as predicted by Theorem 2.3.2.

Following this new eigenvalue information, an iterative approach to solving systems with $\mathcal{A}$ as coefficient matrix is explored in the remaining sections of this chapter.

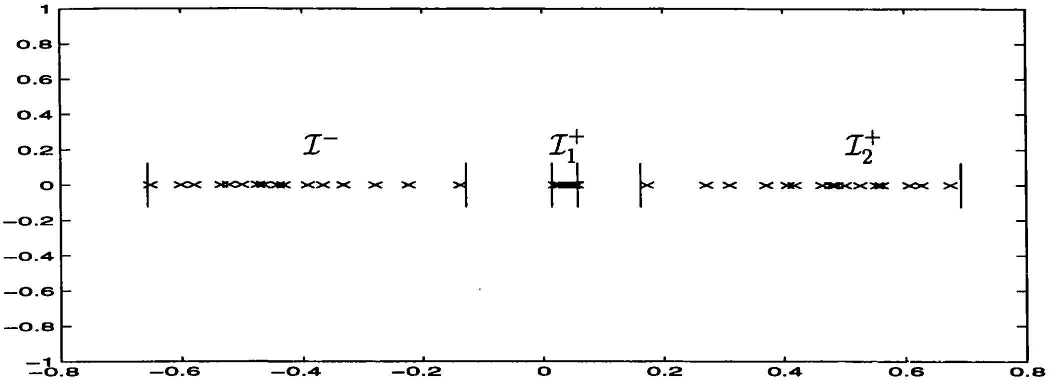


Figure 2-1: Bounding the eigenvalues of a groundwater flow matrix. Vertical lines indicate the bounds obtained from Theorem 2.3.1

2.4 Least-squares characterisation of residual polynomials

Recall from §2.2 that iterative methods of the form

$$z_k = z_{k-1} + d_k, \tag{2.20}$$

where

$$d_k = \sum_{i=0}^{k-1} \psi_i^{(k)} r_i,$$

for the solution of $Mz = f$, can be formed by defining the associated residual polynomials. The residuals r_k satisfy $r_k = p_k(M)r_0$ so that $||r_k|| \leq ||p_k(M)||\,||r_0||$. In order to quickly reduce the norm of the residual, the quantity $||p_k(M)||$ needs to be made small in some sense and an obvious characterisation of this property is to set the residual polynomial to be the optimal polynomial on an inclusion set of the eigenvalues of M. In §2.2 this approach was seen to run into difficulty when the inclusion set Ω for the eigenvalues of M was anything other than a simple connected interval on the real line not containing the origin. In this section an alternative characterisation which follows the treatment in [66] is proposed. The interpolatory constraint (2.10) of the residual polynomial implies that $p_k(t)$ can be expressed as

$$p_k(t) = 1 - ts_{k-1}(t) \tag{2.21}$$

where $s_{k-1}(t) \in P_{k-1}$. The polynomial s_{k-1} will be referred to as the *solution polynomial* since it is easy to show that

$$z_k = z_0 + s_{k-1}(M)r_0. \tag{2.22}$$

The minimax representation of p_k could be expressed by setting s_{k-1} to be the solution of the minimisation

$$\min_{s \in P_{k-1}} \|1 - ts(t)\|_\infty , \tag{2.23}$$

where $\|\cdot\|_\infty$ denotes the uniform norm on Ω. As remarked in §2.2 the uniform norm is difficult to work with in general, and the minimisation (2.23) could be made easier if a different norm were used. Notice that changing the norm results in a loss of the optimal polynomial property of p_k but this will be replaced by a similar condition as is explained below.

Consider the inner product

$$\langle f \, , \, g \rangle_w = \int_\Omega f(t)g(t)w(t) \, dt, \tag{2.24}$$

where $w : \Omega \longrightarrow \mathbb{R}_0^+$ is a non-negative (and non-zero) weight function. Denote the norm associated with the inner product (2.24) by

$$\|f\|_w = \langle f \, , \, f \rangle_w^{\frac{1}{2}} ,$$

and instead of (2.23) consider the minimisation

$$\min_{s \in P_{k-1}} \|1 - ts(t)\|_w . \tag{2.25}$$

This type of minimisation problem is considered in [66] for linear systems whose eigenvalues lie in two intervals. It will be seen that the approach generalises to systems whose eigenvalues are contained in three intervals. The weight function taken in [66] is a 'Chebyshev weight' on each of the two intervals and is zero elsewhere. Here the weight will be taken to be piecewise constant over Ω, the weight will be constant when restricted to one interval and zero outside of the three eigenvalue intervals. This type of weight has been found by the author to provide better numerical results than one of

Chebyshev type.

It is clear from (2.25) that $ts_{k-1}(t)$ is the least-squares approximation to 1 from the space $P'_k = \{ts(t) \mid s \in P_{k-1}\}$ with respect to the w-norm, and hence (see [63, Theorem 2.1]) the residual $1 - ts_{k-1}(t)$ is orthogonal to the space P'_k with respect to the w-inner product. i.e.

$$\langle 1 - ts_{k-1}(t) \, , \, tq(t) \rangle_w = 0, \ \ \forall tq(t) \in P'_k,$$

so that

$$\langle ts_{k-1}(t) \, , \, tq(t) \rangle_w = \langle 1 \, , \, tq(t) \rangle_w \, , \ \ \forall tq(t) \in P'_k. \tag{2.26}$$

Equation (2.26) provides a mechanism for calculating the w-inner product of any member of P'_k with $ts_{k-1}(t)$ without explicitly requiring $ts_{k-1}(t)$.

Suppose that $\{tq_i(t)\}_{i=0}^{k-1}$ is a basis for P'_k. Here $q_i \in P_i$. Then the solution to (2.25) can be written as

$$ts_{k-1}(t) = \sum_{i=0}^{k-1} \varphi_i t q_i(t),$$

for some scalar coefficients φ_i, $i = 0, \ldots, k-1$. Using this expression and putting $q(t) = q_i(t)$, $i = 0, \ldots, k-1$ in (2.26) it can be seen that

$$\sum_{j=0}^{k-1} g_{ij} \varphi_j = c_i, \ \ i = 0, \ldots, k-1, \tag{2.27}$$

where $g_{ij} = \langle tq_i(t) \, , \, tq_j(t) \rangle_w$ and $c_i = \langle 1 \, , \, tq_i(t) \rangle_w$. The matrix $G = (g_{ij})$ is called the *moment matrix of order* k (see [39]) and (2.27) are the normal equations for $\varphi_0, \ldots, \varphi_{k-1}$. Solving (2.27) (and hence (2.25)) is easy for small values of k but the cost of solution increases rapidly with k since the matrix G is, in general, full. The goal is to make the normal equations (2.27) easy to solve. This can be achieved if the set $\{tq_i(t)\}_{i=0}^{k-1}$ is taken to be orthonormal, that is

$$\langle tq_i(t) \, , \, tq_j(t) \rangle_w = \delta_{ij}, \tag{2.28}$$

i.e. $\int_\Omega tq_i(t) tq_j(t) w(t) \, dt = \delta_{ij}$. This choice of $\{tq_i(t)\}_{i=0}^{k-1}$ uncouples the normal equations (2.27) (the moment matrix becomes the identity) to give

$$\varphi_i = \langle 1 \, , \, tq_i(t) \rangle_w \, . \tag{2.29}$$

The φ_i are called *modified moments.* Therefore, provided that an orthonormal basis, $\{tq_i(t)\}_{i=0}^{k-1}$, can be found for P_k' and provided (2.29) is easily calculated, the expansion of the solution polynomial s_{k-1} in terms of $\{q_i\}_{i=0}^{k-1}$ can be formed. In the following sections it will be shown that such a set of orthonormal polynomials can be generated using three-term recurrences and an algorithm for calculating the recurrence coefficients will be presented. The key to being able to perform the calculation (2.29) is that the polynomials $\{tq_i(t)\}_{i=0}^{k-1}$ will themselves be expressed in terms of orthonormal polynomials over each of the eigenvalue intervals so that only the 0th degree term in the expansion of each $tq_i(t)$ in terms of these orthonormal polynomials will contribute to the inner product (2.29).

2.4.1 Orthogonal polynomials on one interval

Here a brief discussion of the properties of orthogonal polynomials on one interval is presented. Suppose that $I = [-1, 1]$ and that $w : I \longrightarrow \mathbb{R}_0^+$, and consider the inner product

$$\langle f \, , \, g \rangle_w = \int_{-1}^{1} f(t)g(t)w(t) \, dt.$$

Then there is a set of polynomials $\{p_k\}_{k=0}^{\infty}$, where $p_k \in P_k$, which are orthogonal with respect to this inner product and further, the set of polynomials satisfies a three term recurrence. i.e. $\exists \, \{\alpha_k\}_{k=0}^{\infty}$ and $\{\beta_k\}_{k=1}^{\infty}, \{\gamma_k\}_{k=0}^{\infty}$ such that

$$\gamma_{k+1} p_{k+1}(t) = (t - \alpha_k) p_k(t) - \beta_k p_{k-1}(t).$$

The proof of this result is constructive, first set $\gamma_0 p_0(t) = 1$ and $\gamma_1 p_1(t) = (t - \alpha_0) p_0$ where $\alpha_0 = \langle tp_0 \, , \, p_0 \rangle_w / \langle p_0 \, , \, p_0 \rangle_w$. Then clearly $\langle p_1(t) \, , \, p_0(t) \rangle_w = 0$. Inductively suppose that $\{p_i\}_{i=0}^{k}$ is orthogonal and set

$$\alpha_k = \frac{\langle tp_k(t) \, , \, p_k(t) \rangle_w}{\langle p_k(t) \, , \, p_k(t) \rangle_w},$$

and

$$\beta_k = \frac{\langle tp_k(t)\,,\, p_{k-1}(t)\rangle_w}{\langle p_{k-1}(t)\,,\, p_{k-1}(t)\rangle_w},$$

and define $\gamma_{k+1}p_{k+1}$ using the three-term recurrence formula. Then clearly p_{k+1} is orthogonal to $\{p_i\}_{i=0}^{k-2}$ using the orthogonality of p_k and p_{k-1}, and $\langle p_{k+1}\,,\, p_k\rangle_w = \langle p_{k+1}\,,\, p_{k-1}\rangle_w = 0$ by construction. As can be seen the choice of γ_{k+1} serves only to normalise p_{k+1} in some way. For example γ_{k+1} could be chosen to scale the coefficient of the leading term in p_{k+1} to unity (to make p_{k+1} monic), in which case $\gamma_{k+1} = 1\ \forall k \geq 0$. Choosing γ to make the set of polynomials orthonormal implies that $\gamma_{k+1} = \beta_{k+1}$.

An important set of orthogonal polynomials are the *Jacobi polynomials* on I, which have a weight function defined by $w(t) = (1-t)^\theta(1+t)^\phi$, where $\theta, \phi > -1$. The choice $\theta = \phi = 0$ implies that $w(t) \equiv 1$. The set of orthogonal polynomials with respect to the constant weight 1 are the *Legendre polynomials* on I. In this case the subscript w on the inner product will be dropped. Taking $\theta = \phi = \frac{1}{2}$ gives $w(t) = (1-t^2)^{-\frac{1}{2}}$ which produces the familiar set of Chebyshev polynomials.

Since constant weights are to be assumed on each of the eigenvalue intervals of $\mathcal{A}$ found in Theorem 2.3.1 it will be useful to study the Legendre polynomials a little further. The set of orthonormal Legendre polynomials on I will be denoted by $\{\widetilde{l}_k\}_{k=0}^{\infty}$. It is easy to prove by induction that $\widetilde{l}_k$ is an odd function for odd k and is an even function for even k. (This result is actually true for any set of Jacobi polynomials with $\theta = \phi$, and is due to the fact that in this case the weight $w(t)$ is an even function - see [63]). Defining the recurrence coefficients by $\widetilde{\alpha}_k$ and $\widetilde{\beta}_k$ it is then clear that $\widetilde{\alpha}_k = \left\langle t\widetilde{l}_k(t)\,,\, \widetilde{l}_k(t)\right\rangle = 0\ \forall k$ since the inner product is the usual $L_2(-1,1)$ inner product of an odd function and an even function. Hence the recurrence formula for these polynomials is of the form

$$\widetilde{\beta}_{k+1}\widetilde{l}_{k+1}(t) = t\widetilde{l}_k(t) - \widetilde{\beta}_k\widetilde{l}_{k-1}(t),$$

where $l_0(t) = (1/2)^{\frac{1}{2}}$. It can also be shown that $\widetilde{\beta}_k = k/(4k^2-1)^{\frac{1}{2}}$, and hence no integrations are required to form this set of polynomials. The first few Legendre polynomials are depicted in Figure 2-2.

It will later be necessary to express the polynomials $ts_{k-1}(t)$ in terms of constant-

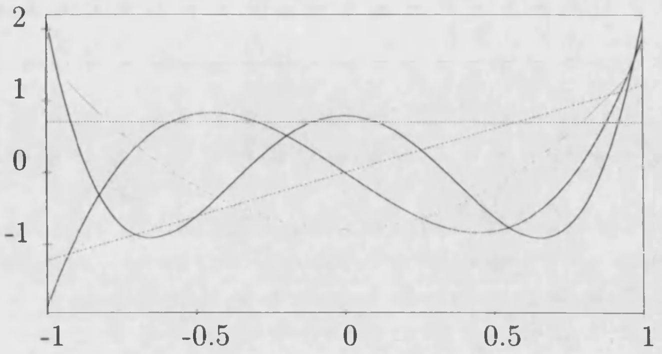


Figure 2-2: The first few orthonormal Legendre polynomials

weighted Legendre polynomials on arbitrary intervals $[a,b]$. Let the constant weight on $[a,b]$ be w, and denote these polynomials by $\{l_k\}_{k=0}^{\infty}$. To determine a recurrence for these polynomials notice that

$$\xi = \xi(t) = \frac{2t-(b+a)}{b-a},$$

is a bijection from $[a,b]$ to $[-1,1]$ (it is the same bijection used in §2.2.1) and $t = c + d\xi$ where $c = (b+a)/2$ is the centre of the interval and $d = (b-a)/2$ is its radius. Then

$$\langle l_i\,,\, l_j\rangle_w = \int_{-1}^{1} (wd)^{\frac{1}{2}} l_i(c+d\xi)(wd)^{\frac{1}{2}} l_j(c+d\xi)\, d\xi,$$

and so the required orthonormality condition can be achieved by setting $(wd)^{\frac{1}{2}} l_k(c + d\xi) = \widetilde{l}_k(\xi)$, i.e.

$$l_k(t) = \left(\frac{1}{wd}\right)^{\frac{1}{2}} \widetilde{l}_k\left(\frac{t-c}{d}\right).$$

Using the recurrence relation for $\{\widetilde{l}_k\}_{k=0}^{\infty}$ it can then be seen that

$$\beta_{k+1} l_{k+1}(t) = (t-c) l_k(t) - \beta_k l_{k-1}(t), \tag{2.30}$$

where $\beta_k = d\widetilde{\beta}_k = dk/(4k^2-1)^{\frac{1}{2}}$ and $l_0(t) = \widetilde{l}_0(t)/(wd)^{\frac{1}{2}}$. Again no actual numerical integrations are required.

The next section discusses using these shifted Legendre polynomials to devise an algorithm for computing orthonormal polynomials over three intervals.

2.5 Generating orthonormal polynomials over three intervals

Now the problem of generating orthonormal polynomials with respect to an inner product defined by (2.24) is treated where the weight function takes the form

$$w(t) = \begin{cases} w_1 & t \in \mathcal{I}^-, \\ w_2 & t \in \mathcal{I}_1^+, \\ w_3 & t \in \mathcal{I}_2^+, \\ 0 & t \notin \Omega, \end{cases} \tag{2.31}$$

where w_1, w_2, w_3 are positive constants and $\Omega = \mathcal{I}^- \cup \mathcal{I}_1^+ \cup \mathcal{I}_2^+$ is the inclusion set for the eigenvalues of $\mathcal{A}$ given by Theorem 2.3.1. This weight differs to that taken in [66], there

$$w(t) = \begin{cases} w_1(t) & t \in \mathcal{I}^-, \\ w_2(t) & t \in \mathcal{I}^+, \\ 0 & t \notin \mathcal{I}^- \cup \mathcal{I}^+, \end{cases}$$

where $\mathcal{I}^- = [c_1 - d_1, c_1 + d_1]$, $\mathcal{I}^+ = [c_2 - d_2, c_2 + d_2]$ and $w_i(t) = (2/\pi)[d_i^2 - (t - c_i)^2]^{-\frac{1}{2}}$ are shifted Chebyshev weights. The reason for taking Legendre type weights of form (2.31) here is that a simple constant weighting on each interval can be used, and it has been observed by the author that such a weight gives better numerical performance on the problems attempted in this section than a weight of Chebyshev type. Varying constant weights on each interval have been allowed so that varying significance can be attached to specific intervals.

Several authors have treated the problem of generating orthogonal polynomials over disjoint intervals (see for example [21, 27, 87]) although the motivation in these cases is usually that of determination of Gaussian quadrature formulae rather than solution of linear systems. Few authors other than Saad [66] appear to have used orthogonal polynomial methods in connection with solution methods other than when forming preconditioning strategies (which is discussed in §2.9).

The standard approaches to generating orthonormal polynomials (on more than one interval) are those of *Stieltjes procedure* type or methods of *modified moments*. In a

Stieltjes procedure, the recurrence coefficients $\{a_k\}_{k=0}^{\infty}$ and $\{b_k\}_{k=1}^{\infty}$ for the orthonormal polynomials $\{p_k\}_{k=0}^{\infty}$ with respect to the weight $w(t)$, where

$$b_{k+1}p_{k+1}(t) = (t - a_k)p_k(t) - b_k p_{k-1}(t),$$

are generated by determining the recurrence coefficients a_k and b_k with the inner products $a_k = \langle tp_k(t) \ , \ p_k(t)\rangle_w$ and $b_k = \langle tp_k(t) \ , \ p_{k-1}(t)\rangle_w$. See [25, 21] for more details. It is usually the case that the inner products can be performed at little cost without resorting to numerical integration.

Methods of modified moments are discussed extensively in [21]. The modified moments approach has the advantage that, provided the modified moments,

$$\nu_k = \langle 1 \ , \ q_k\rangle_w = \int_{\Omega} q_k(t)w(t)\, dt,$$

are known explicitly for some set of polynomials $\{q_k\}_{k=0}^{L}$, where L is the number of iteration steps, the complexity of the procedure does not grow with the number of intervals considered. The method dates back to one first devised by Chebyshev who took $q_k(t) = t^k$. The resulting algorithm requires the computation of the Cholesky factorisation of the associated moment matrix $G = (g_{ij}) = (\langle q_i \ , \ q_j\rangle)$ of order k at the kth step, and competitive algorithms use fast methods to compute this factorisation. The results given in [21] suggest that the modified moment approaches are less stable than the standard Stieltjes approaches and since stability is paramount in iterative solvers the Stieltjes procedure is favoured here. Since only three intervals are being considered, the complexity saving of the modified moment approach is not of great advantage.

Although an orthonormal basis for P_k' is sought, the problem of generating an orthonormal basis for P_k will first be considered since it is a more natural problem and the extension to the P_k' case is simple.

2.5.1 Generalisation of Saad's Stieltjes procedure to the generation of orthonormal polynomials over three intervals

Let $I_1, I_2, I_3 \subset \mathbb{R}$ be three intervals (specifically, $I_1 = \mathcal{I}^-$, $I_2 = \mathcal{I}_1^+$ and $I_3 = \mathcal{I}_2^+$, the change of notation is to aid simplicity) defined by $I_i = [c_i - d_i \ c_i + d_i]$. Here c_i is the centre of the interval I_i and d_i is its radius. Consider the weight (2.31). On defining

$$\langle f\,,\,g\rangle_{w_i} = \int_{I_i} f(t)g(t)w_i\,dt, \tag{2.32}$$

it is clear that

$$\langle f\,,\,g\rangle_w = \sum_{i=1}^{3} \langle f\,,\,g\rangle_{w_i}. \tag{2.33}$$

Now suppose that $\{p_k\}_{k=0}^{\infty}$ is an orthonormal sequence of polynomials with respect to (2.31). The Stieltjes procedure is required to determine the recurrence coefficients $\{a_k\}_{k=0}^{L}$, $\{b_k\}_{k=1}^{L+1}$ such that

$$b_{k+1}p_{k+1}(t) = (t-a_k)p_k(t) - b_k p_{k-1}(t) =: \widetilde{p}_{k+1}(t), \quad k = 0, \ldots L, \tag{2.34}$$

where L is finite and corresponds to the number of iteration steps.

Since any linearly independent set of polynomials of degree L forms a basis for P_L, and since the shifted Legendre polynomials defined by (2.30) are obviously linearly independent, p_k can be expressed in terms of either of the three bases $\{l_i^{(h)}\}_{i=0}^{k}$, $h = 1, 2, 3$, corresponding to the shifted Legendre polynomials on the intervals I_1, I_2 and I_3. It will actually be convenient to express p_k in terms of *each* of these bases since this will provide a simple method for calculating the Stieltjes inner products without integration. Write

$$p_k(t) = \begin{cases} \sum_{i=0}^{k} \gamma_i^{(k)} l_i^{(1)}(t), \\ \sum_{i=0}^{k} \delta_i^{(k)} l_i^{(2)}(t), \\ \sum_{i=0}^{k} \eta_i^{(k)} l_i^{(3)}(t). \end{cases} \tag{2.35}$$

Notice that as (2.35) provides three alternative expansions of $p_k(t)$, two of the expansions are redundant since they could be obtained from the first. However each expansion will be specifically required to calculate the Stieltjes inner products, and it will be cheaper to update each expansion rather than calculate it through a change of basis from the first.

Using (2.33) and the fact that the p_k's are orthonormal, the recurrence coefficients are given by the inner products

$$\begin{aligned} a_k &= \langle tp_k(t)\,,\, p_k(t)\rangle_w\,, \qquad \left(= \sum_{i=1}^{3} \langle tp_k(t)\,,\, p_k(t)\rangle_{w_i}\right), & (2.36)\\ b_{k+1} &= \langle \widetilde{p}_{k+1}(t)\,,\, \widetilde{p}_{k+1}(t)\rangle_w\,. & (2.37)\end{aligned}$$

The component contributed by I_1 to the sum (2.36) is

$$\langle tp_k(t)\,,\, p_k(t)\rangle_{w_1} = \left\langle \sum_{i=0}^{k} \gamma_i^{(k)} t l_i^{(1)}(t)\,,\, \sum_{i=0}^{k} \gamma_i^{(k)} l_i^{(1)}(t) \right\rangle_{w_1},$$

and using the recurrence formulae (2.30) for $\{l_i^{(1)}\}_{i=0}^{k}$ to remove $tl_i^{(1)}(t)$ from the first entry in the inner product, it can be seen that

$$\langle tp_k(t)\,,\, p_k(t)\rangle_{w_1} = \sigma_k^{(1)} + 2\tau_k^{(1)},$$

where $\sigma_k^{(1)} = c_1 \sum_{i=0}^{k} \gamma_i^{(k)^2}$, $\tau_k^{(1)} = \sum_{i=0}^{k-1} \beta_{i+1}^{(1)} \gamma_i^{(k)} \gamma_{i+1}^{(k)}$ and $\{\beta_i^{(1)}\}$ is the set of recurrence coefficients for the Legendre polynomials on the interval I_1. Similarly defining $\sigma_k^{(2)}, \sigma_k^{(3)}, \tau_k^{(2)}, \tau_k^{(3)}$ in terms of $c_2, c_3, \beta_i^{(2)}, \beta_i^{(3)}, \delta_i^{(k)}, \eta_i^{(k)}$ and using (2.33), gives the formula

$$a_k = \sum_{i=1}^{3} \sigma_k^{(i)} + 2\tau_k^{(i)}. \tag{2.38}$$

Equation (2.38) provides a method for calculating the recurrence coefficients a_k without performing any numerical integrations. Using (2.34) and the recurrence relation (2.30), $\widetilde{p}_{k+1}$ can be expressed in terms of $\{l_i^{(1)}\}_{i=0}^{k+1}$ by

$$\begin{aligned} \widetilde{p}_{k+1}(t) &= \textstyle\sum_{i=0}^{k} \gamma_i^k \left(\beta_{i+1}^{(1)} l_{i+1}^{(1)}(t) + c_1 l_i^{(1)}(t) + \beta_i^{(1)} l_{i-1}^{(1)}(t)\right)\\ &\quad -a_j \textstyle\sum_{i=0}^{k} \gamma_i^{(k)} l_i^{(1)}(t) - b_j \sum_{i=0}^{k-1} \gamma_i^{(k-1)} l_i^{(1)}(t). \end{aligned}$$

Hence if the expansion of $\widetilde{p}_{k+1}$ in terms of $\{l_i^{(1)}\}_{i=0}^{k+1}$ is written as $\widetilde{p}_{k+1}(t) = \sum_{i=0}^{k+1} \widetilde{\gamma}_i^{k+1} l_i^{(1)}(t)$, using the orthogonality of the shifted Legendre polynomials it can be seen that

$$\widetilde{\gamma}_i^{(k+1)} = \beta_i^{(1)} \gamma_{i-1}^{(k)} + \beta_{i+1}^{(1)} \gamma_{i+1}^{(k)} + (c_1 - a_k)\gamma_i^k - b_k \gamma_i^{(k-1)}, \tag{2.39}$$

(where any undefined terms are set to zero). Then using orthonormality of the shifted

Legendre polynomials again,

$$\langle \widetilde{p}_{k+1}\,,\, \widetilde{p}_{k+1} \rangle_{w_1} = \sum_{i=0}^{k+1} \widetilde{\gamma}_i^{(k+1)^2}.$$

Defining $\nu_{k+1}^{(1)} = \sum_{i=0}^{k+1} \widetilde{\gamma}_i^{(k+1)^2}$, $\nu_{k+1}^{(2)} = \sum_{i=0}^{k+1} \widetilde{\delta}_i^{(k+1)^2}$, and $\nu_{k+1}^{(3)} = \sum_{i=0}^{k+1} \widetilde{\eta}_i^{(k+1)^2}$, where $\widetilde{\delta}_i^{(k+1)}$ and $\widetilde{\eta}_i^{(k+1)}$ are the coefficients of $\widetilde{p}_{k+1}$ in terms of the shifted Legendre polynomials on I_2 and I_3 respectively and are defined by an analogous recurrence to (2.39), it can be seen that

$$b_{k+1} = \left(\nu_{k+1}^{(1)} + \nu_{k+1}^{(2)} + \nu_{k+1}^{(3)} \right)^{\frac{1}{2}}, \tag{2.40}$$

and so again this Stieltjes inner product can be performed without any numerical integration. Once b_{k+1} is determined, setting

$$p_{k+1}(t) = \frac{1}{b_{k+1}} \widetilde{p}_{k+1}(t), \tag{2.41}$$

(i.e. setting $\gamma_i^{(k+1)} = \widetilde{\gamma}_i^{(k+1)}/b_{k+1}$, $i = 0, \ldots, k+1$) completes the inductive step.

It is not difficult to show that the initial polynomial p_0 is defined by the coefficients

$$\gamma_0^{(0)} = \frac{h_1}{(h_1+h_2+h_3)^{\frac{1}{2}}}, \quad \delta_0^{(0)} = \frac{h_2}{(h_1+h_2+h_3)^{\frac{1}{2}}}, \quad \eta_0^{(0)} = \frac{h_3}{(h_1+h_2+h_3)^{\frac{1}{2}}},$$

where $h_i = (1/(w_i d_i))^{\frac{1}{2}}$. The steps (2.38), (2.39), (2.40) and (2.41) then comprise the required algorithm for computing the orthonormal polynomials on $I_1 \cup I_2 \cup I_3$ with respect to the weight (2.31). It is stressed that at no point is a polynomial specifically formed, instead only the recurrence coefficients a_k and b_{k+1} and the coefficients of the polynomials in terms of the shifted Legendre bases are known. One simplification can be made since it is not actually necessary to form the sum for σ_{k+1}, instead

$$\sigma_{k+1}^{(i)} = \frac{c_1}{b_{j+1}^2} \nu_{k+1}^{(i)}, \quad i = 1, 2, 3$$

should be used.

The Stieltjes procedure can be seen to proceed at a cost which depends on the iteration number since the elements which make up the new recurrence coefficients a_k and b_{k+1} are sums of k or $k+1$ terms. The cost per interval of the kth iteration is approximately $8k$ multiplications and $5k$ additions, which, for modest k, is small compared to an inner product of the size of the linear systems that are being considered. The whole process will parallelise well, if three processors are available it would be wise to assign the calculations on one interval (i.e. all the γ, δ or η calculations) to each processor, and the combining operations are small. Obviously if more processors were available the work could be shared accordingly. Possibly more importantly, the process is also particularly suited to vector machines since operations (2.39) and (2.41) are easy to implement with vector multiplications and additions, and the sums for the σ, τ and ν terms can be implemented as pointwise vector multiplications.

2.5.2 An orthonormal polynomial basis for P'_k

Recall (equation (2.28)), that a set of polynomials $\{tq_i(t)\}_{i=0}^{k-1}$ is required such that

$$\langle tq_i(t)\ ,\ tq_j(t)\rangle_w = \delta_{ij},$$

i.e. the set is an orthonormal basis for the set $P'_k = \{ts(t) \mid s \in P_{k-1}\}$. Then the solution polynomial $s_{k-1}(t)$ can be expressed as the linear combination

$$s_{k-1}(t) = \sum_{i=0}^{k-1} \varphi_i q_i(t), \tag{2.42}$$

where the coefficients, φ, are calculated using (2.29),

$$\varphi_i = \langle 1\ ,\ tq_i(t)\rangle_w .$$

Again using $\{a_i\}_{i=0}^{k-1}$ and $\{b_i\}_{i=1}^{k-1}$ to denote recurrence coefficients, the three term recurrence of these polynomials is written

$$b_{k+1}tq_{k+1}(t) = (t - a_k)tq_k(t) - b_k tq_{k-1}(t) =: t\widetilde{q}_{k+1}(t). \tag{2.43}$$

Here

$$a_k = \langle t\,(tq_k(t))\ ,\ tq_k(t)\rangle_w , \tag{2.44}$$

$$b_{k+1} = \langle t\widetilde{q}_{k+1}(t)\ ,\ t\widetilde{q}_{k+1}(t)\rangle_w^{\frac{1}{2}} . \tag{2.45}$$

Notice that (2.43) can be divided by t to show that the polynomials $\{q_i\}_{i=0}^{k-1}$ also satisfy a three term recurrence,

$$b_{k+1}q_{k+1}(t) = (t - a_k)q_k(t) - b_k q_{k-1}(t), \tag{2.46}$$

with the same recurrence coefficients, although this set is not necessarily orthogonal. This property will be useful later when forming an iterative solver.

It is clear that on setting

$$tq_k(t) = \begin{cases} \sum_{i=0}^{k+1} \gamma_i^{(k)} l_i^{(1)}(t), \\ \sum_{i=0}^{k+1} \delta_i^{(k)} l_i^{(2)}(t), \\ \sum_{i=0}^{k+1} \eta_i^{(k)} l_i^{(3)}(t), \end{cases} \tag{2.47}$$

(with notation similar to that in §2.5.1) the algorithmic details of computing these orthonormal polynomials will be exactly the same as is that in §2.5.1 except that summations are altered to include $k+1^{\text{th}}$ terms. The detail is omitted here. The interpolation property (that each $tq_k(t)$ must pass through zero at the origin) is inherited from the recurrence (2.43) and so causes no additional problems.

Again it is not difficult to calculate the initial polynomial $tq_0(t)$ to initialise the iteration. If its coefficients are $\gamma_0^{(0)}, \gamma_1^{(0)}, \delta_0^{(0)}, \delta_1^{(0)}, \eta_0^{(0)}$ and $\eta_1^{(0)}$ in terms of the shifted Legendre polynomials on I_1, I_2 and I_3 respectively. It can be shown that

$$\begin{aligned} \gamma_0^{(0)} &= \frac{3^{\frac{1}{2}} c_1 (w_1 d_1)^{\frac{1}{2}}}{K}, & \gamma_1^{(0)} &= \frac{d_1}{3^{\frac{1}{2}} c_1} \gamma_0^{(0)}, \\ \delta_0^{(0)} &= \frac{3^{\frac{1}{2}} c_2 (w_2 d_2)^{\frac{1}{2}}}{K}, & \delta_1^{(0)} &= \frac{d_2}{3^{\frac{1}{2}} c_2} \delta_0^{(0)}, \\ \eta_0^{(0)} &= \frac{3^{\frac{1}{2}} c_3 (w_3 d_3)^{\frac{1}{2}}}{K}, & \eta_1^{(0)} &= \frac{d_3}{3^{\frac{1}{2}} c_3} \eta_0^{(0)}, \end{aligned} \tag{2.48}$$

where $K = \left(\sum_{i=1}^{3} \left(d_i^2 + 3c_i^2\right)(w_i d_i)\right)^{\frac{1}{2}}$.

As an example, this algorithm has been used to generate the orthonormal polynomial basis for P_k' with respect to $\Omega = [-10, -5] \cup [1, 3] \cup [10, 20]$, with unit weights on each interval. The first few polynomials are depicted in Figure 2-3.

Notice that the interpolation property holds true as mentioned above. The w-inner products of the first few generalised Legendre polynomials are represented in the (upper

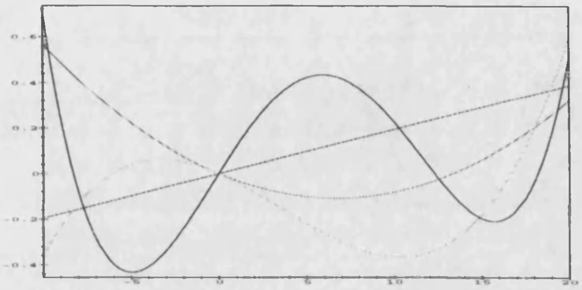


Figure 2-3: The first four orthonormal basis polynomials for P_k' with $\Omega = [-10, -5] \cup [1, 3] \cup [10, 20]$ and $w_i = 1$, $i = 1, 2, 3$

triangular part of the) moment matrix,

$$\begin{bmatrix} 1 & -4 \times 10^{-16} & -9 \times 10^{-16} & 1 \times 10^{-15} & -3 \times 10^{-16} & -1 \times 10^{-15} \\ & 1 & 4 \times 10^{-16} & -3 \times 10^{-15} & 6 \times 10^{-16} & -2 \times 10^{-15} \\ & & 1 & -1 \times 10^{-15} & -3 \times 10^{-15} & 1 \times 10^{-15} \\ & & & 1 & 8 \times 10^{-16} & -4 \times 10^{-15} \\ & & & & 1 & -1 \times 10^{-15} \\ & & & & & 1 \end{bmatrix}.$$

In accordance with the conclusions of [21] the Stieltjes procedure is seen to be very stable. For this choice of Ω and w the maximum value of $\langle tq_i(t)\ ,\ tq_j(t)\rangle_w$ (for $i \neq j$) after 100 iterations is 1.18×10^{-13} and the greatest deviation from 1 of $\langle tq_i(t)\ ,\ tq_i(t)\rangle_w$ is of size 4.44×10^{-16}.

2.5.3 Calculation of the modified moments φ_i

The final hurdle that stands in the way of being able to compute the solution polynomial $s_{k-1}(t) = \sum_{i=0}^{k-1} \varphi_i q_i(t)$ is the calculation of the modified moments φ_i where

$$\varphi_i = \langle 1\ ,\ tq_i(t)\rangle_w\,,$$

and using (2.33),

$$\varphi_i = \sum_{j=0}^{3} \langle 1\ ,\ tq_i(t)\rangle_{w_j}\,.$$

Expanding $tq_i(t)$ in terms of the shifted Legendre polynomials on I_j it is clear that the only contribution the the inner product with 1 (a 0th degree polynomial) will come from the $l_0^{(j)}$ term for $j = 1, 2, 3$. Then using the fact that

$$2^{\frac{1}{2}}\,(w_j d_j)^{\frac{1}{2}}\,l_0^{(j)} = 1,$$

and the orthonormality of the shifted Legendre polynomials it can be seen that

$$\varphi_i = 2^{\frac{1}{2}}\left((w_1 d_1)^{\frac{1}{2}}\gamma_0^{(i)} + (w_2 d_2)^{\frac{1}{2}}\delta_0^{(i)} + (w_3 d_3)^{\frac{1}{2}}\eta_0^{(i)}\right), \tag{2.49}$$

is a formula for the ith modified moment.

2.6 Iterative solution of augmented systems via least-squares residual polynomials

Recall (equation (2.22)) that the approximation z_k to the solution of $Mz = f$ when $r_k = p_k(M)r_0$ is given by

$$z_k = z_0 + s_{k-1}(M)r_0,$$

and that $s_{k-1}(t) = \sum_{i=0}^{k-1}\varphi_i q_i(t)$. It is not desirable to have to form the solution polynomial at each step. Instead, Saad [66] shows that it is possible to update the solution approximations using a three term recurrence similar to that of q_{k-1}. Notice that

$$s_{k-1}(t) = \varphi_{k-1}q_{k-1}(t) + s_{k-2}(t),$$

and hence

$$z_k = z_{k-1} + \varphi_{k-1}u_{k-1},$$

where u_{k-1}, the update direction at the kth step, is given by

$$u_{k-1} = q_{k-1}(M)r_0.$$

Using the three-term recurrence, (2.46), of the (nonorthogonal) polynomials q_{k-1} it is clear that

$$b_{k-1}u_{k-1} = (M - a_{k-2}I)u_{k-2} - b_{k-2}u_{k-3}. \tag{2.50}$$

Here $u_0 = q_0(M)r_0 = (3/2)^{\frac{1}{2}}(1/K)r_0$ (using the expression for $tq_0(t)$ in (2.48)). The residuals r_k can be updated in a similar fashion,

$$p_k(t) = 1 - ts_{k-1}(t) = p_{k-1}(t) - \varphi_{k-1}tq_{k-1}(t),$$

and hence

$$r_k = r_{k-1} - \varphi_{k-1}v_{k-1},$$

where $v_{k-1} = Mu_{k-1}$ $(= Mq_{k-1}(M)r_0)$. With this definition, (2.50) becomes

$$b_{k-1}u_{k-1} = v_{k-2} - a_{k-2}u_{k-2} - b_{k-2}u_{k-3}, \tag{2.51}$$

and

$$\begin{aligned} z_k &= z_{k-1} + \varphi_{k-1}u_{k-1}, & (2.52) \\ r_k &= r_{k-1} - \varphi_{k-1}v_{k-1}, & (2.53) \end{aligned}$$

are the update formulae for the solution approximation and residual. This method based on the least-squares polynomials over three intervals will be referred to as the *generalised Legendre method* and the corresponding algorithm is denoted LS(3). The cost per iteration at the kth step of LS(3) can be seen to be

• Matrix-vector products	$1\ (n+m \times n+m)$,
• Scalar-vector products	$4\ (n+m)$,
• Vector additions	$3\ (n+m)$,
• Additional operations	$3{\times}8k$ mult., $3{\times}5k$ add. ,

where the additional operations are those required to calculate the next basis polynomial (§2.5.1). Here n is the dimension of the (square) matrix A and $B \in \mathbb{R}^{n\times m}$. Only three vector additions are needed since it is not necessary to update the residual as will be explained shortly. If it is decided to update the residual then an additional scalar-vector product and vector addition must be performed. This operation count should be compared with that of CGNR (an implementation of the conjugate gradient algorithm on the normal equations),

• Matrix-vector products	$2\ (n+m \times n+m)$,
• Scalar-vector products	$3\ (n+m)$,
• Vector additions	$3\ (n+m)$,
• Vector inner products	$2\ (n+m)$,

and SYMMLQ,

• Matrix-vector products	$1\ (n+m \times n+m)$,
• Scalar-vector products	$7\ (n+m)$,
• Vector additions	$5\ (n+m)$,
• Vector inner products	$2\ (n+m)$.

It is important to notice that no vector inner products are performed in the generalised Legendre algorithm, which are a usual feature of standard Krylov subspace methods. This feature makes the approach attractive to parallel machines where vector inner products are avoided where possible as they can cause bottlenecks in computation. However the cost per iteration of the generalised Legendre algorithm does depend on the iteration number but since the relative size of the iteration number compared to the size of the system is assumed to be small, this additional cost is small in relation to a vector inner product of size $n+m$.

An obvious stopping criterion for the iteration is that $||r_k||$ becomes sufficiently small. However as mentioned above, choosing not to update the residual can result in a saving in computation. In this case new stopping criteria must be found. In the following section it will be seen that an estimate of the current residual (in a special norm) is available at each step, which is more appropriate for use in a stopping criterion.

2.6.1 A minimisation property

When considering iterative methods for solving linear systems it is important to understand what function is being performed by the iteration. For example if the iterative method is a minimisation method, such as the conjugate gradient method for positive-definite systems, [36], it is important to understand what quantity is actually being minimised (in the case of CG it is the M-norm of the error). As may be expected from (2.25), the generalised Legendre approach considered here will minimise some generalisation of the w-norm of the residual vector. To understand this consider the following definition of the w-norm of a vector. Suppose that $v \in \mathcal{K}^k(M, r_0)$, where $\mathcal{K}^k(M, r_0)$ denotes the Krylov subspace $span\{r_0, Mr_0, \ldots, M^{k-1}r_0\}$, has the representation

$$v = \sum_{i=0}^{k-1} \mu_i M^i r_0.$$

Define the polynomial $k_v(t)$ by

$$k_v(t) = \sum_{i=0}^{k-1} \mu_i t^i.$$

Then the function $g(v) = ||k_v||_w$ defines a norm on $\mathcal{K}^k(M, r_0)$, denoted by $||v||_w$. Now consider $||r_k||_w$. Since $r_k = p_k(M)r_0$,

$$||r_k||_w^2 = ||p_k(t)||_w^2 = ||1 - ts_{k-1}(t)||_w^2,$$

and expanding $s_{k-1}(t)$ using (2.42) it can be seen that

$$||r_k||_w^2 = ||r_{k-1}||_w^2 - \varphi_{k-1}^2, \tag{2.54}$$

where φ_{k-1}^2 is known ((2.49) and $||r_0||_w = ||1||_w = \int_\Omega w(t)\,dt = 2(w_1d_1 + w_2d_2 + w_3d_3)$. $\int_\Omega w(t)\,dt$ is called the *first moment*. As the quantity $||1 - ts_{k-1}(t)||_w$ is minimised at each step over the span of the basis for P_k' by construction, $||r_k||_w$ is minimised over $\mathcal{K}^k(M, r_0)$ at each step. The cost of the calculation of $||r_k||_w$ from (2.54) is negligible and hence provides a cheap stopping criterion for the iteration.

2.7 Convergence of the generalised Legendre method

Before proceeding to present numerical results it must first be shown that the generalised Legendre method actually converges to a solution of (2.2), since this is not necessarily implied by (2.54). Since the least-squares polynomials are calculated in a Legendre-sense, the results given here are subtly different to those in [66] where the least-squares polynomials are calculated in a Chebyshev sense. Since all norms on finite dimensional spaces are equivalent, the first task is to calculate the 'equivalence coefficients' for the w and uniform norms. This is summarised in Lemma 2.7.1. Before proceeding to the lemma a preliminary result concerning the uniform norm of the Legendre polynomials is required.

It is customary to normalise the Jacobi polynomials with weight $w(t) = (1-t)^{\theta}(1+t)^{\phi}$ by insisting that the jth Jacobi polynomial takes the value $(\theta+1)(\theta+1)\dots(\theta+j)/j!$ at 1 (see [63]). Hence the Legendre polynomials are customarily scaled to take the value 1 at 1. Denoting these Legendre polynomials by $\{L_i\}_{i=0}^{\infty}$ it can be seen that

$$||L_i||_{\infty} = L_i(1) = 1,$$

(see [1]), and that

$$\int_{-1}^{1} L_i(t)^2 \, dt = \frac{2}{2i+1},$$

(see [63]). Since the orthonormal Legendre polynomials satisfy $\int_{-1}^{1} \tilde{l}_i(t)^2 \, dt = 1$,

$$\tilde{l}_i(t) = \left(\frac{2i+1}{2}\right)^{\frac{1}{2}} L_i(t),$$

and hence

$$\left|\left|\tilde{l}_i\right|\right|_{\infty} = \left(\frac{2i+1}{2}\right)^{\frac{1}{2}},$$

with the maximum value being taken at the ends of the interval $[-1, 1]$. The lemma can now be stated.

Lemma 2.7.1 *If $h_n \in P_n$,*

$$\begin{aligned} ||h_n||_w &\leq \left(2\textstyle\sum_{i=1}^3 w_i d_i\right)^{\frac{1}{2}} ||h_n||_\infty , \\ ||h_n||_\infty &\leq C(n+1)\,||h_n||_w , \end{aligned}$$

where $||\cdot||_\infty$ denotes the uniform norm on Ω and C is a constant which is independent of n.

Proof The first result is simple. Since $||h_n||_w^2 = \sum_{i=1}^3 ||h_n||_{w_i}^2$, it is clear that

$$||h_n||_w^2 \leq 2\sum_{i=1}^3 w_i d_i \, ||h_n||_\infty^2 ,$$

and the first result follows.

Now, writing $\Omega = \mathcal{I}_1 \cup \mathcal{I}_2 \cup \mathcal{I}_3$,

$$||h_n||_\infty^2 = \max\left\{ \max_{t\in\mathcal{I}_1}\left|\sum_{i=0}^n \gamma_i l_i^{(1)}(t)\right|^2, \max_{t\in\mathcal{I}_2}\left|\sum_{i=0}^n \delta_i l_i^{(2)}(t)\right|^2, \max_{t\in\mathcal{I}_3}\left|\sum_{i=0}^n \eta_i l_i^{(3)}(t)\right|^2, \right\}, \tag{2.55}$$

where $\{\gamma_i\}_{i=0}^n$, $\{\delta_i\}_{i=0}^n$ and $\{\eta_i\}_{i=0}^n$ are the coefficients in the expansion of h_n in terms of the orthonormal Legendre polynomials on $\mathcal{I}_1$, $\mathcal{I}_2$ and $\mathcal{I}_3$ respectively. Using the Cauchy-Schwartz inequality,

$$\left|\sum_{i=0}^n \gamma_i l_i^{(1)}(t)\right|^2 \leq \left(\sum_{i=0}^n \gamma_i^2\right)\left(\sum_{i=0}^n l_i^{(1)}(t)^2\right). \tag{2.56}$$

Recall

$$l_i^{(k)}(t) = \left(\frac{1}{w_k d_k}\right)^{\frac{1}{2}} \tilde{l}_i\left(\frac{t-c_k}{d_k}\right),$$

hence

$$\begin{aligned} \max_{t\in\mathcal{I}_k}\left(\sum_{i=0}^n l_i^{(k)}(t)^2\right) &\leq \sum_{i=0}^n \max_{t\in\mathcal{I}_k} l_i^{(k)}(t)^2 \\ &\leq \frac{1}{w_k d_k}\sum_{i=0}^n \frac{2i+1}{2}, \\ &= \frac{1}{w_k d_k}\frac{(n+1)^2}{2}, \end{aligned} \tag{2.57}$$

using the result above and standard formulae for sums arithmetic progressions. Combining (2.55), (2.56) and (2.57),

$$\begin{aligned} \|h_n\|_\infty^2 &= \tfrac{(n+1)^2}{2} \max\left\{ \tfrac{1}{w_1 d_1} \textstyle\sum_{i=0}^n \gamma_i^2, \tfrac{1}{w_2 d_2} \sum_{i=0}^n \delta_i^2, \tfrac{1}{w_3 d_3} \sum_{i=0}^n \eta_i^2 \right\}, \\ &\leq \tfrac{(n+1)^2}{2} \tfrac{1}{w^* d^*} \textstyle\sum_{i=0}^n \gamma_i^2 + \delta_i^2 + \eta_i^2, \end{aligned}$$

where $w^*d^* = \min\{w_1d_1, w_2d_2, w_3d_3\}$. Since the sum here is equal to $\|h_n\|_w^2$, the second result now follows with $C = 1/(2w^*d^*)^{\frac{1}{2}}$. □

Using the scaled Chebyshev polynomials with non-varying weights the $(n+1)$ in Lemma 2.7.1 can be replaced by $(n+1)^{\frac{1}{2}}$, however since the result is only useful to prove convergence, and is far from being a tight bound, this is not of concern. Lemma 2.7.1 together with the results of §2.2.1 can be used to prove the following convergence result for the generalised Legendre method. The method of proof is the same as that given by Saad [66] although the result is again subtly different.

Theorem 2.7.2 *Let $\Omega \subset \mathcal{E} = [-b, -a] \cup [a, b]$ where $b > a > 0$. Then the generalised Legendre method converges and the residuals, r_n, satisfy*

$$\frac{\|r_n\|}{\|r_0\|} \leq \frac{2}{(w^*d^*)^{\frac{1}{2}}} \left(\sum_{i=0}^{3} w_i d_i \right)^{\frac{1}{2}} (n+1) \left(\frac{\kappa - 1}{\kappa + 1} \right)^{\frac{n'}{2}},$$

where w^*d^* is defined in C, the constant from Lemma 2.7.1, $\kappa = b/a$ and n' is the largest even integer less than or equal to n.

Proof Since $r_n = p_n(A) r_0$,

$$\frac{\|r_n\|}{\|r_0\|} \leq \|p_n\|_\infty \leq C(n+1)\, \|p_n\|_w,$$

using Lemma 2.7.1, and since p_n minimises $\|p_w\|$ over degree n polynomials satisfying the interpolatory constraint $p(0) = 1$,

$$\frac{\|r_n\|}{\|r_0\|} \leq C \left(2 \sum_{i=0}^{3} w_i d_i \right)^{\frac{1}{2}} (n+1)\, \|p\|_\infty, \quad \forall p \in P_n,\ p(0) = 1, \tag{2.58}$$

using Lemma 2.7.1 again. Now let $\mathcal{E}$ be an inclusion set for Ω comprising of two intervals spaced symmetrically about the origin as above and let n' be the largest even integer less than or equal to n. The define

$$p(t) = s_{\frac{n'}{2}}(q_2(t)),$$

where $s_{\frac{n'}{2}}$ and q_2 are as defined in §2.2.1. Then by Lemma 2.2.3,

$$\|p\|_\infty \leq 2\left(\frac{\kappa-1}{\kappa+1}\right)^{\frac{n'}{2}},$$

where $\kappa = b/a$. With this choice of p in (2.58),

$$\frac{\|r_n\|}{\|r_0\|} \leq 2^{\frac{3}{2}}C\left(\sum_{i=0}^{3} w_i d_i\right)^{\frac{1}{2}}(n+1)\left(\frac{\kappa-1}{\kappa+1}\right)^{\frac{n'}{2}}. \tag{2.59}$$

The bound on the residual then follows from the definition of C in Lemma 2.7.1. Since $\kappa > 1$, the right hand side of (2.59) can be made arbitrarily small for sufficiently large n, so that the generalised Legendre method converges as stated. □

Comparing with the remarks of §2.2.1 it would appear that $2k$ steps of the generalised Legendre method would give a similar residual reduction to that of k steps of CGNR applied to (2.2), multiplied by a factor of $2k$. Again it is stressed that the the convergence result is merely intended to be a means of demonstrating convergence of the generalised Legendre method and is not designed to be a tight bound on convergence or a method for predicting the minimum number of iterations required to achieve a required residual reduction. Indeed results presented in §2.8 suggest that, per iteration, the performance of the generalised Legendre method and CGNR are similar although the smaller iteration cost of the Legendre method makes it far more favourable (c.f. figures 2-12 and 2-13).

2.8 Numerical results

In this section the results of some numerical experiments with the generalised Legendre method are presented. All the numerical experiments were performed on a Sun SPARC server 1000 with 4 processors.

Example 1

As a first test of the effectiveness of the generalised Legendre method as an iterative solution method, a matrix whose eigenvalues lie in three intervals which are 'very' disjoint is considered. The matrix A in (2.1) is chosen to be diagonal with n eigenvalues spaced evenly in$[\lambda_1, \lambda_n]$, and B is chosen to have diagonal upper-square part and zero lower part with m singular values spaced evenly in $[\sigma_1, \sigma_m]$. Similar eigenvalue distributions are considered in [66] except that the eigenvalues of $\mathcal{A}$ are taken to be evenly distributed instead of those of A and B. The values $\lambda_1 = 10$, $\lambda_n = 20$, $\sigma_1 = 200$ and $\sigma_m = 300$ are taken with $n = 200$ and $m = 50$, and f is a constant vector of ones in the top n terms and a vector of zeros in the bottom m terms. This results in the three eigenvalue intervals, $\Omega \subset \mathcal{I}^- \cup \mathcal{I}_1^+ \cup \mathcal{I}_2^+ \approx [-295, -190] \cup [10, 20] \cup [205, 310]$. The weight $w(t) = 1 \; \forall t \in \Omega$ is taken. The generalised Legendre method on three intervals, LS(3), is compared with the generalised Legendre method on two intervals, LS(2), (where the positive interval is the smallest inclusion interval for the two positive intervals in Ω), and the SYMMLQ iterative method [56] for symmetric linear systems. Since each method performs a different minimisation, the Euclidean norm of the residual is taken to be the measure in the comparison. Numerical results comparing the residual from the three methods at each iteration are shown in Figure 2-4.

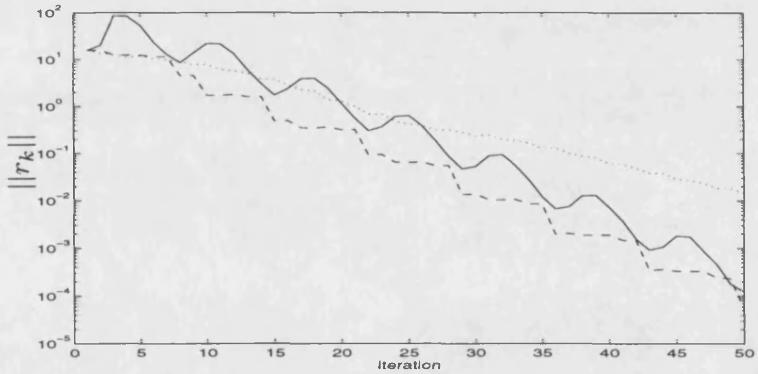


Figure 2-4: Example 1 : Comparing the residual reduction at each iteration of the LS(3) (−−), SYMMLQ (—) and LS(2) ($\cdots$) methods on a system with spectrum contained in three disjoint intervals

As can be seen, after approximately 25 iterations both LS(3) and SYMMLQ surpass the LS(2) residual reduction and the LS(3) method is slightly better than SYMMLQ. Plotting the norm of the residual against time is more revealing however since it is expected that the Legendre-based methods have a much faster iteration time for small iteration numbers. Figure 2-5 shows a plot of the results in Figure 2-4 on this new scale. It can be seen that the Legendre-based methods are indeed faster than the SYMMLQ method in terms of iteration time. It is clear that the residual reduction of the LS(2) method is competitive with SYMMLQ in time. The LS(3) method comprehensively beats both the other methods to achieve a far greater residual reduction than LS(2) in an almost equal iteration time, and a similar residual reduction to SYMMLQ in a much smaller time.

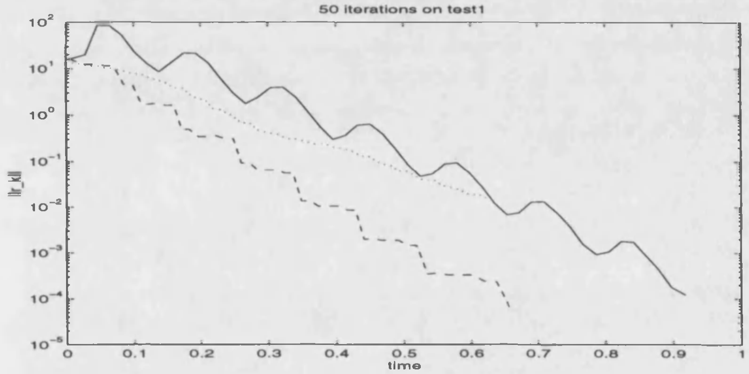


Figure 2-5: Example 1 : A plot of residual reduction against time for 50 iterations of the LS(3) (-- line), SYMMLQ (— line) and LS(2) (··· line) methods on a system with spectrum contained in three disjoint intervals.

The advantage in using LS(3) over LS(2) can be explained by analysing the three-interval residual polynomials, $r_i^{(3)}$, and the two-interval residual polynomials, $r_i^{(2)}$. The polynomials $r_i^{(3)}$ and $r_i^{(2)}$ for $i = 6, 8, \ldots, 14$ are plotted in figures 2-6 through 2-10.

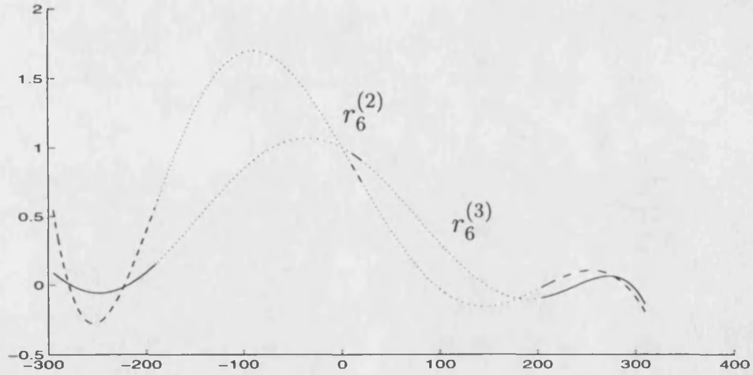


Figure 2-6: Example 1 : degree 6 residual polynomials. Solid (—) and dashed (--) portions indicate $r_6^{(3)}(\Omega)$ and $r_6^{(2)}(\Omega)$ respectively on $\mathcal{I}^- \cup \mathcal{I}_1^+ \cup \mathcal{I}_2^+$.

In allowing the $r_i^{(3)}$ polynomials to grow in the space between the two positive intervals, better approximation of zero is obtained over Ω, and since the spectrum of

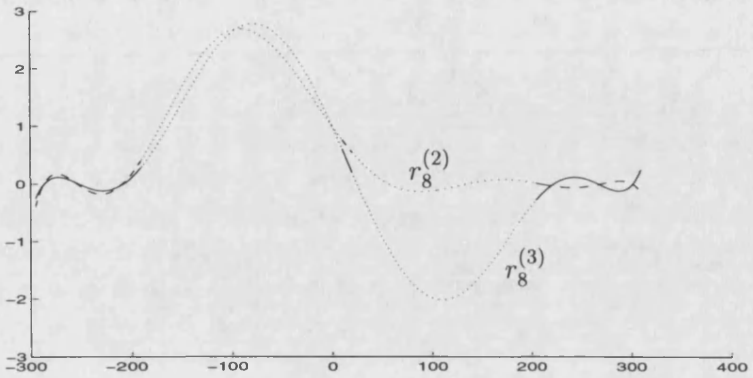


Figure 2-7: Example 1 : degree 8 residual polynomials. Solid (—) and dashed (– –) portions indicate $r_8^{(3)}(\Omega)$ and $r_8^{(2)}(\Omega)$ respectively on $\mathcal{I}^- \cup \mathcal{I}_1^+ \cup \mathcal{I}_2^+$.

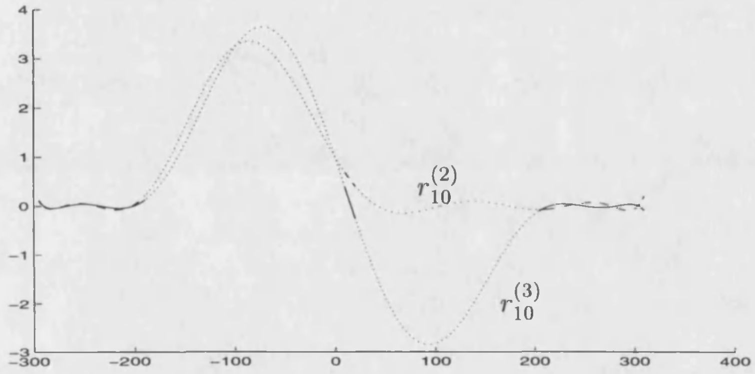


Figure 2-8: Example 1 : degree 10 residual polynomials. Solid (—) and dashed (– –) portions indicate $r_{10}^{(3)}(\Omega)$ and $r_{10}^{(2)}(\Omega)$ respectively on $\mathcal{I}^- \cup \mathcal{I}_1^+ \cup \mathcal{I}_2^+$.

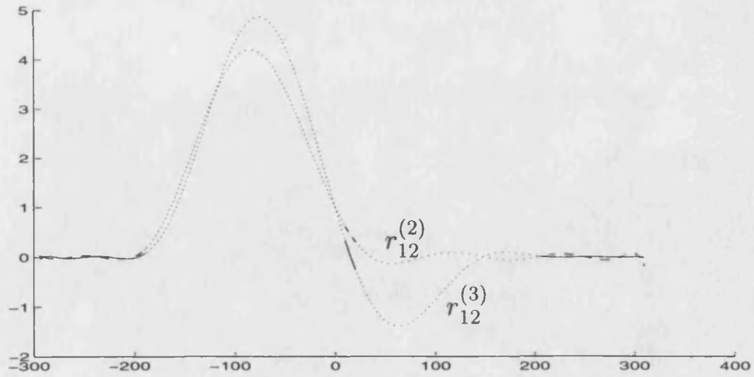


Figure 2-9: Example 1 : degree 12 residual polynomials. Solid (—) and dashed (– –) portions indicate $r_{12}^{(3)}(\Omega)$ and $r_{12}^{(2)}(\Omega)$ respectively on $\mathcal{I}^- \cup \mathcal{I}_1^+ \cup \mathcal{I}_2^+$.

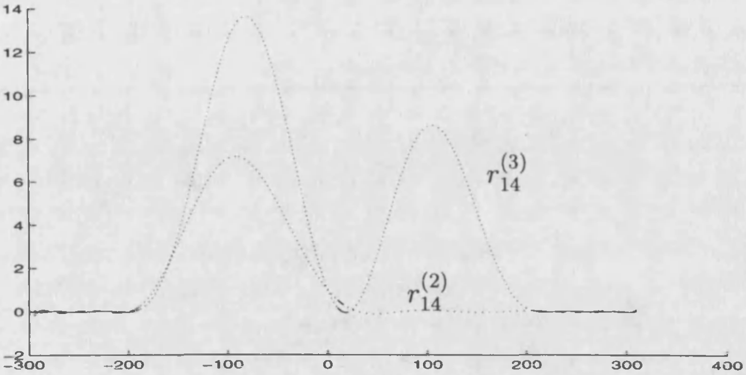


Figure 2-10: Example 1 : degree 14 residual polynomials. Solid (—) and dashed (−−) portions indicate $r_{14}^{(3)}(\Omega)$ and $r_{14}^{(2)}(\Omega)$ respectively on $\mathcal{I}^- \cup \mathcal{I}_1^+ \cup \mathcal{I}_2^+$.

i	$\int_\Omega r_i^{(2)^2}$	$\int_\Omega r_i^{(3)^2}$
1	241.251	220.069
2	33.323	21.429
3	40.026	21.344
4	21.064	10.510
5	12.409	9.837
6	14.411	9.346
7	7.869	4.048
8	7.030	4.038
9	5.967	.620
10	3.714	.618
11	3.845	.386
12	2.542	.173

Table 2.1: Example 1 : The squares of the $L_2(\Omega)$ norms of the residual polynomials based in two ($r_i^{(2)}$) and three ($r_i^{(3)}$) intervals.

M is contained in Ω better convergence is obtained. The $L_2(\Omega)$ norms of the $r_i^{(3)}$ and $r_i^{(2)}$ polynomials are shown in Table 2.1 and it is obvious from the entries that the $r_i^{(3)}$ polynomials are smaller on Ω.

Example 2

The next example is similar to the first. Again the eigenvalues of A are taken to lie between $\lambda_1 = 10$ and $\lambda_n = 20$ and the singular values of B are taken between $\sigma_1 = 200$ and $\sigma_m = 300$, but now instead of being spaced evenly they are distributed randomly. The matrices are no longer diagonal, they have a density of 0.2 (i.e. 20% of the entries are non-zero). This was achieved using the matlab routines *sprandsym* and *sprandn*. The eigenvalue distribution of $\mathcal{A}$ is shown in Figure 2-11. As can be seen there is some clustering of the eigenvalues in the outer intervals although the inner interval containing the majority of the eigenvalues still contains a fairly even spread. The eigenvalue clustering is favourable to the Lanczos based solvers like SYMMLQ and CGNR. Figure 2-12 shows the performance of these algorithms and the two generalised Legendre methods on this system.

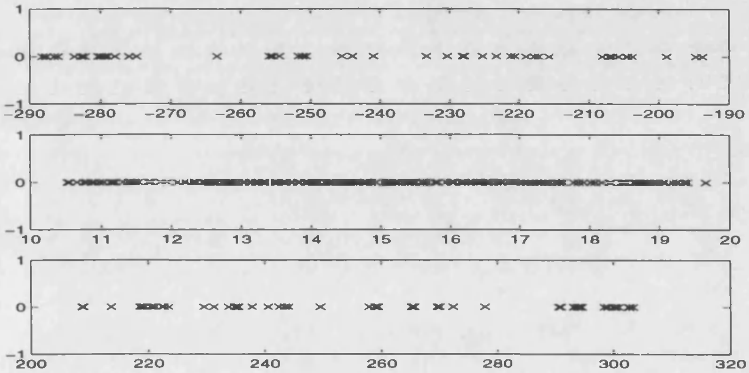


Figure 2-11: Example 2 :The three eigenvalue intervals of $\mathcal{A}$.

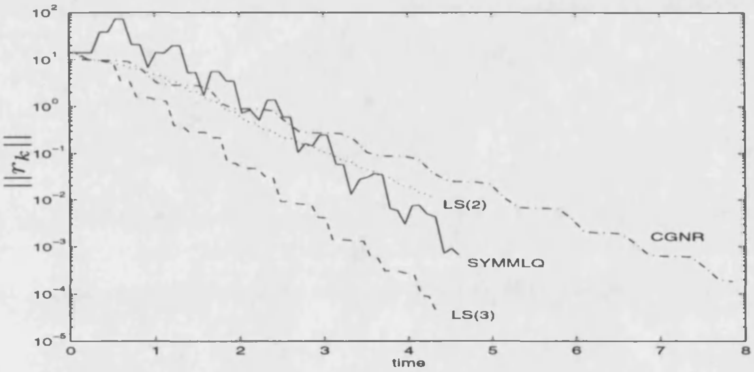


Figure 2-12: Example 2 : Solution-time plots - 50 iterations.

In this example 50 iterations of each method were performed. SYMMLQ beats LS(2) although again LS(3) exhibits the fastest residual reduction of these three methods. The poor performance of CGNR should lay to rest any fears from Theorem 2.7.2 which only promised that the residual reduction of LS(3) would be at most a constant multiple of $n + 1$ times as bad as CGNR applied to the system with half as many steps. It was however stressed that Theorem 2.7.2 was far from being a tight bound on convergence! In fact the reduction per iteration of both methods is approximately the same for this example, as can be seen in Figure 2-13. The reason that LS(3) beats CGNR so comprehensively in Figure 2-12 is that CGNR requires twice as many matrix-vector operations per iteration and since this is the most time consuming operation (since the matrices A and B are fairly dense), CGNR is slower than all of the other methods.

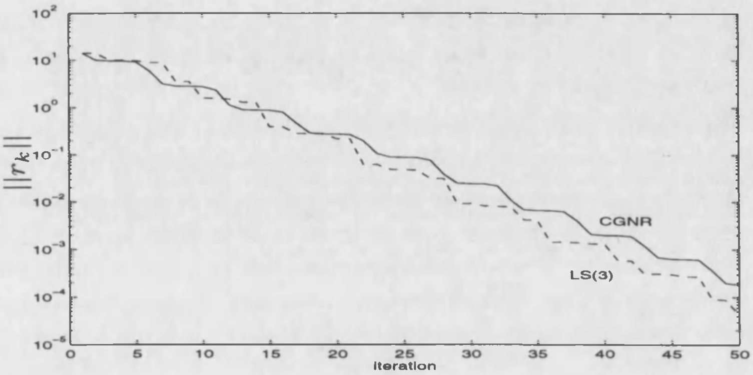


Figure 2-13: Example 2 : Comparing the residual reduction per iteration of CGNR and LS(3) on the second example.

Example 3

As a final example of this sort, eigenvalues of A and B were taken to lie randomly between $\lambda_1 = 10$, $\lambda_n = 20$ and $\sigma_1 = 100$, $\sigma_m = 300$. The values of n and m are 1000 and 500, and a density of 0.005 was chosen for both A and B. This resulted in an average of approximately 4.8 entries per row in A and 2.4 entries per row in B. The results of applying 100 iterations the four algorithms to this system (again f was taken to be a constant vector of ones in the top n terms and a column of zeros in the bottom m terms) can be seen in Figure 2-14.

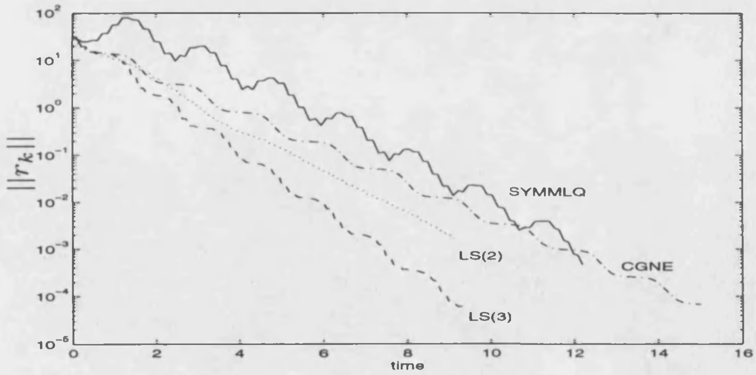


Figure 2-14: Example 3 : 100 iterations

Again it can be seen that LS(3) is the quickest to reduce the residual and this time LS(2) comprehensively beats SYMMLQ. In terms of iterations, SYMMLQ has the worst performance of any of the algorithms and is only competitive with CGNR on this example due to its shorter iteration time. Since the matrices in this example are very sparse, the extra matrix-vector product per iteration in CGNR slows the algorithm to a lesser extent, and the main time saving in using the LS algorithms is due to the lack of vector inner products compared to the Lanczos based methods.

From these last examples it seems sensible to conclude that the generalised Legendre algorithm gives a larger reduction in the residual for a fixed number of iterations than both SYMMLQ and CGNR provided that the gap between $\mathcal{I}_1^+$ and $\mathcal{I}_2^+$ is approximately an order of magnitude wide or more, and that the eigenvalues of $\mathcal{A}$ are not separated into a small number of clusters (which the Lanczos based algorithms will be quick to find). Such systems may arise in discretisations of p.d.e.s where the dependence on the spatial term is $O(h^{-k})$ in A and $O(h^{-(k-1)})$ in B. The LS(3) iterations are also significantly faster when the matrices A and B are large and very sparse so that vector inner product costs are large factor in the cost of one iteration step of SYMMLQ and CGNR. This can be seen in Figures 2-5 and 2-14. Figure 2-12 shows that when the matrices A and B are relatively full the saving in iteration time when using LS(3) instead of SYMMLQ is small as the iteration cost of both algorithms is dominated by the matrix-vector product, and vector inner products are small by comparison, although the greater residual reduction is given by LS(3). In every instance discussed LS(3) is superior to LS(2) and it can be seen that the extra computational cost of performing LS(3) is negligible and is not dependent on the size of the system to be solved. Since both LS(2) and LS(3) will perform similarly when the gap between $\mathcal{I}_1^+$ and $\mathcal{I}_2^+$ becomes small, no experiments of this type are displayed. It can also be seen that choosing between LS(2) and SYMMLQ is not easy since both methods give approximately the same reduction in Figure 2-5. In Figure 2-12, SYMMLQ is the winner, and LS(2) wins in Figure 2-14. As would probably be expected, CGNR is the poor relation in nearly all of these experiments although it performs surprisingly well in Figure 2-14 with better results than SYMMLQ for most of the iteration time, since the matrices considered are very sparse so that vector inner products dominate the iteration cost.

All of the experiments so far have used exact bounds for the eigenvalue intervals in Theorem 2.3.1. This is in fact not necessary, any bounds which can be obtained on the intervals $\mathcal{I}^-, \mathcal{I}_1^+$ and $\mathcal{I}_2^+$ can be used in the algorithm, provided that the upper bound on $\mathcal{I}^-$ is negative and the lower bound on $\mathcal{I}_1^+$ is positive (so that the constraint point at 0 lies outside). The closer these approximate bounds are to the actual bounds will lead to faster convergence. This is discussed in more detail in the next section. It may still be the case that the Lanczos based algorithms can outperform the generalised Legendre algorithms for these types of coefficient matrices when they are suitably preconditioned. One method of preconditioning (which is again attractive for parallel architectures) is that of *polynomial preconditioning.* This approach is also discussed in the following section.

2.9 Inexact eigenvalue bounds and least-squares polynomial preconditioners for augmented systems

Polynomial preconditioning is a popular strategy for solving symmetric linear systems and was introduced in [65]. Here, instead of solving the system $Mz = f$, the (left) preconditioned system

$$\Psi(M)\,Mz = \Psi(M)\,f, \tag{2.60}$$

is solved where Ψ is a polynomial (of small degree). In the case that M is a symmetric positive definite matrix, the motivation for preconditioning is that the conjugate gradient algorithm applied to (2.60) (notice that $\Psi(M)\,M$ can be made symmetric positive definite for suitable choices of Ψ, e.g. take Ψ to have all coefficients positive) is likely to converge more quickly than if it were applied to the original system if the condition number of $\Psi(M)\,M$ is less than that of M. This is a due to the fact the the standard bound on the M-norms of the errors associated with the conjugate gradient algorithm [29, Theorem 10.2.5] is made smaller on reducing the condition number of the system. The motivation behind the more general case when M is a symmetric matrix is that $\Psi(M)$ is an approximation to M^{-1}. Some more adventurous preconditionings attempt to cluster the eigenvalues of the preconditioned system so that iterative methods will converge in a small number of steps (see [52] and §B.1). A detailed discussion on the motivation behind polynomial preconditioning is given in [2].

It is important to appreciate that polynomial preconditioning is only a practical method of preconditioning on parallel or vector machines and is not suitable for scalar machines since, for symmetric positive definite problems, the conjugate gradient algorithm is an optimal algorithm with respect to the number of matrix-vector operations required. A polynomial preconditioned conjugate gradient algorithm will require more matrix-vector operations than the conjugate gradient algorithm on the unpreconditioned system. For very large and sparse systems, the small number of vector inner products required for the polynomial preconditioned conjugate gradient algorithm may occasionally make this approach attractive on a scalar machine but the saving made is likely to be small.

One of the first methods suggested for approximating the inverse of a matrix using polynomials was to use the truncated Neumann series,

$$\Psi(M) = \left(I + F^{-1}G + (F^{-1}G)^2 + \ldots + (F^{-1}G)^{l-1}\right) F^{-1},$$

where $M = F - G$ is a splitting of M. See [15]. This method will only be applicable if the splitting satisfies $||F^{-1}G|| \leq 1$ (as was the case for the Chebyshev semi-iterative method in §2.1) and will be more effective for smaller values of $||F^{-1}G||$.

Another method of defining a polynomial which satisfies $\Psi(M) \approx M^{-1}$ and which doesn't rely on a splitting of M is to set Ψ to be the solution of

$$\min_{\Psi \in P_{l-1}} |||I - \Psi(M)M||| .$$

Here $l-1$ is the maximum permissible degree of Ψ and, for the moment, the norm $|||\cdot|||$ is unspecified. It is clear that such a Ψ will satisfy $\Psi(M)M \approx I$. This definition of Ψ could be reinterpreted by insisting that $\Psi(t)t$ should be the closest approximation to 1 on the spectrum of M which, as usual, can be loosened to insisting that $\Psi(t)t$ is the closest approximation of 1 on some inclusion set of $\sigma(M)$. The above minimisation then becomes

$$\min_{\Psi \in P_{l-1}} |||1 - \Psi(t)t||| , \tag{2.61}$$

where intuitively the minimisation must be focussed on some inclusion set of the eigenvalues of M by choosing an appropriate norm. Following equations (2.23) and (2.25) it would seem appropriate to choose $|||\cdot|||$ to be either the uniform or w-norm. Saad [67] considers solving symmetric positive definite problems using a polynomial preconditioned conjugate gradient method, and takes $|||\cdot|||$ to be the w-norm (where w is a Chebyshev weight). He shows that it is unnecessary to find a good approximation to the smallest eigenvalue of M in this case and uses the least-squares polynomial on $[0, b]$, where b is the Gershgorin estimate of the largest eigenvalue of M, to precondition the system. In [3], M is again assumed to be symmetric positive definite and a comparison of least squares and optimal polynomial preconditionings is discussed, which results in the conclusion that each polynomial preconditioner is suited to a particular type of eigenvalue distribution.

Since $\mathcal{A}$ is indefinite, following the remarks in §2.2 it is only realistic to consider least-squares polynomial preconditioners. Then it would seem obvious to choose $\Psi(t) = s_{l-1}(t)$, where $s_{l-1}(t)$ is the solution polynomial of degree $l-1$ for $\mathcal{A}$ on three intervals, and can be calculated using the LS(3) algorithm described in sections 2.5.2 and 2.5.3. Then it is very easy to calculate $\Psi(t)$ with the (negligible) cost of calculating the preconditioner being independent of the size of the system. If unit weight is attached to each interval,

$$||r_{l-1}||_w = \left|\left|1 - \Psi_{l-1}^{(3)}(t)t\right|\right|_{L_2(\Omega)},$$

where the values $||r_{l-1}||_w$ are essentially free as was described in §2.6.1, and so the norm $|||\cdot|||$ above is taken to be the usual $L_2(\Omega)$ norm. If different constant weights are attached to each interval, $|||\cdot|||$ is simply a weighted $L_2(\Omega)$ norm.

The effectiveness of the LS(3) solution polynomial as a preconditioner will be compared with that of the solution polynomial on two intervals, the $l-1^{\text{th}}$ degree solution polynomials on two and three intervals being denoted $\Psi_{l-1}^{(2)}$ and $\Psi_{l-1}^{(3)}$ respectively. Since $\Psi(M)M$ is indefinite the polynomials will be used to precondition the SYMMLQ algorithm. Before presenting any numerical results the assumption of the previous section, that the eigenvalue intervals are known exactly, will be loosened.

2.9.1 Gershgorin bounds on three intervals

In this section it will not be assumed that the exact values of the bounds on the three eigenvalue intervals of $\mathcal{A}$ are known (which is equivalent to knowing the largest and smallest eigenvalues / singular values of A and B respectively). Instead it will be assumed that it is possible to find a set $\widetilde{\Omega} = \widetilde{\mathcal{I}}^- \cup \widetilde{\mathcal{I}}_1^+ \cup \widetilde{\mathcal{I}}_2^+$ with the property that

$\mathcal{I}^- \subset \widetilde{\mathcal{I}}^-$, $\mathcal{I}_1^+ \subset \widetilde{\mathcal{I}}_1^+$ and $\mathcal{I}_2^+ \subset \widetilde{\mathcal{I}}_2^+$. Least squares polynomials on $\widetilde{\Omega}$ can then be used in the iteration. Figure 2-15 shows that it is not possible to apply Gershgorin's theorem directly to the matrix $\mathcal{A}$, with $\Omega \approx [-292, -199] \cup [14, 20] \cup [216, 310]$, to find such a set $\widetilde{\Omega}$, since there are no three disconnected unions of Gershgorin disks (see [49, Ch. III 2.2.5]), in fact the largest Gershgorin disk contains all of the others. (The plot was generated using the *gersh* routine of [37]). Hence a different method must be found.

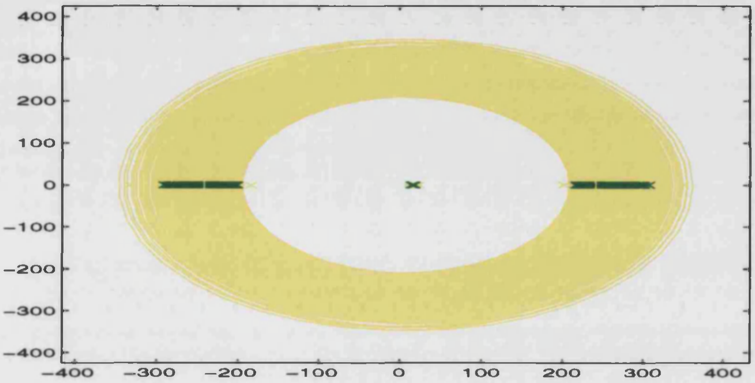


Figure 2-15: The Gershgorin disks of $\mathcal{A}$.

Notice that if $\varepsilon_1, \varepsilon_n, \delta_1, \delta_m > 0$ and if $\widetilde{\mathcal{I}}_1^+$ and $\widetilde{\mathcal{I}}_2^+$ are defined by

$$\begin{aligned}
\widetilde{\mathcal{I}}_1^+ &= [\lambda_1 - \varepsilon_1, \lambda_n + \varepsilon_n], \\
\widetilde{\mathcal{I}}_2^+ &= \left[\tfrac{\lambda_1-\varepsilon_1}{2} + \sqrt{\tfrac{(\lambda_1-\varepsilon_1)^2}{4} + (\sigma_1-\delta_1)^2}, \tfrac{\lambda_n+\varepsilon_n}{2} + \sqrt{\tfrac{(\lambda_n+\varepsilon_n)^2}{4} + (\sigma_m+\delta_m)^2}\right]
\end{aligned}$$

then $\mathcal{I}_1^+ \subset \widetilde{\mathcal{I}}_1^+$ and $\mathcal{I}_2^+ \subset \widetilde{\mathcal{I}}_2^+$ and the lower bound on both intervals is positive for $\varepsilon_1 < \lambda_1$. The two intervals are not necessarily non-overlapping, see §A.2 for a discussion of this phenomenon. Now consider the functions

$$\begin{aligned}
f(\varepsilon, \delta) &= \tfrac{\lambda_1-\varepsilon}{2} - \sqrt{\tfrac{(\lambda_1-\varepsilon)^2}{4} + (\sigma_m+\delta)^2}, \\
g(\varepsilon, \delta) &= \tfrac{\lambda_n+\varepsilon}{2} - \sqrt{\tfrac{(\lambda_n+\varepsilon)^2}{4} + (\sigma_1-\delta)^2}.
\end{aligned}$$

Then it is easy to show that $\frac{\partial f}{\partial \varepsilon} < 0$ for $0 \le \varepsilon < \lambda_1$ and $\frac{\partial f}{\partial \delta} < 0$ for $\delta \ge 0$. Similarly $\frac{\partial g}{\partial \varepsilon} > 0$ for $\varepsilon \ge 0$ and $\frac{\partial g}{\partial \delta} > 0$ for $0 \le \delta < \sigma_1$. Since the functions f and g define the upper and lower bounds on an interval $\widetilde{\mathcal{I}}^-$ corresponding to the same perturbation above, the following lemma has been proven.

Lemma 2.9.1 *If* $0 < \widetilde{\lambda}_1 \le \lambda_1 \le \lambda_n \le \widetilde{\lambda}_n$ *and* $0 < \widetilde{\sigma}_1 \le \sigma_1 \le \sigma_m \le \widetilde{\sigma}_m$, *then the* *eigenvalue intervals* $\mathcal{I}^-, \mathcal{I}_1^+, \mathcal{I}_2^+$ *of* $\mathcal{A}$ *where* $\rho(A) \subset [\lambda_1, \lambda_n]$ *and* $\sigma(B) \subset [\sigma_1, \sigma_m]$ *satisfy*

$$\mathcal{I}^- \subset \widetilde{\mathcal{I}}^-, \quad \mathcal{I}_1^+ \subset \widetilde{\mathcal{I}}_1^+, \quad \text{and } \mathcal{I}_2^+ \subset \widetilde{\mathcal{I}}_2^+,$$

where $\widetilde{\mathcal{I}}^-, \widetilde{\mathcal{I}}_1^+$*, and* $\widetilde{\mathcal{I}}_2^+$ *are the eigenvalue intervals found by replacing* $\lambda_1, \lambda_n, \sigma_1$ *and* σ_m *by* $\widetilde{\lambda}_1, \widetilde{\lambda}_m, \widetilde{\sigma}_1$*, and* $\widetilde{\sigma}_m$ *in Theorem 2.3.1.*

This result implies that, provided positive lower bounds can be found on the smallest eigenvalue of A and singular value of B and any upper bound can be found for the largest eigenvalue of A and singular value of B, the intervals found by substituting the eigenvalue / singular value bounds in Theorem 2.3.1 will contain the eigenvalue intervals of $\mathcal{A}$. Hence the Gershgorin estimates of the largest and smallest eigenvalues of A and singular values of B can be used to generate the three intervals provided that the estimates of the small eigenvalues and singular values are positive. Then the intervals $\widetilde{\mathcal{I}}^-, \widetilde{\mathcal{I}}_1^+$ and $\widetilde{\mathcal{I}}_2^+$ will be called the Gershgorin intervals. It will automatically be the case that the lower bounds obtained from Gershgorin's theorem on the smallest eigenvalue and singular values are positive if the matrices A and B^TB are strictly diagonally dominant. The Gershgorin intervals for the matrix $\mathcal{A}$ with Gershgorin disks in Figure 2-15 are $\widetilde{\Omega} \approx [-331, 183] \cup [14, 23] \cup [216, 309]$, and are shown in Figure 2-16. Hereinafter all numerical experiments are performed on the Gershgorin intervals unless otherwise stated. It should be noted that Gershgorin eigenvalue bounds are rarely tight and many improved results exist, see for example [88, 40]. As will be seen however the Gershgorin bounds give fair numerical performance for their simplicity and the use of tighter bounds is only expected to improve on these results.

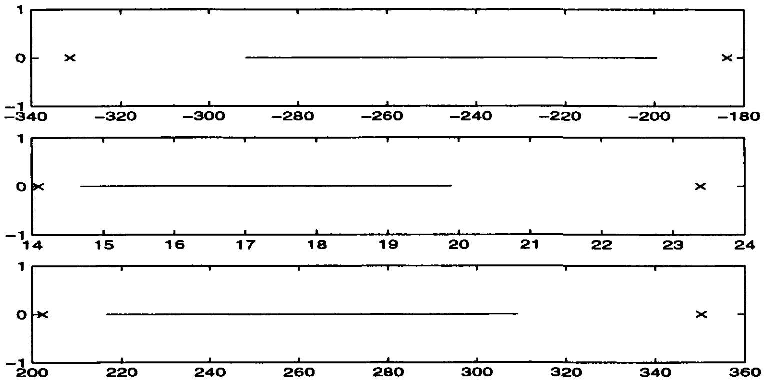


Figure 2-16: The three interval bounds (crosses) calculated from the Gershgorin bounds on the eigenvalues of A and singular values of B for the matrix $\mathcal{A}$ shown in Figure 2-15.

The Gershgorin intervals can be used in the algorithm LS(3), the resulting algorithm will be denoted G-LS(3). Since the least-squares residual polynomials of G-LS(3) are calculated over a larger set than is actually required, the performance G-LS(3) is expected to be slower than LS(3) on the exact intervals. In Figure 2-17, the difference between using exact eigenvalue bounds and the Gershgorin estimates can be seen. For small iteration numbers, which correspond to small degree residual polynomials, the difference is not large, although for some 'unfortunate' choices of degree of polynomial the difference can be larger, for example the 18th degree residual polynomial on the exact intervals gives a residual reduction of approximately one order of magnitude better than that on the Gershgorin intervals. Since only low order polynomials are being considered as preconditioners the difference is not too worrying.

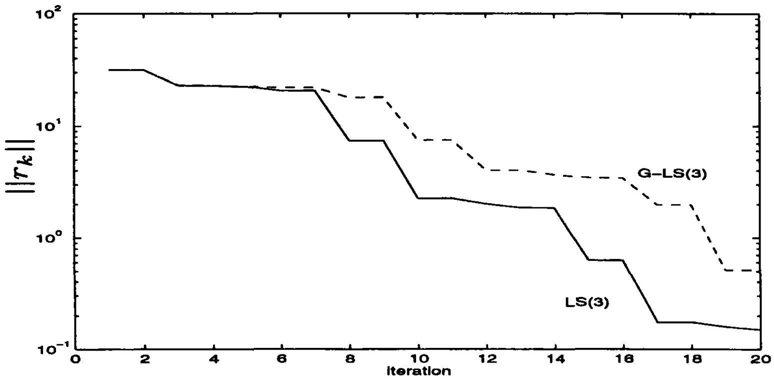


Figure 2-17: Comparing LS(3) and G-LS(3) where G-LS(3) uses the Gershgorin bound on the eigenvalue intervals of $\mathcal{A}$.

2.9.2 Polynomial preconditioning results

Since it has not been possible to perform experiments on a parallel machine, the time effect of the polynomial preconditioning has been simulated by dividing the time taken for any matrix-vector multiplications by the number of processors assumed. This will hopefully give a qualitative (but possibly optimistic and probably not a quantitative), representation of the actual performance that might be expected on a parallel machine, since these matrix-vector multiplications contribute the majority of the operations required in the algorithms. Effects such as increasing the degree of the solution polynomial on a fixed number of processors will not be greatly affected by this assumption, only when varying numbers of processors are being considered will results be affected. Graphs corresponding to time calculations on this basis will carry the label pseudo-time and the corresponding computer will be said to have n pseudo-processors.

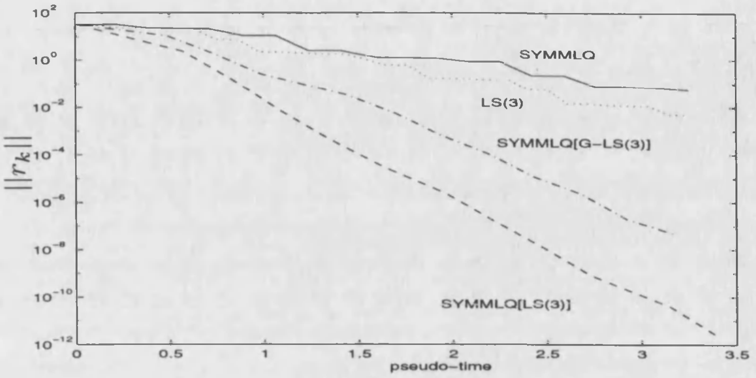


Figure 2-18: A comparison of preconditioned and unpreconditioned methods for $\mathcal{A}$ on 10 pseudo processors. SYMMLQ and LS(3) denote the unpreconditioned algorithms. SYMMLQ[LS(3)] denotes SYMMLQ preconditioned by the degree 10 solution polynomial on the actual eigenvalue intervals, SYMMLQ[G-LS(3)] denotes SYMMLQ preconditioned by the degree 10 solution polynomial on the Gershgorin intervals.

The LS(3) and G-LS(3) residual polynomials of degree 10 have been used to precondition SYMMLQ in Figure 2-18. The algorithms are applied to a variant of the matrix described in Example 3 of §2.8 where A and $B^T B$ are diagonally dominant so that the Gershgorin bounds on the eigenvalues of A and singular values of B satisfy the assumptions of Lemma 2.9.1. It is assumed that the machine being used has 10 pseudo-processors, and the unpreconditioned SYMMLQ and LS(3) residuals have been plotted for comparison. Again the LS(3) algorithm is faster than SYMMLQ although both of the preconditioned algorithms are considerably quicker. As would be expected the LS(3) preconditioned algorithm, SYMMLQ[LS(3)], is the fastest of the four, with the G-LS(3) preconditioned algorithm, SYMMLQ[G-LS(3)], being not quite as fast. The improvement of SYMMLQ[G-LS(3)] over both SYMMLQ and LS(3) is considerable, and proves the point made at the end of §2.8, that the Lanczos based algorithms can outperform the LS(3) algorithm when suitably preconditioned.

Figure 2-19 demonstrates the effect altering the number of pseudo-processors assumed when preconditioning SYMMLQ by the degree 10 G-LS(3) polynomial. It can be seen that there is little improvement over unpreconditioned SYMMLQ when only two pseudo-processors are used, since the amount of computing time taken to apply the preconditioner almost outweighs any gain in performance made by iterating on the preconditioned system. By increasing the number of pseudo-processors to five and large increase in performance is made, the gain in increasing to ten pseudo-processors is not so large since with this many processors, the effects of operations other than matrix-vector products begin to be noticed.

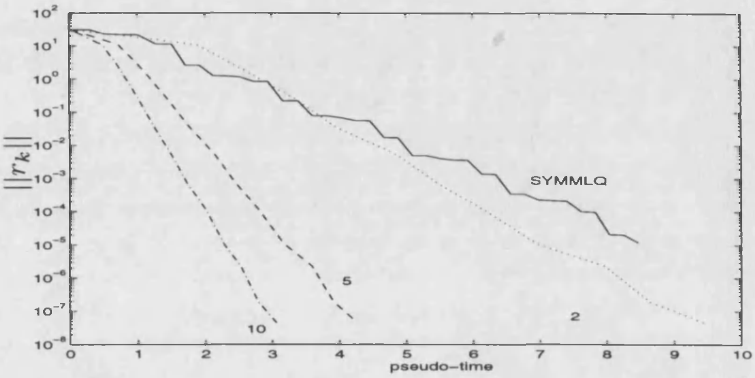


Figure 2-19: SYMMLQ preconditioned by the degree 10 Gershgorin solution polynomial on a range of pseudo processors.

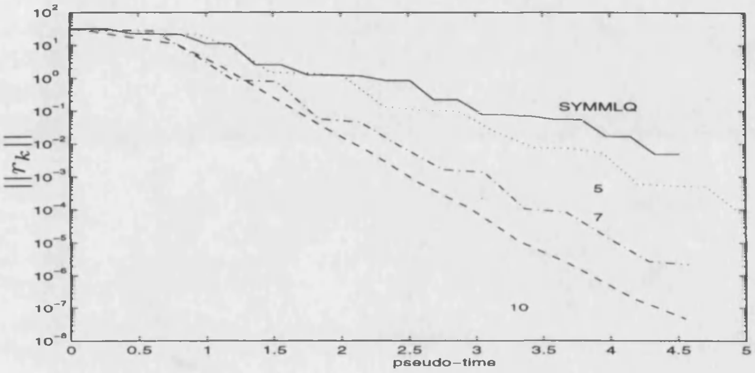


Figure 2-20: SYMMLQ preconditioned by degree 5,7 and 10 Gershgorin polynomials on a 4 pseudo processors.

The effect of increasing the degree of the preconditioner is demonstrated in Figure 2-20, here 4 pseudo-processors are assumed. Here it can be seen that the degree 5 preconditioner gives a slight improvement over the unpreconditioned algorithm, with degree 7 and 10 preconditioners performing faster. Increasing the degree of the preconditioner past 10 does not greatly increase the speed of solution since the non matrix-vector operations become more dominant.

Finally, Figure 2-21 compares using the G-LS(3) and G-LS(2) polynomials as preconditioners for a range of degrees. Here the number of processors is irrelevant as the time to apply the preconditioner is independent of whether it was calculated on two intervals or three. It is surprising to see that the degree 5 G-LS(3) preconditioner performs worse than the corresponding G-LS(2) preconditioner for most of the iteration time, although the difference is small. The result is doubly surprising when it is considered that the matrix residual from two intervals is larger than that from three, $\left\|I - \Psi_5^{(2)}(\mathcal{A})\mathcal{A}\right\| = 1.2826$ whereas $\left\|I - \Psi_5^{(3)}(\mathcal{A})\mathcal{A}\right\| = 1.0672$. Hence matrix residuals by themselves are not enough to determine whether a given preconditioner is a good one, indeed it is true that the matrix residual using the degree 5 solution polynomial from the exact eigenvalue intervals has value $||I - \Psi(M)M|| = 1.1795$ which is larger than that from the Gershgorin intervals, but Figure 2-17 shows that both G-LS(3) and LS(3) give approximately the same residual reduction at the 5th iteration. With slightly higher degree preconditioners the clear conclusion is that the G-LS(3) polynomials perform better than the G-LS(2) polynomials by a considerable margin for preconditioning SYMMLQ on these problems, indeed the degree 10 G-LS(3) preconditioner on 10 pseudo-processors performs as well as the degree 15 G-LS(2) preconditioner on 15 pseudo-processors.

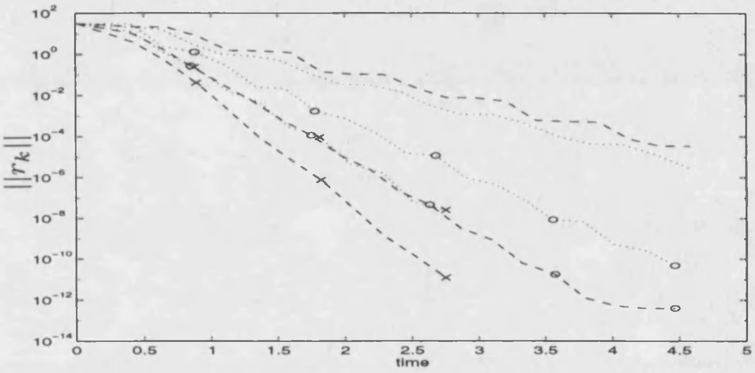


Figure 2-21: Comparing LS(3) and LS(2) preconditioners of degree 5, 10, and 15 on 5, 10 and 15 pseudo processors respectively. (— — —) indicates LS(3), (···) indicates LS(2) polynomials of degree 5. (o) indicates degree 10 and (x) degree 15.

The poor performance of the degree 5 G-LS(3) polynomial may be explained by considering the residual polynomials calculated on two and three intervals which are plotted in Figure 2-22. Although the G-LS(3) polynomial is smaller than G-LS(2) on most of the domain Ω, it is larger on the interval $\widetilde{\mathcal{I}}_1^+$, and this may be causing the poorer performance. This is where the variable Legendre weight allowed in the LS(3) algorithm can be applied, if a greater weight is attached to this interval then it might be expected that better performance could be obtained from SYMMLQ[G-LS(3)] as this will act to improve the approximation of 0 on $\widetilde{\mathcal{I}}_1^+$.

Figure 2-23 shows the effect of increasing the weight on the central interval from

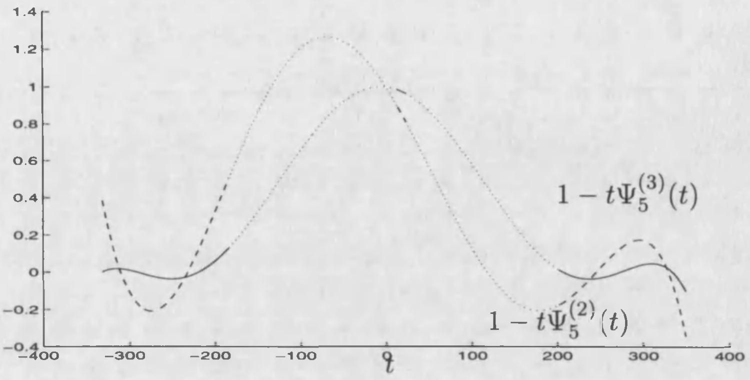


Figure 2-22: The polynomials $1 - t\Psi_5^{(2)}(t)$ and $1 - t\Psi_5^{(3)}(t)$. Solid and dashed sections indicate the intervals which define the domain Ω.

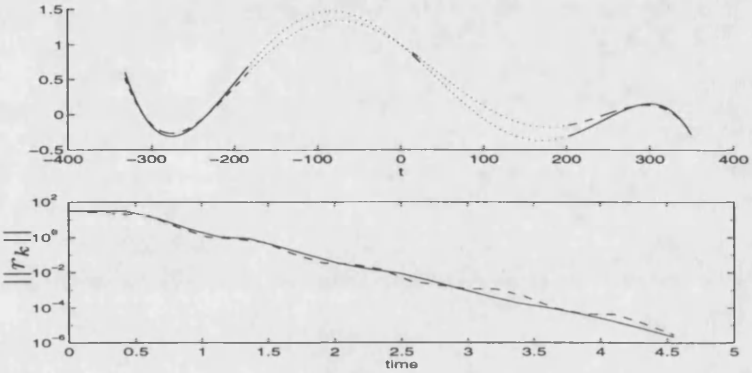


Figure 2-23: Altering the weight on $\widetilde{\mathcal{I}}_1^+$ from 1 to 20. The upper graph shows the G-LS(3) (—) and G-LS(2) (− − −) residual polynomials. With greater weight on $\widetilde{\mathcal{I}}_1^+$ the G-LS(3) polynomial is smaller on this interval (c.f. Figure 2-22). The lower graph shows the performance of SYMMLQ[G-LS(3)] and SYMMLQ[G-LS(2)] with the same notation.

1 to 20. Then the value of $r_6^{(3)}(t)$ on $\widetilde{\mathcal{I}}_1^+$ is then slightly smaller than that of $r_6^{(2)}(t)$ on the same interval. Almost equal performance is displayed by SYMMLQ[G-LS(3)] and SYMMLQ[G-LS(2)]. It may be expected that, because of this result, only the eigenvectors corresponding to eigenvalues in $\widetilde{\mathcal{I}}_1^+$ have components in the direction of the right hand side vector. However, increasing the value of the weight on $\widetilde{\mathcal{I}}_1^+$ further does not increase performance as the better approximation of zero by $r_6^{(3)}(t)$ on $\widetilde{\mathcal{I}}_1^+$ is at the expense of a loss in accuracy on the remaining two intervals, and the residual improvement is lost. For very large weights ($O(10^3)$) on $\widetilde{\mathcal{I}}_1^+$, the solution diverges, so that care needs to be taken when choosing appropriate weights. Choosing an optimal weighting strategy is an interesting problem, but ultimately the best choice of weights will depend on the eigencomponents of the right hand side vector which are very unlikely

to be known *a priori*, and if they were, far better solution strategies could be found.

2.10 Summary

The main point of this chapter was to explore the eigenvalue result in Theorem 2.3.1, and see if the extra information it gives about the spectrum of the coefficient matrices considered here, could be used in a numerical solver. The method for generating orthogonal polynomials over three intervals is a generalisation of Saad's work in [66]. The devised iterative method, LS(3), which has a residual polynomial which is minimal in an appropriate norm, and is constructed from a basis of orthogonal polynomials over the three intervals compares favourably with the standard iterative methods CGNR and SYMMLQ for indefinite symmetric systems in terms of the number of operations required. LS(3) needs one less matrix-vector multiply per iteration than CGNR and two fewer scalar-vector multiplications and vector additions than SYMMLQ per iteration. Furthermore the LS(3) algorithm requires no vector inner products since the orthogonality conditions are on polynomials, as opposed to vectors as is the case for most Krylov methods, and are enforced using polynomial inner products which are much cheaper than vector inner products. This lack of vector inner products makes the approach attractive for parallel computers where inner products can cause bottlenecks in computation.

Numerical experiments comparing LS(3) with SYMMLQ, CGNR and LS(2), the algorithm based on two eigenvalue intervals from [66], showed LS(3) to behave similarly to CGNR in terms of residual reduction per iteration and better than SYMMLQ. However with a much shorter time per iterate than both SYMMLQ and CGNR, the LS(3) algorithm converged much more quickly. The algorithm was also considerably faster than LS(2) and the cost of calculating orthogonal polynomials on three intervals rather than two was negligible.

In §2.9 it was seen that the exact eigenvalue intervals bounds assumed previously were not necessary, and conditions such that approximate eigenvalue intervals based on Gershgorin type estimates of the individual spectra of the matrices A and B defined three intervals which contained the three eigenvalue intervals were given in Lemma 2.9.1. It was seen that the algorithm G-LS(3) based on the approximate eigenvalue information compared well with the LS(3) algorithm (based on exact eigenvalue information). The G-LS(3) algorithm was seen to be a good preconditioner for the SYMMLQ algorithm, and final numerical experiments were performed in an attempt to examine the type of performance which might be expected if the algorithms were implemented on a parallel machine.

Whether the LS(3) algorithm is an effective method of solving discretisations of groundwater flow problems remains to be seen. For the algorithm LS(3) to be effective it is imperative that the three eigenvalue intervals defined in Theorem 2.3.1 are disjoint, otherwise it would be more sensible to apply the LS(2) algorithm, or any of the other methods mentioned here which do not rely on the coefficient matrix having a spectrum contained in three intervals. In order that the three intervals are disjoint, a simple rule is that the eigenvalues of the matrix A must be a lot smaller than the singular values of the matrix B. This will not be the case for a general mixed finite element discretisation of a groundwater flow problem, since typically the matrix A will have large and small eigenvalues, depending on the geometry of the underlying problem. However it may be the case that with a suitable preconditioning, the coefficient matrix may be made to have a spectrum of the required form (for example by scaling A by a small parameter - see Appendix A), and then the LS(3) algorithm will become competitive.

Chapter 3

A generalised least squares approach to solving augmented systems

3.1 Introduction

It can be seen (§3.2) that the y component of the solution of

$$\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix}, \tag{3.1}$$

is the solution of the *generalised least-squares* (see for example [42]) problem

$$minimise_{\ y\in\mathbb{R}^m} \ ||By - b||_{A^{-1}}, \tag{3.2}$$

where $||v||^2_{A^{-1}} = v^T A^{-1} v$. The LSQR algorithm [58] introduced in §3.3 is a method for solving least-squares problems of the form $minimise_{\ y\in\mathbb{R}^m} \ ||Cy - c||$ and hence could be used naïvely to solve the generalised least-squares problem above, if the Cholesky factorisation $A = LL^T$ were known, upon setting $C = L^{-1}B$ and $c = L^{-1}b$. This approach would require one multiplication with each of L^{-1} and L^{-T} per iteration step. In §3.4 it is shown that a variation of the LSQR algorithm exists that can be applied to the generalised least-squares problem in which the solves with L and L^T are replaced with a single solve with A. In cases where the level of fill-in in L is high, operations involving L can be expensive whereas an iterative method which calculates the effect of A^{-1} using only operations involving A can be relatively inexpensive, especially if a good preconditioner for A is known. It is essential that L be computed accurately if it is to be used to form a linear-least squares problem as described above. This can be impractical if A is large. However if a good preconditioner for A is available it may still be reasonable to expect to be able to operate with A^{-1}. For example consider the case when A is the mass matrix associated with a piecewise linear basis on an arbitrary triangular grid. Wathen [82] (see also §5.4) has shown that if $H_D = \operatorname{diag}(A)$ then $\kappa = \kappa(H_D^{-1}A) = 4$ so that the parameter

$$\alpha = \left(\frac{1 - \kappa^{\frac{1}{2}}}{1 + \kappa^{\frac{1}{2}}} \right),$$

which is a (pessimistic) upper bound on the convergence rate of the preconditioned conjugate gradient method, is small and more importantly is independent of the mesh parameter. The conjugate gradient iterates satisfy

$$||z - z_k||_A \leq \alpha^k \, ||z - z_0||_A ,$$

see [29], so that the preconditioned conjugate gradient method will be fast to solve large systems of this form. For example, the conjugate gradient method on the diagonally scaled mass matrix above would require at most $k/\log 9$ steps to reduce the energy norm of the error by a factor of 10^k.

Assuming that it is possible to calculate the effect of A^{-1} is equivalent to assuming that it is possible to precondition (3.1) with a preconditioner that contains A^{-1}. This assumption is made in [22] where the authors conclude that if it is feasible to operate with A^{-1} then an efficient way to solve (3.1) using a minimum residual approach (see for example [56, 69]) is to solve the *Schur complement* equations

$$B^T A^{-1} B y = B^T A^{-1} b. \tag{3.3}$$

using the MINRES algorithm [56] (and then recover x). In §3.5 this approach is compared with the generalised least-squares approach outlined above, and it is noted that since the Schur complement system is simply the normal equations for the generalised least-squares problem it is more favourable to solve (3.1) using a least-squares approach.

3.2 The generalised least-squares connection

Notice that x can be eliminated from (3.1) to form (3.3), an equation in y only. The matrix $B^TA^{-1}B$ is often referred to as the Schur complement. If y solves (3.3) the solution $(x^T \;\; y^T)^T$ of (3.1) can be recovered using $x = A^{-1}(b - By)$. From (3.3),

$$p^TB^TA^{-1}By = p^TB^TA^{-1}b \quad \forall p \in \mathbb{R}^m,$$

so that, on defining the inner-product $\langle \cdot \, , \, \cdot \rangle$ on $\mathbb{R}^n$ by $\langle u \, , \, v \rangle = u^TA^{-1}v$, (3.3) can be reformulated as

$$\textit{Find } y \in \mathbb{R}^m \textit{ such that} \quad \langle Bp \, , \, By \rangle = \langle Bp \, , \, b \rangle \quad \forall p \in \mathbb{R}^m. \tag{3.4}$$

Setting $r = b - By$ implies that if y solves (3.4), $\langle Bp \, , \, r \rangle = 0 \quad \forall p \in \mathbb{R}^m$, so that the residual r is orthogonal to the span of the columns of B, denoted $span(B)$, with respect to $\langle \cdot \, , \, \cdot \rangle$. This condition implies that By is the closest representation of b in $span(B)$ with respect to the norm induced by $\langle \cdot \, , \, \cdot \rangle$, i.e. that y is determined by the generalised least-squares problem,

$$\textit{Find } y \in \mathbb{R}^m \textit{ such that } \min_{p \in \mathbb{R}^m} ||b - Bp||_{A^{-1}} = ||b - By||_{A^{-1}}, \tag{3.5}$$

where $||u||_{A^{-1}} = \langle u \, , \, u \rangle^{\frac{1}{2}} = (u^TA^{-1}u)^{\frac{1}{2}}$. The reverse argument, that (3.5) is equivalent to (3.3), can be seen by minimising the functional $F(p) = \frac{1}{2} ||b - Bp||^2_{A^{-1}}$.

3.3 Golub-Kahan bidiagonalisation and LSQR

As is the case for many iterative solution methods for symmetric systems (for example CG [36], SYMMLQ and MINRES [56]), the LSQR algorithm [58] for the solution of least-squares problems can be seen to have its roots buried in the Lanczos process [46] for reducing a symmetric matrix to tridiagonal form. In §3.3.1 the Lanczos process is reintroduced and in §3.3.2 it is explained how Golub and Kahan used the Lanczos process to reduce a given matrix to bidiagonal form, and then how Paige and Saunders extended this bidiagonalisation process to form an iterative solution method for least-squares problems.

3.3.1 The Lanczos process

Given a symmetric matrix M and a starting vector b, the Lanczos process [46] reduces M to tridiagonal form as follows.

The Lanczos process

```
Set $\hat{\beta}_1\hat{v}_1 = b$,

For $i = 1, 2, \dots$
    $w_i = M\hat{v}_i - \hat{\beta}_i\hat{v}_{i-1}$,
    $\hat{\alpha}_i = \hat{v}_i^T w_i$,
    $\hat{\beta}_{i+1}\hat{v}_{i+1} = w_i - \hat{\alpha}_i\hat{v}_i$,
```

where $\hat{v}_0 := 0$ and $\hat{\beta}_i \geq 0$ is chosen so that $||\hat{v}_i|| = 1, \;\; i = 1, 2, \dots$. The recurrences of the Lanczos process can be written more succinctly as

$$M\hat{V}_k = \hat{V}_k T_k + \hat{\beta}_{k+1}\hat{v}_{k+1}e_k^T = \hat{V}_{k+1}H_k, \tag{3.6}$$

where $\hat{V}_k = [\hat{v}_1, \dots, \hat{v}_k]$ and

$$T_k = \begin{bmatrix} \hat{\alpha}_1 & \hat{\beta}_2 & & & \\ \hat{\beta}_2 & \hat{\alpha}_2 & \hat{\beta}_3 & & \\ & \cdot & \cdot & \cdot & \\ & & \cdot & \cdot & \hat{\beta}_k \\ & & & \hat{\beta}_k & \hat{\alpha}_k \end{bmatrix} \quad \text{and } H_k = \begin{bmatrix} T_k \\ \hat{\beta}_{k+1}e_k^T \end{bmatrix}.$$

The set $\{\hat{v}_i\}_{i=1}^k$ can be shown to be constructed so that $\hat{v}_i^T\hat{v}_j = \delta_{ij}$ (in the absence of rounding error) so that $\hat{V}_k^T\hat{V}_k = I_k$. Further, since $\hat{v}_{k+1} \in span\{\hat{v}_{k-1}, \hat{v}_k, M\hat{v}_k\}$, inductively it can be seen that $\{\hat{v}_i\}_{i=1}^k$ is an orthonormal basis for the *Krylov subspace* $\mathcal{K}^k(b, M) = span\{b, Mb, \dots, M^{k-1}b\}$.

As mentioned above, many iterative solution methods for the system $Mz = b$ can be derived from the Lanczos process. For example if $\hat{p}_k$ is the solution of the system

$$T_k\hat{p}_k = \hat{\beta}_1 e_1, \tag{3.7}$$

and if $z_k := \hat{V}_k\hat{p}_k$ then (3.6) gives

$$Mz_k = b + \hat{\beta}_{k+1}\hat{\eta}_k\hat{v}_{k+1},$$

where η_k is the last entry in $\hat{p}_k$. Hence z_k will solve $Mz = b$ whenever $\hat{\beta}_{k+1}\hat{\eta}_k$ is negligible. Defining $r_k = b - Mz_k$ implies $r_k = \hat{\beta}_{k+1}(e_k^T\hat{p}_k)\hat{v}_{k+1}$ and hence

$$\hat{V}_k^T r_k = 0.$$

That is, the residual r_k is orthogonal to the span of the vectors $\hat{v}_i,\ i = 1,\dots,k$ and hence is orthogonal to the span of the Krylov subspace $\mathcal{K}^k(b, M)$. This is often referred to as a *Ritz-Galerkin* condition. When M (and therefore T_k) is positive definite, using the Cholesky decomposition of T_k to solve (3.7) (which is easily obtained from the Cholesky decomposition of T_{k-1} at trivial cost) is the foundation of the conjugate gradient method.

Subproblems involving H_k give rise to the related SYMMLQ and MINRES algorithms. Consider the subproblem

$$\text{minimise } ||t_{k+1}|| \text{ subject to } H_k^T t_{k+1} = \hat{\beta}_1 e_1. \tag{3.8}$$

Then defining $z_k = \hat{V}_{k+1}t_{k+1}$ implies that

$$\begin{aligned} \hat{V}_k^T M z_k &= H_k^T\left(\hat{V}_{k+1}^T\hat{V}_{k+1}\right)t_{k+1} \\ &= \hat{\beta}_1 e_1 \\ &= \hat{V}_k^T b \end{aligned}$$

Hence it is clear that

$$\hat{V}_k^T r_k = 0,$$

and so again the residual is orthogonal to the Krylov subspace $\mathcal{K}^b(k, M)$. Using LQ factorisations of H_k to solve (3.8) is the idea behind the SYMMLQ algorithm.

The subproblem

$$\text{minimise } \left|\left|H_k t_k - \hat{\beta}_1 e_1\right|\right|, \tag{3.9}$$

is that of the MINRES algorithm, so called because setting $z_k = \hat{V}_k t_k$ implies

$$\begin{aligned} ||Mz_k - b|| &= \left|\left|\hat{V}_{k+1}\left(H_k t_k - \hat{\beta}_1 e_1\right)\right|\right|, \\ &= \left|\left|H_k t_k - \hat{\beta}_1 e_1\right|\right|, \end{aligned}$$

and hence at each step of the algorithm the norm of the residual over the Krylov subspace $\mathcal{K}^k(b, M)$ is minimised. The minimisation (3.9) is performed using the QR decomposition of H_k.

Since the LQ and QR factorisations do not require that the matrix M is positive definite, the SYMMLQ and MINRES algorithms can be applied to arbitrary symmetric linear systems, whereas the CG algorithm can only be applied to symmetric positive-definite systems. The SYMMLQ and MINRES iterative methods are discussed in detail in [56] and [70].

3.3.2 Golub-Kahan bidiagonalisation and LSQR

The *singular value decomposition* of a matrix B is $B = U\Sigma V^T$ where Σ is a rectangular, diagonal matrix having the same dimensions as B with non-negative entries, and U, V are unitary matrices of appropriate size. In [28] Golub and Kahan suggest an iterative procedure for generating a bidiagonal matrix, E, having the same singular values as B. The matrix E can be seen to be the result of applying the Lanczos process to the matrix

$$\mathcal{A}_{\text{LS}} = \begin{bmatrix} I & B \\ B^T & 0 \end{bmatrix}$$

as follows.

If the Lanczos process is applied to $\mathcal{A}_{\text{LS}}$ with starting vector $\begin{bmatrix} b^T & 0^T \end{bmatrix}^T$ it can be seen that the orthonormal vectors $\hat{v}_i$ are of the form

$$\hat{v}_i =: \begin{cases} \begin{bmatrix} u_j \\ 0 \end{bmatrix} & i = 2j-1, \quad j \in \mathbb{N} \\ \\ \begin{bmatrix} 0 \\ v_j \end{bmatrix} & i = 2j, \quad j \in \mathbb{N} \end{cases}$$

and that

$$\hat{\alpha}_i =: \begin{cases} 1 & i = 2j-1, \quad j \in \mathbb{N} \\ 0 & i = 2j, \quad j \in \mathbb{N} \end{cases}.$$

Notice that since $\{\hat{v}_i\}_{i=1}^{k}$ is an orthonormal set, both $\{u_j\}_{j=1}^{[\frac{k+1}{2}]}$ and $\{v_j\}_{j=1}^{[\frac{k}{2}]}$ (where $[\cdot]$ denotes 'integer part of') are orthonormal sets. It can also be verified that $\{u_j\}_{j=1}^{l}$ spans the Krylov space $\mathcal{K}^l(BB^T, b)$ whilst $\{v_j\}_{j=1}^{l}$ spans the space $\mathcal{K}^l(B^TB, B^Tb)$. Renaming the $\hat{\beta}_i$'s as

$$\hat{\beta}_i =: \begin{cases} \beta_j & i = 2j-1, \quad j \in \mathbb{N} \\ \alpha_j & i = 2j, \quad j \in \mathbb{N} \end{cases}$$

allows the tridiagonal matrix T_{2l+1} at the $2l + 1^{\text{st}}$ step of the Lanczos process on $\mathcal{A}_{\text{LS}}$ to be written as

$$T_{2l+1} = \begin{bmatrix} 1 & \alpha_1 & & & & & \\ \alpha_1 & 0 & \beta_2 & & & & \\ & \beta_2 & 1 & \alpha_2 & & & \\ & & \cdot & \cdot & \cdot & & \\ & & & \cdot & \cdot & \cdot & \\ & & & & \alpha_l & 0 & \beta_{l+1} \\ & & & & & \beta_{l+1} & 1 \end{bmatrix}.$$

The unit entries in T_{2l+1} can be grouped together into the top-left block by operating on T_{2l+1} with a permutation matrix P_{2l+1}. If P_{2l+1} is designed to keep the entries α_i

and β_i in order then

$$P_{2l+1}T_{2l+1}P_{2l+1}^T = \begin{bmatrix} I_{l+1} & E_l \\ E_l^T & 0 \end{bmatrix}, \tag{3.10}$$

where $I_{l+1} \in \mathbb{R}^{l+1 \times l+1}$ and $E_l \in \mathbb{R}^{l+1 \times l}$ has the form

$$E_l = \begin{bmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \cdot & \cdot & \\ & & \cdot & \alpha_l \\ & & & \beta_{l+1} \end{bmatrix}.$$

Hence the matrix $\mathcal{A}_{\text{LS}}$ can be reduced to the form in (3.10) by a Lanczos process and a permutation. In comparison with the standard Lanczos process which is usually thought of as a tridiagonalisation procedure, this process on $\mathcal{A}_{\text{LS}}$ can be considered as a reduction of B to the bidiagonal form E_l, a *bidiagonalisation process.* These observations allow the Lanczos process for $\mathcal{A}_{\text{LS}}$ to be expressed in the following form, to be referred to as the Golub-Kahan lower-bidiagonalisation procedure (GKLB). (The original derivation was not based on a Lanczos process, see [28], although the equivalence was noted).

The Golub-Kahan lower-bidiagonalisation procedure (GKLB)

```
Set $\beta_1 u_1 = b$,   $\alpha_1 v_1 = B^T u_1$,
For $j = 1, 2, \ldots$
    $\beta_{j+1} u_{j+1} = B v_j - \alpha_j u_j$,
    $\alpha_{j+1} v_{j+1} = B^T u_{j+1} - \beta_{j+1} v_j$,
```

where $\alpha_j, \beta_j > 0$ are chosen so that $||u_j|| = ||v_j|| = 1$ respectively.

The original motivation for reduction to bidiagonal form was that the singular values of E_l can be used to approximate the singular values of B, in much the same way that the Lanczos process can be used to approximate eigenvalues of symmetric matrices. See for example [59, 68].

Notice that one step of GKLB is equivalent to performing two Lanczos steps on $\mathcal{A}_{\text{LS}}$ (since two Lanczos vectors are found at each step of GKLB) but requires only two multiplications involving B as opposed to four. Defining $U_l = [u_1, \dots, u_l]$, $V_l = [v_1, \dots, v_l]$ allows the recurrence relations of GKLB to be expressed as

$$\begin{aligned} U_{l+1}(\beta_1 e_1) &= b, \\ BV_l &= U_{l+1}E_l, \\ B^T U_{l+1} &= V_l E_l^T + \alpha_{l+1} v_{l+1} e_{l+1}^T. \end{aligned}$$

The matrix E_l represents the restriction of B to the space spanned by the columns of V_l with respect to the basis formed by the columns of U_{l+1}.

The algorithm GKLB owes the special lower-bidiagonal form of the matrix E_l to the starting vector $\begin{bmatrix} b^T & 0^T \end{bmatrix}^T$. In fact if the starting vector $\begin{bmatrix} 0^T & B^T b \end{bmatrix}^T$ was instead used to start the Lanczos process for $\mathcal{A}_{\text{LS}}$ a similar algorithm to GKLB would result which reduces B to *upper-bidiagonal* form. This algorithm will be referred to as Golub-Kahan upper-bidiagonalisation (GKUB), the detail of its derivation is omitted since it is essentially the same as that above. The recurrences of GKUB are as follows.

The Golub-Kahan upper-bidiagonalisation procedure (GKUB)

Set $\theta_1 v_1 = B^T b, \quad \rho_1 p_1 = B^T v_1,$

For $j = 1, 2, \dots$

$$\begin{aligned} \theta_{j+1} v_{j+1} &= B^T p_j - \rho j v_j, \\ \rho_{j+1} p_{j+1} &= B v_{j+1} - \theta_{j+1} p_j, \end{aligned}$$

with $\theta_j, \rho_j \geq 0$ chosen so that $||v_j|| = ||p_j|| = 1$. If $P_l = [p_1, \dots, p_l]$ and

$$R_l = \begin{bmatrix} \rho_1 & \theta_2 & & & \\ & \rho_2 & \theta_3 & & \\ & & \cdot & \cdot & \\ & & & \cdot & \theta_l \\ & & & & \rho_l \end{bmatrix}.$$

then

$$\begin{aligned} V_l(\theta_1 e_1) &= B^T b, \\ BV_l &= P_l R_l, \\ B^T P_l &= V_l R_l^T + \theta_{l+1} v_{l+1} e_l^T, \end{aligned}$$

hold and $P_l^T P_l = I_l$.

There is an important connection between the matrices E_l and R_l. Using the orthogonality properties of GKLB and GKUB it is simple to deduce that

$$E_l^T E_l = R_l^T R_l.$$

Now suppose that $Q_l E_l = \begin{bmatrix} \hat{R}_l \\ 0^T \end{bmatrix}$ is the QR decomposition of E_l where $\hat{R}_l$ has positive diagonal entries (so that $\hat{R}_l$ is unique [73]). Then $\hat{R}_l^T \hat{R}_l = R_l^T R_l$. Further since the Cholesky decomposition of a positive definite matrix is unique upon insisting that the diagonal entries of the Cholesky factor are positive [73], it follows that $\hat{R}_l = R_l$. Hence the the matrix R_l of GKUB applied to $\mathcal{A}_{\text{LS}}$ is the upper-bidiagonal matrix obtained in the QR decomposition of E_l. It will be shown in §3.3.3 that the entries of R_l can be obtained trivially from the entries of E_l without requiring GKUB to be carried out.

3.3.3 The LSQR algorithm

The solution of the system

$$\begin{bmatrix} I & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix} \tag{3.11}$$

is now considered. Since the Lanczos vectors satisfy $\hat{V}_k^T M \hat{V}_k = T_k$, the GKLB vectors have the property that

$$\begin{bmatrix} u_1^T & 0^T \\ 0^T & v_1^T \\ u_2^T & 0^T \\ 0^T & v_2^T \\ \cdot & \cdot \\ \cdot & \cdot \\ u_{l+1}^T & 0^T \end{bmatrix} \begin{bmatrix} I & B \\ B^T & \end{bmatrix} \begin{bmatrix} u_1 & 0 & \dots & u_{l+1} \\ 0 & v_1 & \dots & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & \alpha_1 & & & & & \\ \alpha_1 & 0 & \beta_2 & & & & \\ & \beta_2 & 1 & \alpha_2 & & & \\ & & \cdot & \cdot & \cdot & & \\ & & & \cdot & \cdot & \cdot & \\ & & & & \alpha_l & 0 & \beta_{l+1} \\ & & & & & \beta_{l+1} & 1 \end{bmatrix}$$

and by permuting rows and columns the above can be rewritten in the form

$$\begin{bmatrix} U_{l+1}^T & \\ & V_l^T \end{bmatrix} \begin{bmatrix} I & B \\ B^T & \end{bmatrix} \begin{bmatrix} U_{l+1} & \\ & V_l \end{bmatrix} = \begin{bmatrix} I & E_l \\ E_l^T & \end{bmatrix},$$

so that the Lanczos subproblem (3.7) with $k = 2l + 1$ corresponds to

$$\begin{bmatrix} I & E_l \\ E_l^T & 0 \end{bmatrix} \begin{bmatrix} s_{l+1} \\ t_l \end{bmatrix} = \beta_1 \begin{bmatrix} e_1 \\ 0 \end{bmatrix}, \tag{3.12}$$

where $\begin{bmatrix} s_{l+1}^T & t_l^T \end{bmatrix}^T$ is the permutation of $\hat{p}_{2l+1}$. Hence t_l is the solution of the least-squares problem

$$\text{minimise}_{\, t \in \mathbb{R}^l} \, ||E_l t - \beta_1 e_1||, \tag{3.13}$$

and $y_l = V_l t_l$, $x_l = U_{l+1}s_{l+1}$ are approximations to y and x from the spans of V_l and U_{l+1} respectively. Solving the minimisation (3.13) is equivalent to solving (3.12) and hence equivalent to the subproblem (3.7) associated with the CG algorithm. The resulting algorithm will be an implementation of CG for the system (3.11), in the sense that the standard CG algorithm cannot be applied directly to this non-positive-definite system as T_k will not necessarily be a positive definite matrix for all k, and hence will not necessarily permit a Cholesky decomposition for all k, but the solutions of the subproblem (3.7) of CG and subproblem (3.13) of LSQR are equal.

The connection between the matrix R_l of GKUB and the QR decomposition of E_l provided Paige and Saunders with an attractive method for the solution of (3.13). First define $f_l = (\varphi_1, \dots, \varphi_l)^T$ and $\overline{\varphi}_{l+1}$ by

$$Q_l(\beta_1 e_1) = \begin{bmatrix} f_l \\ \overline{\varphi}_{l+1} \end{bmatrix}, \tag{3.14}$$

where Q_l is the orthonormal matrix in the QR decomposition of E_l. Then

$$\begin{aligned} ||E_l t - \beta_1 e_1|| &= ||Q_l(E_l t - \beta_1 e_1)||, \\ &= \left|\left| \begin{bmatrix} R_l \\ 0^T \end{bmatrix} t - \begin{bmatrix} f_l \\ \overline{\varphi}_{l+1} \end{bmatrix} \right|\right|, \end{aligned}$$

and so upon noticing $f_l \in span(R_l)$, t_l is defined by

$$R_l t_l = f_l,$$

and it is trivial to show that

$$s_{l+1} = Q_l^T \begin{bmatrix} 0 \\ \overline{\varphi}_{l+1} \end{bmatrix}.$$

Since t_l changes elementwise it would appear that $y_l = V_l t_l$ needs to be recalculated at every step. This is not desirable, it is more convenient to rewrite the above method in such a way that solution estimates can be updated. To this end notice that

$$[R_l \;\; f_l] = \begin{bmatrix} R_{l-1} & \vdots & f_{l-1} \\ \cdots & \cdot & \cdot \end{bmatrix}$$

where the dots indicate the inclusion of a new row and column, so that

$$y_l = V_l t_l = (V_l R_l^{-1}) f_l =: D_l f_l,$$

where $D_l = [d_1, \dots, d_l]$. Defining $d_0 := 0$, the rule

$$d_l = \frac{1}{\rho_l}(v_l - \theta_l d_{l-1}),$$

is obtained for developing D_l, whence if $y_0 := 0$,

$$y_l = y_{l-1} + \varphi_l d_l,$$

is the update rule for solution approximations.

The final observation before the LSQR algorithm of Paige and Saunders can be presented is the following. The matrix Q_l in the QR decomposition of E_l can be expressed as a product of plane rotations, $Q_l = Q_{l,l+1} \cdots Q_{2,3} Q_{1,2}$, where each $Q_{j,j+1}$ operates on the transformed E_l to destroy the sub-diagonal β_{j+1} term. Hence c_l and s_l, the non-trivial entries in $Q_{l,l+1}$ must satisfy

$$\begin{bmatrix} c_l & s_l \\ s_l & -c_l \end{bmatrix} \begin{bmatrix} \overline{\rho}_l & \\ \beta_{l+1} & \alpha_{l+1} \end{bmatrix} = \begin{bmatrix} \rho_l & \theta_{l+1} \\ & \overline{\rho}_{l+1} \end{bmatrix}, \tag{3.15}$$

where $\overline{\rho}_l = (R_l)_{ll}$. Notice also that (3.14) implies that

$$\begin{bmatrix} c_l & s_l \\ s_l & -c_l \end{bmatrix} \begin{bmatrix} \overline{\varphi}_l \\ 0 \end{bmatrix} = \begin{bmatrix} \varphi_l \\ \overline{\varphi}_{l+1} \end{bmatrix}. \tag{3.16}$$

Using (3.15) , (3.16) and the fact that the c_l, s_l matrix is unitary it is easy to verify that given $\alpha_{l+1}, \beta_{l+1}, \overline{\rho}_l$ and $\overline{\varphi}_l$, the new coefficients $\rho_l, c_l, s_l, \theta_{l+1}, \overline{\rho}_{l+1}, \varphi_l$ and $\overline{\varphi}_{l+1}$ can be calculated at trivial cost by the rules given in algorithm LSQR below.

The LSQR algorithm

```
Set $y_0 = 0$, $\beta_1 u_1 = b$, $\alpha_1 v_1 = B^T u_1$, $d_1 = v_1$, $\overline{\varphi}_1 = \beta 1$, $\overline{\rho}_1 = \alpha_1$

For j=1,2, ...
    (Bidiagonalisation step)
    $\beta_{j+1} u_{j+1} = B v_j - \alpha_j u_j$
    $\alpha_{j+1} v_{j+1} = B^T u_{j+1} - \beta_{j+1} v_j$
    (Orthogonal transformation step)
    $\rho_j = \left(\overline{\rho}_j^2 + \beta_{j+1}^2\right)^{\frac{1}{2}}$
    $c_j = \overline{\rho}_j / \rho j$
    $s_j = \beta_{j+1} / \rho_j$
    $\theta_{j+1} = s_j \alpha_{j+1}$
    $\overline{\rho}_{j+1} = -c_j \alpha_{j+1}$
    $\varphi_j = c_j \overline{\varphi}_j$
    $\overline{\varphi}_{j+1} = s_j \overline{\varphi}_j$
    (Solution update)
    $y_j = y_{j-1} + (\varphi_j / \rho_j) d_j$
    $d_{j+1} = v_{j+1} - (\theta_{j+1} / \rho_j) d_j$
```

After each solution update a convergence test needs to be carried out and the iteration terminated if some stopping criteria is satisfied. An analysis of suitable stopping criteria can be found in [58].

It should be noted that it is also possible to reduce the system (3.11) to a bidiagonal problem by using GKUB instead of GKLB. The resulting algorithm for the solution of (3.11) is then the LSCG algorithm of Paige [54]. This algorithm corresponds to solving a version of the normal equations associated with (3.11) and is therefore not as computationally attractive as the LSQR algorithm above.

3.4 Solution of generalised least-squares problems

In this section the bidiagonalisation procedure GKLB will be applied to a particular matrix associated with the generalised least-squares problem (3.5) to obtain a bidiagonalisation procedure for B with different orthogonality properties. In the same way that LSQR was constructed from GKLB, an iterative solution method for generalised least-squares problems will then be derived from the new bidiagonalisation procedure.

3.4.1 A bidiagonalisation procedure with a different inner-product

Recall from §3.2 that solving the system (3.1) is equivalent to solving the generalised least-squares problem (3.5),

$$\textit{Find } y \in \mathbb{R}^m \textit{ such that } \min_{p\in\mathbb{R}^m} ||b - Bp||_{A^{-1}} = ||b - By||_{A^{-1}}.$$

Typically to solve such problems it is first assumed that the Cholesky factorisation $A = LL^T$, where L is lower-triangular, is available. Then the problem (3.5) can be recast as the linear least-squares problem,

$$\textit{Find } y \in \mathbb{R}^m \textit{ such that } \min_{p\in\mathbb{R}^m} \left|\left|L^{-1}(b - Bp)\right|\right| = \left|\left|L^{-1}(b - By)\right|\right|, \tag{3.17}$$

which is then solved using any favourite method for linear least-squares. An example of solving (3.5) in this way is given in [55].

From §3.2 it is obvious that (3.17) is equivalent to the linear system

$$\begin{bmatrix} I & L^{-1}B \\ B^TL^{-T} & 0 \end{bmatrix} \begin{bmatrix} z \\ y \end{bmatrix} = \begin{bmatrix} L^{-1}b \\ 0 \end{bmatrix}, \tag{3.18}$$

and clearly LSQR could be the method used to solve (3.17). The solution $\begin{bmatrix} x^T & y^T \end{bmatrix}^T$ of (3.1) is then recovered by evaluating $z = L^{-1}(b - By)$, $x = L^{-T}z$. However there is another way in which an LSQR type method can be applied to solve (3.5) which avoids using the Cholesky factor of A, and which sacrifices some of the Euclidean orthonormality properties of the LSQR vectors for a similar orthonormality condition.

Defining the vectors $w_i := Mu_i$, where the choice of the matrix M will be discussed shortly, the GKLB procedure can be applied to the matrix $L^{-1}B$ as follows

GKLB applied to $L^{-1}B$ with transformed u_j

```
Set $\beta_1 w_1 = ML^{-1}b$,   $\alpha_1 v_1 = B^TL^{-T}M^{-1}w_1$,

For $j = 1, 2, \ldots$
    $\beta_{j+1}w_{j+1} = ML^{-1}Bv_j - \alpha_j w_j$,
    $\alpha_{j+1}v_{j+1} = B^TL^{-T}M^{-1}w_{j+1} - \beta_{j+1}v_j$,
```

where again $\alpha_j \geq 0$ is chosen so that $||v_j|| = 1$ and now $\beta_j \geq 0$ is chosen such that

$||w_j||_{(MM^T)^{-1}} = ||u_j|| = 1$. Since the vectors u_j form an orthonormal set it is true that

$$w_i^T \left(MM^T\right)^{-1} w_j = \delta_{ij}. \tag{3.19}$$

It is clear that judicious choices for M are $M \in \{L, L^{-T}\}$. Choosing $M = L$ will cancel all applications of L^{-1} and all occurrences of $L^{-T}M^{-1}$ become applications of A^{-1} Similarly choosing $M = L^{-T}$ cancels all L^{-T} operations and ML^{-1} becomes A^{-1}. In both cases the Cholesky factor L is completely removed from the picture. The lower-bidiagonalisation procedure resulting from the choice $M = L$ will be referred to as GKLB(A^{-1}) (since the vectors w_j are A^{-1}-orthonormal by (3.19)) and that resulting from the choice $M = L^{-T}$ will be referred to as GKLB(A) (since then the w_j are A-orthonormal). These two algorithms are presented below.

GKLB(A^{-1})

Set $\beta_1 w_1 = b, \quad \alpha_1 v_1 = B^T A^{-1} w_1,$

For $j = 1, 2, \ldots$

$$\begin{aligned} \beta_{j+1} w_{j+1} &= Bv_j - \alpha_j w_j, \\ \alpha_{j+1} v_{j+1} &= B^T A^{-1} w_{j+1} - \beta_{j+1} v_j, \end{aligned}$$

GKLB(A)

Set $\beta_1 w_1 = A^{-1}b, \quad \alpha_1 v_1 = B^T w_1,$

For $j = 1, 2, \ldots$

$$\begin{aligned} \beta_{j+1} w_{j+1} &= A^{-1}Bv_j - \alpha_j w_j, \\ \alpha_{j+1} v_{j+1} &= B^T w_{j+1} - \beta_{j+1} v_j, \end{aligned}$$

where in both algorithms $\alpha_j \geq 0$ is chosen so that $||v_j|| = 1$ and with $\beta_j \geq 0$ chosen so that $||w_j||_{A^{-1}} = 1$ in GKLB(A^{-1}) and $||w_j||_A = 1$ in GKLB(A) . The recurrences of GKLB(A^{-1}) are more nicely expressed as

$$\begin{aligned} W_{k+1}(\beta_1 e_1) &= b, \\ BV_k &= W_{k+1} E_k, \\ B^T A^{-1} W_{k+1} &= V_k E_k^T + \alpha_{k+1} v_{k+1} e_{k+1}^T, \end{aligned}$$

and those of GKLB(A) ,

$$\begin{aligned}
W_{k+1}(\beta_1 e_1) &= A^{-1}b, \\
A^{-1}BV_k &= W_{k+1}E_k, \\
B^T W_{k+1} &= V_k E_k^T + \alpha_{k+1} v_{k+1} e_{k+1}^T.
\end{aligned}$$

The procedure for carrying out the calculation of $\beta_{j+1} w_{j+1}$ in GKLB(A^{-1}) is

$$\begin{aligned}
\hat{w}_{j+1} &= Bv_j - \alpha_j w_j \\
t &= A^{-1} w_{\hat{j}+1} \\
\beta_{j+1} &= (w_{\hat{j}+1}{}^T t)^{\frac{1}{2}} \\
w_{j+1} &= w_{\hat{j}+1}/\beta_{j+1} \\
A^{-1} w_{j+1} &= t/\beta_{j+1}
\end{aligned}$$

so that $A^{-1} w_{j+1}$ is already known before it is required in the calculation of v_{j+1}. The similar calculation for GKLB(A) is

$$\begin{aligned}
t &= Bv_j - \alpha_j A w_j \\
w_{\hat{j}+1} &= A^{-1} t \\
\beta_{j+1} &= (w_{\hat{j}+1}{}^T t)^{\frac{1}{2}} \\
w_{j+1} &= w_{\hat{j}+1}/\beta_{j+1} \\
A w_{j+1} &= t/\beta_{j+1}
\end{aligned}$$

where $Aw_1 = b/\beta_1$. Since both bidiagonalisation procedures are computationally equivalent, only GKLB(A^{-1}) will be considered further, although all the results given are equally true of both methods. The last note on GKLB(A) is the following. At each iteration step, one solve of the form $Af = g$ is required. If $f_k = W_k h_k$ is an approximation to f from $span(W_k)$, then it may be hoped that $AW_k h_k \approx g$ (in some sense). If the matrix W_k is that obtained from GKLB(A) , $h_k = W_k^T g$ and so $f_k = (W_k W_k^T) g$. This observation has not proved useful so far.

3.4.2 Extension of the bidiagonalisation procedure to an iterative solution method

In exactly the same way that LSQR was derived from GKLB, GKLB(A^{-1}) forms the basis for an iterative solution method for generalised least-squares problems. Let $y_k = V_k z_k$ be an approximation to the solution of (3.5). The optimal z_k is then the solution of

$$\min_{z\in\mathbb{R}^k} ||b - BV_k z||_{A^{-1}},$$

and using the recurrences of GKLB(A^{-1}) ,

$$||b - BV_k z||_{A^{-1}} = ||b - W_{k+1}E_k z||_{A^{-1}}.$$

Defining $F(z) = \frac{1}{2}||b - W_{k+1}E_k z||^2_{A^{-1}}$ it can be seen that $\frac{d}{dz}F(z_k) = 0$ implies

$$E_k^T\left(E_k z_k - W_{k+1}^T A^{-1} b\right) = 0,$$

so that z_k is the solution of $\min_{z\in\mathbb{R}^k}\left|\left|E_k z - W_{k+1}^T A^{-1} b\right|\right|$. Further since $W_{k+1}^T A^{-1} b = \beta_1 e_1$, z_k is the least squares solution of

$$\min_{z\in\mathbb{R}^k} ||E_k z - \beta_1 e_1||,$$

which is exactly the same subproblem which is solved by LSQR (c.f. (3.13)) so that only the bidiagonalisation step of LSQR needs to be swapped for GKLB(A^{-1}) in order to apply an LSQR-type algorithm to the generalised least squares problem (3.5). The algorithm described above will be referred to as LSQR(A^{-1}) and is given below for completeness.

The LSQR(A^{-1}) algorithm

```
Set $y_0 = 0$, $\beta_1 w_1 = b$, $\alpha_1 v_1 = B^T A^{-1} w_1$, $d_1 = v_1$, $\overline{\varphi}_1 = \beta_1$, $\overline{\rho}_1 = \alpha_1$

For j=1,2, ...
    (Bidiagonalisation step)
```

$$\begin{aligned}
\beta_{j+1}w_{j+1} &= Bv_j - \alpha_j w_j \quad \text{with } \beta_{j+1} \text{ chosen so that } \|w_{j+1}\|_{A^{-1}} = 1 \\
\alpha_{j+1}v_{j+1} &= B^T A^{-1} w_{j+1} - \beta_{j+1} v_j
\end{aligned}$$

(Orthogonal transformation step)

$$\begin{aligned}
\rho_j &= \left(\overline{\rho}_j^2 + \beta_{j+1}^2\right)^{\frac{1}{2}} \\
c_j &= \overline{\rho}_j / \rho j \\
s_j &= \beta_{j+1} / \rho_j \\
\theta_{j+1} &= s_j \alpha_{j+1} \\
\overline{\rho}_{j+1} &= -c_j \alpha_{j+1} \\
\varphi_j &= c_j \overline{\varphi}_j \\
\overline{\varphi}_{j+1} &= s_j \overline{\varphi}_j
\end{aligned}$$

(Solution update)

$$\begin{aligned}
y_j &= y_{j-1} + (\varphi_j / \rho_j) d_j \\
d_{j+1} &= v_{j+1} - (\theta_{j+1} / \rho_j) d_j
\end{aligned}$$

3.4.3 Preconditioning

When solving systems of the form

$$Mz = b,$$

where $M \in \mathbb{R}^{n \times n}$, it is usual to introduce a nonsingular preconditioning matrix N, or if N is positive definite its Cholesky factor L, and instead solve one of the systems

$$\begin{aligned}
N^{-1} M z_L &= N^{-1} b && \text{(left preconditioning)} \\
M N^{-1} z_R &= b && \text{(right preconditioning)} \\
L^{-1} M L^{-T} z_C &= L^{-1} b && \text{(central preconditioning)}
\end{aligned}$$

where $z_L = z$, $z_R = Nz$ and $z_C = L^T z$; the idea in all cases being that the matrix $N^{-1}M$ should have better condition than M, or that $N^{-1}M \approx I$ in some sense.

It is clear that of these preconditioning strategies only right preconditioning is applicable to (generalised) least-squares problems since any other form of preconditioning affects the norm in which the minimisation is taking place. Hence preconditioned generalised least-squares problems,

$$\min \left\| B N^{-T} p - b \right\|_{A^{-1}}, \tag{3.20}$$

can be considered when solving (3.1), where $p = N^T y$. Analogous to the extension from least-squares to generalised least-squares, an iterative method for (3.20) can be devised from the Golub-Kahan bidiagonalisation procedures by altering the inner product in which the m-vectors, $\{v_i\}$, are orthonormal. GKLB can be applied to the matrix BN^{-T} with $z_i := Sv_i$ as follows.

The Golub-Kahan lower-bidiagonalisation procedure for BN^{-T} with transformed v_j

```
Set $\beta_1 w_1 = b, \quad \alpha_1 z_1 = SN^{-1}B^T A^{-1} w_1,$

For $j = 1, 2, \dots$
    $\beta_{j+1} w_{j+1} = BN^{-T}S^{-1} z_j - \alpha_j w_j,$
        ($\beta_{j+1}$ chosen so that $||w_{j+1}||_{A^{-1}} = 1$)
    $\alpha_{j+1} z_{j+1} = SN^{-1}B^T A^{-1} w_{j+1} - \beta_{j+1} z_j$
        ($\alpha_{j+1}$ chosen so that $||z_{j+1}||_{(SS^T)^{-1}} = 1$.)
```

If $S = N$ is chosen with $H = NN^T$, the resulting algorithm, to be referred to as GKLB(A^{-1}, H^{-1}), is

GKLB(A^{-1}, H^{-1})

```
Set $\beta_1 w_1 = b, \quad \alpha_1 z_1 = B^T A^{-1} w_1,$

For $j = 1, 2, \dots$
    $\beta_{j+1} w_{j+1} = BH^{-1} z_j - \alpha_j w_j,$
        ($\beta_{j+1}$ chosen so that $||w_{j+1}||_{A^{-1}} = 1$)
    $\alpha_{j+1} z_{j+1} = B^T A^{-1} w_{j+1} - \beta_{j+1} z_j$
        ($\alpha_{j+1}$ chosen so that $||z_{j+1}||_{H^{-1}} = 1$).
```

The procedure for carrying out the calculation of $\alpha_{j+1} z_{j+1}$, similar to that given for calculating $\beta_{j+1} w_{j+1}$ in GKLB(A^{-1}) , is

$$\begin{aligned}
\hat{z}_{j+1} &= B^T A^{-1} w_{j+1} - \beta_{j+1} z_j \\
t &= H^{-1} \hat{z}_{j+1} \\
\alpha_{j+1} &= (\hat{z}_{j+1}^T t)^{\frac{1}{2}} \\
z_{j+1} &= \hat{z}_{j+1} / \alpha_{j+1} \\
H^{-1} z_{j+1} &= t / \alpha_{j+1}
\end{aligned}$$

where $H^{-1}z_{j+1}$ is used in the next iteration step in the calculation of w_{j+2}.

A similar algorithm, GKLB(A^{-1}, H), to GKLB(A^{-1}, H^{-1}) can be obtained on taking $S = N^{-T}$ above. Algorithms GKLB(A, H^{-1}) and GKLB(A, H) are also defined. Any of these bidiagonalisation procedures can be inserted into the bidiagonalisation step of LSQR(A^{-1}) to form an algorithm for the preconditioned system (3.20), the one corresponding to GKLB(A^{-1}, H^{-1}) being denoted LSQR(A^{-1}, H^{-1}) e.t.c. .

Remarks

The only other reference to preconditioning LSQR that the author is aware of is that in [57]. Here it is noted that of the three preconditioning strategies, left, central and right, only right preconditioning is suitable when using LSQR but no further discussion is given other than saying that the effect of N^{-1} must be easy to calculate. Here, preconditioners of the form $H = NN^T$ are being proposed for use in the LSQR algorithm, where H is a preconditioner for B^TB, together with an appropriate change of inner product. It is obvious that this approach requires only one operation with the preconditioner per iteration, compared with two operations per iteration for the standard LSQR preconditioning approach. It is suggested that such preconditioners may be also easier to find since the matrix B^TB is symmetric and positive definite and so open to several possible types of preconditioning strategies such as those of domain decomposition, additive Schwartz type. Whereas, to form a preconditioner for the (rectangular) matrix B, the only obvious options appear to be to diagonally scale B or form a Cholesky factorisation of a preconditioner for B^TB and use this as the preconditioner N. Also, the incomplete factorisations of rectangular matrices discussed in [4, Section 7.1] may also be appropriate. Whilst such an approach is probably an effective preconditioner for B, being limited to these preconditioning strategies is not ideal. If an additive Schwartz type preconditioner were to be used to precondition in the manner proposed in [57], not only would the preconditioner need to be calculated, but so would its (complete) Cholesky factorisation. Since additive Schwartz preconditioners are rarely formed in practise, this makes the approach extremely expensive, whereas the approach outlined in this section is no more expensive than preconditioning an equivalent conjugate gradient algorithm. For a more detailed discussion on preconditioning LSQR(A^{-1}) see §4.4.

3.5 Comparing LSQR(A^{-1}) with preconditioned Krylov methods

The main factor influencing whether the LSQR(A^{-1}) method is applicable to a particular system of form (3.1) is that the action of A^{-1} must be easy to perform. In effect this is equivalent to assuming that a specific preconditioning of (3.1) is possible, and hence it would make sense to compare the LSQR(A^{-1}) method with preconditioned Krylov methods, where the preconditioner for the full system contains the matrix A. In [22] preconditioners of the form

$$\begin{bmatrix} \frac{1}{\eta^2}A & 0 \\ 0 & \pm H \end{bmatrix}, \quad \eta \in \mathbb{R}^+, \tag{3.21}$$

are considered, where H is symmetric positive definite and is typically an approximation to the Schur complement $B^TA^{-1}B$. The preconditioner (3.21) can be applied centrally to (3.1) to form the preconditioned system

$$\begin{bmatrix} \eta I & \hat{B} \\ \pm\hat{B}^T & 0 \end{bmatrix} \begin{bmatrix} \hat{x} \\ \hat{y} \end{bmatrix} = \begin{bmatrix} \hat{b} \\ 0 \end{bmatrix}, \tag{3.22}$$

where $\hat{B} = L^{-1}BQ^{-T}$, $\hat{x} = \frac{1}{\eta}L^Tx$, $\hat{y} = Q^Ty$ and $\hat{b} = L^{-1}b$. Here L and Q are the Cholesky factors of A and H respectively. The choice of $\pm H$ in (3.21) can impose different conditions on $\hat{\mathcal{A}}_{\pm}$, the matrix in (3.22). It is easy to see that choosing $+H$ ensures that $\hat{\mathcal{A}}_+$ is symmetric, whilst the identity

$$\begin{bmatrix} u^T & v^T \end{bmatrix} \hat{\mathcal{A}}_- \begin{bmatrix} u \\ v \end{bmatrix} = \eta u^Tu > 0, \;\; u \neq 0,$$

shows that choosing $-H$ gives rise to a positive semi-definite matrix. The dilemma of selecting symmetric or positive semi-definite preconditioners is addressed in [22] and some elegant results are presented, the most relevant being that if MINRES (§3.3.1, [56]) is used to solve the symmetrically preconditioned version of (3.1) with $\hat{\mathcal{A}}_+$ as coefficient matrix and GMRES [69] is used to solve the positive semi-definite problem with $\hat{\mathcal{A}}_-$ as coefficient matrix, then the two resulting solution methods for (3.1) are equivalent in the sense that

$$||r_k^{\text{MINRES}}|| = ||r_k^{\text{GMRES}}||,$$

and further convergence is independent of the choice of η, and

$$\left|\left|r_{2k+1}^{\text{MINRES}}\right|\right| = ||r_{2k}^{\text{MINRES}}|| \;\left(= ||r_{2k}^{\text{GMRES}}|| = \left|\left|r_{2k+1}^{\text{GMRES}}\right|\right|\right),$$

so that both methods make no progress every second step. These observations make MINRES the most attractive of the two methods since it is cheaper to implement than GMRES.

The system (3.22) is equivalent to the preconditioned generalised least-squares problem

$$\min \left|\left|BQ^{-T}\hat{x} - \hat{b}\right|\right|_{A^{-1}},$$

(c.f. (3.20)) and hence LSQR(A^{-1}, H^{-1}) can be applied. The matrix H will certainly affect the convergence of both the least-squares and the MINRES methods of solution, but since only a comparison of these two approaches is required the matrix H plays no part, as will hopefully become clear. Therefore it will be assumed that $H = I$ (and hence $Q = I$). The MINRES approach then corresponds to solving the system

$$\begin{bmatrix} I & L^{-1}B \\ B^TL^{-T} & 0 \end{bmatrix} \begin{bmatrix} L^Tx \\ y \end{bmatrix} = \begin{bmatrix} L^{-1}b \\ 0 \end{bmatrix}. \tag{3.23}$$

Recall now (§3.3.1) that the MINRES algorithm is a development of the Lanczos process, and consider the $2l + 1^{\text{th}}$ step of this Lanczos process. With starting vector $\beta_1\hat{v}_1 = \begin{bmatrix} (L^{-1}b)^T & 0^T \end{bmatrix}^T$ this Lanczos process will have generated $l + 1$ orthonormal vectors of the form $\begin{bmatrix} u_i^T & 0^T \end{bmatrix}^T$ and l orthonormal vectors of the form $\begin{bmatrix} 0^T & v_i^T \end{bmatrix}^T$. At the $2l + 1^{\text{th}}$ step of MINRES on (3.23), the approximation y_{2l+1}^{MINRES} to y is from the span of $V_l = [v_1 \dots v_l]$, whereas an approximation to y from the span of V_l is available from LSQR(A^{-1}) at only the lth step. i.e. the Lanczos minimisation-subproblems used to generate $y_l^{\text{LSQR}(A^{-1})}$ and y_{2l+1}^{MINRES} are performed on subspaces of equal dimension, so that LSQR(A^{-1}) takes half the number of iterations of MINRES on (3.23) to produce an approximation to y from a subspace of the same size. Obviously the two approximations $y_l^{\text{LSQR}(A^{-1})}$ and y_{2l+1}^{MINRES} will not necessarily be the same since each method uses a different minimisation subproblem. It could be argued that MINRES also provides an approximation to x at each step whereas LSQR(A^{-1}) does not. However once $y_l^{\text{LSQR}(A^{-1})}$ is deemed sufficiently accurate, $x_l^{\text{LSQR}(A^{-1})}$ can be recovered at a cost of less than one LSQR(A^{-1}) step using the rule $x_l^{\text{LSQR}(A^{-1})} = A^{-1}(b - By_l^{\text{LSQR}(A^{-1})})$, or by forming an approximation to x from *span*(W_l). Hence the LSQR(A^{-1}) method provides a way of stepping over the redundant MINRES iterations found in [22], by using the GKLB process to step over the redundancies in the Lanczos process on the matrix in (3.23).

It can be seen ([20]) that MINRES on the preconditioned system (3.22) requires

• Matrix-vector products	2 ($n \times m$)
• A^{-1} operations	1
• Vector inner products	2 (n) , 2 (m)
• Additional flops	$12n + 12m$
• Stored vectors	7 (n) , 7 (m)

per step, where ($\cdot$) indicates the dimension of the operation, whereas the operation count for one step of LSQR(A^{-1}) is

• Matrix-vector products	2 ($n \times m$)
• A^{-1} operations	1
• Vector inner products	1 (n) , 1 (m)
• Additional flops	$4n + 8m$
• Stored vectors	1 (n) , 4 (m)

so that the LSQR(A^{-1}) method seems to be more attractive than either of the preconditioned MINRES or GMRES approaches.

In [22], the authors also note that applying MINRES to the matrix

$$\hat{\mathcal{A}}_+(\hat{\mathcal{A}}_+ - \eta I) = \begin{bmatrix} \hat{B}\hat{B}^T & 0 \\ 0 & \hat{B}^T\hat{B} \end{bmatrix}$$

generates the same iterates as MINRES applied the the matrix $\hat{\mathcal{A}}_+$. Notice that if an initial vector of the form $\begin{bmatrix} 0^T & r_0^T \end{bmatrix}^T$ were used to generate a Krylov subspace together with the matrix above then each Krylov vector would retain the zero component in its upper part. Therefore applying MINRES to $\hat{A}_+(\hat{A}_+ - \eta I)$ is equivalent to applying MINRES to $\hat{B}^T\hat{B} = B^TA^{-1}B$ (when $H = I$, the case $H \neq I$ being similar), that is applying MINRES to the Schur complement matrix. Obviously when B is of full column rank the cheaper conjugate gradient method may be preferred. This observation led to the conclusion that the most efficient way to solve (3.1) (if the preconditioner (3.21) is viable) is to apply MINRES to the Schur complement system. However, notice that MINRES applied to the Schur complement system satisfies (at the kth step)

$$\min_{y \in \mathcal{K}_1^k} ||B^TA^{-1}By - B^TA^{-1}b|| = ||B^TA^{-1}By_k^{\text{MINRES}} - B^TA^{-1}b||$$

whereas at the kth step of LSQR(A^{-1}),

$$\min_{y \in \mathcal{K}_2^k} ||By - b||_{A^{-1}} = \left|\left|By_k^{\text{LSQR}(A^{-1})} - b\right|\right|_{A^{-1}},$$

where $\mathcal{K}_1^k$ and $\mathcal{K}_2^k$ are the Krylov subspaces associated with MINRES and LSQR(A^{-1}) respectively. It can be verified that with the same choice of y_0^{MINRES} and $y_0^{\text{LSQR}(A^{-1})}$, the two Krylov subspaces are the same indeed, $\mathcal{K}_1^k = \mathcal{K}_2^k = \mathcal{K}^k(B^TA^{-1}B, B^TA^{-1}b)$. Hence both LSQR($A^{-1}$) and MINRES on the Schur complement system return an approximation to y from the same subspace. Notice further that the Schur complement system

$$B^TA^{-1}By = B^TA^{-1}b$$

is simply the normal equations corresponding to the generalised least-squares problem (3.5). This observation leads to the conclusion that in inexact arithmetic, MINRES on the Schur complement system will be numerically inferior to LSQR(A^{-1}) on ill-conditioned problems since the Schur complement matrix can have worse condition than the coefficient matrix $L^{-1}B$ of the generalised least-squares problem, the condition of the Schur complement being cond$(L^{-1}B)^2$.

3.6 Numerical experiments and implementation

Following from the final comments of §3.5, little difference between applying MINRES to the Schur complement and LSQR(A^{-1}) to the generalised least-squares problems is to be expected when the Schur complement matrix is well conditioned. Experiments have shown this to be the case, with LSQR(A^{-1}) proving slightly better at minimising $||y - y_k||$ and $||By_k - b||_{A^{-1}}$. All experiments were performed on machines with precision $\epsilon \approx 10^{-16}$.

Example 1 : An incompatible problem

The advantage of LSQR(A^{-1}) over the MINRES approach is more obvious when considering more ill-conditioned problems. Figure 3-1 illustrates the performance of LSQR(A^{-1}) and MINRES at minimising the error, $||y - y_k||$, and residual, $||r_k||_{A^{-1}} = ||By_k - b||_{A^{-1}}$, for a finite element discretisation of flow in a porous region containing non-porous 'fingers', as governed by Darcy's law. The resulting generalised least-squares problem is obviously *incompatible* (the solution y satisfies $By \neq b$) since $x = 0$ would correspond to a flow with zero velocity. Similar performance is displayed initially by each method. However after approximately 10 iterations LSQR(A^{-1}) can be seen to reduce the error faster than MINRES. Both methods perform similarly in reducing the residual, with LSQR(A^{-1}) being only slightly faster. The fact that the generalised least-squares problem is so incompatible (in this case $||By - b||_{A^{-1}} \approx 6 \times 10^{-1}$) and that $||r_0||_{A^{-1}}$ is small makes a more precise comparison of each method's residual reducing properties difficult.

For the purpose of these experiments the A^{-1}-norm of the LSQR(A^{-1}) residual has been calculated exactly using an A^{-1} solve at each step of the iteration. In practise, however, the estimate

$$||r_k||_{A^{-1}} = \overline{\varphi}_{k+1} = \beta_1 s_k s_{k-1} \cdots s_1$$

should be used since it is virtually free and has been observed to agree closely with the calculated value of $||r_k||_{A^{-1}}$. This bound is analogous to that given in [58] for $||r_k||$ in

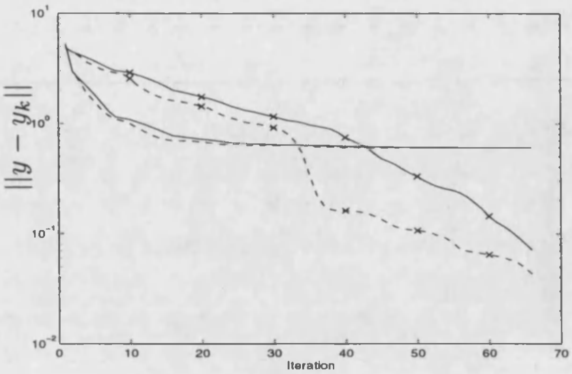


Figure 3-1: Example 1 : Graphs comparing the performance of MINRES on the Schur complement system and LSQR(A^{-1}) on the generalised least-squares problem for a groundwater flow system with $\kappa(B^TA^{-1}B) = 7.5 \times 10^2$. The (two) crossed lines show the errors, $||y - y_k||$, for both methods. The (uncrossed) lines representing the residuals, $||r_k||_{A^{-1}}$, are also plotted. Solid lines (—) show the performance of MINRES against iteration and dashed lines (– –) represent LSQR(A^{-1}) . LSQR(A^{-1}) is faster to reduce the error and slightly faster at reducing the residual norms.

LSQR and its derivation is the same. From the comments following (3.13),

$$\begin{aligned}
r_l &= b - By_l \\
&= U_{l+1}(\beta_1 e_1) - BV_l t_l \\
&= U_{l+1}(\beta_1 e_1) - U_{l+1}E_l t_l \\
&= U_{l+1}s_{l+1} \quad \text{where } s_{l+1} \text{ is the residual of the minimisation subproblem} \\
&= U_{l+1}Q_l^T \begin{bmatrix} 0 \\ \overline{\varphi}_{l+1} \end{bmatrix} \\
&= \overline{\varphi}_{l+1} U_{l+1} Q_l^T e_{l+1}.
\end{aligned}$$

The estimate now follows upon exploiting the orthogonality properties of U_{l+1} and Q_l.

Example 2 : A compatible problem

The system described in Example 1 was used to construct a more compatible problem by altering b so that $||By - b|| < 6 \times 10^{-3}$. Since the system is more compatible, comparison of the abilities of LSQR(A^{-1}) and MINRES at reducing the residual is possible. Figure 3-2 shows the performance of both methods on this system. Again the advantage of LSQR(A^{-1}) is clear after only a small number of iterations.

A similar estimate to that above for $||r_k||_{A^{-1}}$ is available for the residual of the

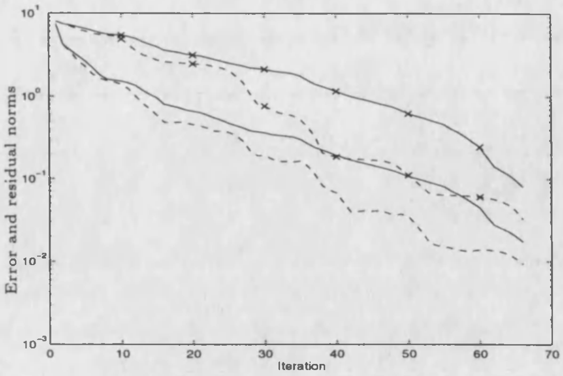


Figure 3-2: Example 2 : Graphs comparing the performance of MINRES on the Schur complement system and LSQR(A^{-1}) for the 'almost compatible' generalised least-squares problem ($||By - b|| < 6 \times 10^{-3}$), $\kappa(B^TA^{-1}B) = 7.5 \times 10^2$ in example 2. The (two) crossed lines show the errors, $||y - y_k||$, for both methods. The (uncrossed) lines representing the residuals, $||r_k||_{A^{-1}}$, are also plotted. Solid lines (—) show the performance of MINRES against iteration and dashed lines (−−) represent LSQR(A^{-1}) . LSQR(A^{-1}) is noticeably faster at reducing the error and residual norms.

Schur complement equations (the normal equations for the generalised-least squares problem),

$$\left|\left|B^TA^{-1}r_k\right|\right| = \overline{\varphi}_{k+1}\alpha_{k+1}|c_k|.$$

This estimate is a useful stopping criterion since $\left|\left|B^TA^{-1}r_k\right|\right|$ should be small for good approximations, y_k, to the solution. It is to be expected that MINRES will be the superior of the two methods at minimising $\left|\left|B^TA^{-1}r_k\right|\right|$ since this is precisely the quantity minimised at each step when MINRES is applied to the Schur complement system. This can be observed in Figure 3-3, which shows $\left|\left|B^TA^{-1}r_k\right|\right|$ for the problem considered in Figure 3-2. As expected, $\left|\left|B^TA^{-1}r_k^{\text{MINRES}}\right|\right|$ is monotonically decreasing, whereas $\left|\left|B^TA^{-1}r_k^{\text{LSQR}(A^{-1})}\right|\right|$ exhibits oscillations. Hence care should be taken when using the residual of the Schur complement equations as a stopping criterion (as MINRES was seen to be inferior when minimising the error in Figure 3-2).

Example 3 : A moderately ill conditioned problem

As a final example it is shown that LSQR(A^{-1}) can succeed in solving a moderately ill-conditioned problem upon which MINRES fails. Figure 3-4 depicts what happens when both methods are applied to a system with $B \in \mathbb{R}^{32\times16}$ and with $\kappa(B^TA^{-1}B) = 5.3 \times 10^4$. Although LSQR($A^{-1}$) requires a greater number of iterates than expected,

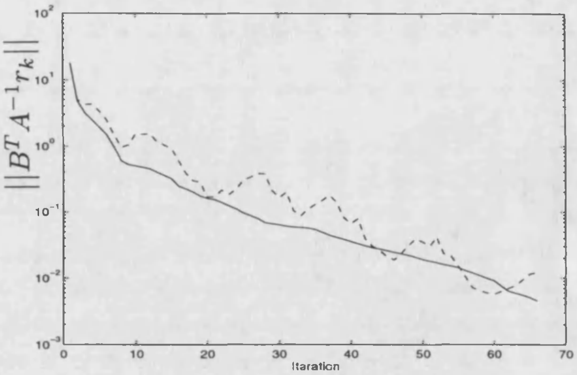


Figure 3-3: Example 2 : Graphs showing $||B^TA^{-1}r_k||$ for the generalised least-squares problem in Figure 3-2. Solid lines (—) show the performance of MINRES against iteration and dashed lines (−−) represent LSQR(A^{-1}) . MINRES performs better at minimising $||B^TA^{-1}r_k||$ as expected from its minimisation property.

it successfully minimises $||y - y_k||$, whereas the MINRES approach fails to noticeably reduce the error. This trend was observed to continue up to 200 iterations, at which point the computation was terminated.

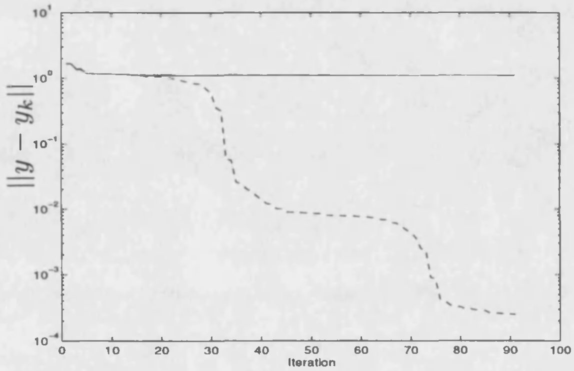


Figure 3-4: Example 3 : Graphs showing $||y - y_k||$ for the moderately ill-conditioned problem, $\kappa(B^TA^{-1}B) = 5.3 \times 10^4$ in example 3. Solid lines (—) show the performance of MINRES against iteration and dashed lines (−−) represent LSQR(A^{-1}) . MINRES fails to reduce the norm of the error, LSQR(A^{-1}) succeeds eventually.

3.7 The case $A = D$

In the case that $A = D$, a diagonal positive definite matrix, it is obvious that (3.1) is equivalent to the weighted least-squares problem,

$$\textit{Find } y \in \mathbb{R}^m \textit{ such that } \quad \min_{p \in \mathbb{R}^m} ||Bp - b||_{D^{-1}} = ||By - b||_{D^{-1}}, \tag{3.24}$$

which could be solved by applying the LSQR algorithm to the least-squares problem

$$\min_{p\in\mathbb{R}^m} \left|\left|D^{-\frac{1}{2}}Bp - D^{-\frac{1}{2}}b\right|\right|.$$

Notice however that this method of solution would first require calculation of the entries of $D^{-\frac{1}{2}}$, and requires two operations involving $D^{-\frac{1}{2}}$ per iteration, compared with one operation involving D^{-1} per iteration if (3.24) were to be solved using LSQR(D^{-1}). As was noted in Chapter 1, systems with badly scaled weight matrices D occur frequently. An interesting result due to Stewart [74] is that regardless of how badly scaled the weight matrix becomes, the solution of (3.24) can be bounded independently of D. The solution can be bounded by

$$||y|| \leq \left|\left|(B^TD^{-1}B)^{-1}B^TD^{-1}\right|\right| \, ||b||.$$

Stewart's theorem provides a bound on $\left|\left|(B^TD^{-1}B)^{-1}B^TD^{-1}\right|\right|$ which is independent of D.

Theorem 3.7.1 (Stewart [74]) *Let B be of full column rank and $D \in \mathcal{D}_+$, the set of diagonal matrices with positive diagonal entries. Then $\exists$ constants χ and $\overline{\chi}$ which are independent of D such that*

$$\begin{aligned} \left|\left|(B^TD^{-1}B)^{-1}B^TD^{-1}\right|\right| &\leq \chi \\ \textit{and } \left|\left|B(B^TD^{-1}B)^{-1}B^TD^{-1}\right|\right| &\leq \overline{\chi}. \end{aligned}$$

This theorem does not extend to the generalised least squares case when A is non-diagonal positive definite.

To bound $||y||$ independently of D it is sufficient to bound $\overline{\chi}$ independently of D since $||y|| \leq \left|\left|(B^TB)^{-1}B^T\right|\right| \, ||b|| \, \overline{\chi}$. A second theorem due to Stewart [74] provides a lower bound on $\overline{\chi}$ which O'Leary [53] later proved to be tight.

Theorem 3.7.2 (O'Leary [53]) *Let the columns of U form an orthonormal basis for span(B) and let U_I denote any submatrix formed from a set of rows of U. Then*

$$\overline{\chi} = (\min \sigma^+_{min}(U_I))^{-1},$$

where $\sigma^+_{min}(U_I)$ denotes the smallest non-zero singular value of U_I.

It would seem consistent to expect that the solutions of the weighted least-squares subproblems of LSQR(D^{-1}) can also be bounded independently of D since they correspond to solving (3.24) over a subspace of $\mathbb{R}^m$. The extension of Theorem 3.7.1 to this result would be obvious were it not for the fact that the subspace of $\mathbb{R}^m$ in question is itself dependent on D, namely it is the Krylov subspace $\mathcal{K}^k(B^TD^{-1}B, B^TD^{-1}b)$. The next theorem shows that despite of this, Theorem 3.7.1 does hold for the LSQR(D^{-1}) subproblems.

Theorem 3.7.3 *The solutions of the weighted least-squares subproblems of LSQR(D^{-1}) can be bounded independently of D in exact arithmetic.*

Proof At the kth step of LSQR(D^{-1}) the subproblem

$$\min_{z\in\mathbb{R}^k} ||BV_kz - b||_{D^{-1}},$$

is solved (see §3.4.2). Hence the solution z_k satisfies

$$||z_k|| \leq \left|\left|\left((BV_k)^TD^{-1}(BV_k)\right)^{-1}(BV_k)^TD^{-1}\right|\right|\,||b||. \tag{3.25}$$

Writing $S_1 = \left((BV_k)^TD^{-1}(BV_k)\right)^{-1}(BV_k)^TD^{-1}$, $S_2 = (BV_k)S_1$ and $S_3 = \left((BV_k)^T(BV_k)\right)^{-1}(BV_k)^T$ it is clear that

$$||S_1|| \leq ||S_2||\,||S_3|| \tag{3.26}$$

and hence bounding both $||S_2||$ and $||S_3||$ independently of D provides a bound on $||z_k||$ which is independent of D.

Consider S_3. Clearly $||S_3|| \leq \left|\left|\left((BV_k)^T(BV_k)\right)^{-1}\right|\right|\,||(BV_k)^T||$. Now since $||(BV_k)^T|| = ||(BV_k)||$ (see [73]) and

$$\begin{aligned} ||(BV_k)|| &= \max_{z\neq 0}\frac{||(BV_k)z||}{||z||}, \\ &= \max_{z\neq 0}\frac{||(BV_k)z||}{||V_kz||}, \quad \text{since } V_k^TV_k = I, \\ &\leq \max_{y\neq 0}\frac{||By||}{||y||}, \end{aligned}$$

$||(BV_k)^T||$ can be bounded by

$$||(BV_k)^T|| \leq \sigma_{\max}(B). \tag{3.27}$$

Now let $u = (BV_k)^T(BV_k)z \neq 0$, then

$$\begin{aligned} z^T u &= \frac{||(BV_k)z||^2}{||V_k z||^2}||V_k z||^2 \\ &\geq \min_{z \neq 0} \frac{||(BV_k)z||^2}{||V_k z||^2}||z||^2 \\ &\geq (\sigma^+_{\min}(B))^2\,||z||^2 \end{aligned}$$

and since $|z^T u| \leq ||z||\,||u||$, $||z|| \leq (\sigma^+_{\min}(B))^{-2}\,||u||$ and hence

$$||((BV_k)^T(BV_k))^{-1}|| \leq (\sigma^+_{\min}(B))^{-2}. \tag{3.28}$$

Combining (3.27) and (3.28) gives a bound on $||S_3||$ which is independent of D,

$$||S_3|| \leq (\sigma^+_{\min}(B))^{-2}\sigma_{\max}(B). \tag{3.29}$$

It remains to bound $||S_2||$, this can be achieved by an analogous proof of that given by Stewart ([74, Theorem 1]). First define

$$\begin{aligned} \mathcal{S} &= \{z \in span(BV_k) \mid ||z|| = 1\} \\ \text{and } \mathcal{T} &= \{z \mid \exists D \in \mathcal{D}_+ \text{ such that } (BV_k)^T Dz = 0\}. \end{aligned}$$

Then $\mathcal{S} \cap \overline{\mathcal{T}} = \emptyset$, for if this is not the case $\exists\{z_j\} \subset \mathcal{T}$ and $\{D_j\} \subset \mathcal{D}_+$ such that $z_j \to z \in \mathcal{S}$ and $(BV_k)^T D_j z_j = 0$. Since $z \in \mathcal{S}$, $\sum_{i=1}^n z_i d_{ii}^{(j)} z_i^{(j)} = 0$, but as $z_j \to z$ it must be the case that for sufficiently large j, $z_i z_i^{(j)} > 0$ (whenever $z_i > 0$) which is a contradiction. Hence $\mathcal{S} \cap \overline{\mathcal{T}} = \emptyset$ and since $\mathcal{S}$ is closed and bounded it follows that

$$\rho_{(BV_k)} := \inf_{z_1 \in \mathcal{S}, z_2 \in \mathcal{T}} ||z_1 - z_2|| > 0.$$

Using another result of Stewart ([74, Theorem 2]), if the columns of $U^{(BV_k)}$ form an orthonormal basis for $span(BV_k)$ and if $U_I^{(BV_k)}$ is any submatrix formed from a set of rows of $U^{(BV_k)}$ then

$$\rho_{(BV_k)} \leq \min \sigma^+_{\text{min}}(U_I^{(BV_k)})$$

and further by O'Leary's result [53],

$$\rho_{(BV_k)} = \min \sigma^+_{\text{min}}(U_I^{(BV_k)}).$$

Now extend $U^{(BV_k)}$ to form an orthonormal basis, U, for $span(B)$. Then $U = [U^{(BV_k)}, U^E]$ for some orthonormal matrix U^E and any submatrix formed from a set of rows of U is of the form $U_I = [U_I^{(BV_k)}, U_I^E]$. Then

$$\begin{aligned}
\sigma^+_{\text{min}}(U_I) &= \min^+_{\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \neq 0} \frac{\left\| [U_I^{(BV_k)}, U_I^E] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \right\|}{\left\| \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \right\|} \\
&\leq \min^+_{\begin{bmatrix} x_1 \\ 0 \end{bmatrix} \neq 0} \frac{\left\| [U_I^{(BV_k)}, U_I^E] \begin{bmatrix} x_1 \\ 0 \end{bmatrix} \right\|}{\left\| \begin{bmatrix} x_1 \\ 0 \end{bmatrix} \right\|} \\
&= \min^+_{x_1 \neq 0} \frac{\left\| U_I^{(BV_k)} x_1 \right\|}{||x_1||} \\
&= \sigma^+_{\text{min}}(U_I^{(BV_k)})
\end{aligned}$$

where $\min^+$ denotes smallest positive value. Hence defining $\rho_B = \min \sigma^+_{\text{min}}(U_I)$,

$$\rho_B \leq \rho_{(BV_k)}. \tag{3.30}$$

Using Stewart's proof [74, Theorem 1] and O'Leary's result,

$$\rho_B = \inf_{z_1 \in \mathcal{S}', z_2 \in \mathcal{T}'} ||z_1 - z_2|| > 0,$$

where $\mathcal{S}' = \{z \in span(B) \mid ||z|| = 1\}$, $\mathcal{T}' = \{z \mid \exists D \in \mathcal{D}_+ \text{ such that } B^T D z = 0\}$, and ρ_B is independent of D.

Lastly suppose that $x = (BV_k)((BV_k)^TD^{-1}(BV_k))^{-1}(BV_k)^TD^{-1}z$. Then $z - x$ satisfies $(BV_k)^TD^{-1}(z - x) = 0$ so that $z - x \in \mathcal{T}$. Writing $v = z - x$ and $t = 1/\,||x||$ implies $tv + tx = tz$ and since $tv \in \mathcal{T}$ and $tx \in \mathcal{S}$,

$$\rho_{(BV_k)} \leq ||tv - (-tx)|| = ||tz||,$$

and so

$$||x|| \leq \rho^{-1}_{(BV_k)}\,||z||.$$

Hence

$$||S_2|| = \left|\left|(BV_k)((BV_k)^TD^{-1}(BV_k))^{-1}(BV_k)^TD^{-1}\right|\right| \leq \rho^{-1}_{(BV_k)} \leq \rho_B^{-1} \tag{3.31}$$

by (3.30). Combining (3.25), (3.26), (3.29) and (3.31) results in the bound

$$||z_k|| \leq \rho_B^{-1}(\sigma^+_{\min}(B))^{-2}\sigma_{\max}(B)\,||b||$$

on the solution of the kth LSQR(D^{-1}) subproblem which is independent of D as required.

□

This result implies that every solution approximation y_i given by LSQR(D^{-1}) should be bounded independently of D. Such a result may be useful when the system (3.1) is not needed to be solved exactly, perhaps when only a small reduction in the residual is required. Then Theorem 3.7.3 implies that even a very poor approximate solution is still bounded independently of D, and hence this particular quality of the solution is not lost by a poor approximation.

3.8 Summary

It has been demonstrated that considering augmented systems of the form (3.1) as generalised least-squares problems of the form (3.2) can be advantageous when constructing iterative solution methods. In §3.4 it was seen that the standard LSQR algorithm for least-squares problem could be extended to algorithm LSQR(A^{-1}) for generalised least-squares problems without assuming availability of the Cholesky decomposition of the matrix A, which would be required if the LSQR algorithm were to be applied naïvely to the problem. This fact also results in a saving when the weight matrix is diagonal, the LSQR(A^{-1}) method requiring one less diagonal matrix multiplication per iteration than the LSQR approach. The results of §3.5 show that the LSQR(A^{-1}) method for (3.2) is essentially twice as fast as the corresponding preconditioned MINRES method for (3.1) as it is able to step over the redundant steps in the MINRES approach. Since the condition of the generalised least-squares problem is better than that of the equivalent Schur complement system, it has been seen (§3.6) that the LSQR(A^{-1}) method also has nicer numerical properties than MINRES applied to the Schur complement system.

In conclusion, if it is reasonable to precondition (3.1) using the matrix A, algorithm LSQR(A^{-1}) should be used to solve (3.2) in favour of MINRES on the Schur complement equations since it is more reliable in practise.

Further, the preconditioning strategy described in §3.4.3 is far more effective than that originally given in [57], since it requires only one matrix-vector operation per iteration as opposed to two, and it allows many preconditioners which were impractical under the original preconditioning approach to be applied to the problem. Hence this strategy is suggested as the correct way to precondition not just LSQR(A^{-1}) , but also the original LSQR algorithm.

Chapter 4

Vavasis type finite elements, LSQR(D^{-1}) and preconditioning

In this chapter, a low order finite element method for partial differential equations of the form of the groundwater flow equations will be described which gives rise to discretisations whose matrix equations are ideally suited for solution with the LSQR(A^{-1}) method introduced in Chapter 3.

First recall that systems of the form

$$\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix}, \tag{4.1}$$

where A is symmetric positive definite, are (generalised) least-squares problems in disguise. Namely, (4.1) is equivalent to

$$\text{Find } y \in \mathbb{R}^m \text{ such that } \min_{p \in \mathbb{R}^m} ||Bp - b||_{A^{-1}} = ||By - b||_{A^{-1}}, \tag{4.2}$$

in the sense that the unique minimiser, y, of (4.2) is the same as the component y of the solution of (4.1). The LSQR(A^{-1}) method, introduced in Chapter 3, is a method for solving (4.2) (and hence (4.1)) which relies on the fact that the operation of A^{-1} is easy to compute. When this is true, the LSQR(A^{-1}) method is a fast and stable method of solving (4.2) (it is stable since it is effectively an implementation of the Lanczos algorithm applied to (4.1) with a change of variables, see Chapter 3).

The question naturally arises, which types of problems in applied mathematics give rise to systems of the form (4.1) where the operation of A^{-1} is easy to compute, and taking this question to extremes, which types of problems can give rise to linear systems of the form (4.1) where the matrix A is diagonal? In this case the change of notation $D := A$ will be preferred. It is well known that weighted least-squares problems can generate systems (4.1) with A diagonal, also interior point methods in minimisation and discretisations of electrical networks, see for example [16].

The goal here however is to solve the groundwater flow equations, and it will not, in general, be possible to choose a mixed finite element approximation that generates a linear system with diagonal A, since this would imply that an $\mathrm{L}^2(\Omega)$-orthogonal basis for the velocity trial space is known.

Vavasis [79] has described a finite element method for the pressure equation which gives rise to linear systems of the form (4.1) with A diagonal. This is achieved by introducing dummy variables in the finite element method for the pressure equation, and it will be seen that these dummy variables can be taken to be an approximation to the Darcy velocity. This formulation is not a true mixed formulation since there is essentially no choice in the selection of the velocity space, it is always taken to be the space spanned by the derivatives of the pressure basis functions and therefore doesn't necessarily satisfy any continuity conditions. This velocity approximation is of lower order than the pressure approximation, which is in contrast to the usual mixed finite element approximations of the groundwater flow equations which are usually more concerned with an accurate velocity approximation which satisfies some continuity requirement (see Chapter 5).

The Vavasis formulation is used as a template for analysing preconditioners, H, in the LSQR(A^{-1}, H^{-1}) algorithm (with diagonal A) since the formulation makes standard preconditioners for the conjugate gradient algorithm applied to the discretised pressure equation, such as additive Schwarz and incomplete factorisations, an obvious choice.

In §4.1, the standard finite element method for the pressure equation is reviewed together with some results on the accuracy of piecewise polynomial approximations. Vavasis' approach to discretising the pressure equation is described in §4.2 and it is shown how a Darcy velocity approximation can also be obtained. Piecewise-linear pressure approximations are considered throughout this chapter, although possible extensions to higher order approximations are discussed in §4.3. A brief review of preconditioning the LSQR(A^{-1}) algorithm is given in §4.4, and then additive Schwarz and incomplete factorisation preconditioners are described in §§4.5,4.6. Graham and Hagger [34, 33] have examined the effects of the permeability function k when using additive Schwarz preconditioners in the conjugate gradient solution of the finite element discretised pressure equation, and their results are presented in §4.5.3. These results are shown to also apply for additive Schwarz preconditioned LSQR(D^{-1}, H^{-1}) applied to the Vavasis-discretised problem. Numerical results presented in §4.7 tend to indicate that the results of Graham and Hagger also hold for suitable incomplete Cholesky preconditioned solvers. This result is shown to hold for Manteuffel's shifted incomplete Cholesky factorisation (see §4.6) in theorems 4.7.1 and 4.7.2, which is an analogue of the result [33, Theorem 4.4] for additive Schwarz preconditinoers.

4.1 Finite element methods for elliptic problems

Notice that the variable u in the groundwater flow equations (1.2) (with $\mu = 1$)

$$\begin{aligned} u + k\nabla p &= 0 \text{ in } \Omega \subset \mathbb{R}^n, \\ \nabla \cdot u &= 0 \text{ in } \Omega, \end{aligned} \tag{4.3}$$

with boundary conditions

$$\begin{aligned} p &= f \quad \text{on } \Gamma_1, \\ u \cdot n &= 0 \quad \text{on } \Gamma_2, \end{aligned} \tag{4.4}$$

where $\partial\Omega = \Gamma_1 \cup \Gamma_2$, Γ_1 is not empty and f is the restriction of an $\mathrm{H}^1(\Omega)$ function to Γ_1, can be eliminated by operating with *div*, to give

$$\nabla \cdot (k\nabla p) = 0. \tag{4.5}$$

Here, without loss of generality, the (constant) viscosity term is taken to be 1.

Throughout this chapter it will be assumed that Ω is a convex subset of $\mathbb{R}^n$ ($n = 2$, or 3), and that

$$\overline{\Omega} = \bigcup_{i=1}^{N} \overline{\Omega}_i,$$

where each Ω_i is open, $\Omega_i \cap \Omega_j = \emptyset$ for $i \neq j$ and

$$k|_{\Omega_i} = k_i,$$

where k_i is a constant. It will also be assumed that the boundaries of $\Omega, \Omega_1, \dots, \Omega_N$ are piecewise polygonal (piecewise-linear in two dimensions). The set $\overline{\Omega}_i \cap \overline{\Omega}_j$ (where $\overline{\Omega}_i \cap \overline{\Omega}_j \neq \emptyset$) may be referred to as the *interface* between Ω_i and Ω_j. It is immediately obvious from (4.5) that $p \in \mathrm{H}^1(\Omega)$. The above geometrical description allows a little more information about the derivative of p to be obtained. Suppose for simplicity that $N = 2$, so that $\overline{\Omega} = \overline{\Omega}_1 \cup \overline{\Omega}_2$, denote by Σ the interface between Ω_1 and Ω_2, and let

$$p_i = p|_{\overline{\Omega}_i}, \quad i = 1, 2.$$

Then obviously,

$$\nabla \cdot (k_i \nabla p_i) = 0 \ \text{ in } \Omega_i,$$

furthermore $p_1|_\Sigma = p_2|_\Sigma$ and $p_i|_{\partial\Omega_i} = p|_{\partial\Omega_i}$. Now using Green's formula, if $q \in \mathrm{H}^1(\Omega)$,

$$\int_{\partial\Omega_i} q k_i \frac{\partial p_i}{\partial n_i} d\gamma = \int_{\Omega_i} q \nabla \cdot (k_i \nabla p_i)\, dx + \int_{\Omega_i} k_i \nabla p_i \cdot \nabla q dx, \quad i = 1, 2, \tag{4.6}$$

where n_i is the unit outward normal to Ω_i and γ is the unit of arclength along $\partial\Omega_i$. Summing (4.6) with $i = 1, 2$,

$$\begin{aligned} \sum_{i=1}^{2} \int_{\partial\Omega_i} q k_i \frac{\partial p_i}{\partial n_i} d\gamma &= \int_{\Omega} q \nabla \cdot (k \nabla p)\, dx + \int_{\Omega} k \nabla p \cdot \nabla q dx \\ &= \int_{\partial\Omega} q k \frac{\partial p}{\partial n} d\gamma, \end{aligned}$$

using Green's formula again. Here n is the unit outward normal to Ω. Noticing that the integral on the right hand side is contained in the integral on the left hand side it immediately follows that

$$\int_{\Sigma} q \left(k_1 \frac{\partial p}{\partial n_1} + k_2 \frac{\partial p}{\partial n_2} \right) d\gamma = 0, \quad \forall q \in \mathrm{H}^1(\Omega),$$

and hence

$$k_1 \frac{\partial p}{\partial n_1} + k_2 \frac{\partial p}{\partial n_2} = 0 \text{ a.e. on } \Sigma. \tag{4.7}$$

Equation (4.7) provides a condition on the continuity of the derivatives of p along normals to the interface between Ω_1 and Ω_2, and can be interpreted as $[k\partial p/\partial n] = 0$ on all interfaces in Ω, where $[\cdot]$ is a jump function. Hence (4.5) on the domain Ω is equivalent to,

$$\begin{aligned} \nabla \cdot (k_i \nabla p_i) &= 0, \ i = 1, 2, \\ p_1|_\Sigma &= p_2|_\Sigma, \\ k_1 \tfrac{\partial p}{\partial n_1} + k_2 \tfrac{\partial p}{\partial n_2} &= 0. \end{aligned}$$

The case $N > 2$ is similar.

A standard finite element for method (4.5) with boundary conditions

$$\begin{aligned} p &= f \quad \text{on } \Gamma_1, \\ \nabla p \cdot n &= 0 \quad \text{on } \Gamma_2, \end{aligned} \tag{4.8}$$

then proceeds as follows. Let Π and Π^0 denote the test and trial spaces,

$$\begin{aligned} \Pi &= \{q \in \mathrm{H}^1(\Omega) \mid q|_{\Gamma_1} = f\}, \\ \Pi^0 &= \{q \in \mathrm{H}^1(\Omega) \mid q|_{\Gamma_1} = 0\}. \end{aligned}$$

Then p can be expressed as $p = p^0 + p^f$ where $p^f \in \Pi$ is any extension of f from Γ_1 onto Ω (i.e. $p^f|_{\Gamma_1} = f$) and $p^0 \in \Pi^0$. A similar analysis to that above yields,

$$\begin{aligned} 0 &= \textstyle\int_\Omega q \nabla \cdot (k \nabla p) \ dx \\ &= \textstyle\int_{\partial\Omega \cup \text{Interfaces}} q \, (k \nabla p) \cdot n \ d\gamma - \int_\Omega k \nabla p \cdot \nabla q \ dx, \end{aligned}$$

where $d\gamma$ is the element of arclength along $\partial\Omega$ and n is the unit outward normal to $\partial\Omega$ and to each interface. Hence, using the boundary conditions, the interface condition (4.7) and the fact that $q \in \Pi^0$,

$$\int_\Omega k \nabla p^0 \cdot \nabla q \ dx = - \int_\Omega k \nabla p^f \cdot \nabla q \ dx,$$

and so the following variational form of (4.5) and (4.8) is obtained,

$$\textit{Find } p^0 \in \Pi^0 \textit{ such that } a(p^0, q) = L(q) \;\; \forall q \in \Pi^0. \tag{4.9}$$

Here $a(p,q) = \langle k\nabla p \, , \, \nabla q\rangle_{L^2(\Omega)^n}$ and $L(q) = -\int_\Omega k\nabla p^f \cdot \nabla q \; dx$. A simple application of the Cauchy-Schwartz inequality on $L^2(\Omega)^n$ shows that the symmetric, bilinear form $a(\cdot,\cdot)$ is continuous,

$$|a(p,q)| \le C_1 \, ||p||_{H^1(\Omega)} \, ||q||_{H^1(\Omega)} \;\; \forall p, q \in \Pi^0,$$

whilst the coercivity condition,

$$a(q,q) \ge C_2 \, ||q||^2_{H^1(\Omega)} \;\; \forall q \in \Pi^0,$$

follows from an application of the Poincaré inequality. Lastly the linear form $L(\cdot)$ is also continuous,

$$|L(q)| \le ||k||_\infty \, |p^f|_{H^1(\Omega)} |q|_{H^1(\Omega)} \le C_3 \, ||q||_{H^1(\Omega)} \, ,$$

when $||k||_\infty$ is assumed to be finite. The constants C_1 and C_3 can be seen to depend on the maximum of k on Ω whilst C_2 depends on the minimum of k on Ω. As a consequence of the Lax-Milgram lemma [13, Theorem 1.1.3], the solution p of (4.9) is unique and standard variational theory states that p satisfies the stability estimate

$$||p||_{H^1(\Omega)} \le \frac{C_3}{C_2}.$$

Furthermore, if $\mathcal{T}$ is a conforming triangulation of Ω with triangles of maximum radius h, $\mathcal{S}_h \subset H^1(\Omega)$ is a finite dimensional subspace, and

$$\begin{aligned} \Pi_h &= \{q \in \mathcal{S}_h \mid q \text{ interpolates } f \text{ at all nodes on } \Gamma_1\}, \\ \Pi_h^0 &= \{q \in \mathcal{S}_h \mid q \text{ is zero at all nodes on } \Gamma_1\}, \end{aligned}$$

then $p_h \in \Pi_h$ can be expressed as $p_h = p_h^0 + p_h^f$ where $p_h^0 \in \Pi_h^0$ and $p_h^f \in \Pi_h$ is any function in Π_h (typically p_h^f is the function which interpolates f at nodes on Γ_1 and is zero at all other nodes). The discrete analogue of (4.9) is then

$$\textit{Find } p_h^0 \in \Pi_h^0 \textit{ such that } a(p_h^0, q) = L(q) \;\; \forall q \in \Pi_h^0. \tag{4.10}$$

If $\{\psi_i\}_{i=1}^m$ is a basis for Π_h^0 then (4.10) becomes

$$\textit{Find } p_h^0 \in \Pi_h^0 \textit{ such that } a(p_h^0, \psi_i) = L(q) \;\; i = 1, \ldots, m,$$

and if ψ_0 is a function which interpolates f on Γ_1 and is zero at all other nodes then, with $p_h^f = \varphi_0(x)$, p_h can be expressed as

$$p_h(x) = \psi_0(x) + \sum_{i=1}^m y_i \psi_i(x).$$

The discrete variational form can be expressed as the linear system to be solved for the unknown nodal values $y = [y_1 \ldots y_m]^T$,

$$Ky = \tilde{b},$$

where $K = [K_{ij}] = [a(\psi_i, \psi_j)]$ and $\tilde{b} = [\tilde{b}_i] = [a(\psi_i, \psi_0)]$. K will be referred to as a *scaled stiffness matrix* since the bilinear form a is an inner product of $\nabla\psi_i$ and $\nabla\psi_j$ terms which are scaled by the permeability term k.

It can be shown that the error estimate

$$||p - p_h||_{\mathrm{H}^1(\Omega)} \leq \frac{C_2}{C_3} ||p - q||_{\mathrm{H}^1(\Omega)} \quad \forall q \in \Pi_h, \tag{4.11}$$

holds, and in the energy norm, $||q||_e = a(q,q)^{\frac{1}{2}}$, p_h can be shown to be the optimal approximation to p, i.e.

$$||p - p_h||_e \leq ||p - q||_e \quad \forall q \in \Pi_h. \tag{4.12}$$

See [13, 41] for more details.

Error estimates such as (4.11) and (4.12) are useful theoretical tools, but give little indication of what types of errors can be expected in practise. Instead, some idea of how the choice of Π_h and the mesh size affects the error is needed.

Here it will be assumed that $\Omega \subset \mathbb{R}^2$ although the results presented will generalise to higher dimensions. Suppose $\Omega = \bigcup_{T\in\mathcal{T}} T$ is a triangulation of Ω that resolves the discontinuities in k (i.e. the discontinuities in k lie along triangle edges only so that k is constant within each triangle in $T \in \mathcal{T}$), that

$$\Pi_h = \{q \mid \text{q is continuous on } \Omega \text{ and } q|_T \text{ is linear } \forall T \in \mathcal{T} \text{ and } q \text{ interpolates } f \text{ on } \Gamma_1\},$$

and that $\{\psi_i\}_{i=1}^m$ are the hat functions centred on the interior nodes and nodes on the Neumann boundary, Γ_2, of the triangulation. Clearly $\Pi_h \subset \Pi$, and if ψ_0 is the piecewise linear interpolant of f on the Dirichlet boundary Γ_1 and is zero at all other nodes of the triangulation, then $\{\psi_i\}_{i=0}^m$ is a basis for Π_h. In [17] it is shown that if $q \in \mathrm{H}^2(T)$ and πq is the linear interpolation of q on the triangle T (which coincides with q at the vertices of T), then

$$||q - \pi q||_{\mathrm{L}^2(T)} \le C h_T^2 |q|_{\mathrm{H}^2(T)}, \tag{4.13}$$

$$|q - \pi q|_{\mathrm{H}^1(T)} \le C \frac{h_T^2}{r_T} |q|_{\mathrm{H}^2(T)}, \tag{4.14}$$

where $|\cdot|_{\mathrm{H}^j(T)}$ denotes the $\mathrm{H}^j(T)$ seminorm, h_T is the length of the longest side of T, and r_T is the diameter of the greatest circle in T. If $\pi_h q$ is used to denote the piecewise linear interpolation of q on the triangulation $\mathcal{T}$ (so that $\pi_h q$ coincides with q at all nodes in the triangulation) then, summing (4.13) over all triangles in $\mathcal{T}$,

$$\begin{aligned} ||q - \pi_h q||^2_{\mathrm{L}^2(\Omega)} &= \textstyle\sum_{T\in\mathcal{T}} ||q - \pi q||^2_{\mathrm{L}^2(T)}, \\ &\le \textstyle\sum_{T\in\mathcal{T}} C^2 h_T^4 |q|^2_{\mathrm{H}^2(T)}, \\ &\le C^2 \max_{T\in\mathcal{T}} h_T^4 \textstyle\sum_{T\in\mathcal{T}} |q|^2_{\mathrm{H}^2(T)}. \end{aligned}$$

Hence, writing $h = \max_{T\in\mathcal{T}} h_T$, the interpolation error on the whole of the triangulation,

$$||q - \pi_h q||_{\mathrm{L}^2(\Omega)} \le C h^2 (\sum_{T\in\mathcal{T}} |q|^2_{\mathrm{H}^2(T)})^{\frac{1}{2}}, \tag{4.15}$$

is obtained. If it is also assumed that the triangles T are not allowed to become too thin as the mesh size decreases, that is there exists a constant β which is independent of the triangulation and which satisfies

$$\frac{r_T}{h_T} \geq \beta \quad \forall T \in \mathcal{T}, \tag{4.16}$$

then the error estimate in the $\mathrm{H}^1(\Omega)$ seminorm,

$$|q - \pi_h q|_{\mathrm{H}^1(\Omega)} \leq \frac{Ch}{\beta}(\sum_{T\in\mathcal{T}} |q|^2_{\mathrm{H}^2(T)})^{\frac{1}{2}}, \tag{4.17}$$

can be found by summing (4.14) over $\mathcal{T}$ in the same way that (4.15) was obtained from (4.13). Notice that (4.17) is equivalent to

$$||\nabla q - \nabla(\pi_h q)||_{\mathrm{L}^2(\Omega)} \leq \frac{Ch}{\beta}(\sum_{T\in\mathcal{T}} |q|^2_{\mathrm{H}^2(T)})^{\frac{1}{2}}, \tag{4.18}$$

so that a bound on the gradient of the interpolation error in $\mathrm{L}^2(\Omega)$ is known. Hence the interpolation error and the interpolation error of the gradient in the $\mathrm{L}^2(\Omega)$ norm when interpolating a function which lies in $\mathrm{H}^2(T)\forall T \in \mathcal{T}$ with piecewise linear functions on $\mathcal{T}$ are $O(h^2)$ and $O(h)$ respectively.

Now consider (4.11) with $q = \pi_h p$. Then (4.15) and (4.17) imply

$$||p - p_h||_{\mathrm{H}^1(\Omega)} \leq Ch(\sum_{T\in\mathcal{T}} |p|^2_{\mathrm{H}^2(T)})^{\frac{1}{2}}, \tag{4.19}$$

(since k is constant in each $T \in \mathcal{T}$, it is known that $p \in \mathrm{H}^2(T)$ for each $T \in \mathcal{T}$) and hence

$$||\nabla p - \nabla p_h||_{\mathrm{L}^2(\Omega)} \leq Ch(\sum_{T\in\mathcal{T}} |p|^2_{\mathrm{H}^2(T)})^{\frac{1}{2}}. \tag{4.20}$$

Here the constants C depend on β, which is independent of k and independent of the mesh size. Equation (4.15) suggests that an $O(h^2)$ error of $p - p_h$ in the $\mathrm{L}^2(\Omega)$ norm may be possible. Indeed the following theorem, which is described in [14, Ch. VII §3], shows this to be true, but only in the case that $k \in C^\infty(\Omega)$.

Theorem 4.1.1 *Provided that the domain Ω is smooth or is polygonal and convex,* *and* $k \in C^{\infty}(\Omega)$, *then*

$$||p - p_h||_{L^2(\Omega)} \leq Ch^2 |p|_{H^2(\Omega)}. \tag{4.21}$$

Proof See [14]. □

Hence the finite element method applied to (4.5) with boundary conditions (4.8) with piecewise linear elements on the triangulation $\mathcal{T}$ gives errors of the form

$$\begin{aligned} ||\nabla p - \nabla p_h||_{\mathrm{L}^2(\Omega)} &= O(h), & (4.22)\\ ||p - p_h||_{\mathrm{L}^2(\Omega)} &= O(h^2), & (4.23) \end{aligned}$$

when k is smooth. i.e. setting $u_h = \nabla p_h$ gives an $O(h)$ approximation to the solution variable u, whilst the approximation to p is $O(h^2)$. In the case that k is piecewise constant, the solution p of (4.5), (4.8) will not generally be an $\mathrm{H}^2(\Omega)$ function and so the theory above will cease to apply. However it is obvious from (4.19) that at least an $O(h)$ bound on $||p - p_h||_{\mathrm{L}^2(\Omega)}$ exists. Since (4.19) is actually an $O(h)$ bound on $||p - p_h||_{\mathrm{H}^1(\Omega)}$, it may be expected that $||p - p_h||_{\mathrm{L}^2(\Omega)}$ satisfies a better bound than $O(h)$. Fraenkel [24] has shown that if $\mathcal{S} \subset \Omega$ and $\mathcal{S}$ contains only linear interface sections (not piecewise-linear interface sections) then $p \in \mathrm{H}^s(\mathcal{S})$ for $s < \frac{3}{2}$. A further result of [17] then informs that

$$||p - \pi_h p||_{\mathrm{L}^2(\mathcal{S})} \leq Ch^s |p|_{\mathrm{H}^s(\mathcal{S})}.$$

Therefore the error in the piecewise-linear interpolant of p on any subset $\mathcal{S}$ of Ω, of the form described above, is $O(h^s)$ where $s < \frac{3}{2}$. Hence it might be suspected that the approximation p_h to p may actually be a higher order approximation in $\mathrm{L}^2(\Omega)$ than $O(h)$ (possibly close to $O(h^{\frac{3}{2}})$). The numerical results following §4.2 appear to agree with this supposition. Clearly (4.19) still implies that $||u - u_h||_{\mathrm{L}^2(\Omega)} = O(h)$ in the case that k is piecewise constant.

A drawback to using the approach described in this section is that the assembled stiffness matrix $K = [K_{ij}] = (a(\psi_i, \psi_j))$, in the equation

$$Ky = \underset{\sim}{b}, \tag{4.24}$$

which is to be solved for the unknown nodal values y of p, can be extremely ill-conditioned. The condition number of K is bounded below by the maximum ratio of any two diagonal entries of K so that,

$$\kappa(K) \geq C\frac{\max_{x\in\Omega} k(x)}{\min_{x\in\Omega} k(x)},$$

and hence the conjugate gradient algorithm applied to (4.24) is not expected to be numerically stable for large ratios of $\max_{x\in\Omega} k(x)/\min_{x\in\Omega} k(x)$. The matrix K can be badly scaled throughout, with no entries being independent of $k(x)$.

In the following section a method which simultaneously discretises (4.3), (4.4) to provide an approximation to both p and u will be described which addresses the drawback of the above method, the bad scaling will be confined to a diagonal portion $D = A$ of the coefficient matrix (4.1) which can be handled independently of the unscaled part. The resulting system can be solved using the LSQR(D^{-1}) algorithm which is seen to be more stable than the conjugate gradient algorithm applied to (4.24) in numerical experiments. If required, the approximation u_h can be calculated during the iteration thus rectifying the first point. The approximations p_h and u_h will satisfy the same error bounds found above.

4.2 Mixed elements based on unmixed approximations

The analysis of the previous section implies that any discretisation of (4.5) and (4.8), which is based on piecewise linear elements on a triangulation of Ω (which resolves discontinuities in k), will satisfy the error bounds (4.19) and (4.20). Such a piecewise linear approximation to (4.5) (with purely Dirichlet boundary conditions) is given in [79], where (4.5) is not considered as a reduced form of (4.3) and (4.4), instead it is taken to be a model of the heat equation in a composite material with known temperature distribution on the boundary, where only an approximation to p (the temperature) is required. The method of discretisation used is still applicable in the case of mixed boundary conditions, and introduces an artificial variable which can be taken to be an approximation of u, in the case of (4.5) describing a reduced form of the groundwater flow equations. The linear system which arises from this discretisation is of the form (1.1) with the matrix A being diagonal.

Suppose again that $\{\psi_i\}_{i=1}^{m}$ are the hat functions on the interior and Neumann boundary nodes of the triangulation $\Omega = \bigcup_{T \in \mathcal{T}} T$ which resolves the discontinuities in k. Here $\Omega \subset \mathbb{R}^d$ is assumed, although for simplicity the practical implementation described is given for the case $\Omega \subset \mathbb{R}^2$. The extension to $\Omega \subset \mathbb{R}^d$ is obvious. In terms of these basis functions the solution p_h of (4.10) can be expressed as

$$p_h(x) = \psi_0(x) + \sum_{i=1}^{m} y_i \psi_i(x), \tag{4.25}$$

where y_i is the value of p_h at the node upon which ψ_i is centred, and ψ_0 is a piecewise linear interpolation of f on the Dirichlet boundary nodes and is zero at all other nodes. The approach to solving (4.5) given in [79] is to introduce artificial variables $x_{\mathrm{T}} \in \mathbb{R}^d$ where x_{T} represents the gradient of p_h on T, multiplied by the factor

$$\int_T k(x)\ dx.$$

Since the triangulation is assumed to resolve discontinuities in the piecewise-constant function k this calculation is trivial. An extra scale factor is also used in [79] so that error analysis for the NSHI algorithm, presented there, is possible. The NSHI algorithm is an extension of the NSH algorithm [78] for solving augmented systems of the form (4.1) where the matrix $A = D$ is extremely ill-conditioned, and its derivation is motivated by Stewart's result [74] which was discussed in §3.7. The variables x_{T} are truly artificial since they are not required to be recovered and they are not considered to have any physical relevance. As is mentioned above, in the case that (4.5) represents the reduced form of the groundwater flow equations with velocity terms eliminated, the variables x_{T}, being the scaled gradients of the pressures, provide an approximation to the Darcy velocity and are no longer truly artificial variables since they have some physical relevance.

Since p_h is linear on each T, $T \in \mathcal{T}$, the scaled gradient x_{T} is simple to determine by interpolation at the nodal values (for higher order elements a similar argument will follow and is outlined in §4.3). Ordering $\{x_{\mathrm{T}}\}_{T \in \mathcal{T}}$ into the vector x, the linear relationship between x_{T} and $p_h|_T$ implies that there exist matrices D, B and a vector

b such that

$$Dx = -By + b. \tag{4.26}$$

Here the matrix D is a diagonal scaling matrix whose entries have the form

$$\left[\int_T k(x)\ dx\right]^{-1} \tag{4.27}$$

and are ordered corresponding to the position of x_T in x. The vector $-By+b$ has to represent the (unscaled) gradients of p_h on each triangle T. The procedure given for assembling B and b in [79] is as follows. If $y_\mathrm{T} = \left(y_1^{(T)}, y_2^{(T)}, y_3^{(T)}\right)^T$ denotes the values of p_h at the nodes of T then an element gradient matrix $G^{(T)}$ can be formed so that the gradient of the interpolating plane is given by $G^{(T)}y_\mathrm{T}$. To form $G^{(T)}$ first write

$$G^0 = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \end{bmatrix}.$$

Then $G^0 y_\mathrm{T}$ is a 2$-$vector containing the pressure differences between nodes $y_1^{(T)}$ and $y_3^{(T)}$, and $y_2^{(T)}$ and $y_3^{(T)}$. Now if $v_1^{(T)}, v_2^{(T)}$ and $v_3^{(T)}$ are the corresponding vertices of T and

$$V_T = \left(v_1^{(T)} - v_3^{(T)}, v_2^{(T)} - v_3^{(T)}\right),$$

then V_T^T is a change of basis from the usual Cartesian coordinates in $\mathbb{R}^2$ to the coordinates with basis $\{\overrightarrow{v_3^{(T)}v_1^{(T)}}, \overrightarrow{v_3^{(T)}v_2^{(T)}}\}$. Hence if t is a vector written in terms of the basis $\{\overrightarrow{v_3^{(T)}v_1^{(T)}}, \overrightarrow{v_3^{(T)}v_2^{(T)}}\}$, then

$$s = V_T^{-T} t,$$

is its Cartesian representation. The element gradient matrix $G^{(T)}$ is then formed as

$$G^{(T)} = V_T^{-T} G^0,$$

i.e. G^0 calculates the pressure difference along two edges of T and V_T^{-T} transforms this into the gradient in Cartesian coordinates.

For example suppose that T has vertices $(0,0),(1,0),(\frac{1}{2},\frac{\sqrt{3}}{2})$ and that the pressure at nodes $(0,0)$ and $(\frac{1}{2},\frac{\sqrt{3}}{2})$ is 0 and the pressure at $(1,0)$ is 1. Then the interpolated pressure on T is actually given by $q(x) = x_1 - x_2/\sqrt{3}$ and it is clear that $\nabla q(x) = (1,-1/\sqrt{3})$. Here

$$V_T = \begin{bmatrix} -\frac{1}{2} & \frac{1}{2} \\ -\frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix},$$

and so

$$G^{(T)}y_{\mathrm{T}} = \frac{2}{\sqrt{3}} \begin{bmatrix} -\frac{\sqrt{3}}{2} & \frac{\sqrt{3}}{2} \\ -\frac{1}{2} & -\frac{1}{2} \end{bmatrix} \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ -\frac{1}{\sqrt{3}} \end{bmatrix}.$$

The matrix B can be formed block row-wise (with blocks of size d) where the entries in each block correspond to the relevant, non Dirichlet boundary, elements in each $G^{(T)}$ with the columns of B arranged in keeping with the global node ordering. The Dirichlet boundary entries of each $G^{(T)}$ are multiplied by the corresponding nodal values on the boundary to form the vector b.

With D, B and b defined above, (4.26) holds. However it still remains to enforce (4.10). To do this it suffices that, since p_h solves (4.10),

$$a(p_h, \psi_i) = 0 \quad i = 1, \dots m. \tag{4.28}$$

(4.28) can be rewritten as

$$\sum_{T \in \mathcal{T}} \int_T k(x) \nabla p_h \cdot \nabla \psi_i \; dx = 0, \quad i = 1, \dots m,$$

and since $\nabla p_h|_{\mathrm{T}} = \left(\int_T k(x) \; dx\right)^{-1} x_{\mathrm{T}}$, which is constant, and $\nabla \psi_i$ is constant on T,

$$\sum_{T \in \mathcal{T}} x_{\mathrm{T}}^T \nabla \psi_i|_{\mathrm{T}} = 0, \quad i = 1, \dots m. \tag{4.29}$$

Notice now that if $\psi_i \not\equiv 0$ on T then ψ_i takes the value 1 at one node of T and so $\nabla \psi_i = G^{(T)} e_{\mathrm{T}_i}$ on T, where e_{T_i} is a vector of zeros except for a 1 in position corresponding to node i. On the other hand if $\psi_i \equiv 0$ on T then $\nabla \psi_i = 0$ on T. Hence

$x_{\mathrm{T}}^T \nabla \psi_i = (G^{(T)} e_{\mathrm{T}_i})^T x_{\mathrm{T}}$, which is the inner product of the ith column of B with x, when $\psi \not\equiv 0$ on T and $x_{\mathrm{T}}^T \nabla \psi_i = 0$ otherwise. Thus $\sum_{T \in \mathcal{T}} x_{\mathrm{T}}^T \nabla \psi_i$ is equivalent to the inner product of the ith column of B with x, and so (4.29) becomes

$$B^T x = 0. \tag{4.30}$$

Combining (4.26) and (4.30), the linear equations for the nodal values of p_h, y, and the scaled gradients, x, are then

$$\begin{bmatrix} D & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix}. \tag{4.31}$$

Since both (4.24) and (4.31) derive from (4.10), it is clear that the solution y of (4.24) and y of (4.31) (which are obviously unique from considerations in §4.1) are equal. Hence both systems give the same solution p_h. Considering (4.22) and the fact that u_h obtained from (4.31) is simply the scaled gradient of p_h it is clear that

$$||u - u_h||_{\mathrm{L}^2(\Omega)} = O(h),$$

where the constant in the $O(h)$ term is dependent on $\min_{x \in \Omega} k(x) / \max_{x \in \Omega} k(x)$. The pressure approximation p_h obtained from (4.31) will possibly satisfy a higher order error bound in $\mathrm{L}^2(\Omega)$ following comments made at the end of §4.1, possibly close to $O(h^{\frac{3}{2}})$, but certainly no worse than $O(h)$. Indeed the following results seem to agree with this hypothesis.

Results

Figure 4-1 shows the velocity approximation u_h and contours of the pressure approximation p_h of the solution to the groundwater flow equations for flow in a rectangular region $\Omega \subset \mathbb{R}^2$. The flow direction is indicated by arrows. Shaded regions in the domain indicate areas where $k(x) = 10^{-3}$, whilst white areas have $k(x) = 1$. The pressure is fixed at 100 on the top of the domain and 10 on the bottom. A 3d plot of the pressure p_h is also shown. Taking the solution with nodal values (x^*, y^*) on a 256 × 256 grid (65,535 pressure and 262,144 velocity unknowns) to be close to exact, the order of approximation as the mesh size is refined (by halving) from an 8 × 8 grid to a 64 × 64

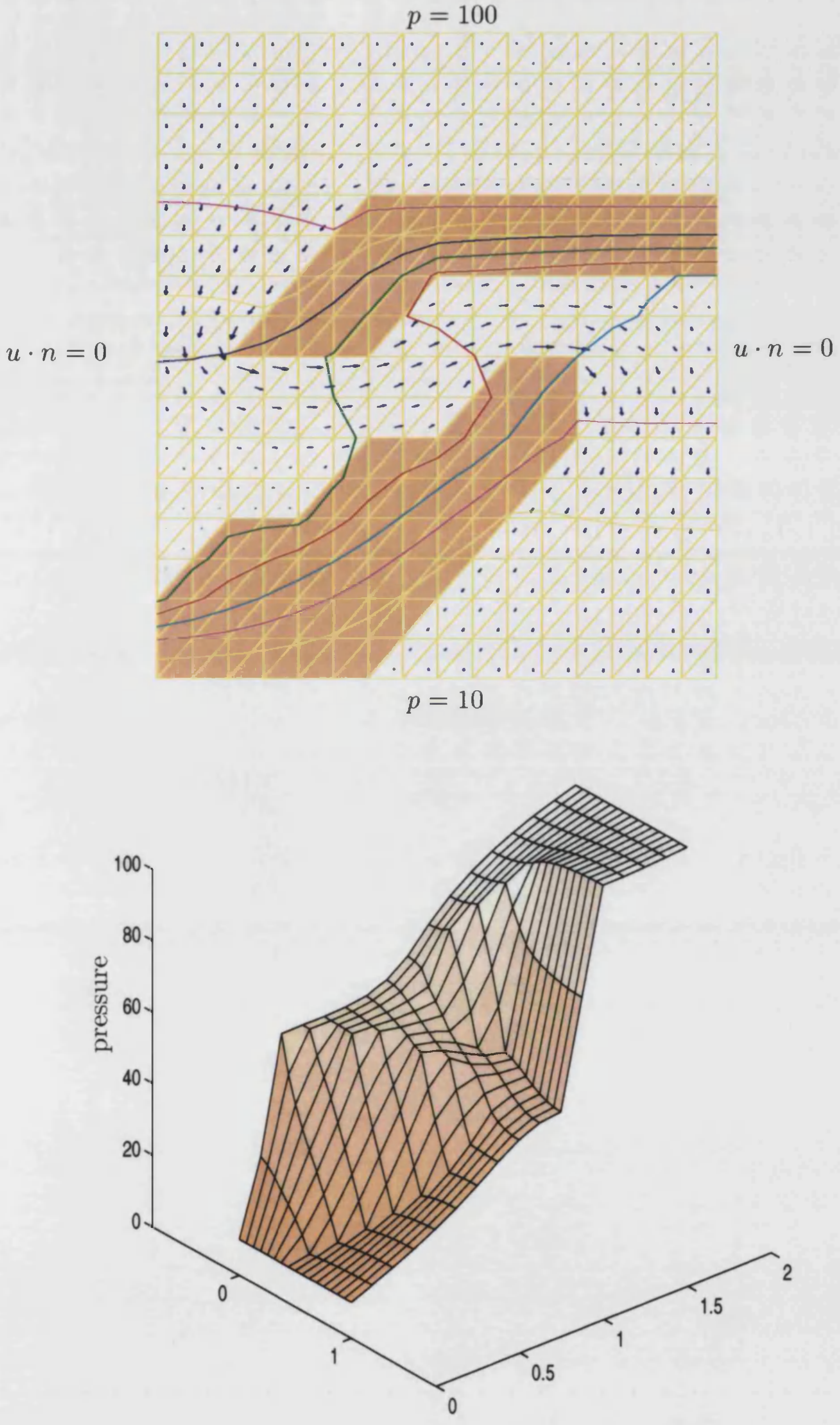


Figure 4-1: Groundwater flow in a 2d region

grid is shown in the following table. The number of velocity and pressure unknowns are denoted by n and m respectively.

h	(n,m)	$\|\|y^* - y_h\|\|_2$	$r_h = \frac{\|\|y^*-y_h\|\|_2}{\|\|y^*-y_{h/2}\|\|_2}$	$\log_2(r_h)$
1/8	(256,63)	352.8540	2.3749	1.2479
1/16	(1024,255)	148.5783	2.4014	1.2639
1/32	(4096,1023)	61.8707	2.5645	1.3587
1/64	(16384,4095)	24.1258		

The norm $||p - p_h||_{L^2(\Omega)}$ is not calculated, rather the l_2 norm of the nodal values is used. Since $||p - p_h||_{L^2(\Omega)} = \left((y - y_h)^T M(y - y_h)\right)^{\frac{1}{2}}$ where M is the pressure mass matrix, and

$$ch^2 \, ||y - y_h||_2^2 \leq (y - y_h)^T M(y - y_h) \leq Ch^2 \, ||y - y_h||_2^2$$

(see [41, (7.46)]) where c and C are constants, the l_2 norm is acceptable for the task. It is clear from the last column that the order of convergence is better than $O(h)$, but smaller than $O(h^{\frac{3}{2}})$ as was suggested in the comments made at the end of §4.1.

4.3 Higher order Vavasis type elements

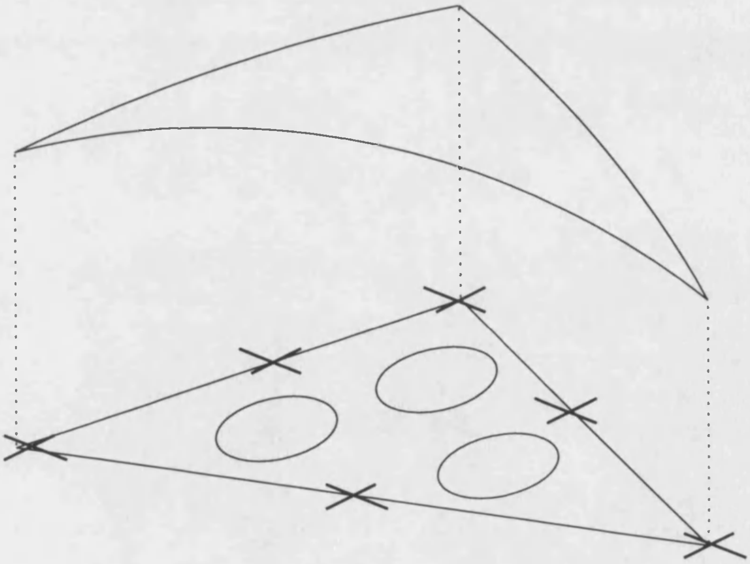


Figure 4-2: A typical element T_K

Taking higher degree finite element spaces for the pressure variable in the standard finite element method for the elliptic equation (§4.1) obviously leads to higher degrees of precision of the approximation. If continuous piecewise polynomial functions of degree s are used on triangulations obeying (4.16) then it can be shown that

$$\begin{aligned} \|p - \pi_h p\|_{\mathrm{L}^2(T)} &\leq Ch^{s+1}|p|_{\mathrm{H}^{s+1}(T)}, \\ \|\nabla p - \nabla \pi_h p\|_{\mathrm{L}^2(T)} &\leq Ch^{s}|p|_{\mathrm{H}^{s+1}(T)}, \end{aligned}$$

provided that p is sufficiently smooth. (See [17]). These interpolation errors can be used in an analogous way as in §4.1 to prove approximation errors of the form

$$\|p - p_h\|_{\mathrm{L}^2(\Omega)} = O(h^{s+1}), \tag{4.32}$$

$$\|u - u_h\|_{\mathrm{L}^2(\Omega)} = O(h^{s}), \tag{4.33}$$

where $u_h = -k\nabla p_h$, for the case that $k \in C^\infty(\Omega)$.

Again, Vavasis type elements can be used to discretise the elliptic problem and still retain the structure (4.31). To give some idea of how this may be achieved in the case of piecewise quadratic functions consider figure 4-2. The crosses at the vertices of the triangle and midpoints indicate nodes of the pressure discretisation, and the circles indicate points where the gradient of the pressure must be calculated. These three gradient nodal values can be used to determine the linear velocity planes. Such a discretisation will satisfy an $O(h^3)$ $\mathrm{L}^2(T)$ error in the pressure variable and an $O(h^2)$ $\mathrm{L}^2(T)$ error in the velocity variable, in the case that k is smooth, however the velocity approximation will still not necessarily be continuous on Ω.

4.4 Preconditioning LSQR(D^{-1})

It is well known that iterative methods based on a Lanczos process work well on matrices that are well conditioned or have only a few distinct eigenvalues. The eigenvalue distribution of the coefficient matrix in

$$\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix}. \tag{4.34}$$

was described in §2.3, specifically in Theorems 2.3.1 and 2.3.2, and a polynomial preconditioner for (4.34) based on its eigenvalue distribution was discussed in §2.9. The spectrum of a preconditioned version of (4.34) will be considered in §B.1.

Since the (diagonal) entries in the symmetric positive definite matrix A can become arbitrarily ill conditioned when (4.34) represents a discretisation of an elliptic operator with variable scaling (and when triangular elements become long and thin, see §5.3), it is obvious from Theorem 2.3.1 that (4.34) can be made arbitrarily ill conditioned and so some form of preconditioning is appropriate. Here attention is restricted to the case that (4.34) represents a discretisation of the elliptic equation (4.3) with boundary conditions (4.4) by Vavasis type finite elements (§4.2) so that $A = D$. Then the preconditioned algorithm LSQR(D^{-1}, H^{-1}) can be applied to (4.34) as outlined in §3.4.3. The case in which (4.34) arises from more general discretisations is left until §5.2.

The preconditioned algorithm, LSQR(D^{-1}, H^{-1}) , solves

$$\min_{p \in \mathbb{R}^m} \left|\left|BN^{-T}p - b\right|\right|_{D^{-1}}, \tag{4.35}$$

where $H = NN^T$ and the solution of (4.34) (with $A = D$), is given by $y = N^{-T}p$. Here by 'the solution', the y component in (4.34) is being referred to. Since D is diagonal the x component is easy to recover. Of course it is never actually necessary to form the Cholesky factor N of H, since the LSQR(D^{-1}, H^{-1}) algorithm operates in the D^{-1} and H^{-1} inner products as opposed to the standard Euclidean inner products of LSQR. The solution y is obtained naturally from the algorithm with only one H^{-1} operation and a normalisation step per iteration extra to the cost of the LSQR(D^{-1}) algorithm. The normal equations of (4.35) are

$$N^{-1}B^TD^{-1}BN^{-T}p = N^{-1}B^TD^{-1}b, \tag{4.36}$$

and so the preconditioner H in LSQR(D^{-1}, H^{-1}) can be any preconditioner which can be used to centrally precondition the unpreconditioned normal equations. Now since

$$x = D^{-1}(b - By),$$

the unpreconditioned normal equations are equivalent to $B^Tx = 0$ and hence by (4.30), equivalent to (4.28). Therefore the preconditioner H that is required is any (good) preconditioner for the stiffness matrix K that arises from the usual primal form of the finite element method for (4.5),(4.8). Three such preconditioners, diagonal scaling, additive Schwarz preconditioning and incomplete Cholesky preconditioning will be considered. Since the stiffness matrix arising in Vavasis finite element discretisation of (4.5) is equal to the Schur complement of (4.34) the notation $K = B^T D^{-1} B$ will be assumed throughout.

4.5 Diagonal scaling and additive Schwarz preconditioners

4.5.1 Diagonal scaling

Diagonal scaling is the simplest form of preconditioning possible, here $H = H_D$ is defined by

$$H_D = \mathrm{diag}(K),$$

so that the coefficient matrix in the preconditioned normal equations (4.36) has unit diagonal elements. This approach was introduced in [23], where it was shown that if K has 'property-A' (K can be partitioned into 2-by-2 block form with diagonal matrices in the $(1,1)$ and $(2,2)$ block), then diagonal scaling is the best scaling in terms of minimising $\kappa(H^{-1}K)$ over all diagonal matrices H. In [10] it is observed that for a variety of matrices, diagonal scaling typically has the effect of scaling large eigenvalues to near unity but often leaves small eigenvalues which cause poor convergence, this phenomenon will be discussed shortly for the case that K is a scaled stiffness matrix. Hence diagonal scaling is not expected to be a particularly effective preconditioner for discretisations of elliptic operators. In the case of preconditioning mass matrices the story is different however, diagonal scaling is particularly effective (see [82]) and is discussed in §5.4.

4.5.2 Additive Schwarz preconditioning

Additive Schwarz preconditioning is a type of domain decomposition preconditioning, see [29, 10.3.4]. Suppose that $\mathcal{T}$ is a triangulation of Ω which resolves the discontinuities in k, and that the set $\{\Omega_j\}_{j=1}^{s}$ is a partition of Ω into s (non-overlapping) subsets and that each $\Omega_j \subset \Omega'_j$ where $\partial\Omega'_j$ coincides with the edges of the triangulation. The assumption that $\mathcal{T}$ resolves the discontinuities in k is not actually necessary as is explained in [34], although the simple case suffices here since it is assumed that the discontinuities in k lie along the edges of polyhedra and not curved surfaces. If $\mathcal{N}(\mathcal{S})$ is used to denote the subset of nodes of the triangulation that are contained in $\mathcal{S}$ then the inclusion operator

$$R_j^T : \mathcal{N}(\Omega'_j) \longrightarrow \mathcal{N}(\Omega),$$

is defined to transform the global vector y_j in $\mathcal{N}(\Omega'_j)$ to the vector $y \in \mathcal{N}(\Omega)$ which takes the value of y_j on $\mathcal{N}(\Omega'_j)$ and is zero elsewhere. The restriction of the action of K to $\mathcal{N}(\Omega'_j)$ is then defined to be the matrix $K_j : \mathcal{N}(\Omega'_j) \longrightarrow \mathcal{N}(\Omega)$ where

$$K_j = R_j K R_j^T.$$

The classical additive Schwarz preconditioner also includes a solve on a coarse grid. It is assumed that a coarse triangulation of the domain is also available and that $K_0 = R_0 K R_0^T$ is a restriction of K to the coarse mesh. Here R_0 is a projection from the fine mesh to the coarse mesh and is usually taken to be the piecewise linear interpolant on the coarse mesh. Finally the additive Schwarz preconditioner is then defined by

$$H_{AS}^{-1} = \sum_{j=0}^{s} R_j^T K_j^{-1} R_j,$$

and it is shown in [12] that the condition of $H_{AS}^{-1}K$ can be bounded by

$$\kappa(H_{AS}^{-1}K) \leq C\left(1 + \frac{h^*}{\delta}\right)^2,$$

where h^* is the maximum diameter in the coarse triangulation and δ is the minimum distance between $\partial\Omega_j$ and $\partial\Omega'_j$. C is a constant which is independent of the triangulation diameters but depends on k and grows as the ratio of the maximum and minimum values of k grows. Additive Schwarz methods for other types of finite elements for (4.5), (4.8) with $k = 1$ throughout the domain (Laplace's equation) are discussed in [35] and similar bounds on the condition number of the preconditioned system are found. The analysis in [33] shows that the condition of $H_{AS}^{-1}K$ does not really tell the full story however, and that H_{AS}^{-1} is a more effective preconditioner than $\kappa(H_{AS}^{-1}K)$ suggests. This is the subject of the next section.

4.5.3 Spectral dependence on k

The dependence of the spectrum of $H_{AS}^{-1}K$ on k is explained in [33] and some of the results therein are presented below. The results are found by comparing the spectrum of $H_{AS}^{-1}K$ with that of $H_{D}^{-1}K$. Such a comparison is possible since diagonal scaling is an extreme form of additive Schwarz preconditioning corresponding to a partition of the domain into subsets containing only one node, and no multilevel step. Hence the diagonal scaling operator can be expressed as a sum of inverses of K^{-1} restricted to subdomains containing only one node. It is shown in [33] that the eigenvalues of $H_{AS}^{-1}K$ satisfy

$$\lambda_j(H_D^{-1}K) \leq C_1\lambda_j(H_{AS}^{-1}K) \leq C_2, \ \ j = 1,\dots,m, \tag{4.37}$$

where C_1 and C_2 are independent of k (in 2 dimensions C_1 can be taken to be 4, see [11]) so that any lower bounds for $\lambda_j(H_D^{-1}K)$ will also be lower bounds for $\lambda_j(H_{AS}^{-1}K)$ (up to multiplication by a constant). It is then shown that the eigenvalues of $H_D^{-1}K$ can be bounded below, independently of k provided that, if k is piecewise constant on a finite number of open, disjoint polyhedral regions then the closure each of these regions has a non-empty intersection with the Dirichlet boundary. In this case, equation (4.37) implies that the eigenvalues of $H_{AS}^{-1}K$ are also bounded independently of k. In practice the Dirichlet boundary often doesn't intersect each of the level sets of k in the way described above, the domain depicted in Figure 4-3 is a typical case. Here $k(x) = 10^c$ in the dark region and $k(x) = 1$ in the light region. For large values of c the condition of $H_D^{-1}K$ grows as a single eigenvalue approaches zero. The reason for this is that in the limit as $c \longrightarrow \infty$, the nodes on the boundary of the dark region and their neighbouring nodes in the light region become disconnected due to the large jump in k. This results in an interior Dirichlet type boundary in the light region and a Neumann type boundary on the dark region. In [33] it is proved that if j corresponds to a node in the dark region and i corresponds to a node in the interior of the light region then the ij^{th} entry of $H_D^{-1}K$ satisfies

$$(H_D^{-\frac{1}{2}} K H_D^{-\frac{1}{2}})_{ij} = O(10^{-\frac{c}{2}}) \text{ as } c \longrightarrow \infty.$$

Otherwise $H_D^{-1}K$ acts like a scaled stiffness matrix when both i and j correspond to nodes in the same region. As there is no connection between the two regions as $c \longrightarrow \infty$, $(H_D^{-1}K)_{ij} \longrightarrow 0$ for ij corresponding to nodes in different regions, and so $H_D^{-1}K$ tends to a block diagonal structure (for a suitable nodal ordering) of two blocks corresponding to the light and dark regions. The block corresponding to the dark region is essentially a stiffness matrix for a purely Neumann problem which is known to have a single zero eigenvalue. This result generalises so that the number of eigenvalues approaching zero is equal to the number of dark regions which fail the boundary criterion above. To state the results of [33] in detail is technical but the main idea can be summed up by the following theorem.

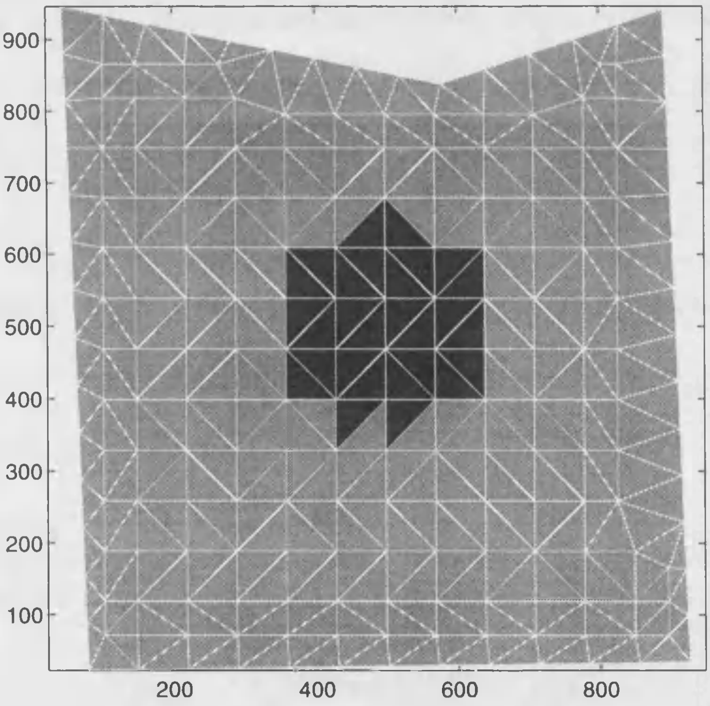


Figure 4-3: A triangulated domain with $k = 10^c$ in the central region and $k = 1$ elsewhere

Theorem 4.5.1 (Graham and Hagger) *Suppose that the domain Ω is subdivided into light and dark regions, and k is allowed to tend to ∞ in the dark regions. Then* *with K and H_D defined as above, the number of eigenvalues of $H_D^{-1}K$ which approach zero as $k \longrightarrow \infty$ is equal to the number of disjoint dark regions which have empty intersection with the Dirichlet boundary of Ω. The remaining eigenvalues of $H_D^{-1}K$ can be bounded independently of k.*

Theorem 4.5.1 may explain the comment made in [10], that diagonal scaling typically has the effect of leaving small eigenvalues for the types of coefficient matrix considered. The same theory indicates that for values of k approaching zero in the dark region, no eigenvalues will approach zero since, as $k \longrightarrow 0$ in the dark region, the disconnected dark region will correspond to a purely Dirichlet problem with no bad eigenvalues. This eigenvalue behaviour will also be true of $H_{AS}^{-1}K$ because of (4.37).

Small values of k are typical in groundwater flow applications where regions of low permeability are present in the interior of the domain. Such regions tend to act like obstacles in the domain which the fluid must flow around. The results in [33] are useful since they imply that the number of iterations required by LSQR(D^{-1}, H_{AS}^{-1}) should be independent of the number of such regions. Large interior values of k arise in highly permeable regions which act as a conduit for the flow. Here LSQR(D^{-1}, H_{AS}^{-1}) will obviously not perform independently of the number of these regions (or the value of k), although the performance will certainly not be as bad as $\kappa(H_{AS}^{-1}K)$ suggests since Theorem 4.5.1 implies that only a small number of eigenvalues are approaching zero, the rest remain bounded independently of k.

As was described in §2.2.1, estimates on the convergence of iterative methods, and hence a lower bound on the number of iterations required to reduce an error estimate by a given tolerance, are usually found by choosing an inclusion set for the eigenvalues upon which a suitably chosen polynomial can be made small. Then using the minimisation property of the algorithm in question, the error estimate can be bounded in terms of the uniform norm of the chosen polynomial on the inclusion set. By inserting the eigenvalues of $H_{AS}^{-1}K$ in an inclusion set of the union of points corresponding to eigenvalues approaching zero and an interval containing the remaining eigenvalues, such a polynomial argument will give a bound on, for example, the number of conjugate gradient iterations on $H_{AS}^{-1}K$ required to reduce the energy norm of the error to a given tolerance. This number is shown in [33] to grow linearly with $\log(k_{\max}/k_{\min})$. Since the LSQR(D^{-1}, H^{-1}) algorithm is equivalent to the preconditioned conjugate gradient

c	$\lambda_1(H_D^{-1}K)$	$\lambda_2(H_D^{-1}K)$	$\kappa(H_D^{-1}K)$	LSQR(D^{-1})	LSQR(D^{-1}, H_D^{-1})	CG	PCG(H_D^{-1})
12	1.1×10^{-13}	0.102	2.8×10^{13}	345	63	>400	64
11	1.1×10^{-12}	0.102	2.8×10^{12}	321	60	>400	60
10	1.1×10^{-11}	0.102	2.8×10^{11}	289	58	>400	58
9	1.1×10^{-10}	0.102	2.8×10^{10}	266	55	361	55
8	1.1×10^{-9}	0.102	2.8×10^{9}	238	52	305	52
7	1.1×10^{-8}	0.102	2.8×10^{8}	211	48	244	48
6	1.1×10^{-7}	0.102	2.8×10^{7}	184	46	206	46
5	1.1×10^{-6}	0.102	2.8×10^{6}	157	42	169	42
4	1.1×10^{-5}	0.102	2.8×10^{5}	131	40	135	40
3	1.1×10^{-4}	0.101	2.8×10^{4}	105	37	104	37
2	1.1×10^{-3}	0.101	2.8×10^{3}	76	33	76	33
1	9.2×10^{-3}	0.095	2.8×10^{2}	43	29	43	29
-1	0.042	0.067	50.1	35	28	35	28
-2	0.043	0.065	50.3	63	29	63	29
-3	0.043	0.065	50.5	79	29	79	29
-4	0.043	0.065	50.5	91	29	91	29
-5	0.043	0.065	50.5	106	29	103	29
-6	0.043	0.065	50.5	122	29	122	29
-7	0.043	0.065	50.5	136	29	136	29
-8	0.043	0.065	50.5	149	29	149	29
-9	0.043	0.065	50.5	162	29	165	29
-10	0.043	0.065	50.5	181	29	181	29
-11	0.043	0.065	50.5	193	29	193	29
-12	0.043	0.065	50.5	209	29	208	29

Table 4.1: Iteration counts for the unpreconditioned and diagonally preconditioned LSQR(D^{-1}) and CG methods for the problem described in Figure 4-3

method for K in exact arithmetic, this bound on the iteration number will also hold for LSQR(D^{-1}, H^{-1}) . A bound of this type will also hold for the Euclidean norm of the error, see [33] for more details.

By way of demonstration, the domain shown in Figure 4-3 has been discretised using Vavasis type elements on an unstructured grid. The value of $k = 10^c$ is taken in the central region, $k = 1$ is taken elsewhere and Dirichlet boundary conditions are assumed on the whole of $\partial\Omega$. There are 153 pressure unknowns and 716 velocity unknowns. The number of LSQR(D^{-1}, H_D^{-1}) and preconditioned conjugate gradient iterations required to reduce the norm of the Euclidean norm of the error by a factor of 10^4 are displayed in Table 4.1 together with the number of iterations required by unpreconditioned versions of both algorithms. Also shown are the two smallest eigenvalues of $H_D^{-1}K$. Since there is only one level set of k which does not intersect the Dirichlet boundary, only one eigenvalue approaches zero as c grows, as predicted in the theory above. Notice that for values of $c > 5$, the LSQR(D^{-1}) algorithm begins to take fewer iterations than the conjugate gradient algorithm to converge, with a discrepancy of over one hundred iterations for larger values of c. This would tend to suggest that the LSQR(D^{-1}) approach with Vavasis type elements is more stable than the standard primal formulation - conjugate gradient approach to solving the problem when k is large (although both methods require more steps than the number of pressure unknowns which is not desirable). This discrepancy is not as evident in the preconditioned versions of both algorithms since both algorithms are 'stabilised', to an extent, by the diagonal scaling. Only for large values of c is the LSQR(D^{-1}, H_D^{-1}) more competitive, for smaller values the convergence of both algorithms is the same. For values of c larger than 13 both algorithms cease to converge to the solution of (4.34), with PCG(H_D^{-1}) failing to converge to anything. The LSQR(D^{-1}, H_D^{-1}) algorithm has however been observed to converge to solutions which are close to the actual solution, for example in the case $c = 13$, $||B^T D^{-1}(b - By_{\text{LS}})|| = O(10^1)$, which should be compared to a residual of $O(10^{13})$ after 400 preconditioned conjugate gradient iterations. This is disappointing but not entirely unexpected as the conjugate gradient algorithm is well known to fail to converge for problems with large ratios of k and a 'stable approach' to solving the system must be taken. Such stable approaches are discussed in [78, 79, 38]. These algorithms, although attractive for their stability properties, are not particularly appropriate for the large systems which arise in CFD applications since their first step invariably is to calculate a basis for the nullspace of $B^T D^{-\frac{1}{2}}$ (usually via a QR factorisation of $B^T D^{-\frac{1}{2}}$ or a related matrix) which, for anything but small systems, is unrealistic in terms of computation time and storage requirements (c.f. the ΠCG algorithm of Appendix B). The iteration count for LSQR(D^{-1}, H_D^{-1}) is plotted against $\log(k_{\max}/k_{\min})$ in Figure 4-4. The linear relationship between $\log(k_{\max}/k_{\min})$ and iteration number expected can be clearly seen.

4.6 Incomplete LU preconditioners

If it were possible to compute the LU decomposition of the coefficient matrix K then, setting $H = K$ and performing the LU decomposition to enable the action of H to be computed would clearly be an effective method of preconditioning in terms of the number of preconditioned iterations. This approach is not practical for large sparse

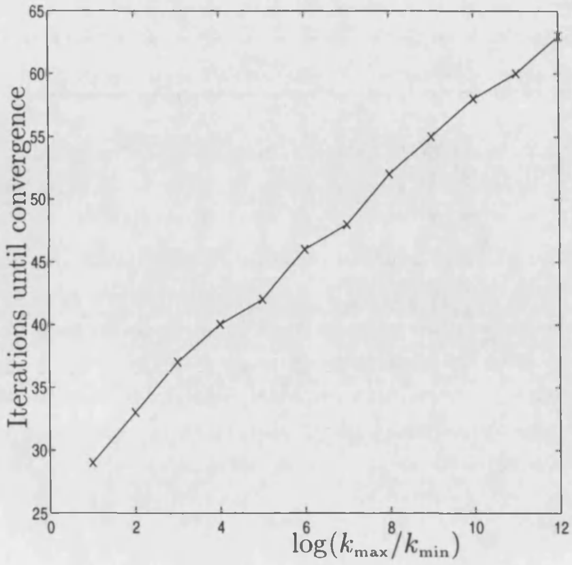


Figure 4-4: Iteration count of LSQR(D^{-1}, H_D^{-1}) against ratio of k values

matrices K, since the cost of the LU factorisation is prohibitive and since the LU factors are, in general, less sparse than K (L and U are usually full in the bandwidth of K - see Figure 4-5) so that the backward and forward solves when applying the preconditioner are expensive. Instead, if it were possible to calculate approximations $\tilde{L}$ and $\tilde{U}$ to L and U, which have a similar sparsity structure to K, in a cheap and efficient manner, then solves with $\tilde{L}$ and $\tilde{U}$ would be cheap to perform and it might be hoped that $H = \tilde{L}\tilde{U}$ would be a good preconditioner for K. The factorisation $H = \tilde{L}\tilde{U}$ alluded to above is referred to as an *incomplete factorisation* since it is assumed that

$$\tilde{L}\tilde{U} \approx LU = K.$$

Hereafter it will be assumed that all factorisation matrices are incomplete and the tilde notation will be dropped.

Stone [75] was one of the first people to consider incomplete factorisations of matrices. Here matrices L and U are found which are almost as sparse as K and which approximate the LU factorisation as outlined above. In [75] the matrices are used in an iteration called the strongly implicit procedure (SIP) which is similar to the Gauss-Seidel iterative method. A generalisation of the SIP method using the incomplete LU factorisation ILU(p) of Watts [86] is given in [50]. ILU(p) denotes the incomplete LU factorisation with p levels of fill in, higher values of p lead to less sparsity in L, see figures 4-6 and 4-7. (The factorisations of the matrix K represented in figure 4-5 were generated using the ILU(p) code in [50], the sparsity pattern of the full Cholesky decomposition is also shown in Figure 4-5). Using ILU(p), the sparsity pattern of L and U is not easy to predict (unless $p = 0$) and the amount of work required to perform the factorisation usually grows with p. This unpredictability makes the use of fill in methods impractical. A comparison of SIP and incomplete Cholesky preconditioned conjugate gradient (ICCG) methods for groundwater flow problems is given in [45]. Here it is found that the ICCG iterations are cheaper to perform than the SIP iterations and that ICCG converges faster than SIP on most of the test problems presented.

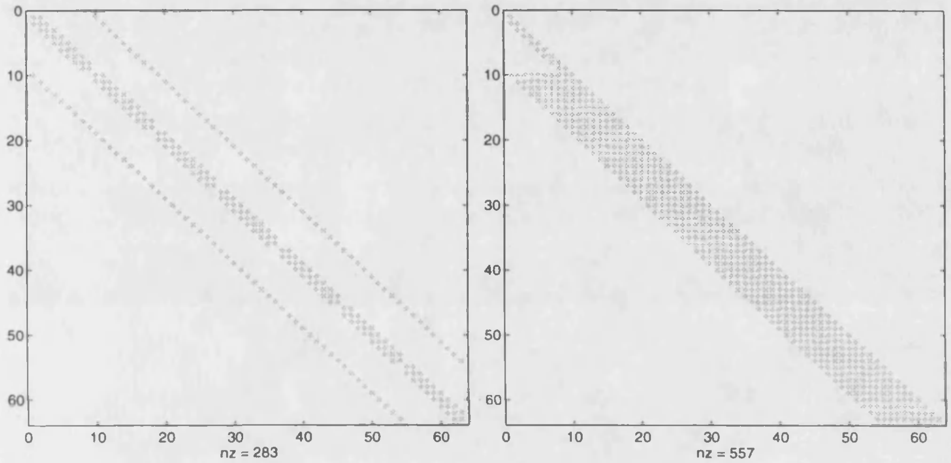


Figure 4-5: Sparsity pattern of K (left) and the full Cholesky factorisation of K (right)

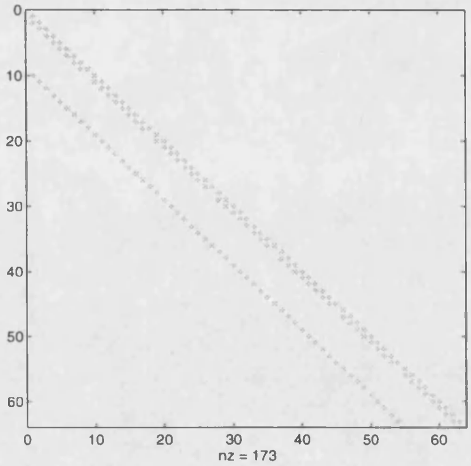


Figure 4-6: L computed with ILU(0)

Incomplete LU factorisations are not guaranteed to exist for all matrices and so care

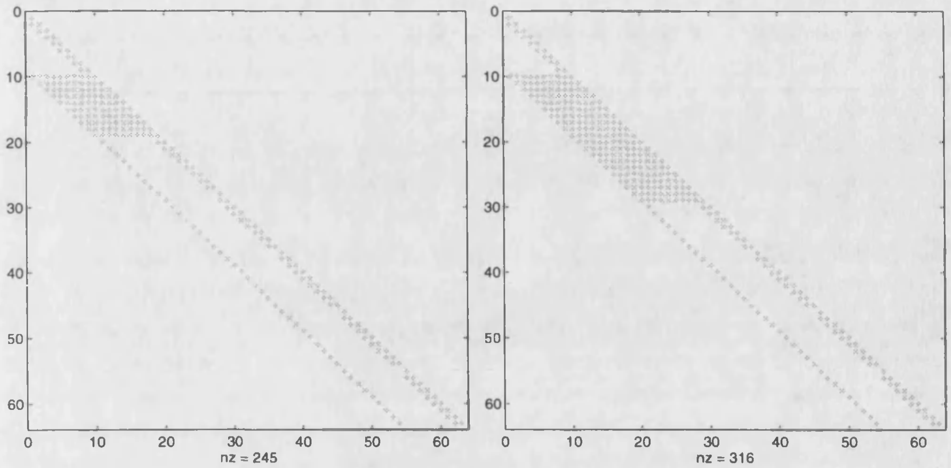


Figure 4-7: L computed with ILU(1000)(left) and ILU(100000) (right)

must be taken when using the ILU(p) algorithm. One situation in which incomplete LU factorisations always exist is that when the matrix K is an M-matrix, this case is presented in [51].

Definition 4.6.1 *A matrix A is said to be an* M-matrix *if $A = (a_{ij})$ where $a_{ij} \leq 0 \ \forall i \neq j$ and*

$$Ax = y \text{ and } y \geq 0 \Longrightarrow x \geq 0. \tag{4.38}$$

(In the case that (4.38) holds the notation $A^{-1} \geq 0$ is used).

The approach taken in [51] is to define the desired sparsity pattern for L and U *a-priori*, which is different to ILU(p) where the final sparsity pattern is unknown until the factorisation is completed. This is achieved by first choosing the set $P_S \subset P$, where

$$P = \{(i, j) \mid i \neq j\}, \tag{4.39}$$

to contain the required sparsity pattern, i.e. entries $(i, j) \in P_S$ will denote positions in which zero entries are desired in the matrices L and U. It is shown in [51] that for any M-matrix A and any subset P_s of P, there are unique matrices L, U and R such that

$$A = LU + R,$$

where L and U are lower and upper triangular respectively and have zeros at all entries contained in P_S, and the residual matrix $R = (r_{ij})$ satisfies $r_{ij} = 0$ for $(i, j) \notin P_S$. Furthermore the splitting is convergent (in the sense of §2.1), that is $[(LU)^{-1}R]^k \longrightarrow 0$ as $k \longrightarrow \infty$. Therefore an iteration of the form

$$LUy_{i+1} = -Ry_i + b, \tag{4.40}$$

is guaranteed to converge to the solution of $Ay = b$ for any choice of y_0. See [77] for more details. Of course when A is a symmetric, positive definite M-matrix the notion of an incomplete Cholesky factorisation arises naturally from the incomplete LU factorisation.

An incomplete LU factorisation can be constructed in many ways, the approach given in [51] is by performing Gaussian elimination on A and subtracting away terms which lie in P_S in the row and column of the current pivot after each column elimination. These subtracted terms are accumulated in the matrix R and U is the result upon completion of the process. As usual, L is the inverse of the product of the column elimination matrices. Using some elementary M-matrix results (see [77, Section 3.5]) it is possible to show that this process of generating L and U is at least as stable as standard Gaussian elimination applied to A (see [51]). The approach taken in the ILU(p) algorithm is to fill in the ILU(0) Cholesky factor from top-left to bottom-right until the fill in tolerance is exceeded. Here ILU(0) denotes the incomplete LU factorisation of K where L and U have the same sparsity pattern as K. This is referred to in [51] as the incomplete LU factorisation of K with no extra diagonals. Algorithms for ILU(0) and incomplete LU with three extra diagonals are given in [51] for matrices K arising in discretisations on uniform grids.

Although the application to standard iterative methods of the form (4.40) is useful, incomplete factorisations have been more widely used as preconditioners for iterative methods, especially the incomplete Cholesky factors together with the conjugate gradient iteration. This is because, for highly sparse choices of L, the incomplete factor is cheap to operate with and compute. Also, no assumptions are made on the geometry of the underlying problem to be solved, the coefficient matrix A is simply fed to the incomplete factorisation algorithm. This is both an advantage and a drawback, the approach is simple to use but the lack of dependence on the initial problem often leads to poor numerical performance.

It should be noted that M-matrices are not the only class of matrices for which incomplete factorisations exist. Classes of positive definite matrices for which incomplete LU factorisations exist are described in [48]. Here incomplete factorisations of *H-matrices* ($K = (k_{ij})$ is an H-matrix if $\widetilde{K} = (\widetilde{k}_{ij})$ is an M-matrix where $\widetilde{k}_{ii} = k_{ii}$ and $\widetilde{k}_{ij} = -|k_{ij}|$) are discussed, and it is shown that incomplete factorisations exist for H-matrices with positive diagonal. This motivates Manteuffel's ***shifted incomplete Cholesky factorisation*** (SIC). Since any diagonally dominant matrix is an H-matrix, and since any positive definite matrix can be transformed into a diagonally dominant matrix by scaling its off-diagonal terms until they are suitably small, the SIC approach is to apply an incomplete factorisation to

$$K(\alpha) = D + \frac{1}{1+\alpha}(F+G),$$

where F is the strictly lower and G the strictly upper triangular parts of K. As $K(\alpha)$ is diagonally dominant for α sufficiently large, an incomplete LU factorisation of $K(\alpha)$ is guaranteed to exist. The idea of SIC is then to use the incomplete factor of $K(\alpha)$ to precondition K. This allows any positive definite matrix to be preconditioned by an incomplete factor of a matrix which is close to the original matrix (provided that α is not too large), and hence is applicable to most matrices arising in the application of the finite element methods. The choice of α is obviously important and can greatly affect the number of iterations required see [48, Section 6(c)].

Another area of current interest is the choice of the sparsity pattern P_S, since this can be a deciding factor in whether an incomplete factorisation exists. Some necessary and sufficient conditions on P_S for the existence of incomplete Cholesky factors of symmetric positive definite matrices can be found in [81] and the references therein.

Since it is assumed here that the matrix $K = B^T D^{-1} B$ arises in the discretisation of (4.3) using Vavasis type elements, and $K = (k_{ij})$ where

$$k_{ij} = \int_\Omega k(x) \nabla\varphi_i \cdot \nabla\varphi_j \, dx \leq 0 \text{ for } i \neq j,$$

provided that no triangle contains an angle greater than 90°, since $\nabla\varphi_i$ is always decreasing (increasing) in a direction in which $\nabla\varphi_j$ is increasing (decreasing) when $i \neq j$ and $\nabla\varphi_i \cdot \nabla\varphi_j \neq 0$, for the piecewise linear elements considered. This observation leads to the conclusion [77, §3.5 Cor. 3.5] that the symmetric positive definite matrix K is an M-matrix so that an incomplete Cholesky factorisation of K exists and the straightforward procedure in [51] for computing the factorisation is stable. For more general finite elements and meshes the approach of Manteuffel [48] is more suitable.

4.7 Numerical results and a note on incomplete preconditioners

Here some problems arising from discretisations of the groundwater flow problem (4.3), with boundary conditions (4.4) in 2 dimensional domains by Vavasis type elements are examined, and the performance of the three preconditioning strategies outlined in §§4.5,4.6 is compared.

Example 1

The first example concerns the domain shown in figure 4-1. Here $k(x) = 10^c$ in the dark regions and $k(x) = 1$ elsewhere. The vertical sides of Ω comprise the no-flow boundary Γ_2, and the pressure is prescribed on the Dirichlet boundaries on the top and bottom of the domain. Discretisations on 8×8 grids (63 pressure and 256 velocity unknowns), 16×16 grids (255 pressure and 1024 velocity unknowns) and 64×64 grids (4095 pressure and 16384 velocity unknowns) were performed. The resulting systems were solved using unpreconditioned, diagonally scaled, additive Schwarz preconditioned and ILU(0) (calculated using the algorithm in [51]) preconditioned versions of LSQR(D^{-1}). The Schwarz preconditioners use the subdivision of Ω shown in figure 4-8 for the 8×8 and 16×16 problems. On the 64×64 grid problems, the cost of calculating each K_j^{-1} on each of the four subdomains in figure 4-8 becomes prohibitive and so the division of Ω into fifteen smaller regions as shown in figure 4-9 was used. The additive Schwarz preconditioner was calculated without a coarse grid solve, the improvement which could be expected with the addition of a coarse grid solve can be guessed by comparison with the results in [34, 33]. In the case that the coarse grid is designed to replicate some of the geometry of the domain (rather than being simply a coarse uniform mesh) the best results would be expected and in this case only a weak dependence on the coefficient ratios is expected (see the conjecture in the results of [34], and the following remarks), i.e. the number of iterations taken for large values of c are likely to be only slightly

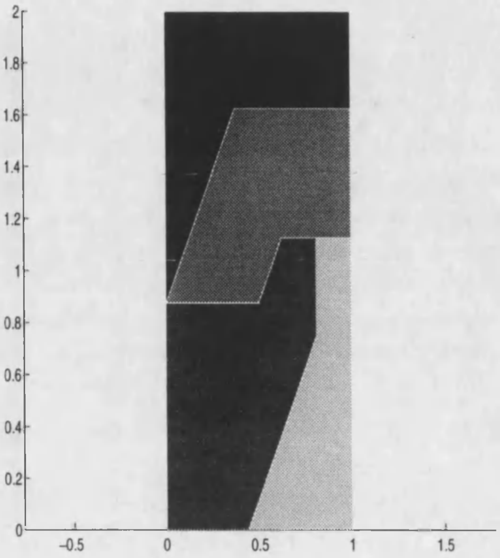

2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-0.5
0
0.5
1
1.5

Figure 4-8: Decomposition of the domain Ω

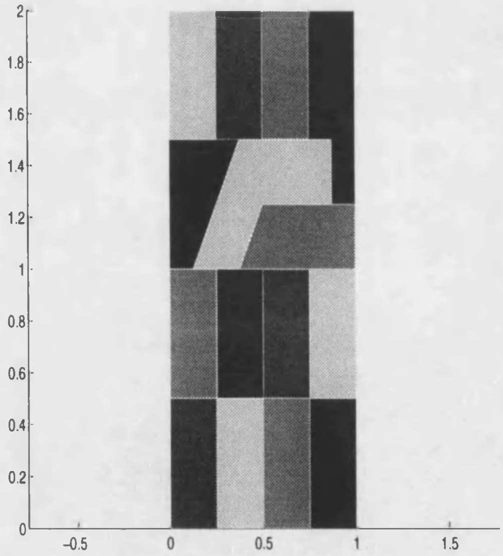

2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-0.5
0
0.5
1
1.5

Figure 4-9: Finer decomposition of the domain Ω

larger than the number of iterations for the case $c = 1$. No coarse mesh was used here since this work seeks only to compare the three preconditioning techniques mentioned and the result is clear without needing the coarse grid solve.

Each of the solution methods outlined above was carried out until the stopping criterion

$$\frac{||y - y_0||}{||y - y_j||} \geq 10^4,$$

where y is the exact solution of (4.34) (obtained by Gaussian elimination), was obtained. Iteration counts for each of the methods are shown in Table 4.2. As was previously mentioned, positive values of c are typical of problems in which a highly permeable conduit is present, and negative values of c are usual when the dark regions represent obstacles of low permeability (which is the case shown in Figure 4-1).

For negative values of c it is clear that the unpreconditioned LSQR(D^{-1}) algorithm struggles to solve the problems as k tends to zero. This is as expected since the unpreconditioned system has several eigenvalues which tend to infinity as k decreases (this is obvious by consideration of (4.27) and Theorem 2.3.1). The behaviour of the preconditioned algorithms is much nicer, each of the three preconditioning strategies have iteration counts which are much smaller than those of LSQR(D^{-1}) . The number of iterations required by LSQR(D^{-1}, H_D^{-1}) grows slightly as as k decreases, whereas LSQR(D^{-1}, H_{AS}^{-1}) requires the same number of iterations for all values of k between -2 and -10 on both of the two smaller systems. The ILU(0) preconditioner also performs independently of k (for small values of k) on the two small systems, and LSQR(D^{-1}, H_{ILU}^{-1}) requires fewer iterations than LSQR(D^{-1}, H_{AS}^{-1}) on the two smaller systems. The story is different for the 64×64 system however, here the LSQR(D^{-1}, H_{AS}^{-1}) algorithm is the fastest although again LSQR(D^{-1}, H_{ILU}^{-1}) has iteration counts which are independent of k for $c < -2$. The growth in the number of iterations required as h is reduced also indicates that these preconditioning strategies are dependent on mesh size.

For positive values of c the expected growth in iteration numbers can be seen for the diagonally preconditioned system. For the small systems the H_{AS}^{-1} preconditioned system appears to have iteration number independent of k although on the larger system the expected behaviour can be seen. The independence of k of LSQR(D^{-1}, H_{AS}^{-1})

c	LSQR(D^{-1})			LSQR(D^{-1}, H_D^{-1})			LSQR(D^{-1}, H_{AS}^{-1})			LSQR(D^{-1}, H_{ILU}^{-1})		
	8 × 8	16 × 16	64 × 64	8 × 8	16 × 16	64 × 64	8 × 8	16 × 16	64 × 64	8 × 8	16 × 16	64 × 64
10	376	>1000	>1000	49	108	500	7	13	102	20	35	134
9	342	>1000	>1000	49	104	429	7	13	98	20	34	118
8	306	>1000	>1000	47	100	416	7	13	95	19	32	113
7	269	>1000	>1000	45	95	397	7	13	89	18	30	107
6	234	>1000	>1000	44	92	382	7	13	84	16	29	103
5	196	863	>1000	42	88	367	7	13	81	14	27	98
4	150	649	>1000	39	82	339	7	13	77	13	25	90
3	120	443	>1000	37	77	317	7	13	71	12	23	83
2	82	247	>1000	35	70	291	8	14	65	11	20	77
1	48	122	554	31	62	256	11	17	58	10	17	69
-1	43	94	383	33	69	283	20	27	59	11	20	76
-2	65	117	778	35	73	301	21	29	64	11	20	81
-3	72	277	>1000	36	74	313	21	29	65	11	20	82
-4	78	395	>1000	38	74	316	21	29	65	11	20	82
-5	112	507	>1000	38	78	316	21	29	65	11	20	82
-6	127	604	>1000	38	79	316	21	29	66	11	20	82
-7	136	722	>1000	38	80	318	21	29	66	11	20	82
-8	150	831	>1000	38	80	328	21	29	66	11	20	82
-9	175	938	>1000	38	80	334	21	29	66	11	20	82
-10	189	>1000	>1000	38	80	336	21	29	66	11	20	82

Table 4.2: Iteration counts for Example 1

on the small systems is probably due to the fact that the Schwarz preconditioner used on the small systems is fairly 'strong', in the sense that Ω is partitioned into only four subdomains which closely replicate the geometry of Ω, and so it might be expected that H_{AS}^{-1} is a very good approximation to K^{-1}, whereas the splitting of Ω into 15 subdomains for the large problem leads to a poorer approximation of K^{-1}. This behaviour is in agreement with the conjecture at the end of [34] mentioned previously, namely that for a well designed mesh which resembles the geometry of the original problem, the number of additive Schwarz preconditioned steps is only very mildly dependent on k. It can also be seen that LSQR(D^{-1}, H_{ILU}^{-1}) exhibits the same type of growth as is expected of LSQR(D^{-1}, H_D^{-1}) and LSQR(D^{-1}, H_{AS}^{-1}) , which would tend to suggest that the eigenvalue distribution of $H_{ILU}^{-1}K$ is similar to that of $H_D^{-1}K$. This observation is partially explained by Theorem 4.7.2, but first a preliminary result due to Manteuffel is required.

Suppose M is a symmetric positive definite matrix with splitting

$$M = D - B,$$

where D is the diagonal part of M and B is the off-diagonal, and let

$$M(\alpha) = D - \frac{1}{1+\alpha}B.$$

Recall (§4.6) that for sufficiently large α, $M(\alpha)$ is diagonally dominant and hence is an H-matrix with positive diagonal, so that an incomplete Cholesky factorisation is guaranteed to exist [48]. This incomplete factorisation is called a shifted incomplete factorisation of M and is denoted $H_{SIC}(\alpha)$ (so that $H_{SIC}(\alpha) = LL^T$ for some L with the desired sparsity pattern), and so

$$M(\alpha) = H_{SIC}(\alpha) + R(\alpha)$$

where non-zero entries in R only occur in some non-zero set P_S. It will be assumed hereonin that P_S has empty intersection with the non-zero set of M. Notice that,

$$\lim_{\alpha \longrightarrow \infty} H_{SIC}(\alpha) = D.$$

We are now in a position to state [48, Theorem 5.1].

Theorem 4.7.1 (Manteuffel) *With the above notation,* $\exists\alpha_{min} > 0$ *such that* $\forall\alpha > \alpha_{min}$,

$$\kappa(H_{SIC}(\alpha)^{-1}M) < \kappa(D^{-1}M).$$

Further if $\lambda_i(H_{SIC}(\alpha)^{-1}M)$ *is the* i^{th} *eigenvalue of* $H_{SIC}(\alpha)^{-1}M$ *and* $\lambda_i(D^{-1}M)$ *the corresponding* i^{th} *eigenvalue of* $D^{-1}M$*, then for* α *sufficiently large either*

$$1 \leq \lambda_i(H_{SIC}(\alpha)^{-1}M) < \lambda_i(D^{-1}M),$$

or

$$\lambda_i(D^{-1}M) < \lambda_i(H_{SIC}(\alpha)^{-1}M) < 1.$$

The following theorem then follows naturally

Theorem 4.7.2 *If* K *is the stiffness matrix arising in (4.24) then for sufficiently large* α*, either*

$$1 \leq \lambda_i(H_{SIC}(\alpha)^{-1}M) < C_2,$$

or

$$\lambda_i(D^{-1}M) < \lambda_i(H_{SIC}(\alpha)^{-1}M) < 1,$$

where C_2 *is a constant.*

Proof Trivial consequence of Theorem 4.7.1 and [11, Theorem 12]. □

Corollary 4.7.3 *The number of eigenvalues of* $H_{SIC}(\alpha)^{-1}K$ *that approach zero is no greater than the number of eigenvalues of* $H_D^{-1}K$ *and* $H_{AS}^{-1}K$ *that approach zero (which can be predicted by the theory in [33]) whenever* α *is sufficiently large. Further the remaining eigenvalues that are bounded away from zero can be bounded above by a constant.*

Using Theorem 4.7.2 and Corollary 4.7.3 it is now obvious that the shifted incomplete Cholesky factorisation preconditioners should exhibit similar behaviour to the diagonal and additive Schwarz preconditioners for a sufficiently large shift parameter. However, no lower bound on the value of the shift parameter is known. For the results in this section, the (non-shifted) incomplete Cholesky factorisation has been used, so that $\alpha = 0$. The experiments have tended to indicate strongly that the results of Theorem 4.7.2 still hold, but no analytical justification why this should be true has been found. For example Figure 4-10 shows the spectrum of $H_D^{-1}K$ and $H_{ILU}^{-1}K$ for the domain shown in Figure 4-11 with $k(x) = 10^{10}$ in the dark region and $k(x) = 1$ in the light region. The spectrum of the incomplete Cholesky preconditioned system can be seen to be very similar to that of $H_D^{-1}K$ with only one small eigenvalue and the remainder bounded away from zero.

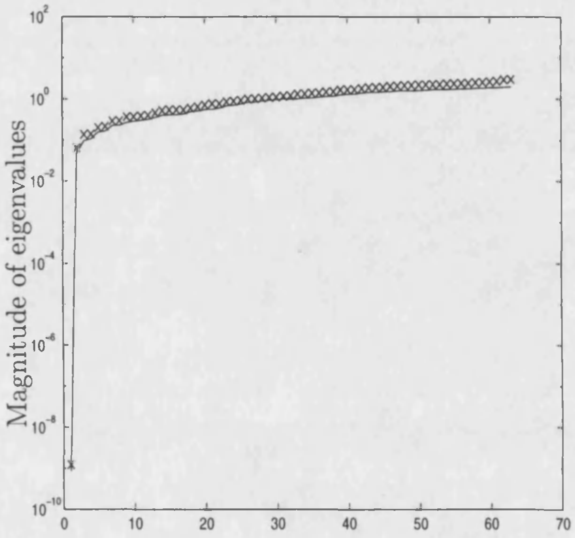


Figure 4-10: Eigenvalues of $H_D^{-1}K$ (points joined by solid line) and $H_{ILU}^{-1}K$ (crosses).

Example 2

The next example relates to the domain shown in figures 4-11 and 4-12. The domain is similar to that of the previous example, except for the fact that the dark region intersecting with the Dirichlet boundary is removed and the remaining dark region no longer intersects the (Neumann) boundary of Ω. Figure 4-11 indicates the type of flow expected for large values of k in the dark region. Here flow tends to be through the dark region which is highly permeable. Figure 4-12 shows a typical groundwater flow around a region which has a small value of k.

The same size meshes used in Example 1 were used to discretise the domain and the resulting systems were again solved by LSQR(D^{-1}) , LSQR(D^{-1}, H_D^{-1}) , LSQR(D^{-1}, H_{AS}^{-1}) and LSQR(D^{-1}, H_{ILU}^{-1}) . The same subdomains as in Example 1 were used to form the additive Schwarz preconditioners, and the same stopping criterion was used to terminate the iteration. The iteration counts for each method are shown in Table 4.3.

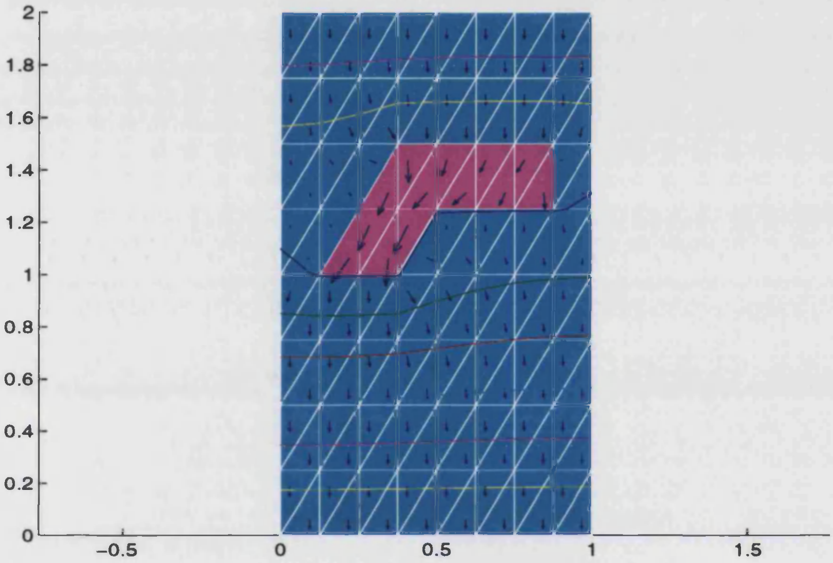


Figure 4-11: Flow and pressure contour plots for k large in the dark region

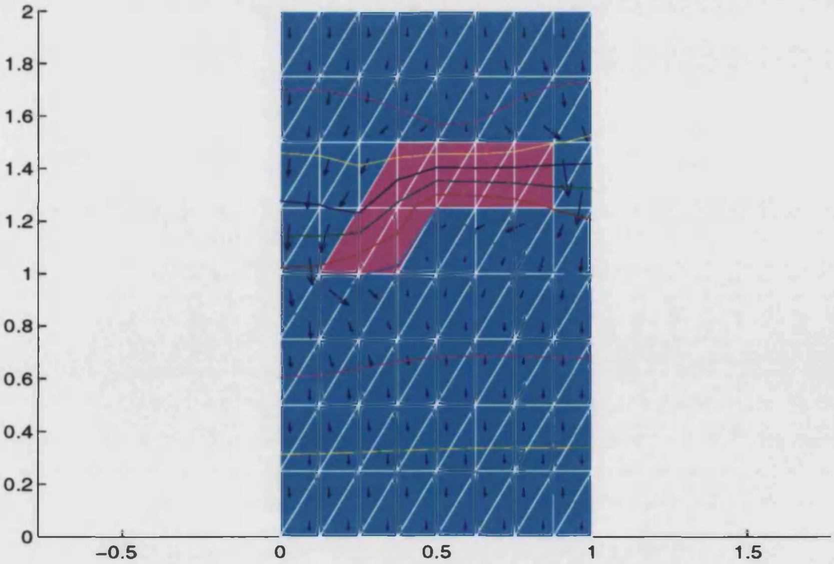


Figure 4-12: Flow and pressure contour plots for k small in the dark region

c	LSQR(D^{-1})			LSQR(D^{-1}, H_D^{-1})			LSQR(D^{-1}, H_{AS}^{-1})			LSQR(D^{-1}, H_{ILU}^{-1})		
	8 × 8	16 × 16	64 × 64	8 × 8	16 × 16	64 × 64	8 × 8	16 × 16	64 × 64	8 × 8	16 × 16	64 × 64
10	219	>1000	>1000	50	101	419	13	18	95	18	33	117
9	198	980	>1000	48	98	402	13	18	90	17	31	113
8	183	871	>1000	46	95	385	13	18	86	16	29	108
7	159	759	>1000	44	90	368	13	18	82	15	28	102
6	142	652	>1000	41	85	346	13	18	77	14	26	98
5	120	552	>1000	39	80	327	13	18	73	13	25	92
4	102	444	>1000	36	74	299	13	18	68	12	23	85
3	82	330	>1000	33	67	276	13	18	62	11	21	79
2	62	210	>1000	31	61	253	13	18	59	10	19	73
1	36	102	558	30	60	241	14	18	53	10	18	70
-1	34	71	287	29	55	228	17	24	56	10	19	70
-2	41	125	509	30	57	241	18	25	56	10	19	70
-3	46	196	>1000	31	57	243	18	25	56	10	19	70
-4	50	274	>1000	31	57	243	18	25	56	10	19	70
-5	54	312	>1000	31	57	243	18	25	56	10	19	70
-6	58	383	>1000	31	57	243	18	25	56	10	19	70
-7	61	442	>1000	31	57	243	18	25	56	10	19	70
-8	64	509	>1000	31	57	243	18	25	56	10	19	70
-9	67	576	>1000	31	57	243	18	25	56	10	19	70
-10	68	623	>1000	31	57	243	18	25	56	10	19	70

Table 4.3: Iteration counts for Example 2

The unpreconditioned algorithm doesn't struggle so much to solve these problems as in Example 1. This is probably due to the fact that fewer eigenvalues are tending to zero when k is large, and infinity when k is small, since the area of the dark region is smaller than in Example 1. However it is still the case that for very large and very small values of k in the dark region, the number of iterations is greater than the size of the system. The expected growth in iteration number for positive c in the diagonally scaled algorithm can be seen. For $k > 1$, the number of iterations of LSQR(D^{-1}, H_D^{-1}) is approximately the same as those for Example 1 which is as expected since the number of eigenvalues approaching zero is the same. For small values of k this problem is easier to solve than Example 1 (in terms of iterations). This is most likely due to the clustering of the eigenvalues in the diagonally scaled systems since the number of bad eigenvalues for both examples is zero when k is small. This is also true of the additive Schwarz preconditioned system for small k, although for large k Example 2 appears harder to solve than Example 1 for the small system sizes. This is perhaps because the subdomains for the small problems more closely resemble the geometry of Example 1 than Example 2. Further evidence for this is that for the large system where the subdomains better represent the geometry of Example 2, Example 2 appears easier to solve. Hence the choice of subdomains can make a considerable difference when using the additive Schwarz preconditioners. For the small systems, the ILU(0) preconditioned system behaves almost exactly the same as for Example 1, although for the large system, Example 1 appears a slightly harder problem.

The iteration counts for each of the three preconditioned systems are plotted against $\log(k_{\max}/k_{\min})$ in figure 4-13. A linear relation is clearly visible for each of the three preconditioned systems which agrees with the remarks made at the end of Example 1.

4.8 Summary

The clear conclusion of this section is that the additive Schwarz preconditioner is the most effective of the three methods tried here in terms of the number of iterations required by LSQR(D^{-1}) applied to the system (4.34) where the system arises from discretisations of groundwater flow type problems by Vavasis type finite elements. The condition number of the preconditioned systems was observed to be a poor indicator of the numerical performance which could be expected, since only a small number of

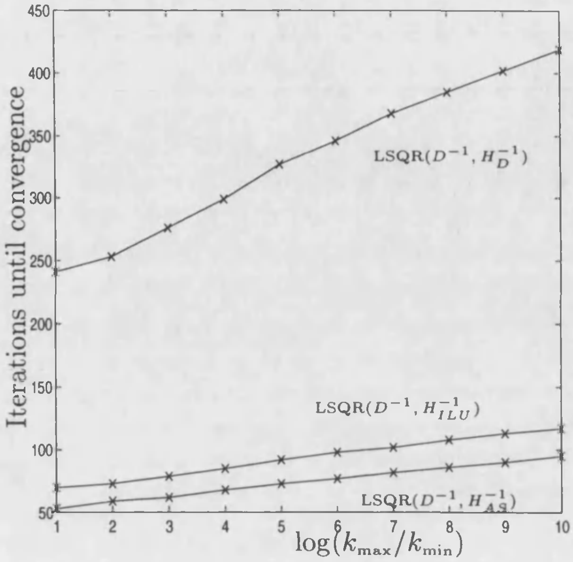


Figure 4-13: Iteration count of LSQR(D^{-1}, H_D^{-1}) , LSQR(D^{-1}, H_{AS}^{-1}) and LSQR(D^{-1}, H_{ILU}^{-1}) against ratio of k values.

eigenvalues approach zero for the case when k is large in regions in Ω which have empty intersection with the Dirichlet boundary, although in the case that k is small the condition number of the system is a more reliable bound since in this case there are no isolated eigenvalues. The eigenvalue distribution of the shifted incomplete Cholesky preconditioned system was shown to have similar dependence on k to that of the diagonally preconditioned system (and hence to that of the additive Schwarz preconditioned system) in Theorem 4.7.1, and similar properties were observed for the (non shifted) incomplete Cholesky preconditioner. Depending on the machine architecture it may well be the case that the ILU preconditioner is the more effective of the two in terms of time taken, since the cost of applying the preconditioner may be smaller. The ILU preconditioner requires two sparse triangular backsolves whereas the additive Schwarz requires several solves on smaller subdomains. It is impossible to predict which preconditioner will be more effective in a general setting, but on the serial machine used for the examples in this chapter the ILU iteration was usually the fastest.

It should be mentioned that at no time has the expense of calculating the preconditioner been considered. Obviously the diagonal preconditioner is the easiest to compute, but the distinction between calculating the additive Schwarz and ILU preconditioners is not clear. An advantage of calculating the additive Schwarz preconditioner is that it is not actually necessary to form the matrix K, whereas in computation of the ILU preconditioner specific elements of K need to be accessed. Experimentally it has been found that, provided the subdomains of Ω are suitably small there is little difference in the time taken to form each of the preconditioners. Another advantage of the additive Schwarz preconditioner is that the procedure parallelises naturally, whereas parallel implementations of ILU are not so simple or well understood. ILU has the advantage however when the geometry of the underlying problem is unknown since then it is more difficult to design an effective additive Schwarz preconditioner. The ILU preconditioner should give similar performance to that of an additive Schwarz preconditioner which replicates the geometry of the underlying problem if the behaviour observed in the experiments of this section is typical.

In [83, 84], element by element (EBE) preconditioners for discretisations of Laplace's equation using piecewise bilinear (2-dimensions) and piecewise trilinear (3-dimensions) elements are discussed and it is shown that EBE preconditioned systems are spectrally equivalent to H_D^{-1} preconditioned systems. Hence it is expected that the EBE preconditioners would exhibit similar behaviour to the three preconditioners discussed above. For 2-dimensional problems [83] advocates the use of incomplete factorisation preconditioners since they have a more rapid convergence rate, but for 3-dimensional problems the EBE method is more competitive. The fact that the incomplete factorisations have more rapid convergence than the EBE method indicates that the additive Schwarz method should also be more competitive than the EBE method in 2 dimensions. In three dimensions it may be the case that the EBE method will be the more competitive.

Chapter 5

Mixed finite elements for the groundwater flow equations

In this chapter the theory of mixed finite element methods for the groundwater flow equations is reviewed. It was seen in Chapter 4 that the velocity variable in the groundwater flow equations could be eliminated, so that standard finite element methods applied to the scaled Laplace equation could be used to provide an approximation to the pressure component of the solution of the groundwater flow equations. An approximation to the Darcy velocity component of the solution could then be obtained using some numerical differentiation, since the Darcy velocity is simply the scaled derivative of the pressure. The velocity approximation obtained in this way will always be of lower order than the pressure approximation, and no control over the continuity of the approximate velocity approximation was possible. Mixed finite element methods address this shortcoming by using more than one approximation space. In the case of the groundwater flow equations two approximation spaces are used, one for the velocity component of the solution and the other for the pressure component. Typically, accurate approximations to velocity are more desirable than accurate approximations to pressure, and so the velocity approximation space is usually chosen to be larger than the pressure approximation space. These spaces have to be chosen carefully so that the solution approximation converges to the actual solution as the mesh size decreases to zero.

In §5.1 the abstract mixed variational form of the groundwater flow equations is studied, and in §5.2 the discretised variational form is considered. One popular choice of mixed finite element for the groundwater flow equations is the Raviart-Thomas element, and this is the subject of §5.3. Since the LSQR(A^{-1}) method requires a solve with A^{-1} at each step, a good preconditioner for A is required so that the inner solves can be performed effectively. For mixed finite element approximations to the groundwater flow equations, A is a scaled mass matrix of velocity basis functions. Wathen [82] has considered preconditioning A by its diagonal, and in §5.4 it will be seen that the resulting preconditioned matrix is independent of the bad scaling due to the permeability function in the groundwater flow equations. In §5.5, attention returns to the computation of the Raviart-Thomas mixed finite element matrices and in particular, computation of velocity mass matrices. Following §5.4 the diagonally preconditioned scaled mass matrix will have condition which is independent of the permeability scaling. However due to the geometry of typical groundwater flow domains, it can be expected that domain elements with large aspect ratios will be present, the effects of which are discussed in §5.5.1 for the particular choice of the Raviart-Thomas elements. It is seen that although existing theory predicts that large aspect ratios will give rise to poorly conditioned mass matrices, the situation for Raviart-Thomas elements is not nearly as bad as expected. In §5.6 the choice of preconditioner H in the LSQR(A^{-1}, H^{-1}) algorithm is considered. This draws on ideas from Chapter 4 and §5.4, and the preconditioned algorithm is applied to one of the Harwell test problems. A disadvantage with the LSQR(A^{-1}) algorithms is that they require an A^{-1} operation at every iteration. A method which cuts down the number of A^{-1} operations by first solving a system with A replaced by its diagonal is considered in §5.7, and some surprising results for a MAC finite element discretised groundwater flow problem are given.

5.1 Abstract mixed variational formulation of the groundwater flow equations

The analysis of the variational form of groundwater flow type partial differential equations is abundant in the literature, and a similar treatment to that in the following sections can be found in any of [7, 9, 60, 61], all of which cite the original work of

Brezzi [8] on the subject. Again consider the groundwater flow equations,

$$\begin{aligned} \tfrac{\mu}{k}u + \nabla p &= 0 \text{ in } \Omega, \\ \nabla \cdot u &= 0 \text{ in } \Omega, \end{aligned} \tag{5.1}$$

with the boundary conditions

$$\begin{aligned} p &= f \text{ on } \Gamma_1, \\ u \cdot n &= 0 \text{ on } \Gamma_2 \end{aligned} \tag{5.2}$$

where $\Omega \subset \mathbb{R}^n$, Ω is convex, Ω contains only piecewise-linear interfaces (see §4.1), $f \in \mathrm{L}^2(\Omega)$ and $\partial\Omega = \Gamma_1 \cup \Gamma_2$. The variational formulation of (5.1) and (5.2) uses two Hilbert spaces, one for velocity and one for pressure. The pressure space is simply $\mathrm{L}^2(\Omega)$. That for velocity is a subset of the space $\mathrm{H}_{\mathrm{div}}(\Omega)$, where

$$\mathrm{H}_{\mathrm{div}}(\Omega) = \{v \in \mathrm{L}^2(\Omega)^n \mid \nabla \cdot v \in \mathrm{L}^2(\Omega)\},$$

which satisfies the boundary condition on Γ_2, denoted by $\mathrm{H}^0_{\mathrm{div}}(\Omega)$, i.e.

$$\mathrm{H}^0_{\mathrm{div}}(\Omega) = \{v \in \mathrm{H}_{\mathrm{div}}(\Omega) \mid v \cdot n = 0 \text{ on } \Gamma_2\}.$$

The inner product

$$\langle v \,,\, w \rangle_{\mathrm{H}_{\mathrm{div}}(\Omega)} = \langle v \,,\, w \rangle_{\mathrm{L}^2(\Omega)^n} + \langle \nabla \cdot v \,,\, \nabla \cdot w \rangle_{\mathrm{L}^2(\Omega)} \,,$$

is defined on $\mathrm{H}_{\mathrm{div}}(\Omega)$ and induces the norm

$$||v||_{\mathrm{H}_{\mathrm{div}}(\Omega)} = \left(||v||^2_{\mathrm{L}^2(\Omega)^n} + ||\nabla \cdot v||^2_{\mathrm{L}^2(\Omega)} \right)^{\frac{1}{2}} .$$

For simplicity the notation $V = \mathrm{H}^0_{\mathrm{div}}(\Omega)$ and $\Pi = \mathrm{L}^2(\Omega)$ will be used where appropriate. The variational formulation of (5.1) and (5.2) is then obtained by multiplying the first equation in (5.1) by an arbitrary function $v \in V$, and the second equation in (5.1) by an arbitrary function $q \in \Pi$, and integrating. Hence suppose $v \in V$ and $q \in \Pi$. Then,

$$\int_\Omega \frac{\mu}{k} u \cdot v \, dx + \int_\Omega v \cdot \nabla p \, dx = 0 \;\; \forall v \in V,$$

and so

$$\int_\Omega \frac{\mu}{k} u \cdot v \, dx - \int_\Omega p \nabla \cdot v \, dx = \int_{\Gamma_1} f v \cdot n \, d\gamma \ \forall v \in V,$$

using the boundary conditions (5.2) and the fact that $v \cdot n = 0$ on Γ_2. Here n is the unit outward normal to $\partial\Omega$, and γ is the unit of arclength along $\partial\Omega$. Hence the variational formulation of (5.1), (5.2) is,

Find $(u,p) \in V \times \Pi$ *such that*

$$\begin{aligned} \int_\Omega \tfrac{\mu}{k} u \cdot v \, dx - \int_\Omega p \nabla \cdot v \, dx &= \int_{\partial\Omega} f v \cdot n \, d\gamma \ \forall v \in V, \\ -\int_\Omega q \nabla \cdot u \, dx &= 0 \ \forall q \in \Pi, \end{aligned} \tag{5.3}$$

which can be rewritten in the abstract form,

Find $(u,p) \in V \times \Pi$ *such that*

$$\begin{aligned} a(u,v) + b(v,p) &= F(v) \ \forall v \in V, \\ b(u,q) &= 0 \ \forall q \in \Pi. \end{aligned} \tag{5.4}$$

It is easy to show that both the bilinear forms $a(\cdot,\cdot)$ and $b(\cdot,\cdot)$ (and the linear function $F(\cdot)$) are continuous, i.e. $\exists$ constants C_1 and C_2 such that

$$a(u,v) \le C_1 \left|\left|u\right|\right|_V \left|\left|v\right|\right|_V,$$

$$b(v,p) \le C_2 \left|\left|v\right|\right|_V \left|\left|p\right|\right|_\Pi.$$

Here $C_1 = \mu/k_{\text{min}}$ and $C_2 = 1$.

Notice that $a(\cdot,\cdot)$ is not coercive on the whole of $V = \mathrm{H}_{\text{div}}(\Omega)$, since

$$a(v,v) \ge \frac{\mu}{k_{\text{max}}} \left|\left|v\right|\right|_{\mathrm{L}^2(\Omega)},$$

and the space $\mathrm{H}_{\text{div}}(\Omega)$ cannot be embedded in $\mathrm{L}^2(\Omega)$. However, $a(\cdot,\cdot)$ is coercive on an important subspace of $\mathrm{H}_{\text{div}}(\Omega)$. Define

$$V' = \{v \in \mathrm{H}_{\text{div}}(\Omega) \mid b(v,q) = 0 \, \forall q \in \Pi\}.$$

Then clearly $V' = \{v \in \mathrm{H}_{\text{div}}(\Omega) \mid \nabla \cdot v = 0\}$, i.e. V' is the subspace of V containing divergence-free functions. Suppose now that $v \in V'$. Then

$$a(v,v) \geq \frac{\mu}{k_{\max}} ||v||_{L^2(\Omega)} = \frac{\mu}{k_{\max}} ||v||_V \ \ \forall v \in V', \tag{5.5}$$

so that $a(\cdot,\cdot)$ is coercive on V'. Now since $b(u,q) = 0 \ \forall q \in \Pi$, $u \in V'$ and

$$a(u,v) = F(v) \ \ \forall v \in V'. \tag{5.6}$$

Consequently, the Lax-Milgram lemma [13, Theorem 1.1.3] can be applied to obtain that (5.6) has a unique solution $u \in V'$. Of course it has yet to be shown that the set $V' \neq \{0\}$. This will follow from a coercivity condition for $b(\cdot,\cdot)$.

The coercivity condition on $a(\cdot,\cdot)$,

$$\exists \alpha > 0 \text{ such that } a(v,v) \geq \alpha \, ||v||_V^2 \ \ \forall v \in V', \tag{5.7}$$

which holds by (5.5), implies that $\exists \alpha > 0$ such that $\alpha \, ||v||_V \leq a(v,v)/\, ||v||_V \ \ \forall v \in V'$ and since $v/\, ||v||_V$ is a unit vector and $V' \subset V$,

$$\frac{a(v,v)}{||v||_V} = \sup_{w \in V} \frac{a(w,v)}{||w||_V}.$$

Therefore the coercivity condition (5.7) can be rewritten as,

$$\exists \alpha > 0 \text{ such that } \alpha \, ||v||_V \leq \sup_{w \in V} \frac{a(w,v)}{||w||_V} \ \ \forall v \in V'.$$

The similar coercivity condition for $b(\cdot,\cdot)$,

$$\exists \beta > 0 \text{ such that } \beta \, ||q||_\Pi \leq \sup_{v \in V} \frac{b(v,q)}{||v||_V} \ \ \forall q \in \Pi, \tag{5.8}$$

provides an existence and uniqueness result for the solution of the equation

$$b(v,p) = \widetilde{F}(v) \ \ \forall v \in V, \tag{5.9}$$

where $\widetilde{F}(v) = F(v) - a(u,v)$ and u is the solution of (5.6), via another application of the Lax-Milgram lemma. In addition to the existence and uniqueness theory, (5.8) is also a sufficient condition for the space V' to be nontrivial (i.e. if (5.8) holds then

$V' \neq \{0\}$), and hence the solution u of (5.6) is nontrivial. Hence the following theorem of Brezzi [8, Theorem 2.1] holds.

Theorem 5.1.1 (Brezzi) *If the bilinear forms $a(\cdot,\cdot)$ and $b(\cdot,\cdot)$ and linear function $F(\cdot)$ are continuous and (5.7) and (5.8) hold, then the problem (5.4) has a unique solution $(u,p) \in V \times \Pi$.*

Proof See Brezzi [8]. □

The coercivity conditions (5.7), (5.8) (together with the continuity requirement on $a(\cdot,\cdot)$ and $b(\cdot,\cdot)$) are therefore sufficient conditions for the existence of a unique solution to (5.4). The condition (5.8) is often referred to as the *Babuska-Brezzi* or *inf-sup* condition, since it can be rewritten in the form

$$\exists\, \beta > 0 \text{ such that } \beta \leq \inf_{q\in\Pi} \sup_{v\in V} \frac{b(v,q)}{\|v\|_V \|q\|_\Pi}. \tag{5.10}$$

Inequality (5.7) has already been verified for (5.1), it only remains to be shown that (5.8) holds. Inequality (5.8) becomes,

$$\beta \|q\|_{\mathrm{L}^2(\Omega)} \leq \sup_{v\in \mathrm{H}_{\mathrm{div}}(\Omega)} \frac{\int_\Omega q\nabla\cdot v\, dx}{\|v\|_{\mathrm{H}_{\mathrm{div}}(\Omega)}} \quad \forall q \in \mathrm{L}^2(\Omega).$$

So, suppose that $q \in \mathrm{L}^2(\Omega)$ and let θ solve the Dirichlet problem,

$$\begin{aligned} -\Delta\theta &= q \text{ in } \Omega, \\ \theta &= 0 \text{ on } \partial\Omega. \end{aligned}$$

Then defining $v := -\nabla\theta$, $v \in \mathrm{H}_{\mathrm{div}}(\Omega)$ since $\theta \in \mathrm{H}^2(\Omega)$ and $\nabla\cdot v = q \in \mathrm{L}^2(\Omega)$. Using the regularity result,

$$\|\theta\|_{\mathrm{H}^2(\Omega)} \leq C\|q\|_{\mathrm{L}^2(\Omega)},$$

where C is a constant, it follows that

$$\|v\|_{\mathrm{L}^2(\Omega)} \leq C\|q\|_{\mathrm{L}^2(\Omega)},$$

and hence

$$||v||^2_{\mathrm{H_{div}}(\Omega)} \leq (1 + C^2)\, ||q||^2_{\mathrm{L}^2(\Omega)} .$$

With this choice of v,

$$\begin{aligned} \frac{\int_\Omega q \nabla \cdot v \, dx}{||v||_{\mathrm{H_{div}}(\Omega)}} &= \frac{||q||^2_{\mathrm{L}^2(\Omega)}}{||v||_{\mathrm{H_{div}}(\Omega)}}, \\ &\geq (1 + C^2)^{-\frac{1}{2}}\, ||q||_{\mathrm{L}^2(\Omega)} , \end{aligned}$$

and hence (5.8) holds with $\beta \geq (1 + C^2)^{-\frac{1}{2}}$.

Thus both (5.7) and (5.8) hold, so that a unique solution $(u, p) \in \mathrm{H_{div}}(\Omega) \times \mathrm{L}^2(\Omega)$ of the variational form (5.4) of the groundwater flow equations exists. However the existence and uniqueness of mixed finite element approximations to (5.4) does not follow immediately, as is explained in the following section.

5.2 Abstract mixed finite element approximation

Finite dimensional analogues of the coercivity conditions (5.7), (5.8) give an existence and uniqueness result for mixed finite element approximations of (5.4). Suppose that $\mathcal{T}$ is a triangulation of the domain Ω and that V_h and Π_h are finite dimensional subspaces of V and Π respectively. Here h denotes the dependence of V_h and Π_h on some discretisation parameter which tends to zero. Then the finite dimensional approximation to (5.4) is

Find $(u_h, p_h) \in V_h \times \Pi_h$ *such that*

$$\begin{aligned} a(u_h, v) + b(v, p_h) &= F(v) \ \forall v \in V_h, \\ b(u_h, q) &= 0 \ \forall q \in \Pi_h. \end{aligned} \tag{5.11}$$

Proceeding in a similar fashion to that in the previous section for the variational formulation (5.4), existence and uniqueness of solutions to (5.11) will depend on the finite dimensional, divergence free subset of V_h,

$$V_h' = \{v \in V_h \mid b(v, q) = 0 \ \forall q \in \Pi_h\}.$$

Provided that finite dimensional analogues of (5.7) and (5.8) hold, namely that

$$\exists\alpha > 0 \text{ such that } a(v,v) \geq \alpha \left|\left|v\right|\right|_V^2 \quad \forall v \in V_h', \tag{5.12}$$

and

$$\exists\beta > 0 \text{ such that } \beta \leq \inf_{q\in\Pi_h} \sup_{v\in V_h} \frac{b(v,q)}{\left|\left|v\right|\right|_V \left|\left|q\right|\right|_\Pi}, \tag{5.13}$$

then (5.11) has a unique solution $(u_h, p_h) \in V_h \times \Pi_h$. This follows from Theorem 5.1.1 with V and Π replaced by V_h and Π_h. A further result from Brezzi [8, Theorem 2.1] shows that this approximation to the solution $(u,p) \in V \times \Pi$ of (5.4) satisfies the error bound,

$$\left|\left|u - u_h\right|\right|_V + \left|\left|p - p_h\right|\right|_\Pi \leq C \left(\inf_{v\in V_h} \left|\left|u - v\right|\right|_V + \inf_{q\in\Pi_h} \left|\left|p - q\right|\right|_\Pi \right).$$

With assumptions on the interpolation of the boundary of Ω by the triangulation $\mathcal{T}$, more useful error bounds in other Sobolev norms, and bounds on purely pressure errors, can be found. See for example [7, §10.6].

Since (5.12) is not necessarily implied by (5.7), as $V_h' \not\subset V'$ in general, and (5.13) is obviously not necessarily implied by (5.8), the discrete coercivity conditions are not implied by their non-discrete counterparts and so (5.12) and (5.13) must be checked for the choice of mixed finite elements taken. The mixed finite element approximation of (5.4) will only be stable if (5.12) and (5.13) both hold with constants α and β independent of h. Otherwise it may be possible to find a sequence $\{h_i\}$ and/or a sequence $\{h_j\}$ with $h_i \longrightarrow 0$ and $h_j \longrightarrow 0$ such that $a(v_i, v_i) \longrightarrow 0$ for some sequence of vectors $\{v_i\}$ and/or $\inf_{q_j\in\Pi_{h_j}} \sup_{v_j\in V_{h_j}} b(v_j, q_j)/\left|\left|v_j\right|\right|_V \left|\left|q_j\right|\right|_\Pi \longrightarrow 0$ for some sequences $\{v_j\}$ and $\{q_j\}$, and so in the limit as $h \longrightarrow 0$, the existence and uniqueness of a solution to (5.11) cannot be verified.

Inequality (5.13) is easy to verify by simply taking V_h to be a sufficiently large space with respect to Π_h. The difficulty with this however is that such a space V_h will have a large basis set and hence this approach will lead to a very large matrix equation. Instead, the general idea is to attempt to find a space V_h which is as small as possible with respect to Π_h (but which provides a sufficiently accurate approximation) and is compatible with Π_h in the sense that (5.13) holds.

One popular choice of mixed finite element for the groundwater flow equations for which (5.12) and (5.13) can be verified is the *Raviart-Thomas* mixed finite element. This is the topic of the following section.

5.3 Raviart-Thomas mixed finite elements

In order to construct a stable mixed finite element approximation to (5.4), the aim is to find subspaces $V_h \subset V$ and $\Pi_h \subset \Pi$ such that the discrete coercivity conditions (5.12) and (5.13) hold. Suppose for simplicity that $\Omega \subset \mathbb{R}^2$ is a polygonal domain and that $\mathcal{T}$ is a triangulation of $\overline{\Omega}$ with maximum triangle diameter h and interior angles bounded below by θ_0. Let $T \in \mathcal{T}$ and let n_T denote the outward unit normal to T. Further suppose that v is smooth when restricted to each $T \in \mathcal{T}$. If $v \in \mathrm{H}_{\mathrm{div}}(\Omega)$ then it must be the case that $v \cdot n_\mathrm{T}$ is continuous along the edge of each triangle $T \in \mathcal{T}$. For if not, suppose that T_1 and T_2 are two adjacent triangles with a common edge which is parallel to the x_2 axis along which $x_1 = 0$ (without loss of generality), and that the component $v_1(x)$ of v in the direction normal to the common edge is discontinuous. Then, on the common edge, $\partial v_1/\partial x_1(x)$ is $\delta(x_1)$ up to multiplication by a nonzero function of x_2, and so $\int_\Omega \left|\frac{\partial v_1}{\partial x_1}(x)\right|^2 dx$ is

$$\int\int_\Omega |\delta(x_1)|^2 \; dx_2 dx_1 = C\int |\delta(x_1)|^2 \; dx_1 = \infty,$$

(since the delta function is not in $\mathrm{L}^2(\Omega)$) so that $v \notin \mathrm{H}_{\mathrm{div}}(\Omega)$. Hence $v \in \mathrm{H}_{\mathrm{div}}(\Omega) \implies v \cdot n_\mathrm{T}$ is continuous for each $T \in \mathcal{T}$. Conversely suppose that $v \cdot n_\mathrm{T}$ is continuous on the common edge. Then it is clear that $\partial v_2/\partial x_2$ is bounded along the common edge, since v_2 is smooth in each triangle. Now if $f(x)$ and $g(x)$ are the components of v in the direction x_1 in T_1 and T_2 respectively, then the Taylor expansions of v_1 in each triangle are

$$f(x) = f_0 + xf_0' + \text{h.o.t.} \quad \text{for } x \text{ on the common edge,}$$

and

$$g(x) = f_0 + xg_0' + \text{h.o.t.} \quad \text{for } x \text{ on the common edge,}$$

since the normal component of v is continuous across the common edge, and so on the common edge

$$\begin{aligned}\frac{\partial v_1}{\partial x_1} &= \lim_{h \longrightarrow 0} \frac{f(h) - g(-h)}{2h}, \\ &= \frac{f_0' + g_0'}{2}, \\ &< \infty,\end{aligned}$$

and hence $\partial v_1/\partial x_1$ is also bounded and so $v \in \mathrm{H}_{\mathrm{div}}(\Omega)$. Therefore $v \in \mathrm{H}_{\mathrm{div}}(\Omega) \Longleftrightarrow v \cdot n_{\mathrm{T}}$ is continuous for each $T \in \mathcal{T}$.

The Raviart-Thomas (R-T) mixed finite element spaces [61] are designed to ensure that the normal components of the functions in the space V_h are continuous across triangle edges. This is achieved as follows. Suppose that the restriction of v to T is a polynomial of degree $k+1$ for each $T \in \mathcal{T}$, i.e. $v|_{\mathrm{T}} \in P_{k+1}(T)^n$. Then $v|_{\mathrm{T}}$ is smooth and $\nabla \cdot v \in P_k(T)$. Then also $v \cdot n_{\mathrm{T}} \in P_k(\partial T)\ \forall T \in \mathcal{T}$, so that $v \cdot n_{\mathrm{T}}$ is a kth degree polynomial on each triangle edge. If V_h is the set of all such functions, then it is clear from previous remarks that $V_h \subset V$. A natural choice for the set Π_h is then

$$\Pi_h = \{q \in \mathrm{L}^2(\Omega) \mid q|_{\mathrm{T}} \in P_k(T)\ \forall T \in \mathcal{T}\}.$$

Now if

$$V_h' = \{v \in V_h \mid \nabla \cdot v = 0\},$$

then $V_h' \subset V'$ and so V_h' inherits the coercivity property (5.12) from (5.7). In order to show that the R-T element is stable it only remains to be shown that (5.13) holds. This is a result of [61, Lemma 4] where it is shown that, given $q \in \Pi_h$, $\exists v \in V_h$ such that

$$\nabla \cdot v = q,$$

and

$$||v||_{\mathrm{H_{div}}(\Omega)} \leq C\,||q||_{\mathrm{L}^2(\Omega)},$$

so that

$$\begin{aligned}\frac{\int_\Omega q\nabla\cdot v\,dx}{||v||_{\mathrm{H_{div}}(\Omega)}} &= \frac{||q||^2_{\mathrm{L}^2(\Omega)}}{||v||_{\mathrm{H_{div}}(\Omega)}},\\ &\geq C^{-1}\,||q||_{\mathrm{L}^2(\Omega)},\end{aligned}$$

and so the discrete coercivity condition (5.13) for $b(\cdot,\cdot)$ holds with $\beta \geq C^{-1}$. Thus the R-T element is stable.

The computation of the basis functions associated with the Raviart-Thomas is explained in §5.5, but first a preliminary result on diagonal scaling of mass matrices due to Wathen is described which will be applied to the mass matrices generated in §5.5.

5.4 Diagonal scaling of scaled mass matrices

The result explained in this section is due to Wathen and is found in [82]. Essentially the result is that the maximum and minimum eigenvalues of an arbitrary mass matrix which is preconditioned by its diagonal can be bounded by the largest maximum eigenvalue and smallest minimum eigenvalue of the diagonally preconditioned element mass matrices associated with the triangulation. This result also holds for the scaled mass matrices which arise in groundwater flow applications, and it will be seen that the diagonally preconditioned scaled mass matrix has condition which is independent of the scaling parameter k in the groundwater flow equations. Hence iterative methods for solving mass matrix systems (specifically the mass matrix system arising in the LSQR(A^{-1}) subproblems) can be made to converge independently of the scaling parameter k, c.f. Chapter 4 where it was seen that iterative methods for scaled stiffness matrix systems could not be made to perform independently of k.

As before, let $\mathcal{T}$ be a triangulation of the domain Ω (or any polygonal division of Ω into convex non-overlapping elements) which resolves the discontinuities in the piecewise constant function k, and let $\{\psi_j\}_{j=1}^n$ be a set of basis functions on the n unconstrained nodes in the triangulation (i.e. non-Dirichlet boundary nodes) for the velocity trial space V_h. Then any function $v \in V_h$ (restricted to the non-Dirichlet part of Ω) can be expressed as

$$v(x) = \sum_{i=1}^{n} V_j \psi_j(x).$$

The scaled mass matrix associated with $\mathcal{T}$ and V_h is then

$$M = [m_{ij}] = [\langle k_i^{-1} \psi_i \, , \, \psi_j \rangle], \tag{5.14}$$

where $\langle \cdot \, , \, \cdot \rangle$ denotes the $\mathrm{L}^2(\Omega)$ inner product, so that $\langle k^{-1} v \, , \, v \rangle = (V^T M V)^{\frac{1}{2}}$, and k_i is the restriction of k to the relevant triangle.

The motivation in [82] is as follows. If the element scaled mass matrix on triangle T_j is denoted by M_j, then the scaled mass matrix M can be assembled from the element scaled mass matrices as

$$M = \sum L_j^T M_j L_j, \tag{5.15}$$

where the summation is performed over all $T \in \mathcal{T}$ and $L_j \in \{0,1\}^{n_j \times n}$ is a restriction from the global nodal ordering to the local nodal ordering on element j with n_j unknowns. Defining $L^T = [L_1^T \, L_2^T \ldots]$, (5.15) can be rewritten as

$$M = L^T (diag M_j) L,$$

where $(diag M_j)$ is a block diagonal matrix with the scaled element mass matrices on its diagonal. With the above notation, the diagonal matrix of M can be expressed as

$$D = L^T (diag D_j) L,$$

where D_j is the diagonal matrix of M_j.

Now if λ is an eigenvalue of $D^{-1}M$ then λ is bounded by the Rayleigh quotients,

$$\min_{x \neq 0} \frac{x^T M x}{x^T D x} \leq \lambda \leq \max_{x \neq 0} \frac{x^T M x}{x^T D x},$$

since M is a symmetric positive definite matrix. Using the substitution $y = Lx$, and recognising that $span(L) \subset span(diagM_j)$ the above bound implies

$$\min_{y\neq 0} \frac{y^T(diagM_j)y}{y^T(diagD_j)y} \leq \lambda \leq \max_{y\neq 0} \frac{y^T(diagM_j)y}{y^T(diagD_j)y}, \tag{5.16}$$

and so λ is bounded by the largest maximum eigenvalue and smallest minimum eigenvalue of $D_j^{-1}M_j$ over all triangles in $\mathcal{T}$.

To see that the maximum and minimum eigenvalues of each $D_j^{-1}M_j$ are independent of k, notice that

$$M_l = [m_{l_{ij}}] = [\langle k_l^{-1}\psi_{l_i}\,,\,\psi_{l_j}\rangle] = [k_l^{-1}[\langle \psi_{l_i}\,,\,\psi_{l_j}\rangle],$$

with obvious notation, because k is constant on each triangle, and so

$$M_l = k_l^{-1}\widetilde{M_l},$$

where $\widetilde{M_l}$ is the (unscaled) element mass matrix on triangle l. Similarly $D_l = k_l^{-1}\widetilde{D}_l$ so that

$$D_l^{-1}M_l = k_l\widetilde{D}_l^{-1}k_l^{-1}\widetilde{M_l} = \widetilde{D}_l^{-1}\widetilde{M_l}$$

and $\widetilde{D}_l^{-1}\widetilde{M_l}$ is independent of k. These results are summed up in the following theorem.

Theorem 5.4.1 (Wathen) *The maximum and minimum eigenvalues of the diagonally preconditioned mass matrix (5.14) are independent of k and are bounded by the largest maximum eigenvalue and smallest minimum eigenvalue of the diagonally scaled element mass matrices.*

To illustrate this result, a scaled mass matrix of linear elements on a square region in which k takes values of 1, 10, 100 and 1000 in four layers and $\mu = 1$ has been formed. Here there are 128 unknowns and $\kappa(M) \approx 2.6 \times 10^6$. The eigenvalue distribution of M is shown in Figure 5-1.

Due to the difficult scaling in Figure 5-1, each cross indicates the presence of a cluster of eigenvalues, for example the rightmost cross hides a cluster of 9 distinct eigenvalues. The eigenvalue distribution of $D^{-1}M$ can be seen in Figure 5-2. The condition number of the preconditioned matrix is $\kappa(D^{-1}M) \approx 2.7$ which agrees with

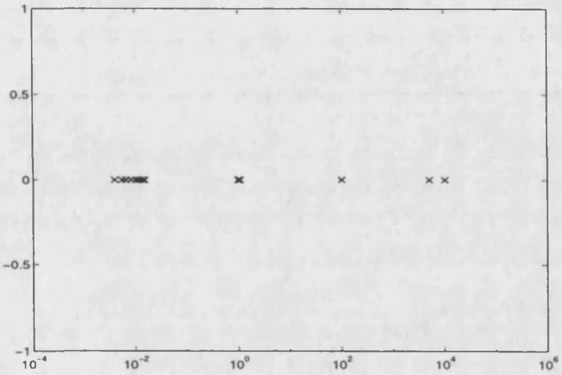


Figure 5-1: Eigenvalue distribution of the scaled mass-matrix M with 128 unknowns.

the bound of 3 for $\kappa(\widetilde{D}^{-1}\widetilde{M})$ given by Wathen in [82] where bounds on the condition of diagonally preconditioned mass matrices are given for some popular choices of finite elements.

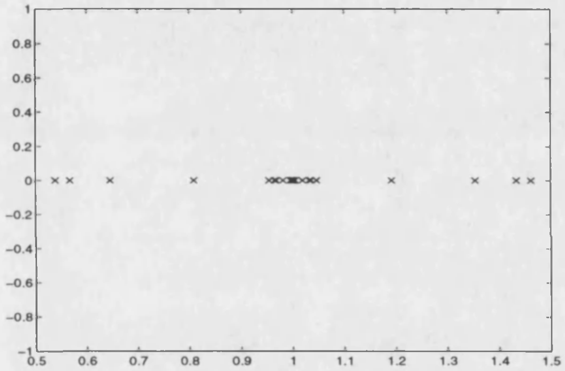


Figure 5-2: Eigenvalue distribution of the preconditioned matrix $D^{-1}M$.

5.5 Computation of Raviart-Thomas basis functions

The usage of the Raviart-Thomas mixed finite element is easiest to visualise in 3 dimensions, and a description of the basis functions is given below for the lowest order R-T element. So, suppose that V_h is the set of functions v satisfying

$$v \cdot n_{\mathrm{T}} \text{ is constant on each face of } T \ \forall T \in \mathcal{T}, \tag{5.17}$$

$$\nabla \cdot v \text{ is constant in each tetrahedron } T \in \mathcal{T}, \tag{5.18}$$

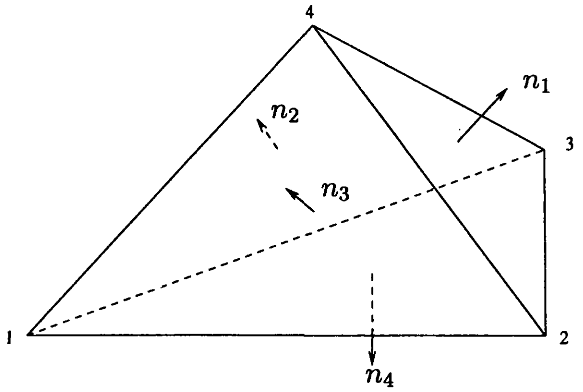


Figure 5-3: A general tetrahedron $T \in \mathcal{T}$

and that

$$\Pi_h = \{q \in \mathrm{L}^2(\Omega) \mid q|_\mathrm{T} \text{ is constant for each } T \in \mathcal{T}\}.$$

Since $v \cdot n_\mathrm{T}$ is specified to be constant on each face, there will be four basis functions with non-empty support in each tetrahedron. Hence let T be an arbitrary tetrahedron with corners and normals to each face numbered as in Figure 5-3 and suppose that $\Psi_1(x), \ldots, \Psi_4(x)$ are the restrictions to T of the four basis functions with non-empty support in T. Given $x \in T$, let $(\lambda_1, \ldots, \lambda_4) = (\lambda_1(x), \ldots, \lambda_4(x))$ be its barycentric coordinates (where $\lambda_i = 1$ at node i). Then $\Psi_1 = \Psi_1(x)$ can be written as

$$\Psi_1 = \lambda_1 a + \lambda_2 b + \lambda_3 c + \lambda_4 d,$$

where a, b, c, d are constant vectors. Since the normal component of each $v \in V_h$ is specified to be continuous along normals to each face, it would be wise to set

$$\Psi_i \cdot n_j = \delta_{ij} \text{ on each face,}$$

in order to satisfy (5.17). Using the fact that $\lambda_i = 0$ on the face with normal n_i, the four equations

$$\begin{aligned}
&\lambda_2(n_1\cdot b)+\lambda_3(n_1\cdot c)+\lambda_4(n_1\cdot d)=1,\\
\lambda_1(n_2\cdot a)\quad &+\lambda_3(n_2\cdot c)+\lambda_4(n_2\cdot d)=0,\\
\lambda_1(n_3\cdot a)+\lambda_2(n_3\cdot b)\quad &+\lambda_4(n_3\cdot d)=0,\\
\lambda_1(n_4\cdot a)+\lambda_2(n_4\cdot b)+\lambda_3(n_4\cdot c)\quad &=0,
\end{aligned}$$

hold. Since the sum of the barycentric coordinates is always equal to 1 (for $x \in T$), the previous equalities can be seen to hold for $a = 0$ and b, c, and d chosen such that Ψ_1 has representation

$$\Psi_1=\lambda_2\frac{n_3\times n_4}{n_1\cdot(n_3\times n_4)}+\lambda_3\frac{n_2\times n_4}{n_1\cdot(n_2\times n_4)}+\lambda_4\frac{n_2\times n_3}{n_1\cdot(n_2\times n_3)}.$$

Ψ_2, Ψ_3, and Ψ_4 are defined similarly. Since $(\lambda_1, \dots, \lambda_4)$ is related linearly to $x = (x_1, x_2, x_3)$ by the mapping,

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14}\\ a_{21} & a_{22} & a_{23} & a_{24}\\ a_{31} & a_{32} & a_{33} & a_{34}\\ 1 & 1 & 1 & 1 \end{bmatrix}\begin{bmatrix}\lambda_1\\ \lambda_2\\ \lambda_3\\ \lambda_4\end{bmatrix}=\begin{bmatrix}x_1\\ x_2\\ x_3\\ 1\end{bmatrix},$$

where node i has coordinates (a_{1i}, a_{2i}, a_{3i}), it is clear that $\nabla \cdot \Psi_i$ is constant on the tetrahedron so that (5.18) holds.

The scaled mass matrices arising from the lowest order Raviart-Thomas elements are analysed in the following subsection. Although it was shown in §5.4 that the effect of the scaling k could be removed by diagonally preconditioning, it will be seen that the condition of mass matrix systems can be expected to be bad when large aspect ratios are present in the triangulation.

5.5.1 Large aspect ratios and ill conditioning

The element mass matrix for the lowest order Raviart-Thomas mixed finite element on an equilateral triangle is

$$M_e = \begin{bmatrix} \frac{5}{18} & \frac{-1}{18} & \frac{-1}{18} \\ \frac{-1}{18} & \frac{5}{18} & \frac{-1}{18} \\ \frac{-1}{18} & \frac{-1}{18} & \frac{5}{18} \end{bmatrix}.$$

Since the diagonal of M_e is constant, preconditioning M_e by its diagonal will have no effect on the condition number. The eigenvalues of M_e are

$$\frac{1}{6}, \frac{1}{3}, \frac{1}{3},$$

and so $\kappa(D_e^{-1}M_e) = \kappa(M_e) = 2$. Therefore, by Theorem 5.4.1, any uniform triangulation of equilateral triangles of the domain Ω with arbitrary permeability function k will have a condition number of 2. If the LSQR(A^{-1}) method were used to solve the resulting mixed finite element matrix problem, this would suggest that the conjugate gradient method would be an effective method of performing the inner solves since its iterates would satisfy

$$\frac{||z - z_k||_A}{||z - z_0||_A} \leq \left(\frac{\sqrt{2}-1}{\sqrt{2}+1}\right)^k = (\sqrt{2}-1)^{2k} \leq \left(\frac{\sqrt{3}}{10}\right)^k,$$

and so an error reduction by a factor of 10^{-p} is guaranteed after only approximately $7p/5$ iterations.

The case of a uniform triangulation of equilateral triangles is obviously an ideal case and will not often arise in practise. In most groundwater flow applications it is reasonable to expect that triangles with very large aspect ratios will be present since the typical domains which are modelled are perhaps only 50m deep but can be many miles wide. This geometry dictates that the types of triangulation used must be very long and thin and this can greatly affect the condition of the element mass matrices. For example consider an isosceles triangle with vertices $(-1, 0), (1, 0), (0, L)$. The element mass matrix for this triangle is

$$M_e = \begin{bmatrix} 1/12\,\frac{3L^2+1}{L^2} & -1/24\,\frac{(L^2-1)(L^2+1)^{\frac{1}{2}}}{L^2} & -1/24\,\frac{(L^2-1)(L^2+1)^{\frac{1}{2}}}{L^2} \\ -1/24\,\frac{(L^2-1)(L^2+1)^{\frac{1}{2}}}{L^2} & 1/48\,\frac{(L^2+7)(1+L^2)}{L^2} & 1/48\,\frac{(L^2-5)(1+L^2)}{L^2} \\ -1/24\,\frac{(L^2-1)(L^2+1)^{\frac{1}{2}}}{L^2} & 1/48\,\frac{(L^2-5)(1+L^2)}{L^2} & 1/48\,\frac{(L^2+7)(1+L^2)}{L^2} \end{bmatrix},$$

and its eigenvalues are

$$\frac{1}{6},\ \frac{1+L^2}{4L^2},\ \frac{(L^2+3)(1+L^2)}{24L^2}.$$

The diagonally preconditioned element matrix is then

$$D_e^{-1}M_e = \begin{bmatrix} 1 & -\frac{L^2-1}{\sqrt{(3L^2+1)(L^2+7)}} & -\frac{L^2-1}{\sqrt{(3L^2+1)(L^2+7)}} \\ -\frac{L^2-1}{\sqrt{(3L^2+1)(L^2+7)}} & 1 & \frac{L^2-5}{L^2+7} \\ -\frac{L^2-1}{\sqrt{(3L^2+1)(L^2+7)}} & \frac{L^2-5}{L^2+7} & 1 \end{bmatrix},$$

whose eigenvalue behaviour is described in the following table.

eigenvalues *of* $D_e^{-1}M_e$	*behaviour* *as* $L \longrightarrow \infty$	*behaviour* *as* $L \longrightarrow 0$
$\frac{12}{L^2+7}$	$= O(L^{-2})$	$\longrightarrow \frac{12}{7}$
$\frac{1}{2}\frac{9L^4+30L^2+9+\sqrt{(3L^2+1)(L^2+3)(11L^4-22L^2+27)}}{(L^2+7)(3L^2+1)}$	$\longrightarrow \frac{1}{2}\left(3+\frac{\sqrt{33}}{3}\right)$	$\longrightarrow \frac{9}{7}$
$\frac{1}{2}\frac{9L^4+30L^2+9-\sqrt{(3L^2+1)(L^2+3)(11L^4-22L^2+27)}}{(L^2+7)(3L^2+1)}$	$\longrightarrow \frac{1}{2}\left(3-\frac{\sqrt{33}}{3}\right)$	$= O(L^2)$

Obviously $D_e^{-1}M_e$ has a single eigenvalue which grows or decays quadratically as the aspect ratio becomes large, and so by Theorem 5.4.1 any triangulation containing such a triangle will have $D^{-1}M$ with condition bounded by order $O(L^2)$. This is not especially promising. The behaviour of the eigenvalues is plotted in Figure 5-4. If the triangulation is *quasi-uniform* (all triangles are of roughly the same size), then the condition of M is actually $O(1)$ for any piecewise polynomial approximation, see [41], so that Theorem 5.4.1 should only be considered in the case of non-uniform grids.

The bound given in Theorem 5.4.1 is tight for many choices of velocity basis. For the Raviart-Thomas element however, the bound can be extremely pessimistic as can be seen in the following example. Consider the domain depicted in Figure 5-5. Then the condition of the element mass matrices in the isosceles triangles is $O(L^{-2})$ for small

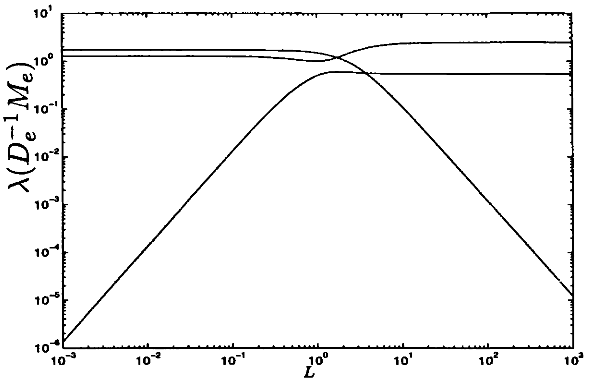


Figure 5-4: Variation of eigenvalues with L

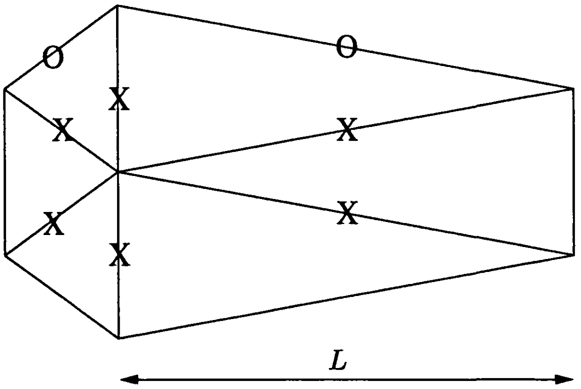


Figure 5-5: Example domain

L and $O(L^2)$ for large L by the remarks above, and that in the equilateral triangles is 2. By an analogous argument to that used to prove Theorem 5.4.1, the condition of the assembled mass matrix on the nodes marked with crosses is expected to be $O(L^{-2})$ for $L << 1$ and $O(L^2)$ for $L >> 1$. This is seen to be the case in Figure 5-6. The bound in Theorem 5.4.1 suggests that the diagonally preconditioned mass matrix still has condition of $O(L^{-2})$ for $L << 1$ and $O(L^2)$ for $L >> 1$, since the minimum eigenvalue in the isosceles diagonally preconditioned element mass matrices is $O(L^2)$ for $L << 1$ and $O(L^{-2})$ for $L >> 1$ and the maximum eigenvalue in both the equilateral and isosceles diagonally preconditioned element mass matrices is $O(1)$. In fact though, the eigenvalues of the diagonally preconditioned mass matrix have been found to be bounded for all values of L as is shown in Figure 5-7, and $\kappa(D^{-1}M)$ is bounded by approximately 4.5. With the inclusion of the Dirichlet nodes indicated by O's in Figure 5-5 the situation is slightly worse, with $\kappa(D^{-1}M)$ seen to be bounded by 18 in Figure 5-8, although the situation is nothing like that predicted by Theorem

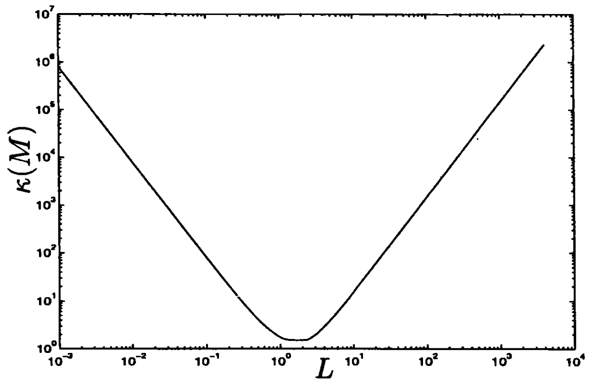


Figure 5-6: $\kappa(M)$ for varying L for the domain shown in Figure 5-5

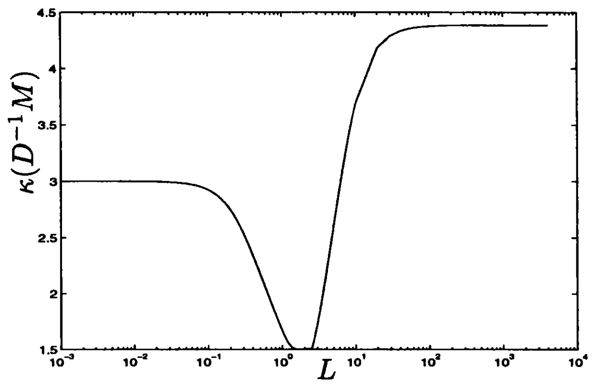


Figure 5-7: $\kappa(D^{-1}M)$ for varying L on the Neumann domain

5.4.1. It is noted that in each of Figures 5-6 - 5-8 the minimum is obtained for the value $L = \sqrt{3}$ when all triangles are equilateral.

Hence, as well making the condition of the scaled mass matrix independent of k, it might also be expected that diagonal scaling for Raviart-Thomas mass matrices on general non-uniform triangulations may lead to well conditioned matrices which have condition independent of the large aspect ratios present. Since convergence bounds for the conjugate gradient algorithm in terms of condition number are notoriously pessimistic, the above remarks lead to the conclusion that the diagonally preconditioned conjugate gradient method may be an effective method for performing the mass-matrix solves at each step of the LSQR(A^{-1}) algorithm for the Raviart-Thomas discretised groundwater flow problem. A preconditioner for the LSQR(A^{-1}) algorithm itself is discussed in the following section.

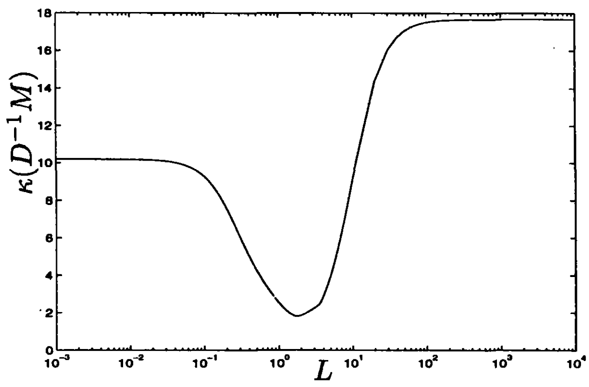


Figure 5-8: $\kappa(D^{-1}M)$ for varying L on the domain with Dirichlet nodes included

5.6 Preconditioning of LSQR(A^{-1}) applied to mixed discretisations of the groundwater flow equations

The preconditioner H which is used in the LSQR(A^{-1}, H^{-1}) iteration can be considered to be of the form NN^T where the matrix $N^{-1}B^TA^{-1}BN^{-T}$ has better condition than the Schur complement $B^TA^{-1}B$, although the matrix N is never explicitly formed. See §3.4.3. Hence the matrix H is ideally a matrix which is spectrally equivalent to $B^TA^{-1}B$. For the case when the coefficient matrix

$$\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \tag{5.19}$$

represents a stable discretisation of a Stokes operator (in the Babuska-Brezzi sense of §5.1), it is not so difficult to find a matrix H which is spectrally equivalent to the Schur complement. In this case A is a discretisation of the Laplace operator and B is a discretisation of *grad* and so, it can be naïvely concluded that the Schur complement is a discretisation of the identity function from a set of m basis functions where $B \in \mathbb{R}^{n\times m}$. Such an identity discretisation is provided by the pressure mass matrix M_p, and in [85] it is shown that

$$\beta^2 \leq \frac{y^TB^TA^{-1}By}{y^TM_py} \leq \gamma^2, \;\; \forall y \in \mathbb{R}^m, y \neq 0,$$

where β is the Babuska-Brezzi constant (in (5.10)) and γ is a continuity bound for B in terms of the M_p and A norms,

$$x^TBy \leq \gamma(x^TAx)^{\frac{1}{2}}(y^TM_py)^{\frac{1}{2}} \;\; \forall x \in \mathbb{R}^n, y \in \mathbb{R}^m.$$

Both the constants β and γ are assumed to be independent of h and so the matrix M_p is spectrally equivalent to the Schur complement, and hence is a natural choice of preconditioner.

The case is not so simple for discretisations of the groundwater flow equations. Here the Schur complement can be thought of as a scaled Laplacian operator and finding a spectrally equivalent matrix is not so simple. Recall that for the case of groundwater flow discretisations with coefficient matrices of the form

$$\begin{bmatrix} D & B \\ B^T & 0 \end{bmatrix},$$

(see Chapter 4), the preconditioner H was chosen to be an approximation to $B^TD^{-1}B$ and in §4.4 good choices were seen to be incomplete LU and additive Schwarz decompositions of $B^TD^{-1}B$. A similar approach is required here, however it is now the case that $B^TA^{-1}B$ cannot be formed explicitly (unlike $B^TD^{-1}B$) since this would involve too many operations with A^{-1} and in general, $B^TA^{-1}B$ would be a full matrix. This also implies that the incomplete factorisations are not applicable decompositions and that the additive Schwarz approach would be expensive. The first step towards forming a preconditioner for $B^TA^{-1}B$ will therefore be to find a sparse matrix which approximates $B^TA^{-1}B$. Then any approximation for the sparse approximation to $B^TA^{-1}B$ will hopefully be a fair preconditioner for $B^TA^{-1}B$ itself. Hence consider the matrix $B^TD^{-1}B$ where now, $D = diag(A)$. It is already known from Theorem 5.4.1 that $\kappa(D^{-1}A)$ can be bounded, and the examples of the previous section would tend to suggest that the bound may be independent of h for suitable elements, although this is not implied by the existing theory. It might be hoped that a similar relation holds between $B^TD^{-1}B$ and $B^TA^{-1}B$, indeed this is true by the following lemma.

Lemma 5.6.1 *If (5.19) is the coefficient matrix of a groundwater flow discretisation* *(so that A is a mass matrix) and $D = diag(A)$, then*

$$\kappa(S_D^{-1}S_A) \leq \kappa(D^{-1}A),$$

where $S_D = B^TD^{-1}B$ and $S_A = B^TA^{-1}B$, and the bound $\kappa(D^{-1}A)$ can be calculated using Theorem 5.4.1.

Proof The largest eigenvalue of $S_A^{-1}S_D$ is given by the Rayleigh quotient,

$$\begin{aligned}\max_{y\in\mathbb{R}^m}\frac{y^TB^TD^{-1}By}{y^TB^TA^{-1}By} &= \max_{z\in span(B)}\frac{z^TD^{-1}z}{z^TA^{-1}z},\\ &\leq \max_{z\in\mathbb{R}^n}\frac{z^TD^{-1}z}{z^TA^{-1}z},\\ &= \lambda_{\max}(D^{-1}A).\end{aligned}$$

An analogous relation holds for the minimum eigenvalue of $(B^TA^{-1}B)^{-1}B^TD^{-1}B$ so that

$$\lambda_{\min}(D^{-1}A) \leq \frac{y^TB^TD^{-1}By}{y^TB^TA^{-1}By} \leq \lambda_{\max}(D^{-1}A),$$

and the result follows. □

Using the result (and notation) of Theorem 5.4.1 the above result could be rewritten

$$\min_j(\lambda_{\min}(D_j^{-1}A_j)) \leq \frac{y^TB^TD^{-1}By}{y^TB^TA^{-1}By} \leq \max_j(\lambda_{\max}(D_j^{-1}A_j).$$

Lemma 5.6.1 implies that (in terms of condition number) $B^TD^{-1}B$ is at least as good a preconditioner for $B^TA^{-1}B$ as D is for A. However it is not realistic to expect to be able to use the preconditioner $B^TD^{-1}B$ in practice since every action of the preconditioner would require a solve of a system with coefficient matrix

$$\begin{bmatrix} D & B \\ B^T & 0 \end{bmatrix}.$$

Instead, good preconditioners for $B^TD^{-1}B$ (i.e. any good preconditioner H which could be used in LSQR(D^{-1}, H^{-1})) will be used to precondition $B^TA^{-1}B$. The matrix $B^TD^{-1}B$ is typically sparse, so that following §3.4.3, preconditioners such as (shifted) incomplete LU and additive Schwarz could be used. It is easy to show that if H is such a preconditioner then

$$\kappa(H^{-1}S_A) \leq \kappa(S_D^{-1}S_A)\kappa(H^{-1}S_D),$$

however this bound is far from tight. For example, the Harwell test problem TTMX2 with Raviart-Thomas elements has $\kappa(S_A) = 288.9$. Preconditioning A by D for this example gives $\kappa(D^{-1}A) = 21.1$ and so Lemma 5.6.1 predicts that $\kappa(S_D^{-1}S_A)$ to be smaller, indeed $\kappa(S_D^{-1}S_A) = 17.7$. If $H = H_{\text{ILU}}$ (where the ILU denotes the ILU(0) factorisation) then $\kappa(H^{-1}S_D) = 194.4$ and so the above bound does not predict that H should be a good preconditioner for S_A at all. However $\kappa(H^{-1}S_A) = 81.3$ which is a fair reduction in the condition number, and is a smaller condition number than $\kappa(H^{-1}S_D)$. It must be said that a condition number of $\kappa(H^{-1}S_A) = 81.3$ is not particularly small, although again it is also true that convergence bounds based on condition number estimates are notoriously pessimistic, especially for preconditioners that give rise to significant eigenvalue clustering.

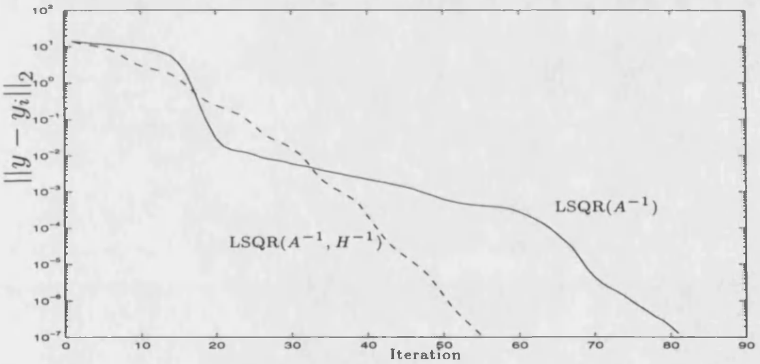


Figure 5-9: Comparison of preconditioned and unpreconditioned iterations

Figure 5-9 shows the unpreconditioned LSQR(A^{-1}) and preconditioned LSQR(A^{-1}, H^{-1}) algorithms applied to the TTMX2 problem with $H = H_{\text{ILU}}(B^T D^{-1} B)$. For the majority of the iteration time the preconditioned algorithm displays the better performance of the two, although for a short time the unpreconditioned algorithm is better when there is a sharp drop in error, probably due to the underlying Lanczos process discovering an important eigenvalue. It is stressed that TTMX2 is not a typical groundwater flow problem, since it is very small ($B \in \mathbb{R}^{752 \times 512}$), and the region considered is homogeneous, and it is not expected that the unpreconditioned algorithm will outperform the preconditioned one at any time on a more realistic problem.

5.7 Reducing the number of A^{-1} operations with a good initial guess

The main cost when using LSQR(A^{-1}) to solve generalised least-squares problems of the form

$$\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix}, \tag{5.20}$$

are the A^{-1} operations at each step. In this section, a method of reducing the number of A^{-1} operations is considered. The approach is to first obtain a good initial guess to the solution of (5.20), by performing a solve with LSQR(D^{-1}) in which the solve-step (a diagonal backsolve) is cheap.

Suppose that

$$\begin{bmatrix} D & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} x_D \\ y_D \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix}, \tag{5.21}$$

where $D = \mathrm{diag}(A)$. Then

$$\begin{aligned} A(x - x_D) + B(y - y_D) &= (D - A)x_D \\ B^T(x - x_D) &= 0 \end{aligned}$$

so that

$$\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} x_U \\ y_U \end{bmatrix} = \begin{bmatrix} b_D \\ 0 \end{bmatrix}, \tag{5.22}$$

where $x_U = x - x_D$, $y_U = y - y_D$ are the solution updates and $b_D = (D - A)x_D$. It is clear that the update $\begin{bmatrix} x_U^T & y_U^T \end{bmatrix}^T$ satisfies

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x_D \\ y_D \end{bmatrix} + \begin{bmatrix} x_U \\ y_U \end{bmatrix}.$$

Hence the solution of the generalised least squares problem (5.20) can be written as the sum of the solution to the weighted least squares problem (5.21) and the generalised least squares problem (5.22). Clearly if $D = A$ the right hand side of (5.22) will be zero so that the update $\left[\begin{array}{cc} x_U^T & y_U^T \end{array}\right]^T$ will also be zero, otherwise (5.22) can be solved to find the solution update required to correct the solution of (5.21) to the solution of (5.20).

The system (5.21) is ideally suited to solution by LSQR(D^{-1}) since inverse operations involving the diagonal matrix D are trivial. Now

$$r^{(k)} = B^T A^{-1}(b - By^{(k)}),$$

is a measure of orthogonality of residual $b - By^{(k)}$ to the span of B (recall that this value will be zero if $y^{(k)}$ solves (5.20)), and suppose that given an initial guess, $\left[\begin{array}{cc} x^{(0)^T} & y^{(0)^T} \end{array}\right]^T = \left[\begin{array}{cc} 0^T & 0^T \end{array}\right]^T$, it is required that iteration should terminate when

$$\frac{||B^T A^{-1} r^{(k)}||}{||B^T A^{-1} r^{(0)}||} < C \tag{5.23}$$

for some constant C.

Now if the initial guess is taken to be zero for both of the systems (5.20) and (5.22) it is simple to observe that

$$B^T A^{-1} r^{(0)} = B^T A^{-1} b$$

for the initial residual on (5.20), whereas

$$\begin{aligned} B^T A^{-1} r_U^{(0)} &= B^T A^{-1}\left((D - A)x_D\right) \\ &= B^T A^{-1}\left(Dx_D\right) \qquad \text{using (5.21)} \\ &= B^T A^{-1}\left(b - By_D\right), \end{aligned}$$

where $r_U^{(0)}$ denotes the initial residual in the least-squares formulation of (5.22) for a zero initial guess. Hence it could be expected that, since By_D is the best approximation to b from the span of B with respect to the D^{-1} inner-product (see [5]),

$$\left|\left|B^T A^{-1} r_U^{(0)}\right|\right| < \left|\left|B^T A^{-1} r^{(0)}\right|\right|,$$

i.e. taking a zero initial guess in (5.22) gives a smaller initial residual than taking a zero initial guess in (5.20) directly, so that the solution of the weighted least squares problem (5.21) should provide a good initial approximation to the solution of the generalised least-squares problem (5.20). It is easy to check that the requirement that $||B^TA^{-1}r^{(k)}|| / ||B^TA^{-1}r^{(0)}|| < C$ can be replaced by

$$\frac{||B^TA^{-1}r^{(k)}||}{\left|\left|B^TA^{-1}r_U^{(0)}\right|\right|} < C\frac{||B^TA^{-1}r^{(0)}||}{\left|\left|B^TA^{-1}r_U^{(0)}\right|\right|}$$

in the iteration for (5.22), and provided that $\left|\left|B^TA^{-1}r_U^{(0)}\right|\right| < ||B^TA^{-1}r^{(0)}||$, it is seen that the residual reduction which is required in the update step (5.22) is C reduced by a factor $||B^TA^{-1}r^{(0)}|| / \left|\left|B^TA^{-1}r_U^{(0)}\right|\right|$.

5.7.1 Numerical results

Test problem

Discretisations of the 2d groundwater flow problem depicted in Figure 5-11 are considered here. The domain is square with a specified high pressure on the top and low pressure on the bottom, and $u \cdot n = 0$ on each of the vertical sides. The discretisation is based on the lowest order MAC finite element, a history and analysis of which is given in [26]. A typical element is shown in Figure 5-10. The velocity is broken into the piecewise linear horizontal (u_1) and vertical (u_2) components, and the pressure is constant on each cell. The normal component of velocity to each cell edge is therefore continuous across cell boundaries so that the velocity space is certainly a subspace of $\mathrm{H}_{\mathrm{div}}(\Omega)$ following remarks in §5.3. Four discretisations in which the mesh size halves each time have been made, starting with an 8 × 8 grid (MAC(8)) and ending with a 64 × 64 grid (MAC(64)).

Results

It is clear that the method described in §5.7 only requires A^{-1} operations when solving the system (5.22), and so it might be expected that the overall iteration time in solving (5.21) and (5.22) will be smaller than solving (5.20) directly. This is indeed the case as can be seen in Table 5.1. Here the mesh sizes, linear system sizes and iteration times are shown. Convergence curves for each of the systems MAC(8)-MAC(64) are shown

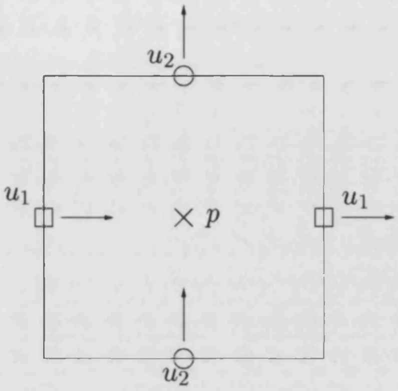


Figure 5-10: Typical MAC element

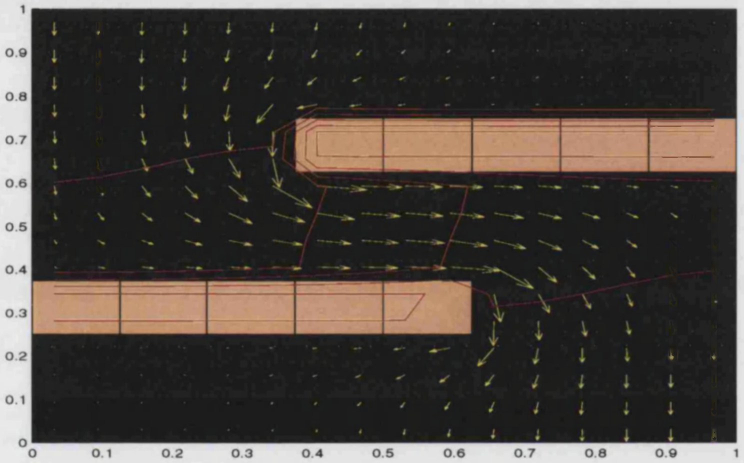


Figure 5-11: Domain, flow and pressure contours

in Figure 5-12. In each case a residual reduction,

$$\frac{||B^T A^{-1} r_k||}{||B^T A^{-1} r_0||} \leq 10^{-4},$$

was taken to be the stopping criterion.

For the small system, MAC(8), there is no advantage in performing the two solves on (5.21) and (5.22) since it is quicker to perform a single solve on the system (5.20). Notice however that the number of A^{-1} operations required in the diagonal system and update approach is fewer, 15 as opposed to 33 for the single solve. Hence the number of A^{-1} operations has reduced as was expected.

For the larger systems, MAC(16) - MAC(64), the behaviour is more unusual. The number of A^{-1} operations required for the update approach is far smaller than the number of A^{-1} operations on (5.20) as was expected. However the number of A^{-1}

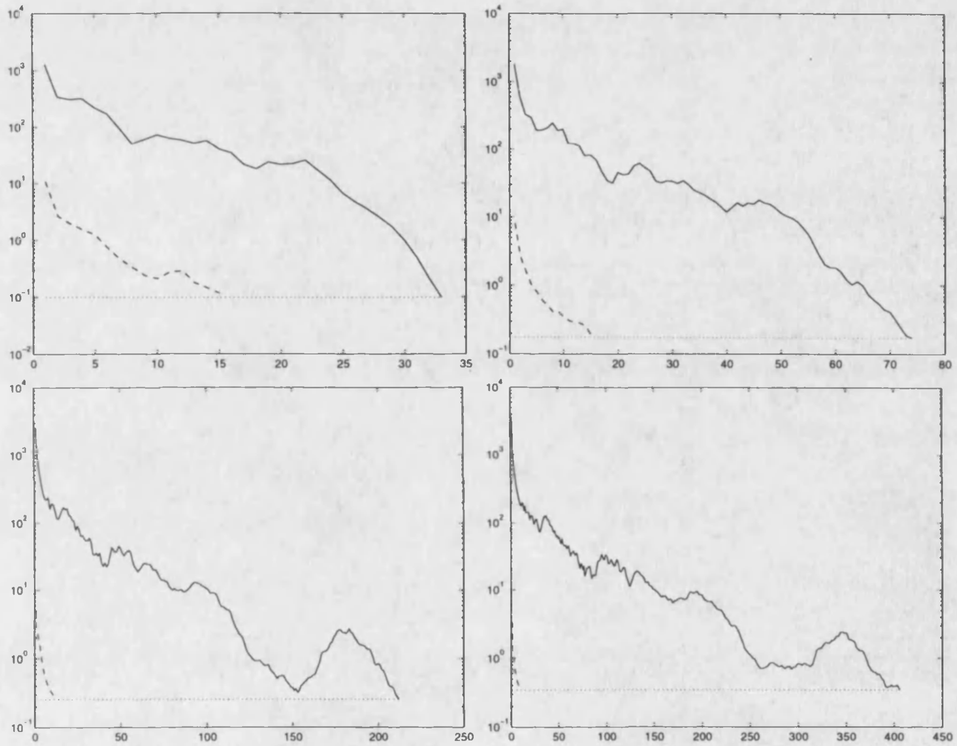


Figure 5-12: $||B^T A^{-1} r_k||$ against iteration for the MAC(8) (top-left), MAC(16) (top-right), MAC(32) (bottom-left) and MAC(64) (bottom-right) systems. Solid line (—) : LSQR(A^{-1}) on (5.20), dotted line ($\cdots$) : LSQR(A^{-1}) on (5.22).

5.7. REDUCING THE NUMBER OF A^{-1} OPERATIONS WITH A GOOD INITIAL GUESS

	Dimensions of B	full LSQR(A^{-1}) (A^{-1} ops.)	LSQR(D^{-1}) (D^{-1} ops.)	LSQR(D^{-1})+update (A^{-1} ops.)
MAC(8)	128×64	1.52 (33)	1.19 (38)	2.01 (15)
MAC(16)	512×256	10.15 (74)	5.28 (75)	7.41 (15)
MAC(32)	2048×1024	194.62 (212)	64.70 (135)	68.97 (11)
MAC(64)	8192×4096	1924.3 (406)	1044.5 (317)	1057.7 (8)

Table 5.1: Total iteration times and counts for LSQR(A^{-1}) on (5.20) and LSQR(A^{-1}) on (5.22) with initial guess from the weighted least-squares system (5.21).

operations required for the update solve on (5.22) actually decreases as the system size increases, from 15 A^{-1} operations on MAC(16) to 8 A^{-1} operations on MAC(64). Of course, the time taken for the solve on (5.21) increases as the mesh size decreases so that the total time taken for the update approach does not decrease, however the total time taken in the update approach is a great deal smaller than that for the solve on (5.20). For the systems MAC(32) and MAC(64) the total time taken was approximately a half of that required for the solve on (5.20).

The reason as to why the number of LSQR(A^{-1}) iterations on (5.22) decreases with the decrease in mesh size is unclear. It is certainly not true that the norm $||\cdot||_{D^{-1}}$ approximates the norm $||\cdot||_{A^{-1}}$ with better accuracy as the system size increases. This is obvious because in each of the graphs in Figure 5-12, after performing the solve on (5.21) there is still a residual reduction of approximately $O(10^2)$ required in each case. If it were true that $||\cdot||_{D^{-1}}$ was becoming a better approximation to $||\cdot||_{A^{-1}}$ then the required residual reduction in the update solve would become smaller. One possible explanation is that the right hand side in the system to be solved for the update,

$$\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} x_U \\ y_U \end{bmatrix} = \begin{bmatrix} (D-A)x_D \\ 0 \end{bmatrix}$$

is a good right hand side in the sense that it quickly generates a Krylov subspace which the solution $[x_U^T \; y_U^T]^T$ lies close to. It is certainly the case that if the right hand side is spanned by only a few eigenvectors of the coefficient matrix then the number of Lanczos iterations required to solve the system will be equal to this number of eigenvectors. However it is not clear that the right hand side above is of this form, and further analysis is necessary.

5.8 Summary

It has been seen in this chapter that the LSQR(A^{-1}) algorithm with suitable preconditioning is an effective method for solving groundwater flow problems arising from mixed finite element discretisations. Two popular choices of finite element, the Raviart-Thomas and MAC elements were introduced. For the Raviart-Thomas discretised problems it was seen that the A^{-1} operations in LSQR(A^{-1}) were easy to perform with the conjugate gradient algorithm since the diagonally preconditioned mass matrix, $D^{-1}A$, has condition which is independent of the bad scaling in the permeability function k (for all choices of finite elements) and is apparently independent of large aspect ratios present in the discretisation.

The problem of choosing a preconditioner H in LSQR(A^{-1}, H^{-1}) was discussed. It was seen that $B^TD^{-1}B$ (where D = diag(A)) was at least as good a preconditioner for $B^TA^{-1}B$ as D is for A which lead to the idea that incomplete factorisation or additive Schwarz preconditioners for $B^TD^{-1}B$ could be used as preconditioners in LSQR(A^{-1}, H^{-1}) . Further numerical experiments are required for the effectiveness of these preconditioners to be fully understood.

A method of reducing the number of A^{-1} operations in LSQR(A^{-1}) by first solving a system with A replaced with D in the coefficient matrix, using LSQR(D^{-1}) , was described. This approach was seen to be very effective for a groundwater flow discretisation on MAC finite elements, and had the unusual property that the number of A^{-1} operations required in the update solve reduces as the mesh size decreases. Again, further experimentation with other types of finite element is required before this result can be said to hold in general.

Appendix A : Further three-interval results

In this section, some further results based on Theorem 2.3.1 are presented. In §A.1, Theorem 2.3.1 is applied to a discretisation of an unsteady Stokes operator to reveal the three eigenvalue intervals in this case. Simple scaling of the matrix A is considered in §A.2 and an optimal scaling value for improving $\kappa(\mathcal{A})$ is found.

A.1 Unsteady Stokes operators

Matrices of the form

$$\mathcal{A}' := \begin{bmatrix} \nu A + \frac{1}{\Delta t} I & B \\ B^T & 0 \end{bmatrix} \tag{A.1}$$

can arise in the discretisation of the unsteady Stokes equations. The following corollary of Theorem 2.3.1 defines the three eigenvalue intervals associated with this matrix.

Corollary A.1.1 *The eigenvalues, λ, of the matrix $\mathcal{A}'$ are contained in three intervals,*

$$\begin{array}{rcccl} \frac{\nu\lambda_1+\frac{1}{\Delta t}}{2} - \sqrt{\frac{(\nu\lambda_1+\frac{1}{\Delta t})^2}{4} + \sigma_m^2} & \leq & \lambda & \leq & \frac{\nu\lambda_n+\frac{1}{\Delta t}}{2} - \sqrt{\frac{(\nu\lambda_n+\frac{1}{\Delta t})^2}{4} + \sigma_1^2}, \\ \nu\lambda_1 + \frac{1}{\Delta t} & \leq & \lambda & \leq & \nu\lambda_n + \frac{1}{\Delta t}, \\ \frac{\nu\lambda_1+\frac{1}{\Delta t}}{2} + \sqrt{\frac{(\nu\lambda_1+\frac{1}{\Delta t})^2}{4} + \sigma_1^2} & \leq & \lambda & \leq & \frac{\nu\lambda_n+\frac{1}{\Delta t}}{2} + \sqrt{\frac{(\nu\lambda_n+\frac{1}{\Delta t})^2}{4} + \sigma_m^2}, \end{array}$$

where $\lambda_1 \leq \ldots \leq \lambda_n$ are the eigenvalues of A and $\sigma_1 \leq \ldots \leq \sigma_m$ are the singular values of B.

Proof Trivial consequence of Theorem 2.3.1.

Notice that for very small time steps the three eigenvalue intervals will approach the sets

$$\begin{aligned} \mathcal{I}^- &\longrightarrow \left[\frac{-\sigma_m^2}{\nu\lambda_1+\frac{1}{\Delta t}}, \frac{-\sigma_1^2}{\nu\lambda_n+\frac{1}{\Delta t}}\right], \\ \mathcal{I}_1^+ &\longrightarrow \left[\nu\lambda_1+\frac{1}{\Delta t}, \nu\lambda_n+\frac{1}{\Delta t}\right], \\ \mathcal{I}_2^+ &\longrightarrow \left[\nu\lambda_1+\frac{1}{\Delta t}+\frac{\sigma_1^2}{\nu\lambda_1+\frac{1}{\Delta t}}, \nu\lambda_n+\frac{1}{\Delta t}+\frac{\sigma_m^2}{\nu\lambda_n+\frac{1}{\Delta t}}\right]. \end{aligned}$$

A.2 Simple scaling

Since $\lambda(\mathcal{A}) \subset \mathcal{I}^- \cup \mathcal{I}_1^+ \cup \mathcal{I}_2^+$, where the intervals $\mathcal{I}^-$, $\mathcal{I}_1^+$ and $\mathcal{I}_2^+$ are defined in Theorem 2.3.1, an upper bound for $\kappa(\mathcal{A})$ is simply

$$\kappa(\mathcal{A}) \leq \frac{\max\{c_2, |a_1|\}}{\min\{|a_2|, b_1\}},$$

where $\mathcal{I}^- = [a_1, a_2]$, $\mathcal{I}_1^+ = [b_1, b_2]$ and $\mathcal{I}_2^+ = [c_1, c_2]$ and $a_1, a_2, b_1, b_2, c_1, c_2$ are defined in Theorem 2.3.1. It is known that the condition of $\mathcal{A}$ can be improved by scaling the matrix A by a constant ν,

$$\mathcal{A}(\nu) = \begin{bmatrix} \nu A & B \\ B^T & 0 \end{bmatrix}.$$

Hence if it is desired to solve $\mathcal{A}x = b$ using a direct method, it is better to first solve the improved-condition system $\mathcal{A}(\nu)z(\nu) = f$ and then rescale the upper component in $x(\nu)$ to recover x. The next theorem provides an optimal scaling value which minimises the value of the bound on the condition number.

Theorem A.2.1 *If* $\mathcal{I}^-(\nu) = [a_1(\nu), a_2(\nu)]$, $\mathcal{I}_1^+(\nu) = [b_1(\nu), b_2(\nu)]$ *and* $\mathcal{I}_2^+(\nu) = [c_1(\nu), c_2(\nu)]$ *are the eigenvalue intervals for* $\mathcal{A}(\nu)$ *provided by Theorem 2.3.1, then the bound*

$$\kappa(\mathcal{A}(\nu)) \leq f(\nu) = \frac{\max\{c_2(\nu), a_1(\nu)\}}{\min\{|a_2(\nu)|, b_1(\nu)\}},$$

is minimised at the value ν_{opt} *where*

$$\nu_{opt} = \frac{\sigma_1}{\lambda_1}\sqrt{\frac{1}{1+\kappa_A}}.$$

λ_i, σ_i *are as defined in Theorem 2.3.1, and* $\kappa_A = \kappa(A)$.

Proof Notice first that $\max\{c_2(\nu), |a_1(\nu)|\} = c_2(\nu)$, so that $f(\nu) = \frac{c_2(\nu)}{\min\{|a_2(\nu)|, b_1(\nu)\}}$. Let $\gamma > 0$ be such that $-a_2(\gamma) = b_1(\gamma)$. Then

$$\gamma\lambda_1 = -\frac{\gamma\lambda_n}{2} + \sqrt{\frac{\gamma^2{\lambda_n}^2}{4} + \sigma_1^2},$$

so that

$$\gamma = \sqrt{\frac{\sigma_1^2}{\lambda_1(\lambda_1 + \lambda_n)}}.$$

Hence

$$\min\{|a_2(\nu)|, b_1(\nu)\} = \left\{ \begin{array}{rcll} b_1(\nu) & = & \nu\lambda_1 & \text{when } \nu \leq \gamma \\ -a_2(\nu) & = & -\frac{\nu\lambda_n}{2} + \sqrt{\frac{\nu^2{\lambda_n}^2}{4} + \sigma_1^2} & \text{when } \nu > \gamma \end{array} \right.$$

Suppose now that $\nu < \gamma$. Then

$$f(\nu) = \frac{c_2(\nu)}{b_1(\nu)} = \frac{1}{2}\kappa_A + \sqrt{\frac{\kappa_A^2}{4} + \frac{\sigma_m^2}{\nu^2\lambda_1}},$$

and it is easy to show that $\frac{d}{d\nu}\mathcal{A}(\nu)$ is negative so that $f(\nu)$ is monotonically decreasing for $\nu < \gamma$. Now suppose $\nu > \gamma$. Then

$$\begin{aligned} f(\nu) &= \frac{c_2(\nu)}{-a_2(\nu)} = \frac{\lambda_n + \sqrt{{\lambda_n}^2 + \frac{4\sigma_m^2}{\nu^2}}}{-\lambda_n + \sqrt{{\lambda_n}^2 + \frac{4\sigma_1^2}{\nu^2}}} \\ &= \frac{\left(\nu\lambda_n + \sqrt{\nu^2{\lambda_n}^2 + 4\sigma_m^2}\right)\left(\nu\lambda_n + \sqrt{\nu^2{\lambda_n}^2 + 4\mu_1}\right)}{4\sigma_1^2} \end{aligned}$$

and so $f(\nu)$ is clearly monotonically increasing for $\nu > \gamma$. These two observations combined with the fact that f is continuous at γ imply that $\min f(\nu) = f(\gamma)$ so that

$$\nu_{opt} = \gamma = \sqrt{\frac{\sigma_1^2}{\lambda_1(\lambda_1 + \lambda_n)}} = \frac{\sigma_1}{\lambda_1}\sqrt{\frac{1}{1 + \kappa_A}}.$$

□

Hence the bound $f(\nu)$ on $\mathcal{A}(\nu)$ is minimised for the scaling ν_{opt}, so if the eigenvalue intervals $\mathcal{I}^-, \mathcal{I}_1^+, \mathcal{I}_2^+$ are 'tight fitting intervals', $\mathcal{A}(\nu)$ will be minimised in a small neighbourhood of ν_{opt}. An example of this can be seen in Figure A-1.

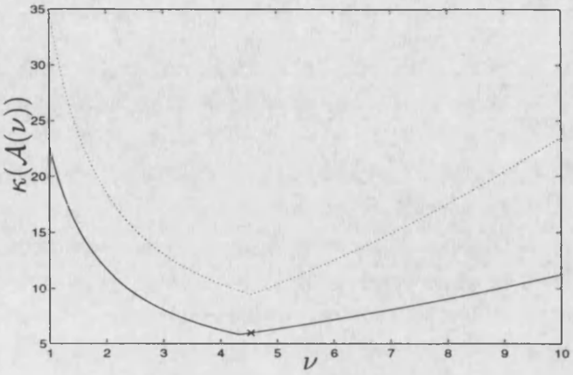


Figure A-1: Bounding $\mathcal{A}(\nu)$. Solid line - $\kappa(\mathcal{A}(\nu))$, dotted line - $f(\nu)$. 'x' marks $\kappa(\mathcal{A}(\nu_{opt}))$

Loose upper bounds on the improvement to be expected upon scaling by ν_{opt} can be obtained as follows. If $\gamma \geq 1$

$$f(1) = \frac{1}{2}\kappa_A + \sqrt{\frac{1}{4}\kappa_A^2 + \frac{\sigma_m^2}{{\lambda_1}^2}},$$

and if $\gamma < 1$,

$$f(1) = \frac{\lambda_n + \sqrt{{\lambda_n}^2 + 4\sigma_m^2}}{-\lambda_n + \sqrt{{\lambda_n}^2 + 4\sigma_1^2}}.$$

Notice also that $f(\gamma) = \frac{1}{2}\kappa_A + \sqrt{\frac{1}{4} + \kappa_B^2(1+\kappa_A)}$ where $\kappa_B = \frac{\sigma_m}{\sigma_1}$. Hence the improvement factor η^- obtained when $\gamma \geq 1$ is

$$\begin{aligned}\eta^- = \frac{f(1)}{f(\gamma)} &= \frac{\frac{1}{2}\kappa_A + \sqrt{\frac{1}{4}\kappa_A^2 + \frac{\sigma_m^2}{{\lambda_1}^2}}}{\frac{1}{2}\kappa_A + \sqrt{\frac{1}{4}\kappa_A^2 + (1+\kappa_A)}} \\ &< \frac{1}{2} + \sqrt{\frac{1}{4} + \frac{\sigma_m^2}{{\lambda_n}^2}}\end{aligned}$$

and so the best improvement to be hoped for when $\gamma \geq 1$ is roughly of order σ_m/λ_n. When $\gamma < 1$, using the alternative expression $f(\gamma) = c_2(\nu)/b_1(\nu)$ it can be shown that the improvement in $\mathcal{A}(\nu)$, η^+ satisfies

$$\eta^+ < \frac{-\frac{1}{2} + \sqrt{\frac{1}{4} + \frac{\sigma_1^2}{{\lambda_n}^2}\frac{1}{\gamma^2}}}{-\frac{1}{2} + \sqrt{\frac{1}{4} + \frac{\sigma_1^2}{{\lambda_n}^2}}}.$$

It should be noted that both of these bounds could be tightened considerably.

The last question addressed is what happens to the eigenvalue 'void' after the scaling of A. Clearly $I_1^+(\nu) \cap I_2^+(\nu) = \emptyset$ whenever $A = \alpha I$ ($\alpha \in \mathbb{R}$) and so it is only necessary to consider the case when $A \neq \alpha I$ (or specifically the case when not all of the eigenvalues of A are 1, so that $\kappa_A > 1$). Suppose that $I_1^+(\nu) \cap I_2^+(\nu) \neq \emptyset$ when $\nu > \nu_t$. Then

$$\nu_t \lambda_n = \frac{\nu_t \lambda_1}{2} + \sqrt{\frac{\nu_t^2 {\lambda_1}^2}{4} + \sigma_1^2},$$

and so

$$\nu_t = \sqrt{\frac{\sigma_1^2}{\lambda_n(\lambda_n - \lambda_1)}}.$$

Recall that $\nu_{opt} = \sqrt{\frac{\sigma_1^2}{\lambda_1(\lambda_1+\lambda_n)}}$, hence

$$\nu_{opt}^2 - \nu_t^2 = \sigma_1^2 \frac{\kappa_A^2 - 2\kappa_A - 1}{\lambda_1 \lambda_n (\kappa_A - 1)(\kappa_A + 1)},$$

so that $\nu_{opt}^2 > \nu_t^2$ whenever $\kappa_A^2 - 2\kappa_A - 1 > 0$ (since the denominator is always positive). This is satisfied whenever

$$\kappa_A > 1 + \sqrt{2} \quad \text{and} \quad \kappa_A > 1 - \sqrt{2},$$

or

$$\kappa_A < 1 + \sqrt{2} \quad \text{and} \quad \kappa_A < 1 - \sqrt{2}.$$

The second case is not possible since $\kappa_A > 1$, and by the same observation the second condition in the first case is satisfied trivially. Hence $\nu_{opt}^2 > \nu_t^2$, and therefore $\nu_{opt} > \nu_t$ when $\kappa_A > 1 + \sqrt{2}$, and so the eigenvalue void is destroyed by optimal velocity scaling for all but very well conditioned A. However the results of Theorem 2.3.2 will obviously still apply.

A.3 Conclusion

Since the scaling parameter ν_{opt} is trivial to calculate (provided that estimates to the extremal eigenvalues of A and the smallest singular value of B are known), it would seem natural to scale A by ν_{opt} if a direct solution method were being used to solve the system. The scaling would not be expected to make a significant difference if an iterative solution method were to be used, since the condition number of the system is only relevant to convergence for positive definite systems (since the condition number is then directly related to an interval containing the eigenvalues of the coefficient matrix, see §2.2.1).

Appendix B : The ΠCG algorithm

In this appendix another algorithm for the solution of augmented systems with coefficient matrix $\mathcal{A}$ is presented. The algorithm can be considered a stablemate of the LSQR(A^{-1}) algorithm of Chapter 3 since it also reduces the size of the $n+m$ dimensional problem to one of smaller dimension by the use of repeated solves of a subsystem associated with $\mathcal{A}$. Whereas the LSQR(A^{-1}) algorithm employs repeated solves with A, the algorithm presented here, ΠCG , uses repeated solves of a system with coefficient matrix

$$\mathcal{B} = \begin{bmatrix} I & B \\ B^T & 0 \end{bmatrix}.$$

The algorithm is motivated firstly by considering $\mathcal{B}$ as a preconditioner for $\mathcal{A}$. The preconditioned system $\mathcal{B}^{-1}\mathcal{A}$ is unsymmetric so it may appear unwise to consider this type of preconditioner, since unsymmetric solvers like GMRES [69] are typically far more expensive the the symmetric system solvers which have the advantage of being able to use three term recurrences to perform vector orthogonalisation rather than a Gram-Schmidt process. However the spectrum of $\mathcal{B}^{-1}\mathcal{A}$ is attractive for iterative methods (§B.1), and it will be seen that the special form of the preconditioned matrix allows the conjugate gradient method to be used as a solver. Several authors have considered the use of solves with matrices of the form $\mathcal{B}$ to speed up an iteration for $\mathcal{A}$. In [18], $\mathcal{B}$ is used in a splitting of $\mathcal{A}$ to construct standard iterative methods for solving systems with $\mathcal{A}$ as coefficient matrix. A similar preconditioner to $\mathcal{B}$ where I is replaced by a diagonal matrix D and the zero-blocks in $\mathcal{A}$ and $\mathcal{B}$ are replaced by a stabilisation matrix is considered in [32] for stabilised approximations to the Navier-

Stokes operator (so that A is unsymmetric). There, the splitting $\mathcal{A} = \mathcal{B} - \mathcal{C}$ is used in an iterative method and the iteration is shown to be convergent if $\left\|I - D^{-\frac{1}{2}}AD^{-\frac{1}{2}}\right\|_2 < 1$, and eigenvalue results for the preconditioned system are given. $\mathcal{B}$ is also used as a preconditioner in the GMRES algorithm [69] for $\mathcal{A}$ and numerical results for varying choices of D are given. In [6], $\mathcal{B}$ is seen to be an effective smoother for the multigrid method applied to the Stokes problem.

Eigenvalue results for the preconditioned system $\mathcal{B}^{-1}\mathcal{A}$ are presented in §B.1, and a brief review of nullspace methods and their relevance is given in §B.2. The action of $\mathcal{B}$ is described in §B.3 and the derivation of the algorithm ΠCG is then motivated in §B.4 by restricting a conjugate gradient iteration for $\mathcal{A}$ to a subspace of $\mathbb{R}^{n \times m}$. Convergence results for ΠCG are given in §B.5 and a generalisation of ΠCG for the groundwater flow equations which combines naturally with the LSQR(D^{-1}) algorithm is discussed in §B.6.

B.1 Preconditioning augmented systems by projection matrices

In this section, preconditioning the matrix $\mathcal{A}$ by $\mathcal{B}$ where

$$\mathcal{B} = \begin{bmatrix} I & \alpha B \\ \alpha B^T & 0 \end{bmatrix}, \quad \alpha > 0. \tag{B.1}$$

is considered. This corresponds to preconditioning using a pressure correction method (to be explained in §B.3). Taking $\alpha = 1$ corresponds to the standard pressure correction idea, the general case with $\alpha \neq 1$ is considered here so that the effect of letting $\alpha \to 0$ can be analysed.

Theorem B.1.1 *Let*

$$\mathcal{A} = \begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix},$$

where $A \in \mathbb{R}^{n \times n}$ is symmetric, positive-definite and $B \in \mathbb{R}^{n \times m}$ is of full column rank, and let $\mathcal{B}$ be defined as in (B.1). Assuming that A has no eigenvectors in the nullspace of B^T, that no columns of B are eigenvectors of A, and that the eigenvalues of A are

$0 < \lambda_1 \le \ldots \le \lambda_n$, *then the eigenvalues,* μ, *of* $\mathcal{A}^{-1}\mathcal{B}$ *are*

$$\begin{array}{ll} \mu = \alpha & \textit{with algebraic multiplicity } 2m \textit{ and geometric multiplicity } m, \\ \textit{and } \frac{1}{\lambda_n} \le \mu \le \frac{1}{\lambda_1} & n - m \textit{ times,} \end{array}$$

unless $\alpha = \frac{1}{\lambda_k}$ *for some* k, *in which case* $\mu = \alpha$ *is an eigenvalue of algebraic and geometric multiplicity* $2m+1$ *(when* λ_k *is simple as an eigenvalue of* A*). The result generalises when* λ_k *is not simple to* $2m+j$ *where* j *is the geometric multiplicity of* λ_k.

Proof Suppose

$$\mathcal{A}^{-1}\mathcal{B}z = \mu z, \tag{B.2}$$

where $z = \begin{bmatrix} x \\ y \end{bmatrix}$. Since $\mathcal{A}^{-1}\mathcal{B}$ is nonsingular there will be $n+m$ (not necessarily distinct) eigenvalues. Then

$$\begin{aligned} x + \alpha By &= \mu Ax + \mu By && \text{(B.3)} \\ \alpha B^T x &= \mu B^T x && \text{(B.4)} \end{aligned}$$

Notice that (B.4) implies either $\mu = \alpha$ or $B^T x = 0$.

First consider the case $\mu = \alpha$. By (B.3)

$$x = \alpha Ax,$$

and so either $x = 0$ or $\alpha = \frac{1}{\lambda_k}$ for some λ_k an eigenvalue of A (with assumed algebraic multiplicity is 1), and $x = x_k$, the associated eigenvector. Solutions to the eigenvalue problem (B.2) corresponding to $\mu = \alpha$ are then

$$\begin{array}{lll} \mu = \alpha \ne \frac{1}{\lambda_k} & z = \begin{bmatrix} 0 \\ e_i \end{bmatrix} & i = 1, \ldots, m \\ \mu = \alpha = \frac{1}{\lambda_k} & z = \begin{bmatrix} x_k \\ 0 \end{bmatrix}, \begin{bmatrix} x_k \\ \pm e_i \end{bmatrix} & i = 1, \ldots, m. \end{array}$$

Notice that $\mu = \alpha = \frac{1}{\lambda_k}$ is an eigenvalue of geometric multiplicity $2m+1$, whilst $\mu = \alpha \ne \frac{1}{\lambda_k}$ is an eigenvalue of geometric multiplicity m. To establish the algebraic multiplicities of these eigenvalues, the generalised eigenvalue problem

$$\left(\mathcal{A}^{-1}\mathcal{B}-\alpha\mathcal{I}\right)\begin{bmatrix} u \\ v \end{bmatrix} = z, \tag{B.5}$$

is examined. If $\mu = \alpha \neq \frac{1}{\lambda_k}$, and $z = \begin{bmatrix} 0 \\ e_i \end{bmatrix}$ then

$$(\mathcal{B}-\alpha\mathcal{A})\begin{bmatrix} u \\ v \end{bmatrix} = \mathcal{A}\begin{bmatrix} 0 \\ e_i \end{bmatrix}.$$

Hence $(I-\alpha A)\,u = Be_i$ and so

$$u = (I-\alpha A)^{-1}\,Be_i.$$

(The inverse is well-defined since $\alpha \neq \frac{1}{\lambda_k}$). Hence the problem (B.5) has solution

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} (I-\alpha A)^{-1}\,Be_i \\ 0 \end{bmatrix} i = 1,\ldots,m.$$

Therefore the eigenvalue $\mu = \alpha \neq \frac{1}{\lambda_k}$ has algebraic multiplicity $2m$ and geometric multiplicity m.

Now suppose $\mu = \alpha = \frac{1}{\lambda_k}$. Then $x = \begin{bmatrix} x_k \\ \pm\gamma e_i \end{bmatrix}$ $(\gamma \in \{0,1\})$ and (B.5) reduces to

$$\begin{bmatrix} I-\alpha A & 0 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} Ax_k+\gamma Be_i \\ B^Tx_k \end{bmatrix}.$$

Hence solutions to (B.5) only exist in this case if

$$u-\alpha Au = \lambda_k x_k + \gamma Be_i.$$

Notice that $u-\alpha Au = u-\frac{1}{\lambda_k}Au$ has no component in the x_k direction so that (B.5) has no solution provided that Be_i is not a multiple of x_k (which is assumed). Therefore the eigenvalue $\mu = \alpha = \frac{1}{\lambda_k}$ has algebraic multiplicity $2m+1$ and geometric multiplicity $2m+1$.

Now suppose that $B^Tx = 0$ and $\mu \neq \alpha$. Recall that either $2m$ or $2m+1$ eigenvalues are known so that $n-m$ or $n-m-1$ eigenvalues are left to locate. By (B.3),

$$(\mu A - I)x + (\mu - \alpha)By = 0.$$

If $\mu \neq \frac{1}{\lambda_k}$,

$$(\mu - \alpha)B^T(\mu A - I)^{-1}By = 0,$$

and since $\mu \neq \alpha$,

$$y^TB^T(\mu A - I)^{-1}By = 0 \tag{B.6}$$

If $\mu > \frac{1}{\lambda_1}$, $(\mu A - I)^{-1}$ is positive definite and if $\mu < \frac{1}{\lambda_n}$, $(\mu A - I)^{-1}$ is negative definite and so in both these cases (B.6) implies $y = 0$, and so by (B.3),

$$\mu Ax = x.$$

If $x \neq x_k$ (a case that has already been discussed), the only solution is the trivial solution $x = 0$. Hence there are no eigenvalues outside $(\frac{1}{\lambda_n}, \frac{1}{\lambda_1})$, and so the remaining $n-m$ or $n-m-1$ eigenvalues must all lie in this interval.

This accounts for all of the eigenvalues of $\mathcal{A}^{-1}\mathcal{B}$ since if $B^Tx = 0$ and $\mu = \alpha$ it must be the case that $\alpha Ax = x$, whence A has an eigenvector in the nullspace of B^T, a contradiction. □

Corollary B.1.2 *If A has k eigenvectors lying in the nullspace of B^T then the eigenvalues of $\mathcal{A}^{-1}\mathcal{B}$ are*

$\mu = \alpha$ *with algebraic multiplicity $2m+k$ and geometric multiplicity $m+k$*

and $\frac{1}{\lambda_n} \leq \mu \leq \frac{1}{\lambda_1}$ *$n-m-k$ times*

unless $\alpha = \frac{1}{\lambda_i}$ in which case $\mu = \alpha$ is an eigenvalue of algebraic and geometric multiplicity $2m+k+1$ (when λ_i is simple as an eigenvalue of A). The result generalises when λ_i is not simple to $2m+k+j$ where j is the geometric multiplicity of λ_i).

Proof Trivial extension of Theorem B.1.1. □

Corollary B.1.3 *If A has no eigenvectors lying in the nullspace of B^T, the eigenvalues of the generalised eigenvalue problem*

$$\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \lambda \begin{bmatrix} I & \alpha B \\ \alpha B^T & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

are

$\lambda = \frac{1}{\alpha}$ *with algebraic multiplicity $2m$ and geometric multiplicity m*

or $\lambda \in [\lambda_1, \lambda_n]$ $n-m$ times

unless $\alpha = \frac{1}{\lambda_k}$ in which case $\lambda = \frac{1}{\alpha}$ is an eigenvalue of algebraic and geometric multiplicity $2m+1$. (When λ_k is simple as an eigenvalue of A. The result generalises when λ_k is not simple to $2m+j$ where j is the geometric multiplicity of λ_k).

Proof Immediate consequence of Theorem B.1.1. □

Corollary B.1.3 generalises trivially to the case when A has eigenvectors in the nullspace of B^T.

Notice the similarity between the eigenvalue intervals of $\mathcal{A}$ and those of $\mathcal{B}^{-1}\mathcal{A}$. Preconditioning by the pressure correction / projection matrix $\mathcal{B}^{-1}$ has the effect of squeezing the intervals $\mathcal{I}^-$ and $\mathcal{I}_2^+$ onto the set $\{\frac{1}{\alpha}\}$.

If standard iterative method such as GMRES were being used on the preconditioned system $\mathcal{B}^{-1}\mathcal{A}$, it would be wise to choose α so that $\frac{1}{\alpha} \in [\lambda_1, \lambda_n]$, this would remove the possibility of there being an isolated eigenvalue outside the interval which may hamper convergence, especially if α were very large, so that $\frac{1}{\alpha}$ is close to zero. Then $\kappa(\mathcal{B}^{-1}\mathcal{A}) \leq \kappa(A)$ so that an iterative method used to solve the positive-definite but unsymmetric preconditioned system $\mathcal{B}^{-1}\mathcal{A}$ should display a similar (or better) convergence rate than the conjugate gradient algorithm applied to a system with A as coefficient matrix, although this will obviously be dependent on the clustering of the eigenvalues of A.

B.2 Nullspace methods

Since the solution $\begin{bmatrix} x^T & y^T \end{bmatrix}^T$ of

$$\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix} \tag{B.7}$$

satisfies $B^T x = 0$, it is known that x lies in the nullspace of B^T, denoted $\mathcal{N}(B^T)$. Methods of solution of (B.7) which make use of this fact are known as nullspace methods. If $z_i,\ i = 1, \ldots, n-m$ is a basis for $\mathcal{N}(B^T)$ and $Z = [z_1\ z_2 \ldots z_{n-m}]$ then x can be written as a linear combination of these basis vectors and so $\exists s \in \mathbb{R}^{n-m}$ such that $x = Zs$. Then

$$\begin{aligned} AZs + By &= b \\ \text{and so} \quad Z^T AZs &= Z^T b, \end{aligned} \tag{B.8}$$

since $Z^T B = 0$. The coefficient matrix in (B.8) is called the projected (or reduced) Hessian. Notice that the system (B.8) is $n - m$ dimensional, i.e. that the dimension of the original problem has been reduced by $2m$, and that (B.8) is independent of y. Although this approach to solving (B.7) may appear attractive, finding a basis for $\mathcal{N}(B^T)$ is difficult for all but very small problems, and the projected Hessian lacks the sparsity of the original problem. For this reason nullspace methods are usually neglected for large systems.

It has been remarked that the coefficient matrix $\mathcal{A}$ in (B.7) is symmetric but indefinite, however $\mathcal{A}$ can be considered positive definite when restricted to a special subspace of $\mathbb{R}^{n \times m}$. Let

$$\mathcal{S} = \mathcal{N}(B^T) \backslash \{0\} \times \mathbb{R}^m,$$

then for $z^T = \begin{bmatrix} x^T & y^T \end{bmatrix}^T \in \mathcal{S}$,

$$\begin{aligned} \begin{bmatrix} x^T & y^T \end{bmatrix}^T \begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} &= x^T Ax + 2y^T \left(B^T x\right), \\ &= x^T Ax \quad \text{since } x \in \mathcal{N}(B^T), \\ &> 0. \end{aligned}$$

Therefore the coefficient matrix in (B.7) is a positive definite self-adjoint operator with respect to the usual Euclidean inner product when restricted to $\mathcal{S}$, and so solution methods which rely on the coefficient matrix being symmetric positive definite, such as the conjugate gradient algorithm, will be applicable provided that the iteration vectors can be constrained to lie $\mathcal{S}$. Notice that if $z_1^T = \begin{bmatrix} x^T & y_1^T \end{bmatrix}^T$ and $z_2^T = \begin{bmatrix} x^T & y_2^T \end{bmatrix}^T$ then $z_1^T \mathcal{A} z_1^T = z_2^T \mathcal{A} z_2^T$ and hence the inner product is independent of the y component.

B.3 The action of $\mathcal{B}$

Notice that the matrix $\mathcal{B}$ can be considered to be a projection matrix, since if

$$\begin{bmatrix} I & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} \overline{v} \\ q \end{bmatrix} = \begin{bmatrix} v \\ 0 \end{bmatrix}, \tag{B.9}$$

then $B^T v = 0$, and if $B^T w = 0$ then

$$w^T(\overline{v} - v) = w^T B q = 0,$$

so that the residual $\overline{v} - v$ is orthogonal to the span of $\mathcal{N}(B^T)$. Hence x is the orthogonal projection of $\overline{v}$ onto $\mathcal{N}(B^T)$.

If $\mathcal{A}$ represents a groundwater flow matrix with a Dirichlet boundary condition then (B.9) could arise as a discretisation of the problem

$$\begin{aligned} \overline{u} - \nabla p &= u \text{ in } \Omega \\ \nabla \cdot \overline{u} &= 0 \text{ in } \Omega \\ p &= 0 \text{ on } \partial\Omega. \end{aligned}$$

Similarly to above, if $v \in \mathrm{H}_0^1(\Omega)$ satisfies $\nabla \cdot v = 0$ then

$$\begin{aligned} \langle v\ ,\ \overline{u} - u \rangle_{\mathrm{L}^2(\Omega)} &= \int_\Omega v \cdot \nabla p \, dx, \\ &= \int_{\partial\Omega} p v \cdot n \, d\gamma \ - \int_\Omega p \nabla \cdot v \, dx, \\ &= 0, \end{aligned}$$

and hence $\overline{u}$ is the orthogonal $\mathrm{L}^2(\Omega)$ projection from u onto the space of divergence free functions. p can be recovered as the solution of a Poisson equation and so if a fast Poisson solver is available, the action of $\mathcal{B}^{-1}$ can be computed implicitly.

It is not difficult to show that $\begin{bmatrix} \overline{v}^T & q \end{bmatrix}^T$ in (B.9) satisfies

$$\begin{bmatrix} \overline{v} \\ q \end{bmatrix} = \begin{bmatrix} \Pi v \\ (B^TB)^{-1}B^Tv \end{bmatrix},$$

where $\Pi = I - B(B^TB)^{-1}B^T$ is the orthogonal projection matrix [29] onto $\mathcal{N}(B^T)$, and hence the update from v to $\overline{v}$ can be calculated at a cost of one projection evaluation. The preconditioned system $\mathcal{B}^{-1}\mathcal{A}z = \mathcal{B}^{-1}f$ can then be written as

$$\begin{bmatrix} \Pi A + (I - \Pi) & 0 \\ (B^TB)^{-1}B^T(A - I) & I \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \Pi b \\ (B^TB)^{-1}B^Tb \end{bmatrix}.$$

Therefore, preconditioning with the projection matrix $\mathcal{B}$ has the effect of removing the dependence on y in the first equation (c.f. LSQR(A^{-1}) which eliminates x dependence). Since $B^Tx = 0$, the equation can be formulated as a problem in x only,

$$\Pi A \Pi x = \Pi b,$$

(since $x = \Pi x$). The coefficient matrix $\Pi A \Pi$ is similar to the projected Hessian in (B.8), however unlike Z^TAZ, $\Pi A \Pi$ is an $n \times n$ singular matrix, and comparing with the theorems of §B.1 it is easy to see that the the nullspace of $\Pi A \Pi$ is m dimensional. Hence, the projection matrix Π allows the dimension of the problem to be reduced from $n + m$ to $n - m$ (although the y solution has not been obtained).

B.4 The ΠCG algorithm

In this section a nullspace algorithm, ΠCG , for (B.7) is derived by first considering the Conjugate Gradient algorithm applied to the system (B.7) on the space $\mathcal{S}$. The Conjugate Gradient algorithm for the solution of $\mathcal{A}z = f$ where $\mathcal{A} \in \mathbb{R}^{n \times n}$ is a symmetric positive definite matrix is given by,

CG algorithm for symmetric, positive definite systems

Pick $z_0 \in \mathbb{R}^n$. Set $r_0 = \mathcal{A}z - f$, $d_0 = -r_0$ and iterate,

1. $z_{i+1} = z_i + \alpha_i d_i, \qquad \alpha_i = -\frac{\langle r_i\,,\,d_i\rangle}{\langle d_i\,,\,d_i\rangle_{\mathcal{A}}}$,
2. $r_{i+1} = \mathcal{A}z_{i+1} - f$,
3. $d_{i+1} = -r_{i+1} + \beta_i d_i, \quad \beta_i = \frac{\langle r_{i+1}\,,\,d_i\rangle_{\mathcal{A}}}{\langle d_i\,,\,d_i\rangle_{\mathcal{A}}}$,

where $\langle\cdot\,,\,\cdot\rangle$ denotes the usual Euclidean inner product in $\mathbb{R}^n$ and $\langle\cdot\,,\,\cdot\rangle_{\mathcal{A}}$ denotes the $\mathcal{A}$ inner product. The vectors r_i are the residual, d_i the search direction, and z_i the approximate solution at the ith step. Note that this is by no means the most efficient implementation of the CG algorithm, it is however simple to observe well known orthogonality properties of the iteration vectors (see [44]) from this version.

Now let $\mathcal{A}$ be the coefficient matrix in (B.7). The ΠCG algorithm can be derived from the CG algorithm as follows. Any iteration vector in the CG algorithm has an x component and a y component. Given an iteration vector g let g^x, g^y denote the x and y components of g respectively. Then the ΠCG algorithm will only be stable if all x components of the iteration vectors lie in $\mathcal{N}(B^T)$. Therefore the first requirement is that $z_0^x \in \mathcal{N}(B^T)$. This can be achieved either by projecting z_0^x onto $\mathcal{N}(B^T)$ by premultiplying by the matrix Π from §B.3 or simply by setting $z_0^x = 0$. For simplicity the second option will be used. Next it must be ensured that $r_0^x \in \mathcal{N}(B^T)$ so let $r_0 = \Pi_I(\mathcal{A}\Pi_I z_0 - f)$, where

$$\Pi_I = \begin{bmatrix} \Pi & 0 \\ 0 & I \end{bmatrix}.$$

Since $d_0 = -r_0$, no projection of the initial search direction vector is required.

Suppose that at step $i+1$ of the iteration, $z_i^x, r_i^x, d_i^x \in \mathcal{N}(B^T)$ and consider the update for the x component of the solution from step 1,

$$z_{i+1}^x = z_i^x + \alpha_i d_i^x.$$

Since $z_i^x, d_i^x \in \mathcal{N}(B^T)$ it follows that $z_{i+1}^x \in \mathcal{N}(B^T)$. Step 2 doesn't necessarily ensure that $r_{i+1}^x \in \mathcal{N}(B^T)$ and so an extra iteration step

$$2.5 \quad r_{i+1}^x = \Pi r_{i+1}^x$$

needs to be inserted between steps 2 and 3. Step 3 then updates the search direction. Since $d_i^x \in \mathcal{N}(B^T)$ and $r_i^x \in \mathcal{N}(B^T)$ (after step 2.5), d_{i+1}^x is automatically lies in $\mathcal{N}(B^T)$. Therefore after each iteration of the algorithm new approximate solution, residual and search direction vectors are found which all have x components which are elements of $\mathcal{N}(B^T)$. Since $z_0^x, r_0^x, d_0^x \in \mathcal{N}(B^T)$ all iteration vectors will have discretely divergence free x components if the extra projection step described above is included in the CG algorithm for $\mathcal{A}$. Notice that one projection needs to be done before iteration can commence, then only one projection step is required per iteration. (Two initial projections are required if z_0 is chosen such that $z_0^x \neq 0$).

As shall be seen there are some redundant operations in this algorithm. To identify them write

$$z_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix}.$$

Then, renumbering the steps, the algorithm becomes,

Componentwise CG algorithm with projections

Choose $\begin{bmatrix} x_0^T & y_0^T \end{bmatrix}^T \in \mathbb{R}^n$. Set $x_0 = \Pi x_0$, $r_0^x = \Pi(Ax_0 + By_0 - b)$, $r_0^y = B^T x_0$ and $\begin{bmatrix} d_0^{x^T} & d_0^{y^T} \end{bmatrix}^T = -\begin{bmatrix} r_0^{x^T} & r_0^{y^T} \end{bmatrix}^T$ and iterate

```
1. $x_{i+1} = x_i + \alpha_i d_i^x,$        $\alpha_i = -\frac{\langle r_i^x , d_i^x \rangle + \langle r_i^y , d_i^y \rangle}{\langle d_i^x , d_i^x \rangle_A + 2\langle B^T d_i^x , d_i^y \rangle},$
   $y_{i+1} = y_i + \alpha_i B^T x_i,$
2. $r_{i+1}^x = A x_{i+1} + B y_{i+1} - b,$
   $r_{i+1}^y = B^T x_{i+1},$
3. $r_{i+1}^x = \Pi r_{i+1}^x,$
4. $d_{i+1}^x = -r_{i+1}^x + \beta_i d_i^x,$        $\beta_i = \frac{\langle r_i^x , d_i^x \rangle_A + \langle B^T r_i^x , d_i^y \rangle + \langle B^T d_i^x , r_i^y \rangle}{\langle d_i^x , d_i^x \rangle_A + 2\langle B^T d_i^x , d_i^y \rangle},$
   $d_{i+1}^y = -r_{i+1}^y + \beta_i d_i^y.$
```

Notice that since it is guaranteed that $x_i, r_i^x, d_i^x \in \mathcal{N}(B^T)$, $B^T x_i = B^T r_i^x = B^T d_i^x = 0$ and so all calculations involving these terms vanish. Then step 1 implies that

$$y_{i+1} = y_i + B^T x_i = y_i,$$

so that the y approximation never gets updated, i.e. the algorithm (as it stands) doesn't solve (B.7) for y (c.f. the LSQR(A^{-1}) algorithm which only solves for y). Therefore it is sensible to take $y_0 = 0$ which results in all y terms in the algorithm becoming redundant (including r_i^y and d_i^y). It will be shown that it is still possible to update a pressure approximation without these terms and at negligible extra cost.

The refined algorithm which results from these simplifications will be referred to as the ΠCG algorithm for velocity and acts as follows.

ΠCG algorithm for velocity

Choose $x_0 \in \mathbb{R}^n$. Set $x_0 = \Pi x_0, r_0 = \Pi(Ax_0 - b), d_0 = -r_0$ and iterate

1. $x_{i+1} = x_i + \alpha_i d_i, \qquad \alpha_i = -\frac{\langle r_i \, , \, d_i \rangle}{\langle d_i \, , \, d_i \rangle_A},$
2. $r_{i+1} = Ax_{i+1} - b,$
3. $r_{i+1} = \Pi r_{i+1},$
4. $d_{i+1} = -r_{i+1} + \beta_i d_i, \quad \beta_i = \frac{\langle r_{i+1} \, , \, d_i \rangle_A}{\langle d_i \, , \, d_i \rangle_A}.$

Notice that all residuals and search directions involve only x components terms so that the superscript x's have been dropped. Once again this is not the most cost effective implementation of the algorithm but merely a 'nice' version, and note that taking $x_0 = 0$ simplifies the first steps of the algorithm.

Attention is now turned to finding an approximation for the pressure at each iteration step. First observe that by steps 1 and 2 of the algorithm,

$$\begin{aligned} r_{i+1} &= A(x_i + \alpha_i d_i) - b \\ &= (Ax_i - b) + \alpha_i A d_i \\ &= r_i + \alpha_i A d_i \end{aligned} \qquad \text{(B.10)}$$

thus only one $A \times$ vector is required at each step of ΠCG and so step 2 of the algorithm is replaced by (B.10). (Calculation of both α_i and β_i only require Ad_i). Step 3 then becomes

$$r_{i+1} = r_i + \alpha_i \Pi A d_i, \qquad \text{(B.11)}$$

since $\Pi r_i = r_i$ at step $i + 1$ of the iteration. Hence the projection operation has been shifted to act on Ad_i in place of r_{i+1}.

Notice that if x_k is the approximation to x at step k of the ΠCG algorithm, a corresponding y approximation can be obtained by setting

$$y_k = (B^TB)^{-1}B^T(b - Ax_k),$$

and so

$$y_{k+1} = (B^TB)^{-1}B^T(b - Ax_{k+1}).$$

Therefore

$$y_{k+1} - y_k = -(B^TB)^{-1}B^TA(x_{k+1} - x_k),$$

and hence

$$\begin{aligned} y_{k+1} &= y_k - \alpha_k(B^TB)^{-1}B^TAd_k, \text{ by step 1 of } \Pi\text{CG }, \\ &=: y_k - \alpha_k t_k. \end{aligned} \tag{B.12}$$

Recall now that the projection step has been shifted to act on Ad_k at the kth step and

$$\begin{aligned} \Pi Ad_k &= (I - B(B^TB)^{-1}B^T)Ad_k, \\ &= Ad_k - Bt_k, \end{aligned} \tag{B.13}$$

hence the pressure can be updated as a preliminary step to forming ΠAd_k. Notice that the extra cost of the pressure update is just one VAXPY operation (of dimension m).

Combining all of the above ideas an efficient version of the algorithm for solving (B.7) is obtained which will be referred to as the ΠCG algorithm.

The ΠCG algorithm

Choose $x_0 \in \mathbb{R}^n$. Set $x_0 = \Pi x_0$, $t_0 = (B^TB)^{-1}B^Tb$, $y_0 = t_0$, $r_0 = \Pi A x_0 - b + Bt_0$, $d_0 = -r_0$ and iterate

```
1. $x_{i+1} = x_i + \alpha_i d_i$        $\alpha_i = -\frac{\langle r_i , d_i \rangle}{\langle d_i , d_i \rangle_A}$
2. $t_i = (B^TB)^{-1}B^T(Ad_i)$
3. $r_{i+1} = r_i + \alpha_i(Ad_i - Bt_i)$
4. $y_{i+1} = y_i - \alpha_i t_i$
5. $d_{i+1} = -r_{i+1} + \beta_i d_i$        $\beta_i = \frac{\langle r_{i+1} , d_i \rangle_A}{\langle d_i , d_i \rangle_A}$
```

From the above development and §B.3 it is clear that the following lemma holds.

Lemma B.4.1 *ΠCG applied to the system (B.7) is equivalent to preconditioning (B.7) with*

$$\begin{bmatrix} I & B \\ B^T & 0 \end{bmatrix}$$

and solving the equation in velocity only by CG, with pressure updates being performed using the coupled pressure and velocity equation.

Notice that taking $x_0 = 0$ makes the projection of x_0 and Ax_0 redundant and that the only extra cost of this algorithm over an efficient implementation of ΠCG for velocities is the one VAXPY operation to update pressure per iteration. Also notice that the number of matrix-vector multiplications per iteration is consistent with CG applied to (B.7), i.e. one $A\times$ vector, one $B\times$ vector and one $B^T\times$ vector per iteration. Also observe that the $n+m$ dimensional inner products of CG have been replaced by n dimensional ones and that the two $n+m$ dimensional VAXPY operations of CG (in updating residuals and search directions) have been replaced by n dimensional operations. Hence the only extra operation of ΠCG applied to (B.7) over CG applied to (B.7) is a solve of the form

$$(B^TB)v = w$$

at each step of the iteration. (c.f. LSQR(A^{-1}) which requires a solve with the matrix A at each step of the iteration).

B.5 Convergence of the ΠCG algorithm

It is well known that when the CG algorithm is applied to symmetric positive definite systems, the residual vectors are conjugate in the usual Euclidean inner product space and the search directions are conjugate in the $\mathcal{A}$–inner product space (in exact arithmetic). i.e. if $g_i = \mathcal{A}z_i - f$ is the residual and h_i is the search direction at step i of the CG algorithm then after k iterations

$$\langle g_i\,,\,g_j\rangle = 0,\ \ \langle h_i\,,\,h_j\rangle_{\mathcal{A}} = 0\ \ \forall i \neq j \leq k.$$

This behaviour leads to the theoretical result that, for exact arithmetic, the CG algorithm should converge to the exact solution in at most l steps, where l is the dimension of the problem (or less if $\mathcal{A}$ has repeated eigenvalues).

The ΠCG algorithm acts on the system

$$\Pi A \Pi x = \Pi b, \tag{B.14}$$

and since $span(\Pi A \Pi)$ is $n-m$ dimensional, similar results to those for g_i and h_i above should lead to the conclusion that, again for exact arithmetic, the ΠCG algorithm applied to (B.14) should converge to the exact solution in at most $n-m$ steps. Notice that since the ΠCG algorithm algorithm also updates pressure vectors, the $n+m$ dimensional system (B.7) is solved in at most $n-m$ steps, compared to $n+m$ steps if the CG algorithm were applied directly to (B.7) (and if it converged). The analogous results for r_i and d_i, the residual and search directions of the ΠCG algorithm follow by some standard CG type analysis and are now given.

Lemma B.5.1 *After l iterations of the ΠCG algorithm,*

$$span\{d_0,\dots,d_l\} = span\{r_0,\dots,r_l\} = span\{r_0, \overline{A}r_0, \dots, \overline{A}^l r_0\},$$

where $\overline{A} = \Pi A \Pi$, and

$$\begin{aligned} &(a)\quad \langle d_i\,,\,d_j\rangle_A = 0\ \ \forall i \neq j \leq l,\\ &(b)\quad \langle r_i\,,\,r_j\rangle = 0\ \ \forall i \neq j \leq l. \end{aligned}$$

Proof Trivial. □

Corollary B.5.2 *The ΠCG algorithm for (B.7) converges in at most $n-m$ steps in exact arithmetic.*

Proof Immediate consequence of Lemma B.5.1 and that $dim(\mathcal{N}(B^T)) = dim(span(\overline{A}) = n-m$. □

It can be shown that the error at the kth iterate of CG for the symmetric positive definite system $\mathcal{A}z = f$ is of the form

$$||z - z_k||_{\mathcal{A}} \leq 2\,||z - z_0||_{\mathcal{A}} \left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^k, \tag{B.15}$$

where κ is the condition number of $\mathcal{A}$ (see [29] and §2.2.1). Similarly, the ΠCG algorithm satisfies an error bound of the form

$$||u - u_k||_A \leq 2\,||u - u_0||_A \left(\frac{\sqrt{\overline{\kappa}}-1}{\sqrt{\overline{\kappa}}+1}\right)^k, \tag{B.16}$$

where

$$\overline{\kappa} = \frac{\lambda_1(\overline{A})}{\lambda_{n-m}(\overline{A})} \leq \kappa(A), \tag{B.17}$$

and $\lambda_1(\overline{A}), \lambda_{n-m}(\overline{A})$ denote the non-zero eigenvalues of smallest and largest modulus of $\overline{A}$ (it is assumed that the eigenvalues of $\overline{A}$ are ordered so the $\lambda_{n-m+1}, \cdots, \lambda_n = 0$). As is the case for similar error bounds for the CG algorithm, this error bound is generally pessimistic in practice. Note that the zero eigenvalues of $\overline{A}$ can be ignored since directions with coefficients in directions of eigenvectors corresponding to zero eigenvalues never occur within ΠCG since all iteration vectors are elements of $\mathcal{N}(B^T)$. Hence the convergence of the ΠCG algorithm is completely determined by A and is independent of B. Compare this, for example, with the error estimate for the pressure terms of the Uzawa algorithm (see [19]),

$$||y - y_k||_2 \leq \left[\rho(I - \alpha B^T A^{-1} B)\right]^k ||y - y_0||_2, \tag{B.18}$$

and the optimal value of $\rho(I - \alpha B^T A^{-1} B) = \frac{\kappa(B^T A^{-1} B)-1}{\kappa(B^T A^{-1} B)+1}$.

B.6 Generalisation of ΠCG and a connection with LSQR(D^{-1})

Since $\kappa(\overline{A})$ governs the convergence of ΠCG it would be wise to make $\kappa(\overline{A})$ as small as possible by preconditioning A, since $\kappa(\overline{A}) \leq \kappa(A)$. In §5.4 it was seen that diagonal preconditioners are effective for the mass matrix A, hence consider the preconditioned system

$$\begin{bmatrix} D^{-\frac{1}{2}}AD^{-\frac{1}{2}} & D^{-\frac{1}{2}}B \\ B^TD^{-\frac{1}{2}} & 0 \end{bmatrix} \begin{bmatrix} D^{\frac{1}{2}}x \\ y \end{bmatrix} = \begin{bmatrix} D^{-\frac{1}{2}}b \\ 0 \end{bmatrix}. \tag{B.19}$$

Then $\kappa(D^{-\frac{1}{2}}AD^{-\frac{1}{2}})$ is independent of k (and h for some choices of finite elements), where k is the permeability function in the groundwater flow equations, and so ΠCG for (B.19) should converge faster than ΠCG for $\mathcal{A}$ and furthermore should converge independently of k. If ΠCG on $\mathcal{A}$ is called 'ΠCG with projection matrix Π' and ΠCG on (B.19) is denoted 'ΠCG(D) with projection matrix Π_D' then Π is given by

$$\Pi = I - B(B^TB)^{-1}B^T,$$

and its action can be computed with LSQR solve on $\mathcal{B}$. On the other hand, the projection matrix Π_D is given by

$$\Pi_D = I - D^{-\frac{1}{2}}B(B^TD^{-1}B)^{-1}B^TD^{-\frac{1}{2}},$$

and it's action is equivalent to that of the inverse of

$$\begin{bmatrix} D & B \\ B^T & 0 \end{bmatrix},$$

which can be computed using the LSQR(D^{-1}) algorithm. Saunders has considered the intricacies of using LSQR to compute projections in [71], and analogous results to all those presented there will hold for LSQR(D^{-1}) .

B.7 Numerical results

The ΠCG algorithm was applied to the Harwell problem TTMX2 described in §5.6, the error $||x - x_k||_2$ can be seen in Figure B-1. To compare convergence, the MINRES algorithm has also been used to solve TTMX2. Clearly the ΠCG algorithm is much faster in terms of iterations, the error (in the x component) after 3 iterations of ΠCG is in fact smaller than the error (in the x component) after 300 iterations of MINRES, although it must be remembered that each ΠCG iteration is much more expensive than each MINRES iteration, since it requires a Poisson solve at each step. For this example the time taken for 10 steps of ΠCG was equivalent to approximately 300 steps of MINRES, although the error after this time is of order 10^{-3} for MINRES whereas the error for ΠCG is of order 10^{-8}.

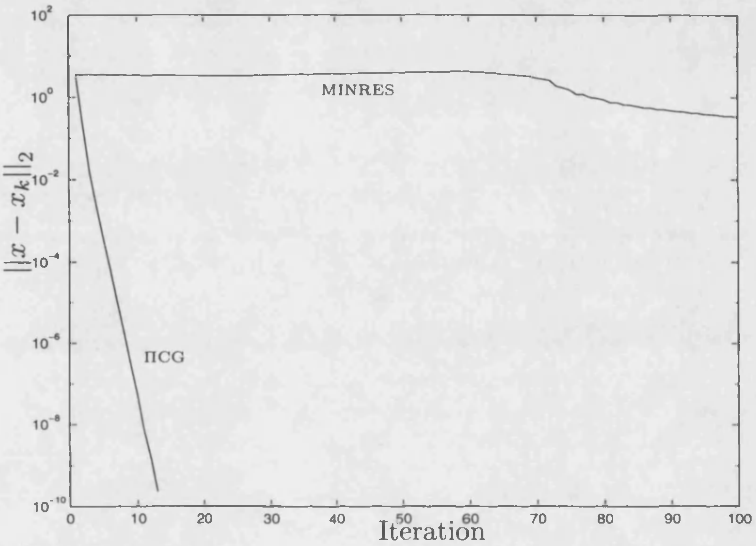


Figure B-1: Comparison of ΠCG and MINRES iterations for the x-error

In order to make ΠCG a realistic algorithm it needs to be combined with an efficient method of solving the Poisson subproblem, for example a multigrid solver.

B.8 Conclusions

The ΠCG algorithm will not be competitive with LSQR(A^{-1}) for the groundwater flow problems, since its inner solves are far too expensive compared with the (relatively) simple mass matrix subproblems involved at each step of LSQR(A^{-1}) . ΠCG has the property of performing a minimisation of velocity solution error, whilst LSQR(A^{-1}) minimises the pressure error norm. For solving Stokes problems it may be the case that ΠCG is more realistic than LSQR(A^{-1}) since there the inner Poisson solves can be thought of as a coarse grid solve compared to the Laplacian matrix A, and so ΠCG can be thought of as a multigrid type algorithm in this case, whereas the LSQR(A^{-1}) inner solves will be of a large Poisson problem. This remains to be investigated.

Bibliography

[1] G. Arfken. *Mathematical Methods for Physicists.* Academic Press, New York, 2nd edition, 1970.

[2] S.F. Ashby. *Polynomial Preconditioning for Conjugate Gradient Methods.* PhD thesis, Dept. of Computer Science, University of Illinois, Urbana, IL, 1987. Tech. Report 1355.

[3] S.F. Ashby, T.A. Manteuffel, and J.S. Otto. A comparison of adaptive Chebyshev and least squares polynomial preconditioning for hermitian positive definite linear systems. *SIAM J. Sci. Statist. Comput.*, 13(1):1–29, January 1992.

[4] O. Axelsson. *Iterative solution methods.* Cambridge University Press, 1994.

[5] S.J. Benbow. Extending LSQR to generalised least-squares problems without resorting to Cholesky decompositions. Preprint 97/03, University of Bath, February 1997.

[6] D. Braess and R. Sarazin. An efficient smoother for the Stokes problem. *Applied Numerical Mathematics*, 23(1):3–19, 1997.

[7] S.C. Brenner and L.R. Scott. *The Mathematical Theory of Finite Element Methods.* Springer-Verlag, 1994.

[8] F. Brezzi. On the existence, uniqueness and approximation of saddle-point problems arising from lagrangian multipliers. *R.A.I.R.O. Anal. Numer.*, R.2:129–151, 1974.

[9] F. Brezzi, L.D. Marini, and P. Pietra. Numerical simulation of semiconductor devices. *Comp. Meth. Appl. Mech. Eng.*, 75:493–514, 1989.

[10] F.F. Campos and J.S. Rollett. Analysis of preconditioners for the Conjugate Gradient method. Technical Report 94/10, Oxford University Computing Laboratory - NAGp, 1994.

[11] T.F. Chan and T.P. Matthew. Domain decomposition algorithms. *Acta Numerica*, pp. 61–143, 1994.

[12] T.F. Chan, B. Smith, and J. Zou. Overlapping schwarz methods on unstructured meshes using non-matching coarse grids. *Numer. Math.*, 73:149–167, 1996.

[13] P.G. Ciarlet. *The Finite Element Method for Elliptic Problems.* North-Holland Publishing Company, 1978.

[14] R. Dautray and J. Lions. *Analyse Mathématique et Calcul Numérique Pour Les Sciences Et Les Techniques*, volume 1. Masson, 1984.

[15] P.F. Dubois, A. Greenbaum, and G.H. Rodrigue. Approximating the inverse of a matrix for use on iterative algorithms on vector processors. *Computing*, 22:257–268, 1979.

[16] I.S. Duff. The solution of augmented systems. In D.F. Griffiths and G.A. Watson, editors, *Proceedings of the 1995 Biennial Conference on Numerical Analysis*, volume 344 of *Pitman Research Notes in Mathematics.* Addison Wesley Longman Ltd, April 1996.

[17] T. Dupont and R. Scott. Polynomial approximation of functions in sobolev spaces. *Math. Comp.*, 34:441–463, 1980.

[18] N. Dyn and W.E Ferguson Jr. The numerical solution of equality-constrained quadratic programming problems. *Math. Comput.*, 41(163):165–170, 1983.

[19] H.C. Elman and G.H. Golub. Inexact and preconditioned Uzawa algorithms for saddle-point problems. *SIAM J. Numer. Anal.*, 31(6):1645–1661, 1994.

[20] B. Fischer. *Polynomial based iteration methods for symmetric linear systems.* Advances in numerical mathematics. Wiley-Teubner, 1996.

[21] B. Fischer and G.H. Golub. On generating polynomials which are orthogonal over several intervals. *Math. Comp.*, 56(194):711–730, April 1991.

[22] B. Fischer, A. Ramage, D. Silvester, and A.J. Wathen. Minimum residual methods for augmented systems. Technical Report 15, University of Strathclyde, June 1995.

[23] G.E. Forsythe and E.G. Strauss. On the best conditioned matrices. In *Proceeding of the Amer. Math. Soc. 6*, pp. 340–345. Amer. Math. Soc., 1955.

[24] L.E. Fraenkel. Personal communication.

[25] W. Gautschi. On generating orthogonal polynomials. *SIAM J. Sci. Statist. Comput.*, 3:289–317, 1982.

[26] V. Girault and H. Lopez. Finite-element error estimates for the MAC scheme. *IMA J. Numer. Anal.*, 16:347–379, 1996.

[27] G.H. Golub and B. Fischer. How to generate unknown orthogonal polynomials out of known orthogonal polynomials. Technical Report NA-91-06, Stanford University, November 1991.

[28] G.H. Golub and W. Kahan. Calculating the singular values and pseudoinverse of a matrix. *SIAM J. Numer. Anal.*, 2:205–224, 1965.

[29] G.H. Golub and C.F. Van Loan. *Matrix Computations.* John Hopkins University Press, 2nd edition, 1989.

[30] G.H. Golub and R.S. Varga. Chebyshev semi-iterative methods, successive overrelaxation iterative methods, and second order richardson iterative methods, part I. *Numer. Math.*, 3:147–156, 1961.

[31] G.H. Golub and R.S. Varga. Chebyshev semi-iterative methods, successive overrelaxation iterative methods, and second order richardson iterative methods, part II. *Numer. Math.*, 3:157–168, 1961.

[32] G.H. Golub and A.J. Wathen. An iteration for indefinite systems and its application to the Navier-Stokes equations. To appear, *SIAM J. Sci. Comput.*, 19(2), March 1998.

[33] I.G. Graham and M.J. Hagger. Unstructured additive schwartz - cg method for elliptic problems with highly discontinuous coefficients. SIAM J. Sci. Comput., to appear.

[34] I.G. Graham and M.J. Hagger. Additive schwartz, cg and discontinuous coefficients. Preprint 96/19, University of Bath, 1996.

[35] Benqui Guo and Weiming Cao. Adaptive schwartz methods for the $h-p$ version of the finite element method in two dimensions. *SIAM J. Sci. Comput.*, 18(5):1267–1288, 1997.

[36] M.R. Hestenes and E.Stiefel. Methods of conjugate gradients for solving linear systems. *J. Res. N.B.S.*, 49:409–436, 1952.

[37] N.J. Higham. The Test Matrix Toolbox for MATLAB (Version 3.0). Technical Report 276, UMIST, September 1995.

[38] P.D. Hough and S.A. Vavasis. Complete orthogonal decomposition for weighted least squares. *SIAM J. Matrix Anal. Appl.*, 18(2):369–392, 1997.

[39] A.S. Householder. *The theory of matrices in numerical analysis.* Dover, 1975.

[40] Liang Jingwei. On the Distribution of Matrix Eigenvalues and Its Applications in Numerical Analysis. Presented at the IMACS symposium.

[41] C. Johnson. *Numerical solution of partial differential equations by the finite element method.* Cambridge University Press, 1987.

[42] J. Johnston. *Econometric methods.* McGraw-Hill, New York, 2nd edition, 1972.

[43] N. Karmarkar. A new polynomial-time algorithm for linear-programming. *Combinatorica*, 4(4):373–395, 1984.

[44] C.T. Kelley. *Iterative Methods for Linear and Nonlinear Equations.* Number 16 in Frontiers in applied mathematics. SIAM, 1995.

[45] L.K. Kuiper. A comparison of the incomplete cholesky-conjugate gradient method with the strongly implicit method as applied to the solution of two-dimensional groundwater flow equations. *Water Resour. Res.*, 17(4):1082–1086, 1981.

[46] C. Lanczos. An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *J. Res. N.B.S.*, 45:255–282, 1950.

[47] V.I. Lebedev. An iteration method for the solution of operator equations with their spectrum lying on several intervals. *Zh. Vych. Mat. i Fiz.*, 9:1247–1252, 1969.

[48] T.A. Manteuffel. An incomplete factorization technique for positive definite linear systems. *Math. Comput.*, 34(150):473–497, 1980.

[49] M. Marcus and H. Minc. *A survey of matrix theory and matrix inequalities.* Dover, 1992.

[50] E.A. Meese. Combined incomplete lu and strongly implicit procedure preconditioning. In *Copper Mountain conference on iterative methods*, volume I, 1996.

[51] 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. Comput.*, 31(137):148–162, 1977.

[52] M.F. Murphy and A.J.Wathen. On preconditioners for the Oseen equations. Technical Report AM 95-07, University of Bristol, 1995.

[53] D.P. O'Leary. On bounds for scaled projections and pseudoinverses. *Linear Algebra Appl.*, 132:115–117, April 1990.

[54] C.C. Paige. Bidiagonalisation of matrices and solution of linear equations. *SIAM J. Numer. Anal.*, 11(1):197–209, March 1974.

[55] C.C. Paige. Fast numerically stable computations for generalised linear least squares problems. *SIAM J. Numer. Anal.*, 16(1):165–171, February 1979.

[56] C.C. Paige and M.A. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM J. Numer. Anal.*, 12(4):617–629, 1975.

[57] C.C. Paige and M.A. Saunders. Algorithm-583 - LSQR - sparse linear-equations and least-squares problems. *ACM Trans. Math. Softw.*, 8(2):195–209, 1982.

[58] C.C. Paige and M.A. Saunders. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Trans. Math. Softw.*, 8(1):43–71, March 1982.

[59] B.N. Parlett. *The symmetric eigenvalue problem.* Prentice-Hall, 1980.

[60] P.A. Raviart. Mixed finite element methods. In D.F. Griffiths, editor, *The mathematical basis of finite element methods.* Oxford University Press, 1983.

[61] P.A. Raviart and J.M. Thomas. A mixed finite element method for second order elliptic problems. In *Mathematical Aspects of the Finite Element Method*, Lecture Notes in Mathematics 606. Springer-Verlag, 1977.

[62] J.K. Reid. The use of conjugate gradients for systems of equations possessing 'Property A'. *SIAM J. Numer. Anal.*, 9:325–332, 1972.

[63] T.J. Rivlin. *An introduction to the approximation of functions.* Dover, 1981.

[64] T. Rusten and R. Winther. A preconditioned iterative method for saddlepoint problems. *SIAM J. Matrix Anal. Appl.*, 13(3):887–904, July 1992.

[65] H. Rutishauser. Theory of gradient methods. In *Refined Iterative Methods for Computation of the Solution and the Eigenvalues of Self-Adjoint Boundary Value Problems*, pp. 24–49. Institute of Applied Mathematics, Zurich, Basel-Stuttgart, 1959.

[66] Y. Saad. Iterative solution of indefinite symmetric linear systems by methods using orthogonal polynomials over two disjoint intervals. *SIAM J. Numer. Anal.*, 20(4):784–811, August 1983.

[67] Y. Saad. Practical use of polynomial preconditionings for the conjugate gradient method. *SIAM J. Sci. Statist. Comput.*, 6(4):865–881, October 1985.

[68] Y. Saad. *Numerical methods for large eigenvalue problems.* Manchester University Press, 1992.

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

[70] M.A. Saunders. Solution of sparse rectangular systems using LSQR and CRAIG. *BIT*, 7(35):588–604, 1995.

[71] M.A. Saunders. Computing projections with LSQR. *BIT*, 37(1):96–104, 1997.

[72] D. Silvester and A.J. Wathen. Fast iterative solution of stabilised Stokes systems. part II: Using general block preconditioners. *SIAM J. Numer. Anal.*, 31(5):1352–1367, 1994.

[73] G.W. Stewart. *Introduction to Matrix Computations.* Academic Press, New York, 1973.

[74] G.W. Stewart. On scaled projections and pseudoinverses. *Linear Algebra Appl.*, 112:189–193, 1989.

[75] H.L. Stone. Iterative solution of implicit approximations of multidimensional partial differential equations. *SIAM J. Numer. Anal.*, 5:530–558, 1968.

[76] G. Strang. *Introduction to applied mathematics.* Wellesley-Cambridge Press, 1986.

[77] R.S. Varga. *Matrix iterative analysis.* Automatic computation. Prentice-Hall, 1965.

[78] S.A. Vavasis. Stable numerical algorithms for equilibrium systems. *SIAM J. Matrix Anal. Appl.*, 15(4):1108–1131, 1994.

[79] S.A. Vavasis. Stable finite elements for problems with wild coefficients. *SIAM J. Numer. Anal.*, 33(3):890–916, 1996.

[80] H.F. Wang and M.P. Anderson. *Introduction To Groundwater Modelling, Finite Difference and Finite Element Methods.* W.H. Freeman and Company, 1982.

[81] X. Wang and R. Bramley. Abstract: A necessary and sufficient symbolic condition for the existence of incomplete cholesky factorisations. In *Copper Mountain conference on iterative methods*, 1996.

[82] A.J. Wathen. Realistic eigenvalue bounds for the Galerkin mass matrix. *IMA J. Numer. Anal.*, 7:449–457, 1987.

[83] A.J. Wathen. An analysis of some element-by-element techniques. *Comput. Meths. Appl. Mech. Engrg.*, 74:271–287, 1989.

[84] A.J. Wathen. Singular element preconditioning for the finite element method. In R. Beauwens and P. de Groen, editors, *Iterative methods in linear algebra.* Elsevier Science Publishers B.V. (North Holland), 1992.

[85] A.J. Wathen and D.J. Silvester. Fast iterative solution of stabilised Stokes systems. part I: Using simple diagonal preconditioners. *SIAM J. Numer. Anal.*, 30:630–649, 1993.

[86] J.A. Watts III. A conjugate-gradient truncated direct method for the iterative solution of the reservoir simulation pressure equation. *Soc. Pet. Eng.*, 21:345–353, 1981.

[87] J.C. Wheeler. Modified moments and gaussian quadrature. *Rocky Mountain J. Math.*, 4:287–296, 1974.

[88] Gu yixi. The distribution of eigenvalues of a matrix. *ACTA MATHEMATICAE APPLICATAE SINICA*, 17(4), 1994.