19 Matrix Preconditioning Techniques and Applications
The Cambridge Monographs on Applied and Computational Mathematics reflects the crucial role of mathematical and computational techniques in contemporary science. The series publishes expositions on all aspects of applicable and numerical mathematics, with an emphasis on new developments in this fast-moving area of research.

State-of-the-art methods and algorithms as well as modern mathematical descriptions of physical and mechanical ideas are presented in a manner suited to graduate research students and professionals alike. Sound pedagogical presentation is a prerequisite. It is intended that books in the series will serve to inform a new generation of researchers.

Also in this series:

1. A Practical Guide to Pseudospectral Methods, Bengt Fornberg
2. Dynamical Systems and Numerical Analysis, A. M. Stuart and A. R. Humphries
3. Level Set Methods and Fast Marching Methods, J. A. Sethian
4. The Numerical Solution of Integral Equations of the Second Kind, Kendall E. Atkinson
5. Orthogonal Rational Functions, Adhemar Bultheel, Pablo González-Vera, Erik Hendriksen, and Olav Njåstad
6. The Theory of Composites, Graeme W. Milton
7. Geometry and Topology for Mesh Generation Herbert Edelsbrunner
8. Schwarz–Christoffel Mapping Tofin A. Driscoll and Lloyd N. Trefethen
10. Practical Extrapolation Methods, Avram Sidi
11. Generalized Riemann Problems in Computational Fluid Dynamics, Matania Ben-Artzi and Joseph Falcovitz
12. Radial Basis Functions: Theory and Implementations, Martin Buhmann
13. Iterative Krylov Methods for Large Linear Systems, Henk A. van der Vorst
14. Simulating Hamiltonian Dynamics, Benedict Leimkuhler and Sebastian Reich
15. Collocation Methods for Volterra Integral and Related Functional Equations, Hermann Brunner
16. Topology for Computing, Afra J. Zomorodian
17. Scattered Data Approximation, Holger Wendland
Dedicated to
Zhuang and Leo Ling Yi
and the loving memories of my late parents Wan-Qing and Wen-Fang
In deciding what to investigate, how to formulate ideas and what problems to focus on, the individual mathematician has to be guided ultimately by their own sense of values. There are no clear rules, or rather if you only follow old rules you do not create anything worthwhile.

Contents

Preface page xiii

Nomenclature xxi

1 Introduction 1
1.1 Direct and iterative solvers, types of preconditioning 2
1.2 Norms and condition number 4
1.3 Perturbation theories for linear systems and eigenvalues 9
1.4 The Arnoldi iterations and decomposition 11
1.5 Clustering characterization, field of values and $\epsilon$-pseudospectrum 16
1.6 Fast Fourier transforms and fast wavelet transforms 19
1.7 Numerical solution techniques for practical equations 41
1.8 Common theories on preconditioned systems 61
1.9 Guide to software development and the supplied Mfiles 62

2 Direct methods 66
2.1 The LU decomposition and variants 68
2.2 The Newton–Schulz–Hotelling method 75
2.3 The Gauss–Jordan decomposition and variants 76
2.4 The QR decomposition 82
2.5 Special matrices and their direct inversion 85
2.6 Ordering algorithms for better sparsity 100
2.7 Discussion of software and the supplied Mfiles 106

3 Iterative methods 110
3.1 Solution complexity and expectations 111
3.2 Introduction to residual correction 112
3.3 Classical iterative methods 113
3.4 The conjugate gradient method: the SPD case 119
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.5</td>
<td>The conjugate gradient normal method: the unsymmetric case</td>
<td>130</td>
</tr>
<tr>
<td>3.6</td>
<td>The generalized minimal residual method: GMRES</td>
<td>133</td>
</tr>
<tr>
<td>3.7</td>
<td>The GMRES algorithm in complex arithmetic</td>
<td>141</td>
</tr>
<tr>
<td>3.8</td>
<td>Matrix free iterative solvers: the fast multipole methods</td>
<td>144</td>
</tr>
<tr>
<td>3.9</td>
<td>Discussion of software and the supplied Mfiles</td>
<td>162</td>
</tr>
<tr>
<td>4</td>
<td>Matrix splitting preconditioners [T1]: direct approximation of $A_{n \times n}$</td>
<td>165</td>
</tr>
<tr>
<td>4.1</td>
<td>Banded preconditioner</td>
<td>166</td>
</tr>
<tr>
<td>4.2</td>
<td>Banded arrow preconditioner</td>
<td>167</td>
</tr>
<tr>
<td>4.3</td>
<td>Block arrow preconditioner from DDM ordering</td>
<td>168</td>
</tr>
<tr>
<td>4.4</td>
<td>Triangular preconditioners</td>
<td>171</td>
</tr>
<tr>
<td>4.5</td>
<td>ILU preconditioners</td>
<td>172</td>
</tr>
<tr>
<td>4.6</td>
<td>Fast circulant preconditioners</td>
<td>176</td>
</tr>
<tr>
<td>4.7</td>
<td>Singular operator splitting preconditioners</td>
<td>182</td>
</tr>
<tr>
<td>4.8</td>
<td>Preconditioning the fast multipole method</td>
<td>185</td>
</tr>
<tr>
<td>4.9</td>
<td>Numerical experiments</td>
<td>186</td>
</tr>
<tr>
<td>4.10</td>
<td>Discussion of software and the supplied Mfiles</td>
<td>187</td>
</tr>
<tr>
<td>5</td>
<td>Approximate inverse preconditioners [T2]: direct approximation of $A^{-1}_{n \times n}$</td>
<td>191</td>
</tr>
<tr>
<td>5.1</td>
<td>How to characterize $A^{-1}$ in terms of $A$</td>
<td>192</td>
</tr>
<tr>
<td>5.2</td>
<td>Banded preconditioner</td>
<td>195</td>
</tr>
<tr>
<td>5.3</td>
<td>Polynomial preconditioner $p_k(A)$</td>
<td>195</td>
</tr>
<tr>
<td>5.4</td>
<td>General and adaptive sparse approximate inverses</td>
<td>199</td>
</tr>
<tr>
<td>5.5</td>
<td>AINV type preconditioner</td>
<td>211</td>
</tr>
<tr>
<td>5.6</td>
<td>Multi-stage preconditioners</td>
<td>213</td>
</tr>
<tr>
<td>5.7</td>
<td>The dual tolerance self-preconditioning method</td>
<td>224</td>
</tr>
<tr>
<td>5.8</td>
<td>Near neighbour splitting for singular integral equations</td>
<td>227</td>
</tr>
<tr>
<td>5.9</td>
<td>Numerical experiments</td>
<td>237</td>
</tr>
<tr>
<td>5.10</td>
<td>Discussion of software and the supplied Mfiles</td>
<td>238</td>
</tr>
<tr>
<td>6</td>
<td>Multilevel methods and preconditioners [T3]: coarse grid approximation</td>
<td>240</td>
</tr>
<tr>
<td>6.1</td>
<td>Multigrid method for linear PDEs</td>
<td>241</td>
</tr>
<tr>
<td>6.2</td>
<td>Multigrid method for nonlinear PDEs</td>
<td>259</td>
</tr>
<tr>
<td>6.3</td>
<td>Multigrid method for linear integral equations</td>
<td>263</td>
</tr>
<tr>
<td>6.4</td>
<td>Algebraic multigrid methods</td>
<td>270</td>
</tr>
<tr>
<td>6.5</td>
<td>Multilevel domain decomposition preconditioners for GMRES</td>
<td>279</td>
</tr>
<tr>
<td>6.6</td>
<td>Discussion of software and the supplied Mfiles</td>
<td>286</td>
</tr>
</tbody>
</table>
Contents

7 Multilevel recursive Schur complements preconditioners [T4] 289
  7.1 Multilevel functional partition: AMLI approximated Schur 290
  7.2 Multilevel geometrical partition: exact Schur 295
  7.3 Multilevel algebraic partition: permutation-based Schur 300
  7.4 Appendix: the FEM hierarchical basis 305
  7.5 Discussion of software and the supplied Mfiles 309

8 Sparse wavelet preconditioners [T5]: approximation of $\tilde{A}_{n\times n}$ and $\tilde{A}^{-1}_{n\times n}$ 310
  8.1 Introduction to multiresolution and orthogonal wavelets 311
  8.2 Operator compression by wavelets and sparsity patterns 320
  8.3 Band WSPAI preconditioner 323
  8.4 New centering WSPAI preconditioner 325
  8.5 Optimal implementations and wavelet quadratures 335
  8.6 Numerical results 336
  8.7 Discussion of software and the supplied Mfiles 338

9 Wavelet Schur preconditioners [T6] 340
  9.1 Introduction 341
  9.2 Wavelets telescopic splitting of an operator 342
  9.3 An exact Schur preconditioner with level-by-level wavelets 346
  9.4 An approximate preconditioner with level-by-level wavelets 352
  9.5 Some analysis and numerical experiments 357
  9.6 Discussion of the accompanied Mfiles 363

10 Implicit wavelet preconditioners [T7] 364
  10.1 Introduction 365
  10.2 Wavelet-based sparse approximate inverse 368
  10.3 An implicit wavelet sparse approximate inverse preconditioner 369
  10.4 Implementation details 371
  10.5 Dense problems 374
  10.6 Some theoretical results 376
  10.7 Combination with a level-one preconditioner 379
  10.8 Numerical results 380
  10.9 Discussion of the supplied Mfile 381

11 Application I: acoustic scattering modelling 383
  11.1 The boundary integral equations for the Helmholtz equation in $\mathbb{R}^3$ and iterative solution 384
  11.2 The low wavenumber case of a Helmholtz equation 397

© Cambridge University Press www.cambridge.org
Contents

1.3 The high wavenumber case of a Helmholtz equation 398
1.4 Discussion of software 399

12 Application II: coupled matrix problems 400
12.1 Generalized saddle point problems 401
12.2 The Oseen and Stokes saddle point problems 403
12.3 The mixed finite element method 405
12.4 Coupled systems from fluid structure interaction 407
12.5 Elasto-hydrodynamic lubrication modelling 410
12.6 Discussion of software and a supplied Mfile 413

13 Application III: image restoration and inverse problems 415
13.1 Image restoration models and discretizations 416
13.2 Fixed point iteration method 429
13.3 Explicit time marching schemes 436
13.4 The Primal-dual method 436
13.5 Nonlinear multigrids for optimization 439
13.6 The level set method and other image problems 442
13.7 Numerical experiments 446
13.8 Guide to software and the supplied Mfiles 447

14 Application IV: voltage stability in electrical power systems 449
14.1 The model equations 450
14.2 Fold bifurcation and arc-length continuation 454
14.3 Hopf bifurcation and solutions 458
14.4 Preconditioning issues 473
14.5 Discussion of software and the supplied Mfiles 473

15 Parallel computing by examples 475
15.1 A brief introduction to parallel computing and MPI 476
15.2 Some commonly used MPI routines 478
15.3 Example 1 of a parallel series summation 481
15.4 Example 2 of a parallel power method 483
15.5 Example 3 of a parallel direct method 489
15.6 Discussion of software and the supplied MPI Fortran files 502

Appendix A: a brief guide to linear algebra 504
A.1 Linear independence 504
A.2 Range and null spaces 505
A.3 Orthogonal vectors and matrices 505
A.4 Eigenvalues, symmetric matrices and diagonalization 505
A.5 Determinants and Cramer’s rule 506
## Contents

A.6  The Jordan decomposition  
A.7  The Schur and related decompositions  

Appendix B: the Harwell–Boeing (HB) data format  

Appendix C: a brief guide to MATLAB®  
C.1  Vectors and matrices  
C.2  Visualization of functions  
C.3  Visualization of sparse matrices  
C.4  The functional Mfile and string evaluations  
C.5  Interfacing MATLAB® with Fortran or C  
C.6  Debugging a Mfile  
C.7  Running a MATLAB® script as a batch job  
C.8  Symbolic computing  

Appendix D: list of supplied M-files and programs  

Appendix E: list of selected scientific resources on Internet  
E.1  Freely available software and data  
E.2  Other software sources  
E.3  Useful software associated with books  
E.4  Specialized subjects, sites and interest groups  

References  
Author Index  
Subject Index
The experiences of Fox, Huskey, and Wilkinson [from solving systems of orders up to 20] prompted Turing to write a remarkable paper [in 1948]. . . In this paper, Turing made several important contributions. . . He used the word “preconditioning” to mean improving the condition of a system of linear equations (a term that did not come into popular use until 1970s).


Matrix computing arises in the solution of almost all linear and nonlinear systems of equations. As the computer power upsurges and high resolution simulations are attempted, a method can reach its applicability limits quickly and hence there is a constant demand for new and fast matrix solvers. Preconditioning is the key to a successful iterative solver. It is the intention of this book to present a comprehensive exposition of the many useful preconditioning techniques.

Preconditioning equations mainly serve for an iterative method and are often solved by a direct solver (occasionally by another iterative solver). Therefore it is inevitable to address direct solution techniques for both sparse and dense matrices. While fast solvers are frequently associated with iterative solvers, for special problems, a direct solver can be competitive. Moreover, there are situations where preconditioning is also needed for a direct solution method. This clearly demonstrates the close relationship between a direct and an iterative method.

This book is the first of its kind attempting to address an active research topic, covering these main types of preconditioners.
<table>
<thead>
<tr>
<th>Type 1</th>
<th>Matrix splitting preconditioner</th>
<th>FEM setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type 2</td>
<td>Approximate inverse preconditioner</td>
<td>FEM setting</td>
</tr>
<tr>
<td>Type 3</td>
<td>Multilevel (approximate inverse) preconditioner</td>
<td>FEM setting</td>
</tr>
<tr>
<td>Type 4</td>
<td>Recursive Schur complements preconditioner</td>
<td>FEM setting</td>
</tr>
<tr>
<td>Type 5</td>
<td>Matrix splitting and Approximate inverses</td>
<td>Wavelet setting</td>
</tr>
<tr>
<td>Type 6</td>
<td>Recursive Schur complements preconditioner</td>
<td>Wavelet setting</td>
</tr>
<tr>
<td>Type 7</td>
<td>Implicit wavelet preconditioner</td>
<td>FEM setting</td>
</tr>
</tbody>
</table>

Here by ‘FEM setting’, we mean a usual matrix (as we found it) often formed from discretization by finite element methods (FEM) for partial differential equations with piecewise polynomial basis functions whilst the ‘Wavelet setting’ refers to wavelet discretizations. The iterative solvers, often called accelerators, are selected to assist and motivate preconditioning. As we believe that suitable preconditioners can work with most accelerators, many other variants of accelerators are only briefly mentioned to allow us a better focus on the main theme. However these accelerators are well documented in whole or in part in the more recent as well as the more classical survey books or monographs (to name only a few)

Preface


Most generally applicable preconditioning techniques for unsymmetric matrices are covered in this book. More specialized preconditioners, designed for symmetric matrices, are only briefly mentioned; where possible we point to suitable references for details. Our emphasis is placed on a clear exposition of the motivations and techniques of preconditioning, backed up by MATLAB® Mfiles, and theories are only presented or outlined if they help to achieve better understanding. Broadly speaking, the convergence of an iterative solver is dependent of the underlying problem class. The robustness can often be improved by suitably designed preconditioners. In this sense, one might stay with any preferred iterative solver and concentrate on preconditioner designs to achieve better convergence.

As is well known, the idea of providing and sharing software is to enable other colleagues and researchers to concentrate on solving newer and harder problems instead of repeating work already done. In the extreme case, there is nothing more frustrating than not being able to reproduce results that are claimed to have been achieved. The MATLAB Mfiles are designed in a friendly style to reflect the teaching of the author’s friend and former MSc advisor Mr Will

---

1 MATLAB is a registered trademark of MathWorks, Inc; see its home page [http://www.mathworks.com](http://www.mathworks.com). MATLAB is an easy-to-use script language, having almost the full capability as a C programming language without the somewhat complicated syntax of C. Beginners can consult a MATLAB text e.g. [135] from [http://www.liv.ac.uk/maths/ETC/matbook](http://www.liv.ac.uk/maths/ETC/matbook) or any tutorial document from the internet. Search [http://www.google.com](http://www.google.com) using the key words: MATLAB tutorial.
McLewin (University of Manchester) who rightly said: ‘in Mathematics, never use the word ‘obviously’.’ A simple and useful feature of the supplied Mfiles is that typing in the file name invokes the help page, giving working examples. (The standard MATLAB reply to such a usage situation is ??? Error…Not enough input arguments.)

The book was born mainly out of research work done in recent years and partly out of a need of helping out graduate students to implement a method following the not-always-easy-to-follow descriptions by some authors (who use the words ‘trivial’, ‘standard’, ‘well-known’, ‘leave the reader to work it out as an exercise’ in a casual way and in critical places). That is to say, we aspire to emphasize the practical implementation as well as the understanding rather than too much of the theory. In particular the book is to attempt a clear presentation and explanation, with the aid of illustrations and computer software, so that the reader can avoid the occasional frustration that one must know the subject already before one can really understand and appreciate a beautiful mathematical idea or algorithm presented in some (maybe a lot of) mathematical literature.

About the solvers and preconditioners.

Chapter 1. (Introduction) defines the commonly used concepts; in particular the two most relevant terms in preconditioning: condition number and clustering. With non-mathematics majors’ readers in mind, we give an introduction to several discretization and linearization methods which generate matrix equations – the idea of mesh ordering affecting the resulting matrix is first encountered. Examples of bounding conditioned numbers by considering norm equivalence (for symmetric systems) are given; these appealing theories are not a guarantee for fast convergence of iterative solvers. Both the fast Fourier transforms (FFT) and fast wavelet transforms (FWT) are introduced here (mainly discrete FWT and the continuous to come later); further discussions of FFT and FWT are in Chapters 2, 4 and 8.

Chapter 2. (Direct methods) discusses the direct Gaussian elimination method and the Gauss–Jordan and several variants. Direct methods are on one hand necessary for forward type preconditioning steps and on the other hand provide various motivations for designing an effective preconditioner. Likewise, for some ill-conditioned linear systems, there is a strong need for scaling and preconditioning to obtain accurate direct solutions – a much less addressed subject. Algorithms for inverting several useful special matrices are then given; for circulant matrices diagonalization by Fourier transforms is explained.
before considering Toeplitz matrices. Block Toeplitz matrices are considered later in Chapter 13. Algorithms for graph nodal or mesh (natural graph) orderings by reverse Cuthill–McKee method (RCM), spiral and domain decomposition methods (DDM) are given. The Schur complements and partitioned LU decompositions are presented together; for symmetric positive definite (SPD) matrices, some Schur properties are discussed. Overall, this chapter contains most of the ingredients for implementing a successful preconditioner.

Chapter 3. (Iterative methods) first discusses the classical iterative methods and highlights their use in multigrid methods (MGM, Chapter 6) and DDM. Then we introduce the topics most relevant to the book, conjugate gradient methods (CGM) of the Krylov subspace type (the complex variant algorithm does not appear in the literature as explicitly as presented in Section 3.7). We elaborate on the convergence with a view on preconditioners’ design. Finally the popular fast multipole expansion method (along with preconditioning) is introduced. The mission of this chapter is to convey the message that preconditioning is relatively more important than modifying existing or inventing new CGM solvers.

Chapter 4. (Matrix splitting preconditioners: Type 1) presents a class of mainly sparse preconditioners and indicates their possible application areas, algorithms and limitations. All these preconditioners are of the forward type, i.e. $M \approx A$ in some way and efficiency in solving $Mx = b$ is assured. The most effective and general variant is the incomplete LU (ILU) preconditioner with suitable nodal ordering. The last two main sections (especially the last one) are mainly useful for dense matrix applications.

Chapter 5. (Approximate inverse preconditioners: Type 2) presents another large class of sparse approximate inverse preconditioners for a general sparse matrix problem, with band preconditioners suitable for diagonally dominant matrices and near neighbour preconditioners suitable for singular operator equations. All these preconditioners are of the backward type, i.e. $M \approx A^{-1}$ in some way and application of each sparse preconditioner $M$ requires a simple multiplication.

Chapter 6. (Multilevel methods and preconditioners: Type 3) gives an introduction to geometric multigrid methods for partial differential equations (PDEs) and integral equations (IEs) and algebraic multigrid method for sparse linear systems, indicating that for PDEs, in general, smoothing is important but can be difficult while for IEs operator compactness is the key. Finally we discuss multilevel domain decomposition preconditioners for CG methods.
Preface

Chapter 7. (Multilevel recursive Schur preconditioners: Type 4) surveys the recent Schur complements based recursive preconditioners where matrix partition can be based on functional space nesting or graph nesting (both geometrically based and algebraically based).

Chapter 8. (Sparse wavelet preconditioners: Type 5) first introduces the continuous wavelets and then considers to how construct preconditioners under the wavelet basis in which an underlying operator is more amenable to approximation by the techniques of Chapters 4–7. Finally we discuss various permutations for the standard wavelet transforms and their use in designing banded arrow (wavelet) preconditioners.

Chapter 9. (Wavelet Schur preconditioners: Type 6) generalizes the Schur preconditioner of Chapter 7 to wavelet discretization. Here we propose to combine the non-standard form with Schur complement ideas to avoid finger-patterned matrices.

Chapter 10. (Implicit wavelet preconditioners: Type 7) presents some recent results that propose to combine the advantages of sparsity of finite elements, sparse approximate inverses and wavelets compression. Effectively the wavelet theory is used to justify the a priori patterns that are needed to enable approximate inverses to be efficient; this strategy is different from Chapter 9 which does not use approximate inverses.

About the selected applications.

Chapter 11. (Application I) discusses the iterative solution of boundary integral equations reformulating the Helmholtz equation in an infinite domain modelling the acoustic scattering problem. We include some recent results on high order formulations to overcome the hyper-singularity. The chapter is concluded with a discussion of the open challenge of modelling high wavenumber problems.

Chapter 12. (Application II) surveys some recent work on preconditioning coupled matrix problems. These include Hermitian and skew-Hermitian splitting, continuous operators based Schur approximations for Oseen problems, the block diagonal approximate inverse preconditioners for a coupled fluid structure interaction problem, and FWT based sparse preconditioners for EHL equations modelling the isothermal (two dependent variables) and thermal (three dependent variables) cases.

Chapter 13. (Application III) surveys some recent results for iterative solution of inverse problems. We take the example of the nonlinear total variation (TV)
equation for image restoration using operator splitting and circulant preconditioners. We show some new results based on combining FWT and FFT preconditioners for possibly more robust and faster solution and results on developing nonlinear multigrid methods for optimization. Also discussed is the ‘matrix-free’ idea of solving an elliptic PDE via an explicit scheme of a parabolic PDE, which is widely used in evolving level set functions for interfaces tracking; the related variational formulation of image segmentation is discussed.

Chapter 14. (Application IV) shows an example from scientific computing that typifies the challenge facing computational mathematics – the bifurcation problem. It comes from studying voltage stability in electrical power transmission systems. We have developed two-level preconditioners (approximate inverses with deflation) for solving the fold bifurcation while the Hopf problem remains an open problem as the problem dimension is ‘squared’!

Chapter 15. (Parallel computing) gives a brief introduction to the important subject of parallel computing. Instead of parallelizing many algorithms, we motivate two fundamental issues here: firstly how to implement a parallel algorithm in a step-by-step manner and with complete MPI Fortran programs, and secondly what to consider when adapting a sequential algorithm for parallel computing. We take four relatively simple tasks for discussing the underlying ideas.

The Appendices give some useful background material, for reference purpose, on introductory linear algebra, the Harwell–Boeing data format, a MATLAB tutorial, the supplied Mfiles and Internet resources relevant to this book.

Use of the book. The book should be accessible to graduate students in various scientific computing disciplines who have a basic linear algebra and computing knowledge. It will be useful to researchers and computational practitioners. It is anticipated that the reader can build intuition, gain insight and get enough hands on experience with the discussed methods, using the supplied Mfiles and programs from

http://www.cambridge.org/9780521838283
http://www.liv.ac.uk/maths/ETC/mpta

while reading. As a reference for researchers, the book provides a toolkit and with it the reader is expected to experiment with a matrix under consideration and identify the suitable methods first before embarking on serious analysis of a new problem.
Acknowledgements. Last but not least, the author is grateful to many colleagues (including Joan E. Walsh, Siamiak Amini, Michael J. Baines, Tony F. Chan and Gene H. Golub) for their insight, guidance, and encouragement and to all my graduate students and research fellows (with whom he has collaborated) for their commitment and hard work, on topics relating to this book over the years. In particular, Stuart Hawkins and Martyn Hughes have helped and drafted earlier versions of some Mfiles, as individually acknowledged in these files. Several colleagues have expressed encouragement and comments as well as corrected on parts of the first draft of the manuscript – these include David J. Evans, Henk A. van der Vorst, Raymond H. Chan, Tony F. Chan, Gene H. Golub and Yimin Wei; the author thanks them all. Any remaining errors in the book are all mine. The handy author index was produced using the authorindex.sty (which is available from the usual L\LaTeX\ sites) as developed by Andreas Wettstein (ISE AG, Zurich, Switzerland); the author thanks him for writing a special script for me to convert bibliitem entries to a bibliography style. The CUP editorial staff (especially Dr Ken Blake), the series editors and the series reviewers have been very helpful and supportive. The author thanks them for their professionalism and quality support. The continual funding in recent years by the UK EPSRC and other UK funding bodies for various projects related to this book has been gratefully received.

Feedback. As the book involves an ambitious number of topics with preconditioning connection, inevitably, there might be errors and typos. The author is happy to hear from any readers who could point these out for future editions. Omission is definitely inevitable: to give a sense of depth and width of the subject area, a search on www.google.com in April 2004 (using keywords like ‘preconditioned iterative’ or ‘preconditioning’) resulted in hits ranging from 19000 to 149000 sites. Nevertheless, suggestions and comments are always welcome. The author is also keen to include more links to suitable software that are readable and helpful to other researchers, and are in the spirit of this book. Many thanks and happy computing.

Ke Chen
Liverpool, September 2004

EMAIL = k.chen@liverpool.ac.uk
URL = http://www.liv.ac.uk/~cmchenke
Nomenclature

All the beautiful mathematical ideas can be found in Numerical Linear Algebra. However, the subject is better to be enjoyed by researchers than to teach to students as many excellent ideas are often buried in the complicated notation. A researcher must be aware of this fact.

Gene H. Golub. Lecture at University of Liverpool (1995)

Throughout the book, capital letters such as $A$ denote a rectangular matrix $m \times n$ (or a square matrix of size $n$), whose $(i, j)$ entry is denoted by $A(i, j) = a_{ij}$, and small letters such as $x, b$ denote vectors of size $n$ unless stated otherwise i.e. $A \in \mathbb{R}^{m \times n}$ and $x, b \in \mathbb{R}^n$.

Some (common) abbreviations and notations are listed here

- $\mathbb{C}^n$ → the space of all complex vectors of size $n$
- $\mathbb{R}^n$ → the space of all real vectors of size $n$
- $\|A\|$ → A norm of matrix $A$ (see §1.5)
- $|A|$ → The matrix of absolute values of $A$ i.e. $|A|_{ij} = |A(i, j)| = |a_{ij}|$
- $A^T$ → The transpose of $A$ i.e. $A^T(i, j) = A(j, i)$, $A$ is symmetric if $A^T = A$.
- $A^H$ → The transpose conjugate for complex $A$ i.e. $A^H(i, j) = \overline{A(j, i)}$. $A$ is Hermitian if $A^H = A$. Some books write $A^* = A^H$.
- $\det(A)$ → The determinant of matrix $A$
- $\text{diag}(a_j)$ → A diagonal matrix made up of scalars $a_j$
- $\lambda(A)$ → An eigenvalue of $A$
- $A \oplus B$ → The direct sum of orthogonal quantities $A, B$
- $A \odot B$ → The biproduct of matrices $A, B$ (Definition 14.3.8)
- $A \otimes B$ → The tensor product of matrices $A, B$ (Definition 14.3.3)
\[ \sigma(A) \rightarrow \text{A singular value of } A \]
\[ \kappa(A) \rightarrow \text{The condition number } \text{cond}(A) \text{ of } A \text{ (in some norm)} \]
\[ \Lambda(A) \rightarrow \text{Eigenspectrum for } A \]
\[ \Lambda_\epsilon(A) \rightarrow \epsilon\text{-Eigenspectrum for } A \]
\[ \Sigma(A) \rightarrow \text{Spectrum of singular values of } A \]
\[ \mathcal{Q}_k \rightarrow \text{Set of all degree } k \text{ polynomials with } q(0) = 1 \text{ for } q \in \mathcal{Q}_k \]
\[ \mathcal{W}(A) \rightarrow \text{Field of values (FoA) spectrum} \]
\[ \text{AINV} \rightarrow \text{Approximate inverse} \] [55]
\[ \text{BEM} \rightarrow \text{Boundary element method} \]
\[ \text{BIE} \rightarrow \text{Boundary integral equation} \]
\[ \text{BCCB} \rightarrow \text{Block circulant with circulant blocks} \]
\[ \text{BTTB} \rightarrow \text{Block Toeplitz with Toeplitz blocks} \]
\[ \text{BPX} \rightarrow \text{Bramble–Pasciak–Xu (preconditioner)} \]
\[ \text{CG} \rightarrow \text{Conjugate gradient} \]
\[ \text{CGM} \rightarrow \text{CG Method} \]
\[ \text{CGN} \rightarrow \text{Conjugate gradient normal method} \]
\[ \text{DBAI} \rightarrow \text{Diagonal block approximate inverse (preconditioner)} \]
\[ \text{DDM} \rightarrow \text{Domain decomposition method} \]
\[ \text{DFT} \rightarrow \text{Discrete Fourier transform} \]
\[ \text{DWT} \rightarrow \text{Discrete wavelet transform} \]
\[ \text{FDM} \rightarrow \text{Finite difference method} \]
\[ \text{FEM} \rightarrow \text{Finite element method} \]
\[ \text{FFT} \rightarrow \text{Fast Fourier transform} \]
\[ \text{FFT2} \rightarrow \text{Fast Fourier transform in 2D (tensor products)} \]
\[ \text{FMM} \rightarrow \text{Fast multipole method} \]
\[ \text{FoV} \rightarrow \text{Field of values} \]
\[ \text{FSAI} \rightarrow \text{Factorized approximate inverse (preconditioner)} \] [321]
\[ \text{FWT} \rightarrow \text{Fast wavelet transform} \]
\[ \text{GMRES} \rightarrow \text{Generalised minimal residual method} \]
\[ \text{GJ} \rightarrow \text{Gauss–Jordan decomposition} \]
\[ \text{GS} \rightarrow \text{Gauss–Seidel iterations (or Gram–Schmidt method)} \]
\[ \text{HB} \rightarrow \text{Hierarchical basis (finite elements)} \]
\[ \text{ILU} \rightarrow \text{Incomplete LU decomposition} \]
\[ \text{LU} \rightarrow \text{Lower upper triangular matrix decomposition} \]
\[ \text{LSAI} \rightarrow \text{Least squares approximate inverse (preconditioner)} \]
\[ \text{MGM} \rightarrow \text{Multigrid method} \]
\[ \text{MRA} \rightarrow \text{Multiresolution analysis} \]
\[ \text{OSP} \rightarrow \text{Operator splitting preconditioner} \]
\[ \text{PDE} \rightarrow \text{Partial differential equation} \]
<table>
<thead>
<tr>
<th>Nomenclature</th>
<th>xxiii</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSM →</td>
<td>Powers of sparse matrices</td>
</tr>
<tr>
<td>QR →</td>
<td>Orthogonal upper triangular decomposition</td>
</tr>
<tr>
<td>SDD →</td>
<td>Strictly diagonally dominant</td>
</tr>
<tr>
<td>SOR →</td>
<td>Successive over-relaxation</td>
</tr>
<tr>
<td>SSOR →</td>
<td>Symmetric SOR</td>
</tr>
<tr>
<td>SPAI →</td>
<td>Sparse approximate inverse [253]</td>
</tr>
<tr>
<td>SPD →</td>
<td>Symmetric positive definite matrix ($\lambda_j(A) &gt; 0$)</td>
</tr>
<tr>
<td>SVD →</td>
<td>Singular value decomposition</td>
</tr>
<tr>
<td>WSPAI →</td>
<td>Wavelet SPAI</td>
</tr>
</tbody>
</table>