Preface

About Matrix Algorithms

Matrix computations are very important to many scientific and engineering disciplines. Many successful public and commercial software packages for the matrix computations, such as LAPACK and MATLAB [1], have been widely used for decades. LAPACK stands for *Linear Algebra PACKage*. It is a Fortran library of routines for solving linear equations, least square of linear equations, eigenvalue problems and singular value problems for dense and band of real and complex matrices. MATLAB stands for *MATrix LABoratory*. It is an interpretive computer language and numerical computation environment. It includes a lot of built-in matrix computation algorithms, most of which are built upon LAPACK. Its powerful sub-matrix indexing capability makes it a good tool for the rapid prototype of numerical algorithms. Vast amount of various computer programs for the matrix computations in different computer languages can also be easily found through the search facilities of the internet. Notably two internet resources, GAMS [2] and NETLIB [3], provide free access to documentation and source codes of many reusable computer software components, most of which are about matrix computations.

In spite of the wide availability of different kinds of computer programs for the matrix computations, the matrix computations remain an active research and development area for many years. New applications, new algorithms and improvements to old algorithms are constantly emerging. Due to their sheer importance, the matrix computations are taught from the middle school to many doctorate majors of science and engineering. There are many good books available for different readers. *'Matrix Computations'* by Gene H. Golub and Charles F. Van Loan [29] builds a systematic framework for the broad discipline of the matrix computations and gives an excellent description of algorithms and in-depth analysis of the properties of algorithms. Furthermore the book contains a vast amount of references to the interested readers. [7, 5] can be consulted for algorithmic matters in equation solutions and eigenvalue solutions. [3, 42, 51] cover the coding matters. For the complete coverage of the field, the two volume treatise by G.W.Stewart [68, 69] can be consulted.

Some of these books present matrix algorithms as MTALAB-like pseudo codes. The first time readers of these books will find the MATLAB pseudo codes useful to gain a preliminary understanding of the algorithms. However, algorithms are best studied through the use of the real computer codes. Through the running and debugging of the real codes, readers can expect to understand the mathematical properties of algorithms and all the subtlety of numerical computations. Readers cannot gain any insight of how an algorithm works by simply studying pseudo codes. On the other hand, many public domain codes, such as LAPACK, are too large to learn. Often the data structure obscures the essence of algorithms.

What is *Matrix Algorithms in MATLAB*

This book tries to shorten the wide gap between the rigorous mathematics of matrix algorithm and complicated computer code implementations. It presents many matrix algorithms using real MATLAB codes. For each algorithm, the presentation starts with a brief but simple mathematical exposure. The algorithm is usually explained with a small matrix, step by step, in the same order that the algorithm is executed

[1] MATLAB is a registered trademark of The MathWorks, Inc.
[2] http://gams.nist.gov
[3] http://www.netlib.org

on a computer. Then the MATLAB codes for the algorithm are listed. Mostly due to its natural notation of the sub-matrices, the MATLAB codes do not look very different from pseudo codes. For the sake of clarity of the representation, most of the MATLAB codes presented in the book are kept within 100 lines. At the beginning of each MATLAB code is some comments about the functions, input and output of the code, which can be accessed by MATLAB's *help* system. Therefore the core codes for each algorithm is even shorter and easier to understand. After the list of codes, a few numerical examples are given. The purpose of the examples has three folds. Firstly, examples verify the code implementation. Secondly, the examples demonstrate the usage of the MATLAB codes. Thirdly, the examples show the mathematical properties of the algorithm.

The algorithms covered in the book include some common algorithms found in MATLAB, such as algorithms in *lu*, *qr*, *eig* and *svd* etc. The MATLAB codes represented in the book for these common algorithms offer more options. For example, restart, different permutations, accuracy and output controls are offered in *lu*, *qr* and the other algorithms. The book contains over 15 different equation solution algorithms, over 10 different eigenvalue algorithms and singular value algorithms. The algorithms covered in the book include many other algorithms not found in MATLAB nor even in LAPACK. For example, the decomposition of symmetric indefinite matrix of [12] and [1], the solution of Vandermonde matrix, various iterative equation solution algorithms with assumed pre-conditioners, iterative eigen solution algorithms of symmetric and non-symmetric matrices. All the MATLAB codes presented in the book are tested with thousands of runs of MATLAB randomly generated matrices. The validity of the computation is verified by the mathematical identity for each computation. No errors are found! The clarity of the presentation of the algorithms is the goal of the MATLAB codes of the book. The efficiency of the codes is the secondary consideration. However, the CPU time of most common algorithms is usually within 5 times of the MATLAB built-in algorithms for same matrices. When running the MATLAB codes, a big part of the CPU time is spent on the interpretation of the codes. Only real matrices are covered in the book. The extension to the complex matrices is usually straight forward for most algorithms. Band and sparse matrices are not treated in most algorithms. But in the iterative equation solution algorithms and eigen solution algorithms, the special considerations of band and sparse matrices can be supplied by user routines as input arguments to the algorithms.

In Chapter 1, *Introduction*, a brief introduction of MATLAB will be made together with most notations used in the book. A summary of the basic theory of matrix computations, a systematic view of different algorithms, the classifications of matrices, the storage of special matrices, reordering of sparse matrices and some utility MATLAB codes will be covered in Chapter 1.

Chapters 2 and 3, *Direct Algorithms of Decompositions of Matrices by Non-orthogonal/Orthogonal Transformations*, address the decompositions of a general matrix and some special matrices. The decomposition of a matrix is itself an important matrix computation problem, but also the foundation of other matrix algorithms. Chapter 2 starts with Gauss zeroing matrix. Then it presents LU and LDU decompositions for general matrices, congruent transformation algorithms (LDLt, LTLt, LBLt decompositions) for symmetric matrices, LLt (Cholesky) for symmetric positive definite matrices and xLLt (a modified Cholesky) for symmetric matrices. It continues with 3 similarity transformation algorithms (LHLi and GTGi) and lastly GTGJGt algorithm that simultaneously transforms one symmetric matrix to a symmetric tri-diagonal and another symmetric matrix to a diagonal of only $\pm$1s and possibly 0s. Chapter 3 is parallel to Chapter 2, but only orthogonal transformations are used in algorithms. Corresponding to Gauss matrix is Householder matrix and Givens matrix. Corresponding to LU, LDU are QR, QLZ and QBZ decompositions. Corresponding to LHLi and GTGi are QHQt and QTQt decompositions. Corresponding to GTGJGt is QHRZ algorithm that simultaneously transforms one matrix to Hessenberg and another matrix to upper triangle. Most algorithms in Chapters 2 and 3 are presented in more than one flavors, transforming, decomposing and multiplying, according to the classifications of matrix algorithms made in Chapter 1.

Chapter 4 presents *Direct Algorithms of Solution of Linear Equations*. The first 3 sections introduce several theoretical tools for the solution of linear equations, pseudo inverse, minimum norm/residual solutions, solutions with linear constraints. It begins the algorithms of 5 special structured matrices: zero matrix, diagonal matrix, orthogonal matrix and lower/upper triangular matrix. Then it presents 4 elimination algorithms: Gauss and Gauss-Jordan eliminations by non-orthogonal transformations, Householder and Givens eliminations by orthogonal transformations. It follows by decomposing algorithms that utilize all the matrix decomposition algorithms of Chapters 2 and 3. Lastly it presents two special algorithms arising from the interpolation problems: Vandermonde matrix and Fast Fourier Transform.

Chapter 5 presents *Iterative Algorithms of Solution of Linear Equations*, which are better suited for large and sparse matrices. Presented algorithms include Jacobi ierations, Gauss-Seidel iterations, 3 algorithms based on Lanczos (Conjugate Gradient, Minimum Residual and Mimimum Error) for symmetric matrices, 11 algorithms based on Arnoldi (Full Orthogonalization, Generalized Minimum Residual, Generalized Minimum Error, and 8 other variants) for unsymmetric matrices, 4 special algorithms for normal equations. The algorithms are complemented by an overview of iterative algorithms and plots of convergence patterns. The last section briefly touches 4 important topics: pre-conditioning, parallel computation, algebraic multigrid method and domain decomposition method.

Chapter 6 presents *Direct Algorithms of Solution of Eigenvalue Problem*. Section 6.1 provides the simple algorithm for 2×2 matrix, which is used in Jacobi iteration algorithms of Section 6.3 and other algorithms as a building block. The third section builds 4 auxiliary tools for the symmetric eigenvalues, to count how many eigenvalues in a given interval and to approximate the lower and upper bounds of a given eigenvalue. Section 6.5 is on the key QR iteration algorithms for both unsymmetric and symmetric matrices. Sections 6.6 and 6.7 are on the calculation of invariant subspace: inverse iteration to calculate an eigenvector by a given eigenvalue and eigenvalue reordering. Section 6.8 presents two more special algorithms for symmetric tri-diagonal matrices: bisection and divide-and-conquer. Sections 6.9–6.11 present 3 algorithms for generalized eigenvalue problems, symmetric positive definite pencils, unsymmetric pencils and symmetric positive indefinite pencils.

Chapter 7 presents *Iterative Algorithms of Solution of Eigenvalue Problem*, which are better suited for large and sparse matrices. The first algorithm is the power/subspace iteration. Sections 7.3 and 7.4 have full presentations of two celebrated algorithms, Lanczos for symmetric matrices and Arnoldi for unsymmetric matrices. All the algorithms are complemented by an overview of iterative algorithms and plots of convergence patterns. The last section discusses briefly 3 important topics: generalized eigenvalue problem, nonlinear eigenvalue problem and Jacobi-Davidson iterations.

Chapter 8 presents *Algorithms of Solution of Singular Value Decomposition*. Section 8.1 discusses the connection between the singular value decomposition and eigenvalue decomposition of a symmetric matrix. The connection shows how to construct the algorithms for singular value decompositions from the eigenvalue algorithms of Chapter 7 and 8. Section 8.2 presents the simple algorithms for row/column vectors and 2×2 matrices. Built upon the simple algorithm of Section 8.2, Section 8.3 presents the relatively simple Jacobi algorithm. Section 8.4 is the counter part of Section 6.5 in singular value decomposition, QR iteration. Section 8.5 presents two special algorithms for a bi-diagonal matrix: bisection and divide-and-conquer. For large and sparse matrices, Section 8.6 discusses the Lanczos singular value algorithm.

Who Reads *Matrix Algorithms in MATLAB*

This book is intended for people working in the field of matrix computations, who need to master, implement and improve the matrix algorithms. Students in computer science, applied mathematics, computer engineering and other engineering disciplines can benefit from studying the book. The book is also useful to researchers and professionals in numerical analysis of engineering and scientific research.

How to Read *Matrix Algorithms in MATLAB*

The book can be used in the traditional way, reading. At first, readers can read the brief discussions of mathematical exposures. Then follow the step-by-step explanation of an algorithm by simple examples. The most efficient way of reading it is to run and debug the examples on computers on which a MATLAB program is installed. With 3–5 rounds of debugging and more numerical experiments by readers, readers can expect to have a solid understanding of the algorithm. Because MATLAB codes are very close to any pseudo codes, no computer programming knowledge is required. For readers with experience in MATLAB, Section 1.2 can certainly be by-passed; otherwise readers can read Section 1.2 to jump start.

Acknowledgments

Writing this book was my personal endeavor to sharpen my skills in numerical computations. Nevertheless, the many supports I received over many years are necessary to make it happen.

First and foremost, I acknowledge the great contribution and sacrifice of my wife, Kathy, and two children, Jeffery and Jennifer. For many days and nights, Kathy shoulders the responsibility of taking care of the family, raising and educating the two children. She fully supports my endeavor of studying mechanics and numerical computations, in either shining or raining days. Her love gives me the strength to persist. Jeffery and Jennifer missed my fatherhood when they needed it the most. I apologize to them my absence and impatience, while I was focusing on my study. Without their contribution and sacrifice, writing this book is impossible. Therefore I dedicate *Matrix Algorithms in MATLAB* to them, my dear wife, Kathy, and two wonderful children, Jeffery and Jennifer.

I am very proud of my humble origin, the son of my uneducated parents, Shang Z. and Allyn. Their unconditional love and belief in me being excellent were my only source of strength for many years. They will live in my heart forever.

I express my great appreciation to my Publisher, Joe Hayton, and my Editorial Project Manager, Kattie Washington, at Elsevier, and to my Production Project Manager, Anusha Sambamoorthy. It was my great honor to have the opportunity to work with Joe, Kattie and Anusha in publishing this book. I am thankful to Maria Ines Cruz at Elsevier for the cover design. I am also thankful to the anonymous manuscript reviewers for their time reviewing the book and making suggestions to improve the book.

Finally I wish to acknowledge two people I have never met: Cleve Moler, the creator of MATLAB and co-founder of Mathworks, for lending me the excellent tool to implement and present matrix algorithms; and Donald Knuth, for lending me latex to write the book.

Ong U. Routh

December 9, 2015