COMPUTER PROGRAMS FOR FINDING EXACT EIGENVALUES OF MATRICES AND TEST MATRICES

By

BIPUL SYAM PURKAYASTHA

DEPARTMENT OF MATHEMATICS

SUBMITTED
IN
FULFILMENT OF THE REQUIREMENTS FOR
THE DEGREE OF
DOCTOR OF PHILOSOPHY

To

NORTH - EASTERN HILL UNIVERSITY
SHILLONG - 793022, INDIA

FEBRUARY, 1996

CERTIFICATE

I certify that the thesis entitled **"COMPUTER PROGRAMS FOR FINDING EXACT EIGENVALUES OF MATRICES AND TEST MATRICES"** submitted by Bipul Syam Purkayastha in fulfilment of the requirements for the degree of Doctor of Philosophy is the outcome of studies undertaken by the candidate. I certify that the sources from which the ideas have been borrowed are duly referred to.

The material in this thesis has not been presented for the award of a degree in any University before.

This thesis may be placed before the examiners for evaluation and necessary formalities.

विजय कुमार

(Prof. Vijai Kumar)

Supervisor

Shillong

The 5th February 1996

Mangesh B. Rege

(Prof. M. B. Rege)

Supervisor

Head of the Department of Mathematics

North Eastern Hill University

Shillong – 793022

INDIA

ACKNOWLEDGEMENTS

The studies for this thesis were undertaken under the guidance of Dr. Vijai Kumar. I wish to express my deepest regards and profound gratitude for his guidance and fatherly affection. Over the years he continued to guide me with a rare sense of patience and understanding which made it much easier for me to write this thesis. I also express my indebtedness to Smt. Sulakshana Rani, his wife, who is encouragement personified. I think that I am immensely fortunate to have had the pleasure of coming across them.

I take this opportunity to express my gratitude to Dr. M. B. Rege, the Head of the Department of Mathematics for his constant help and encouragement. I express my indebtedness to Dr. P. K. Saikia whom I have consulted frequently and who was always ready with valuable suggestions. I owe special thanks to all the other faculty members of the department of Mathematics, NEHU, namely Dr. A. K. Das, Dr. B. K. Dev Sharma, Dr. C. R. Mondal, Dr. H. K. Mukherjee, Dr. S. K. Srivastava, Dr. S. L. Marbaning and Dr. S. S. Khare for their constant encouragement and for their best wishes.

My warmest thanks are also reserved for Dr. S. N. Rai, Director of Computer Centre, NEHU, for allowing me to use the computers in his Centre. My friend Mr. P. P. Dey, Senior System Analyst of the Centre, always saw to this that my work went smoothly and without a hitch.

My colleagues at Shillong College were always ready to extend their helping hands for the smooth completion of my work. Special mention must be made of Mr. T. Maitra, the Principal of the College, Mr. B. C. Goswami,

Mr. K. Choudhury, Mrs. S. Dhar and Mr. H. Dhar, my colleagues in the department of Mathematics of the College, Mrs. M. P. R. Lyngdoh, Mr. R. Dutta and Mr. B. Roy my senior collegues, Mr. M. N. Bhattacherjee, Mr. M. Dey, Miss. V. R. Solomon and Mr. T. S. Rajee, my friends and colleagues. With their constant encouragement my problems and frustrations became more bearable.

I shall like to thank all my friends, specially Mr. A. Das, Mr. H. S. Bhattacherjee, Mr. M. Chakravorty, Mr. P. Chakravorty, Mr. S. K. Singh, Mr. D. Dey and Mrs. R. Bhattacherjee. I consider myself fortunate to have friends like them. Associations with them made my hard days easy.

Special gratitude and heartfelt thanks are also due to Mr. M. Bhattacharjee, with whose help I could avail the library facilities of I.I.T. Kharagpur. I also thank Mr. P. Bhattacherjee for many stimulating discussions I had with him.

I am grateful to my aunt Smt. K. Syam Purkayastha and my late uncle Shri B. B. Syam Purkayastha and for bringing me up as their own child. With much affection and love I remember my mother Smt. A. L. Syam Purkayastha and my late father Shri B. B. Syam Purkayastha. I express my thanks to my elder brothers Mr. B. K. Syam Purkayastha and Mr. B. J. Syam Purkayastha who are extremely caring and who shielded me throughout from the non-academic problems of life. Finally I shall like to thank all my family members, past and present, who shared their lives with me and sustained me in beautiful and sad times. I shall cherish their memories forever.

Bipul Syam Purkayastha

CONTENTS

CONTENTS

Introduction

In order to employ a computer, effective algorithms are needed which can be implemented in the form of computer programs. Endeavours to devise such algorithms often yield new theoretical findings, as a by-product, which are also per se of interest. A basic task, which every mathematician faces, is the task of providing suitable numerical examples and data in order to illustrate the known theorems and in many cases to make up or to support conjectures that may lead to the discovery of new algorithms. However, it is a particular phenomenon of some topics such as Algebraic Number Theory, that even relatively simple computational problems require a great deal of numerical calculations which are not feasible to carry out without a computer [68].

The eigenvalue problem is also 1 such problem. This problem has been the subject of intensive research for many decades with proposals of new algorithms appearing frequently. These algorithms are required to be tested, among others, by providing numerical data. These numerical examples provide a systematic and rational means of evaluating the performance of programs implementing those algorithms executed by high-speed electronic computers. So equal emphasis is given on (**a**) the numerical solution of the problem, such as the computation of the zeros of the polynomial and the exact eigenvalues of the given matrix and (**b**) its converse problem, i.e., generating the problem from its known solution, which means the reconstruction of the polynomial from its known (prescribed) zeros and the generation of

matrices with preassigned eigenvalues. For the determination of eigenvalues the data to be provided is in the form of matrices with known eigenvalues. There exist many algorithms for generating such data. But, as it is known in computer parlance, that there exist algorithms which may be mathematically interesting, but which are certainly not suitable for developing a good software. For testing the accuracy of algorithms it is imperative that the matrices generated as data should be exact. So to provide exact numerical data and examples we have given emphasis on generating these matrices over the unique factorisation domain $\mathbf{Z}$ and over the exact field $\mathbf{Q}$ instead of an imprecise field such as $\mathbf{R}$. Since floating-point computation by nature is inexact, it is not difficult to misuse it in such a way that the computed answers will consist almost entirely of 'noise'. Although the floating-point mode is widely used in scientific calculations and all the major programming languages support floating-point calculations, still it should not be used without reservation because of possible errors in its implementaion. For example, when we try to invert a Hilbert (a very ill-conditioned) matrix, even for a relatively small order, using floating-point computations, we get a result very far from the correct inverse, but it is impossible to locate how and in what way the round-off errors have piled up to such an extent. Because of the complexity of piling up of such errors, it has been reported[64] that certain critical safety systems have prohibited the use of floating-point arithmetic.

There are well-known methods for the computation of eigenvalues. These methods depend on whether we have to compute all the eigenvalues, or some particular eigenvalues, e.g., the power method for the computation of the dominant (largest in magnitude) eigenvalue, and also on the special forms of

the matrices, such as the Jacobi method for the computation of the eigenvalues of a Hermitian matrix. But most of the algorithms require the floating-point mode for their implementation and so the problem of the generation of round-off errors arises with them.

It is known that there does not exist any 'stable' method for obtaining the eigenvalues of a non-symmetric matrix. So for calculating the exact eigenvalues of any matrix, irrespective of being of any special form, such as symmetric or Hermitian, and even as ill-conditioned as it can possibly be, we have adopted the technique of calculating its characteristic polynomial and then computing the zeros of this polynomial, which are known to be the eigenvalues of the given matrix. The major effort in this endeavour is concentrated on error-free computation.

Therefore we have developed algorithms which can be implemented using only fixed-point arithmetic. Consequently there is no possibility of the generation of any round-off error whatsoever in the calculations. However, the fixed-point implementation of such algorithms poses certain problems, such as in many algorithms, the integers become greater than the fixed-point limit for the particular computer on which we are trying to implement the algorithms. So a pre-requisite for the effective implementation of these algorithms is a multi-precision package.

Regarding the algorithms for the multiplication and division of multi-precision integers, possibly the best are due to Schönhage and Strassen which run in $O(n\ ln\ n\ ln\ ln\ n)$ bit operations, which is much faster than $O(n^2)$ as n gets large. These sophisticated algorithms become practical only for numbers having more than several hundred decimal digits and they are not worth-

while for numbers having upto roughly 100 decimal digits [10]. Moreover the arithmetic techniques for use in computer programming for multi-precision numbers differ considerably from the techniques used in the hardware implementation of arithmetic operations [31].

The first chapter of this thesis begins with algorithms for the 4 fundamental arithmetic operations of addition, subtraction, multiplication and division of multi-precision integers. The division of a multi-precision non-negative integer by a multi-precision natural number is the most difficult of the 4 arithmetic operations. The main difficulty in the mechanisation of the pencil-and-paper method of division is the determination of the digits of the quotient. This we do by first finding the trial digit of the quotient and then we find the exact digit, the implementation of which becomes very time consuming and cumbersome. To overcome this difficulty we have given an algorithm and proved that the trial digit $q' \leq 10$ and the exact digit $q=q'$ or $q'-1$ in the 10^d radix division, where $d \in \mathbf{N}$ and $d \geq 2$.

The next problem in this chapter is the evaluation of the square root of a natural number, which is considered to be next in importance to the 4 basic operations, since it finds wide application in many areas of Science and Engineering. Newton-Raphson method is 1 of the most widely used methods for calculating the square root of a positive number. We have used the Newton-Raphson method for the computation of the integer part of the square root of a positive integer x in preference to the technique described in [45] as it has been found that the speed of computation of the latter[45] is proportional to the binary length of the root. It is well-known that with the close initial first estimate of the square root, the quadratic nature of

this algorithm makes it converge fast. But this method does not provide any scheme for guessing the initial first estimate of the square root. We have presented an algorithm (together with a mathematically elegant proof) for obtaining the close initial first estimate of the square root of a positive integer.

The next problem which we have dealt with in the 1st chapter is the purely number-theoretic problem of calculating all the prime numbers from 2 to a given natural number n. We have used these prime numbers for the computation of all the zeros of a given polynomial. The algorithms which purport to solve this problem either require large storage or have comparatively slow speed of computation which is a hindrance to their practical implementation. We have developed an algorithm overcoming these 2 difficulties.

Permutations, combinations and partitions are 3 important problems in combinatorics. They have received much attention and various applications have been found. For example, the calculation of the characteristic polynomial of a square matrix by the direct expansion method requires the computation of all the combinations of the 1st n natural numbers taken r at a time ($1 \leq r \leq n$). As another example, let us consider the set of all the square matrices of a particular order n such that all their eigenvalues are equal to a given number. We must break up this set into mutually disjoint equivalence classes such that all the matrices in a particular equivalence class are similar to each other and no matrix in a particular equivalence class can be similar to any matrix in any other equivalence class. To find the number of such equivalence classes and the Jordan canonical forms representing each equivalence class we require the knowledge of all partitions of n. We have solved

these problems using the control structure proposed by Skordalakis and Papakonstantinou [51], which otherwise would have been solved by altogether different algorithms, e.g., [6,15,60].

The second chapter deals with the numerical solution of polynomial equations and also with the converse problem of reconstructing polynomials from their known zeros. We have started this chapter with the latter problem. The first implementation is of generating the monic polynomial $P(x) = \prod_{i=1}^{n} (x - a_i)$, $a_i \in \mathbf{Z}$, and then we have extended this algorithm to generate the primitive polynomial $Q(x) = \prod_{i=1}^{n} (xq_i - p_i)$, $p_i, q_i \in \mathbf{Z}$, $q_i \neq 0$. Thus we have provided numerical examples for testing the algorithms for the factorisation of polynomials over $\mathbf{Z}$ (or $\mathbf{Q}$). For constructing polynomials with integer coefficients and having zeros of the form $\frac{1}{2}(a \pm \sqrt{b})$, a,b $\in \mathbf{Z}$ and b, if positive, is not a perfect square, we have formed quadratic polynomials with such zeros and developed algorithms for multiplying such polynomials together or such polynomials with linear factors or both. The reconstruction of polynomials from their zeros is useful for testing algorithmic stability (i.e., whether the computed zeros of a given polynomial generate a polynomial which is only slightly different from the original polynomial in the case of performing all calculations in real arithmetic instead of integer arithmetic) [37].

The numerical solution of a polynomial equation is a difficult computational problem. We can view it as a non-linear problem in terms of linear algebra. Many algorithms which work easily on equations with well-separated simple zeros fail to work on more difficult problems [25]. Moreover, most of the algorithms have to work in the field $\mathbf{R}$ of real numbers, which is an im-

precise field due to the reason that we cannot express its irrational elements exactly; we can only express them upto a desired degree of accuracy. Since we are interested only in exact arithmetic, we may not be able to compute the exact zeros of a polynomial (even if they exist) using the existing algorithms. So we have developed algorithms for computing the exact zeros of a polynomial.

In many algorithms it is necessary that the polynomial in question should be square-free; this condition can be fulfilled by dividing the polynomial by the G.C.D. of itself and its derivative. The algorithm for calculating the G.C.D. of polynomials over $\mathbf{Z}$ is also included in this chapter. At the end of this chapter we have given an algorithm for reducing non-monic polynomials over $\mathbf{Z}$ to monic polynomials over $\mathbf{Z}$ (for computing the rational zeros of the original polynomials), and then we have extended this algorithm to generate another algorithm which reduces a polynomial with non-integer rational coefficients to a monic polynomial with integer coefficients. We have also proved that this reduction is possible without inordinately increasing the magnitudes of the coefficients.

In the third chapter the first algorithm, which we have given, is for computing the characteristic polynomial of a prescribed square matrix with integer entries. Efficient methods for this purpose are (**c**) Leverrier-Faddeev method, the computational complexity being n^4, and (**d**) the method of calculating the Hessenberg form of the matrix by a similarity transformation and then computing the characteristic polynomial using a recursive relation, the computational complexity being n^3. But the phenomenon of coefficient explosion neutralises the advantages of an n^3 method (**d**) over an n^4 method

(**c**), when the matrix under consideration is over the base ring **Z or Q**. Moreover the algorithm (**c**) is much simpler than the algorithm (**d**). So we have coded the subroutine for calculating the characteristic polynomial using the method (**c**).

Although, the method of direct expansion, i.e., the process of computing the coefficients $p_1, p_2, p_3, \ldots, p_{n-1}$ of the characteristic polynomial, as the sum of the principal minors of the matrix A, having orders $1,2,3,\ldots,(n-1)$, and the coefficient p_n, as the determinant of A, is not considered to be very efficient, since it requires the evaluation of too many determinants. The number of such determinants of different orders is $2^n - 1$. We have shown that the number of such determinants can be drastically reduced by modifying the Gaussian elimination method of evaluation of determinants.

The eigenvalues of a square matrix, are the zeros of the characteristic polynomial obtained by the above process. The zeros are calculated using the algorithms/subroutines which we have developed in Chapters 1 and 2.

It is well-known that every square matrix, although it need not be diagonalisable, is similar to a matrix whose diagonal entries are its eigenvalues and whose super-diagonal entries consist of only 0's and 1's. This matrix is called the Jordan canonical form of the given matrix and is denoted by J. So for every square matrix M, there exists a non-singular matrix P, and a unique matrix J, such that $PMP^{-1} = J$. We have computed the matrix P using the Gaussian method of solution of a system of linear simultaneous equations. The computation of the Jordan canonical form helps in deciding whether 2 given matrices are similar or not; the calculation of the invariants corresponding to a particular eigenvalue; and so on.

The concluding chapter (i.e., Chapter 4) of this thesis deals with the generation of matrices with known (pre-assigned) spectra, which provides numerical data to the subroutines developed in Chapter 3. For constructing such matrices, we have applied what is known as the 'grand strategy' of similarity transformations in the computation of eigenvalues. The simplest way of generating these matrices accurately is to have the matrices with all integer entries. In the human domain it is a general practice[16,33,34,56] to consider matrices over the base ring **Z** for providing numerical examples for the purposes to which we have referred at the beginning of this introduction. These matrices are very useful in integer programming which, in recent years, has become increasingly popular. It is not possible to obtain such non-trivial matrices with many of the existing algorithms for generating matrices, even if it is known that their spectra are composed of integers, such as the algorithm suggested by Hall and Porsching [22] for the generation of positive test matrices. In certain cases it is not possible to have such a matrix at all, e.g., it is not possible to have a positive matrix (all elements are positive) with integer entries, even with a small order such as 2, with eigenvalues 1,1. In such cases we have deduced necessary conditions which the eigenvalues must satisfy, so as to ensure that the entries of the generated matrix will belong to the set **N** of natural numbers, the principal ideal domain **Z**, the field **Q** and so on, and the matrix will have a special form, such as symmetric, positive, positive symmetric and so on.

The straightforward implementation of the similarity transformation requires $3n^3$ multiplication/division operations. Therefore we have chosen a modal matrix with known inverse, so that the similarity transformation can

be implemented only with addition/subtraction operations. This modal matrix will also facilitate the reduction of propagation of errors if the matrices are to be generated with spectra belonging to $\mathbf{R}$, the field of real numbers (or $\mathbf{C}$, the field of complex numbers).

It is quite evident that if in the similarity transformation PMP^{-1}, $P \in GL_n(\mathbf{Z})$, the group of unimodular matrices with integer entries, then $PMP^{-1} \in M_n(\mathbf{Z})$, the set of matrices with integer entries, provided that $M \in M_n(\mathbf{Z})$. So we have started Chapter 4 with algorithms for generating such unimodular matrices P with integer entries, and then we have extended the algorithms to produce other algorithms which generate a matrix with prescribed determinant, all of whose entries are integers except perhaps the last entry. The unimodular matrix P generated in this process is used in constructing non-trivial matrices with integer entries and prescribed integer spectra. The generated matrices may be of the following different properties:- derogatory, non-derogatory, diagonalisable etc. Since determinants and inverses of square matrices play a major role in these transformations, so the algorithms for error-free computations of determinants and inverses are also included in this chapter.

In the generation of matrices we have paid particular attention so that there is no unnecessary explosion of the entries of the generated matrices which generally accompany such algorithms. As such, these matrices will better meet the basic tasks, which every mathematician faces, such as the tasks of providing numerical examples to illustrate known theorems, and in many cases, to make up or to support conjectures, which may lead to the discovery of new algorithms. The usefulness of such matrices in testing the eigenvalue routines has been demonstrated in [33,34,66,67]. We have writ-

ten programs implementing all the algorithms (The listing of the programs together with sample input/output is given in the appendix.); the systematic testing of the programs for (**a**) the numerical solution of the problems, such as the zeros of a given polynomial and the exact eigenvalues of a given matrix has been accomplished in part with the aid of programs for (**b**) the reconstruction of polynomials from their known (prescribed) zeros and the generation of matrices with pre-assigned eigenvalues; and vice versa. The methods which we have discussed are error-free and work independently of any convergence criteria. This feature amply justifies the small number of extra arithmetic operations. Error-free computation permitted us to test for exactness in all the problems (for example, whether the computed roots generated a polynomial which was exactly the same as the original polynomial, etc.). This fulfilled the objective of this thesis and may also be used for analysing the performance of algorithms which purport to solve these and similar other problems, thereby permitting comparison to be made between theoretical results and computation.