İSTANBUL TECHNICAL UNIVERSITY ★ INFORMATICS INSTITUTE

SOLVING MATRIX EIGENVALUE PROBLEM USING HIGH DIMENSIONAL MODEL REPRESENTATION

M.Sc. THESIS

Esen AKKEMİK

Department: Informatics
Program : Computational Science and Engineering

Supervisor : Prof. Dr. Metin DEMİRALP

August 2003

İSTANBUL TECHNICAL UNIVERSITY ★ INFORMATICS INSTITUTE

SOLVING MATRIX EIGENVALUE PROBLEM USING HIGH DIMENSIONAL MODEL REPRESENTATION

M.Sc. THESIS

Esen AKKEMİK
702011019

Date of submission : 15 August 2003
Date of defence examination : 19 September 2003

Advisor (Chairman) : Prof. Dr. Metin DEMİRALP

Members of the Examining Committee : Assoc. Prof. Dr. Hasan DAĞ

Assoc. Prof. Dr. M. Serdar ÇELEBİ

August 2003

ACKNOWLEDGEMENTS

I would like to thank to my supervisor, Prof. Dr. Metin DEMİRALP, for his guidance, suggestions and comments.

I would also like to thank to my friends who helped me during my studies, especially Ayşegül, who has always been ready to answer my questions and examine my work in patience.

I am thankful to the coordinators of project team in TÜBİTAK-UEKAE for their tolerance during my studies at İTÜ, and also to my colleagues for their help.

I would also thank to State Planning Organization of Turkey for financial support.

I would like to thank Urartu, who always encouraged me with his love and support, and who gave me the strength and the hope to complete this work.

I am so grateful to my family, especially to my mother, who has always been with me all through my life and ready to give the love, support and help at any time I needed.

August 2003 Esen AKKEMİK

CONTENTS

ABBREVIATIONS

HDMR	High Dimensional Model Representation
YBMG	Yüksek Boyutlu Model Gösterilim
GSOP	Gram-Schmidt Orthogonalization Process

LIST OF TABLES

Tables

LIST OF FIGURES

Figures

LIST OF SYMBOLS

$\mathbf{A}$	$n \times n$ matrix
λ	Eigenvalue of $\mathbf{A}$
$\mathbf{x}$	Eigenvector of $\mathbf{A}$
$\mathbf{0}$	Zero vector
$\mathbf{I}$	Identity matrix
$\mathbf{L}$	Lower triangular matrix
$\mathbf{U}$	Upper triangular matrix
$\mathbf{Q}$	Unitary matrix
$\mathbf{R}$	Upper triangular real valued matrix
$\|\mathbf{x}\|$	Norm or magnitude of vector $\mathbf{x}$
$\mathbf{A}^*$	Conjugate transpose of a matrix $\mathbf{A}$
$f(x_1, \ldots, x_N)$	Multivariate function
f_0	Constant term in HDMR expansion
$f_i(x_i)$	Univariate HDMR component
$f_{i_1 i_2}(x_{i_1}, x_{i_2})$	Bivariate HDMR component
$f_{12\ldots N}(x_1, x_2, \ldots, x_N)$	N-variate HDMR component
$W_i(x_i)$	Weight function depending on x_i
$[a_i, b_i]$	Integral interval
$\mathcal{P}_i$	Projection operator
$\mathcal{I}$	Identity projection operator
σ_0	Constancy
$\tilde{\sigma}_0$	Zeroth Order Convergence Measure
σ_i	i.th Order Aditivity Measures
$\tilde{\sigma}_i$	i.th Order Convergence Measure
$\mathbf{u}$	Eigenvector of $\mathbf{A}$
x_i	Independent variables in HDMR,
$\mathbf{A}_i$	Matrix whose non zero element, 1, is in the position of x_i
$\mathbf{u}_0$	Constant term in HDMR expansion of $\mathbf{u}$
$\mathbf{u}_i(x_i)$	Univariate term in HDMR expansion of $\mathbf{u}$

$\mathbf{u}_{i_1 i_2}(x_{i_1}, x_{i_2})$	Bivariate component in HDMR expansion of $\mathbf{u}$
$\mathbf{u}_{i_1 i_2 i_3}(x_{i_1}, x_{i_2}, x_{i_3})$	3-variate component
$\mathbf{\Theta}_i$	$N \times N$ matrix
α_i	Vector of dimension N
$\lambda^{(0)}$	Zeroth Order HDMR Approximation of λ
$\lambda^{(1)}$	First Order HDMR Approximation of λ
$\mathbf{u}_0^{(0)}$	Zeroth Order HDMR Approximation of constant term $\mathbf{u}_0$
$\mathbf{u}_0^{(1)}$	First Order HDMR Approximation of constant term $\mathbf{u}_0$
$\mathbf{u}_k^{(1)}$	First Order HDMR Approximation of univariate HDMR component
$\mathcal{A}$	$N \times N$ matrix depending on x_k and λ
S	The set of indices of the elements on the diagonal
$\mathbf{u}$	The vector whose elements are 1
$\mathbf{e}_i$	i.th standard unit vector
$\mathcal{U}_1$	Matrix depending on λ
$\mathcal{U}_2$	Matrix depending on λ
α_k^d	α_k for diagonal elements
α_k^{od}	α_k for off diagonal elements
$\mathbf{\Gamma}$	Matrix depending on λ
$\mathbf{\Lambda}$	Matrix depending on λ
$\mathbf{\Omega}$	Matrix depending on λ
$\mathbf{\Delta}$	Matrix depending on λ
$\mathbf{\Pi}_1$	Matrix depending on λ
$\mathbf{\Pi}_2$	Matrix depending on λ
$\tilde{\mathbf{A}}$	Augmented matrix of $\mathbf{A}$
C_{ik}	Cofactor of the entry a_{ik} in D
M_{ik}	Determinant of M otained from $\mathbf{A}$ by deleting row and column of entry a_{ik}
Θ	Angle between two vectors
$\hat{\mathbf{y}}_i$	Vector obtained from GSOP

ÖZET

Matris özdeğer problemlerinin çözümünde azımsanmayacak derecede fazla bilgi birikimi vardır. Varolan yöntemler düşük matris boyutlarında iyi çalışır. Eğer çalışılan boyut sonlu ise boyuttan bağımsız olarak toplamlar için yakınsama sorunu yoktur. Ele alınan matris boyutu sonsuzlaştıkça toplamlarda yakınsama sorunu göze çarpar. Aslında sonlu boyuttaki sistemlerde bile sonlu toplamlar boyut arttıkça hata birikimine neden olabilirler. Bu yüzden boyutluluk matris boyutu sonsuza doğru gittikçe önemli hale gelir. Bu durum bilimcileri sonsuz boyut sınırında da iyi çalışabilen çeşitli yakınsama teknikleri geliştirmeye itmiştir. Bu tekniklerden biri de yüksek sayıda bağımsız değişken içeren çok değişkenli bir fonksiyonu bir tane sabit fonksiyonun, N tane sadece bir değişkene bağlı fonksiyonların (N burada değişken sayısı), $N(N-1)/2$ tane sadece iki değişkene bağlı fonksiyonların ve benzer şekilde ilerleyen, basit yapılı fonksiyonların toplamı şekliyle ifade eden açılıma dayalı olan tekniktir. Bu yöndeki ilk çalışma Sobol [17] tarafından yapılmıştır. Bu açılımın amacı bazı istatistiksel uygulamalardı. Daha sonra bu açılım Rabitz ([10], [11], [13], [14], [16]) tarafından yenilenerek ve genişletilerek başka alanlara uygulanmıştır. Bu açılım Yüksek Boyutlu Model Gösterilim (YBMG) olarak isimlendirilmiştir.

Bu çalışmada, $\mathbf{Au} = \lambda\mathbf{u}$ ile belirtilen matris özdeğer probleminin çözümü olan özdeğerin, λ, ve özvektörün, $\mathbf{u}$, matris boyutunun çok büyük olduğu durumlarda bilinen metodların kullanışlılığını yitirmesinden dolayı YBMG ile çözümleri incelenmiştir. Matrisin elemanları değiştikçe özdeğer ve özvektör de değişeceğinden dolayı çalışmada matrisin elemanları YBMG bağımsız değişkenleri olarak ele alınmıştır. Özdeğer, bağımsız değişkenlere bağlı bir fonksiyon olmasına rağmen YBMG'nin doğrusal yapılı problemlerde iyi çalışması nedeniyle bağımsız değişkenmiş gibi görülmüştür ve işlemler sonunda çıkacak özdeğer problemiyle hesaplanmıştır.

Herhangi boyuttaki bir matris için, özvektörün YBMG açılımının matris özdeğer probleminde yerine yerleştirilmesiyle elde edilen denkleme önceden belirlenen aşkın prizma (hyperprism) üzerinde YBMG uygulanarak özvektörün sabit terim ve bir değişkene bağlı YBMG bileşenlerini veren denklemler elde edilmiştir ve özvektöre bağlı olarak özdeğer bulunmuştur. Bazı sayısal örnekler

de verilmiştir. Sayısal sınamalar için simgesel programlama olarak MuPAD 2.5.2 kullanılmıştır ve YBMG yaklaştırımını veren program ek olarak sunulmuştur.

SUMMARY

An enormous knowledge accumulation exists for the solution of matrix eigenvalue equations. The existing routines work well for low or moderate matrix dimensions and there is a strong basis for convergence and error analysis which is mostly based on the norms or similar entities. If the dimension is finite, there are no additive convergence problems for the sums regardless of the magnitude of the dimension. When the matrices under consideration become infinite dimensional a matter of convergence appears in the sums. Even in the case of finite dimensional systems the finite sums may cause error accumulations when the dimension grows up unboundedly. Hence, the dimensionality plays an important role when the dimension of the matrix under consideration grows up to infinity. This urged scientists try to develop various approximation techniques working well at the infinite dimension limit and construct within an additive expansion such that the first contributive term is a constant followed by N number of univariate terms (N is the number of independent variables), $N(N-1)/2$ bivariate terms and so on. The first attempt for the applications toward this direction was made by Sobol [17]. The purpose of the expansion was some statistical applications. This expansion formula has been revisited, revised, generalized and applied to various problems by Rabitz ([10], [11], [13], [14], [16]). The method has been called High Dimensional Model Representation (HDMR). In the solution of the eigenvalue problems of integral operators we can refer to HDMR especially when the number of the independent variables grows unboundedly.

In this work, the solution of algebraic eigenvalue problem $\mathbf{Au} = \lambda\mathbf{u}$, eigenvalue λ and eigenvector $\mathbf{u}$, using HDMR is studied. The elements of the matrix $\mathbf{A}$ are taken as independent HDMR variables, and since the changes in elements results in changes in the eigenvector, u is taken as dependent HDMR variable. Although eigenvalue is also a dependent variable, it is taken as if it were independent variable during the process, due to the power of HDMR in problems having linear form.

HDMR of eigenvector $\mathbf{u}$ is replaced in the equation, and HDMR is applied to the resulting equation over a predetermined hyperprism. The equations

for constant HDMR term and the univariate HDMR functions are determined, and the value of the eigenvalue is determined from the resulting algebraic eigenvalue problem. Moreover, the approximation by using only these two HDMR components is also determined by the help of additivity measures. Also, some numerical experiments are given. In order to get numerical results MuPAD 2.5.2 is used for symbolic programming and the program is given at the end of this work.

SECTION 1

INTRODUCTION

Consider a system of non homogeneous linear algebraic equations:

$$\mathbf{Cx} = \mathbf{b}. \tag{1.1}$$

By using Cramer's rule (see in Appendix D), $\mathbf{x}$ can be found as

$$x_j = \frac{\det(\mathbf{C}^j)}{\det(\mathbf{C})} \qquad \mathrm{j} = 1, 2, \ldots, \mathrm{n}, \tag{1.2}$$

where matrix $\mathbf{C}^j$ is matrix $\mathbf{C}$ with column j replaced by the vector $\mathbf{b}$. In general, $\det(\mathbf{C}) \neq 0$, and unique x_j's values are found.

Consider a system of homogeneous linear algebraic equations:

$$\mathbf{Cx} = \mathbf{0}. \tag{1.3}$$

Solving for $\mathbf{x}$ by using Cramer's rule yields

$$x_j = \frac{\det(\mathbf{C}^j)}{\det(\mathbf{C})} = \frac{0}{|\mathbf{C}|} \qquad \mathrm{j} = 1, 2, \ldots, \mathrm{n} \tag{1.4}$$

Thus, $\mathbf{x} = \mathbf{0}$ if $\det(\mathbf{C}) \neq 0$. In general, $\det(\mathbf{C}) \neq 0$ and the only solution is the trivial solution, which is $\mathbf{x} = \mathbf{0}$. For certain forms of $\mathbf{C}$ that involve an uncertain arbitrary scalar λ, the value of λ can be chosen to force $\det(\mathbf{C}) = 0$, so that a nontrivial solution is possible.

Consider the coefficient matrix

$$\mathbf{C} = \mathbf{A} - \lambda\mathbf{B}, \tag{1.5}$$

where λ is unspecified scalar. Then, multiplying the above equation by $\mathbf{x}$ and equate it to 0, we get

$$\mathbf{Cx} = (\mathbf{A} - \lambda\mathbf{B})\mathbf{x} = \mathbf{0}. \tag{1.6}$$

The λ values can be determined by

$$\det(\mathbf{C}) = \det(\mathbf{A} - \lambda\mathbf{B})\mathbf{x} = \mathbf{0}. \tag{1.7}$$

The corresponding values of λ are called *eigenvalues*.

The general homogeneous system of equations can be written as

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{B}\mathbf{x}. \tag{1.8}$$

In many problems, $\mathbf{B} = \mathbf{I}$, and the above equation becomes

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x}. \tag{1.9}$$

In this work, we will also deal with the case where $\mathbf{B} = \mathbf{I}$.

In problems where $\mathbf{B} \neq \mathbf{I}$, define another matrix $\overline{\mathbf{A}} = \mathbf{B}^{-1}\mathbf{A}$. Then the equation (1.8) becomes

$$\overline{\mathbf{A}}\mathbf{x} = \lambda\mathbf{x},$$

which is again in the form of (1.9).

Alternatively, (1.9) can be written in the form

$$(\mathbf{A} - \lambda\mathbf{I})\mathbf{x} = 0, \tag{1.10}$$

which is the most common form of an eigenproblem.

The procedure to find the eigenvalues is to expand $\det(\mathbf{A} - \lambda\mathbf{I}) = \mathbf{0}$ and to find the roots of the resulting nth degree polynomial, which is called the *characteristic polynomial (equation)*. This procedure can be illustrated in the following discussion.

Consider a system given as

$$(\mathbf{A} - \lambda\mathbf{I})\mathbf{x} = \begin{pmatrix} (8-\lambda) & -2 & -2 \\ -2 & (4-\lambda) & -2 \\ -2 & -2 & (13-\lambda) \end{pmatrix} \mathbf{x} = \mathbf{0}. \tag{1.11}$$

The characteristic equation, $|\mathbf{A} - \lambda\mathbf{I}| = 0$, is

$$(8-\lambda)[(4-\lambda)(13-\lambda) - 4] - (-2)[(-2)(13-\lambda) - 4] + (-2)[4 + 2(4-\lambda)] = 0.$$

$$\lambda^3 - 25\lambda^2 + 176\lambda - 300 = 0 \tag{1.12}$$

The eigenvalues are

$$\lambda_1 = 13.870585 \qquad \lambda_2 = 8.620434 \qquad \lambda_3 = 2.508981.$$

The method for obtaining the roots of the equation (1.12) will not be considered here, because it is out of the scope of this work. But, it can be found in any book containing iterative methods for solution of nonlinear equations.

In summary, eigenproblems arise from homogeneous systems of equations that contain an unspecified arbitrary parameter in the coefficients. The characteristic polynomial is determined by expanding the determinant

$$\det(\mathbf{A} - \lambda\mathbf{I}) = 0,$$

which yields an nth degree polynomial in terms of λ. Solving the characteristic polynomial results in n eigenvalues λ_i $(i = 1, 2, \ldots, n)$. The n eigenvectors $\mathbf{x}_i$ $(i = 1, 2, \ldots, n)$ can be found by substituting the individual eigenvalues into the homogeneous systems of equations, which is then solved for eigenvectors.

In principle, the solution of eigenproblem is straightforward. In practice, when the size of the system of equations is very large, expanding the characteristic determinant to obtain the characteristic equation is difficult. Solving high-degree polynomials for the eigenvalues presents another problem. Consequently, more straightforward procedures for solving eigenproblems are desired.

1.1 Iterative Numerical Methods For Eigenproblems

In next sections, some iterative methods for solving eigenvalues and eigenvectors ([1], [4]) will be given in details and some examples will be given for accuracy.

1.1.1 The Power Method

The *power method* is based on repetitive multiplication of an initial eigenvector by matrix $\mathbf{A}$ with a scaling vector $\mathbf{y}$, so that the scaling vector approaches the *largest* eigenvalue λ and the scaled eigenvector approaches the corresponding eigenvector $\mathbf{x}$. The power method and several variations will be presented in this section.

1.1.1.1 The Direct Power Method

When the largest (in absolute value) eigenvalue of $\mathbf{A}$ is distinct, its value can be determined using an iterative technique called the *direct power method*. The procedure is as follows:

(1) Take an initial vector $\mathbf{x}^{(0)}$ for the eigenvector $\mathbf{x}$. Choose one entry of $\mathbf{x}$ to be unity, i. e. as 1. Designate that entry as the *unity* entry.

(2) Perform the matrix multiplication:

$$\mathbf{A}\mathbf{x}^{(0)} = \mathbf{y}^{(1)}.$$

(3) Scale $\mathbf{y}^{(1)}$ so that the unity component (entry) remains unity:

$$\mathbf{y}^{(1)} = \lambda^{(1)}\mathbf{x}^{(1)}.$$

(4) Repeat steps 2 and 3 with $\mathbf{x} = \mathbf{x}^{(1)}$. Iterate to convergence, at convergence, the value for λ is the largest eigenvalue of $\mathbf{A}$, and the vector $\mathbf{x}$ is the corresponding eigenvector.

The general algorithm for the power method is follows;

$$\mathbf{A}\mathbf{x}^{(k)} = \mathbf{y}^{(k+1)} = \lambda^{(k+1)}\mathbf{x}^{(k+1)}. \tag{1.13}$$

When the iterations indicate that the unity component could be zero, a different entry must be chosen. The method is slow to converge when the magnitudes (in absolute value) of the largest eigenvalues are close to each other. When the largest eigenvalues are of the same magnitude (in absolute value), this method fails.

Example 1 The direct power method

Find the largest (in absolute value) eigenvalue and the corresponding eigenvector of the matrix given by

$$\mathbf{A} = \begin{pmatrix} 8 & -2 & -2 \\ -2 & 4 & -2 \\ -2 & -2 & 13 \end{pmatrix}. \tag{1.14}$$

Assume $\mathbf{x}^{(0)} = [1.0\ 1.0\ 1.0]^T$. This is the initial vector, scale the third component $\mathbf{x}_3$ to unity. Then apply (1.13).

$$\mathbf{A}\mathbf{x}^{(0)} = \begin{pmatrix} 8 & -2 & -2 \\ -2 & 4 & -2 \\ -2 & -2 & 13 \end{pmatrix}\begin{pmatrix} 1.0 \\ 1.0 \\ 1.0 \end{pmatrix} = \begin{pmatrix} 4.0 \\ 0.0 \\ 9.0 \end{pmatrix}, \lambda^{(1)} = 9.0, \quad \mathbf{x}^{(1)} = \begin{pmatrix} 0.444444 \\ 0.000000 \\ 1.000000 \end{pmatrix}, \tag{1.15}$$

$$\mathbf{A}\mathbf{x}^{(1)} = \begin{pmatrix} 8 & -2 & -2 \\ -2 & 4 & -2 \\ -2 & -2 & 13 \end{pmatrix} \quad \begin{pmatrix} 0.444444 \\ 0.000000 \\ 1.000000 \end{pmatrix} = \begin{pmatrix} 1.555555 \\ -2.888888 \\ 12.111111 \end{pmatrix},$$

$$\lambda^{(2)} = 12.111111, \quad \mathbf{x}^{(2)} = \begin{pmatrix} 0.128440 \\ -0.238532 \\ 1.000000 \end{pmatrix}. \tag{1.16}$$

Table 1.1: The Power Method

k	λ	x_1	x_2	x_3
0		1.000000	1.000000	1.000000
1	9.000000	0.444444	0.000000	1.000000
2	12.111111	0.128440	-0.238532	1.000000
3	13.220183	-0.037474	-0.242887	1.000000
4	13.560722	-0.133770	-0.213602	1.000000
5	13.694744	-0.192991	-0.188895	1.000000
...	.....	.....	.....	.....
29	13.870583	-0.291793	-0.143499	1.000000
30	13.870584	-0.291794	-0.143499	1.000000

The results of the first two iterations presented here and subsequent iterations are presented in Table 1.1. The results were taken from [4], and the iterations were continued until λ changed by less than 0.000001 between two subsequent iterations. The final solution for the largest eigenvalue, denoted as λ_1, and the corresponding eigenvector $\mathbf{x}_1$ is

$$\lambda_1 = 13.870584 \qquad \text{and} \qquad \mathbf{x}_1 = [-0.291794 \quad -0.143499 \quad 1.000000]^T.$$

This problem converges so slowly, which is a large number for a 3×3 matrix. A procedure for accelerating the convergence of a slowly converging eigenproblem will be presented later in this chapter.

1.1.1.2 Basis of the Power Method

Before starting, let us give some necessary theorems.

THEOREM 1 Let $\lambda_1, \lambda_2, \ldots, \lambda_k$ be distinct eigenvalues of an $n \times n$ matrix. Then corresponding eigenvectors $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k$ form a linearly independent set.

(You can see the proof of this theorem in Appendix H.)

THEOREM 2 If an $n \times n$ matrix $\mathbf{A}$ has n distinct eigenvalues, then $\mathbf{A}$ has a basis of eigenvectors for $\mathbb{R}^n$.

(You can see the proof of this theorem in Appendix H.)

Assume that $\mathbf{A}$ is an $n \times n$ nonsingular matrix having n eigenvalues, $\lambda_1, \lambda_2, \ldots, \lambda_n$ with n corresponding linearly independent eigenvectors, $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$. Assume that $|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n|$. Since the eigenvectors, $\mathbf{x}_i$ $(i = 1, 2, \ldots, n)$, are linearly independent by Theorem 1, i.e., they span the n-dimensional space by Theorem 2, so any arbitrary vector $\mathbf{x}$ can be written as a linear combination of these eigenvectors. Therefore,

$$\mathbf{x} = C_1\mathbf{x}_1 + C_2\mathbf{x}_2 + \cdots + C_n\mathbf{x}_n = \sum_{i=1}^{n} C_i\mathbf{x}_i \tag{1.17}$$

can be written. Multiplying both sides of the above equation by $\mathbf{A}, \mathbf{A}^2, \ldots, \mathbf{A}^k$, etc., where the superscript denotes the repetitive matrix multiplication, and considering
$\mathbf{A}\mathbf{x}_i = \lambda_i \mathbf{x}_i$, we get

$$\begin{aligned}
\mathbf{A}\mathbf{x} &= \sum_{i=1}^{n} C_i \mathbf{A}\mathbf{x}_i = \sum_{i=1}^{n} C_i \lambda_i \mathbf{x}_i = \mathbf{y}^{(1)}, \\
\mathbf{A}^2\mathbf{x} &= \mathbf{A}\mathbf{y}^{(1)} = \sum_{i=1}^{n} C_i \lambda_i \mathbf{A}\mathbf{x}_i = \sum_{i=1}^{n} C_i \lambda_i^2 \mathbf{x}_i = \mathbf{y}^{(2)}, \\
\ldots\ldots & \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
\mathbf{A}^k\mathbf{x} &= \mathbf{A}\mathbf{y}^{(k-1)} = \sum_{i=1}^{n} C_i \lambda_i^{k-1} \mathbf{A}\mathbf{x}_i = \sum_{i=1}^{n} C_i \lambda_i^k \mathbf{x}_i = \mathbf{y}^{(k)}.
\end{aligned} \tag{1.18}$$

Factoring λ_1^k out of the next to last term in (1.18)

$$\mathbf{A}^k\mathbf{x} = \lambda_1^k \sum_{i=1}^{n} C_i \left(\frac{\lambda_i}{\lambda_1}\right)^k \mathbf{x}_i = \mathbf{y}^{(k)}. \tag{1.19}$$

Since $|\lambda_1| > |\lambda_i|$ for $i = 1, 2, \ldots, n$, the ratios $\left(\frac{\lambda_i}{\lambda_1}\right)^k \to 0$ as $k \to \infty$, and equation (1.19) approaches the limit

$$\mathbf{A}^k\mathbf{x} = \lambda_1^k C_1 \mathbf{x}_1 = \mathbf{y}^{(k)}. \tag{1.20}$$

Equation (1.20) approaches zero if $|\lambda_1| < 1$ and approaches infinity if $|\lambda_1| > 1$. Thus, equation (1.20) must be scaled between the steps.

Scaling can be done by scaling any component of vector $\mathbf{y}^{(k)}$ to unity at each step in the process, as it is done in the previous section. Choose, say, the first component of vector $\mathbf{y}^{(k)}, y_1^{(k)}$, to be that component. Thus, $\mathbf{x}_1 = 1.0$ and the first component of equation (1.20) is

$$y_1^{(k)} = \lambda_1^k C_1.$$

Applying equation (1.20) one more time, i.e. from k to $k+1$ yields

$$y_1^{(k+1)} = \lambda_1^{k+1} C_1. \tag{1.21}$$

Taking the ratio of (1.20) and (1.21), we get

$$\frac{y_1^{(k+1)}}{y_1^{(k)}} = \frac{\lambda_1^{k+1} C_1}{\lambda_1^k C_1} = \lambda_1. \tag{1.22}$$

Thus, if $y_1^{(k)} = 1$, then $y_1^{(k+1)} = \lambda_1$. If $y_1^{(k+1)}$ is scaled by λ_1 so that $y_1^{(k+1)} = 1$, then $y_1^{(k+1)} = \lambda_1$, etc. Consequently, scaling a particular component of vector

$\mathbf{y}$ each iteration essentially factors λ_1 out of vector $\mathbf{y}$, so that (1.20) converges to a finite value. In the limit as $k \to \infty$, the scaling factor approaches λ_1, and the scaled vector $\mathbf{y}$ approaches the eigenvector $\mathbf{x}_1$. However, several restrictions apply to the power method.

(1) The largest eigenvalue must be distinct.

(2) The n eigenvectors must be independent.

(3) The initial guess $\mathbf{x}_i^{(0)}$ must contain some component of eigenvector $\mathbf{x}_i$, so that $C_i \neq 0$.

(4) The convergence rate is proportional to the ratio

$$\frac{|\lambda_i|}{|\lambda_{i-1}|,} \tag{1.23}$$

where λ_i is the largest (in magnitude) eigenvalue and λ_{i-1} is the second largest (in magnitude) eigenvalue [4].

1.1.1.3 The Inverse Power Method

When the smallest eigenvalue (in absolute value) of matrix $\mathbf{A}$ is distinct, its value can be found by a variation of the power method called *inverse power method* [4].

Before starting this method, let us give a useful definition and theorem, the results of which will be used in the next sections.

THEOREM 3 Let $\mathbf{A}$ be an invertible $n \times n$ matrix.

(1) λ is an eigenvalue of $\mathbf{A}$ if, and only if, λ^{-1} is an eigenvalue of $\mathbf{A}^{-1}$.

(2) Let $\mathbf{x} \neq 0$, then $\mathbf{A}\mathbf{x} = \lambda\mathbf{x}$ if, and only if, $\mathbf{A}^{-1}\mathbf{x} = \lambda^{-1}\mathbf{x}$.

(3) Let $s \in \mathbb{R}$, then λ is an eigenvalue of $\mathbf{A}$ if, and only if, $\lambda + s$ is an eigenvalue of $(\mathbf{A} + s\mathbf{I})$.

(4) Let $\mathbf{x} \neq \mathbf{0}$, then $\mathbf{A}\mathbf{x} = \lambda\mathbf{x}$ if, and only if, $(\mathbf{A} + s\mathbf{I})\mathbf{x} = (\lambda + s)\mathbf{x}$

(You can see the proof of this theorem in Appendix H.)

In fact, this involves finding the largest eigenvalue (in magnitude) of the inverse matrix $\mathbf{A}^{-1}$, which is the smallest eigenvalue of matrix $\mathbf{A}$ (from Theorem 3).

Remember the original eigenvalue problem:

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x}. \tag{1.24}$$

Multiplying (1.24) by $\mathbf{A}^{-1}$ gives

$$\mathbf{A}^{-1}\mathbf{A}\mathbf{x} = \mathbf{I}\mathbf{x} = \lambda\mathbf{A}^{-1}\mathbf{x}. \tag{1.25}$$

When equation (1.25) is rearranged, another eigenvalue problem for $\mathbf{A}^{-1}$ yields. Thus,

$$\mathbf{A}^{-1}\mathbf{x} = \left(\frac{1}{\lambda}\right)\mathbf{x} = \lambda_{inverse}\mathbf{x}. \tag{1.26}$$

By the Theorem 3 above, we know that the eigenvalues of the inverse matrix $\mathbf{A}^{-1}$ is the reciprocals of the eigenvalues of matrix $\mathbf{A}$. The power method can be used to find the largest (in absolute value) eigenvalue of matrix $\mathbf{A}^{-1}$, $\lambda_{inverse}$. The reciprocal of that eigenvalue is the smallest eigenvalue of the original matrix $\mathbf{A}$.

In practice, the LU method (see in Appendix E) is used to solve the inverse eigenproblem instead of calculating the inverse matrix $\mathbf{A}^{-1}$. The power method applied to matrix $\mathbf{A}^{-1}$ is given by

$$\mathbf{A}^{-1}\mathbf{x}^{(k)} = \mathbf{y}^{(k+1)}. \tag{1.27}$$

Multiplying (1.27) by $\mathbf{A}$ gives

$$\mathbf{A}\mathbf{A}^{-1}\mathbf{x}^{(k)} = \mathbf{I}\mathbf{x}^{(k)} = \mathbf{x}^{(k)} = \mathbf{A}\mathbf{y}^{(k+1)}, \tag{1.28}$$

which can be rewritten as

$$\mathbf{A}\mathbf{y}^{(k+1)} = \mathbf{x}^{(k)}. \tag{1.29}$$

Equation (1.29) is in the form of $\mathbf{A}\mathbf{x} = b$ where $\mathbf{x} = \mathbf{y}^{(k+1)}$ and $b = \mathbf{x}^{(k)}$. Thus, for a given $\mathbf{x}^{(k)}$, the solution vector $\mathbf{y}^{(k+1)}$ can be determined by Doolittle LU Method [5] (see in Appendix E).

(1) Solve for $\mathbf{L}$ and $\mathbf{U}$ such that $\mathbf{A} = \mathbf{L}\mathbf{U}$ by the Doolittle LU Method.

(2) Take initial vector $\mathbf{x}^{(0)}$, designate a component of $\mathbf{x}$ to be unity.

(3) Solve for $\mathbf{x}'$ by forward substitution using the equation

$$\mathbf{L}\mathbf{x}' = \mathbf{x}^{(0)}. \tag{1.30}$$

(4) Solve for $\mathbf{y}^{(1)}$ by backward substitution using the equation

$$\mathbf{U}\mathbf{y}^{(1)} = \mathbf{x}'. \tag{1.31}$$

(5) Scale $\mathbf{y}^{(1)}$ so that the unity component is unity. Then,

$$\mathbf{y}^{(1)} = \lambda^{(1)}_{inverse}\mathbf{x}^{(1)}. \tag{1.32}$$

(6) Repeat steps 3 to 5 with $\mathbf{x}^{(1)}$. Iterate to convergence, at convergence, $\lambda = 1/\lambda_{inverse}$ and $\mathbf{x}^{(k+1)}$ is the corresponding eigenvector.

The inverse power method can be illustrated as follows [4]:

$$\begin{aligned}
&\mathbf{L}\mathbf{x}' = \mathbf{x}^{(k)} \\
&\mathbf{U}\mathbf{y}^{(k+1)} = \mathbf{x}' \\
&\mathbf{y}^{(k+1)} = \lambda_{inverse}^{(k+1)}\mathbf{x}^{(k+1)}.
\end{aligned}$$

Example 2 The inverse power method

Find the smallest eigenvalue and the corresponding eigenvector of the matrix in Example 1.

Assume $\mathbf{x}^{(0)} = [1.0 \;\; 1.0 \;\; 1.0]^T$. Scale the first component of $\mathbf{x}$ to unity. The first step is to solve $\mathbf{L}$ and $\mathbf{U}$ by the Doolittle LU method. The results are

$$\mathbf{L} = \begin{pmatrix} 1 & 0 & 0 \\ -1/4 & 1 & 0 \\ -1/4 & -5/7 & 1 \end{pmatrix} \quad \text{and} \quad \mathbf{U} = \begin{pmatrix} 8 & -2 & -2 \\ 0 & 7/2 & -5/2 \\ 0 & 0 & 75/7 \end{pmatrix}.$$

Solve for $\mathbf{x}'$ by forward substitution using $\mathbf{L}\mathbf{x}' = \mathbf{x}^{(0)}$.

$$\begin{pmatrix} 1 & 0 & 0 \\ -1/4 & 1 & 0 \\ -1/4 & -5/7 & 1 \end{pmatrix} \begin{pmatrix} x_1' \\ x_2' \\ x_3' \end{pmatrix} = \begin{pmatrix} 1.0 \\ 1.0 \\ 1.0 \end{pmatrix},$$

$$\begin{aligned}
x_1' &= 1.0, \\
x_2' &= 1.0 - (-1/4)(1.0) = 5/4, \\
x_3' &= 1.0 - (-1/4)(1.0) - (-5/7)(5/4) = 15/7.
\end{aligned}$$

Solve for $\mathbf{y}^{(1)}$ by backward substitution using $\mathbf{U}\mathbf{y}^{(1)} = \mathbf{x}'$.

$$\begin{pmatrix} 8 & -2 & -2 \\ 0 & 7/2 & -5/2 \\ 0 & 0 & 75/7 \end{pmatrix} \begin{pmatrix} y_1^{(1)} \\ y_2^{(1)} \\ y_3^{(1)} \end{pmatrix} = \begin{pmatrix} 1.0 \\ 5/4 \\ 15/7 \end{pmatrix},$$

$$\begin{aligned}
y_1^{(1)} &= [1.0 - (-2.0)(0.5) - (-2)(0.2)]/8 = 0.3, \\
y_2^{(1)} &= [5/4 - (-5/2)(0.2)]/(7/2) = 0.5, \\
y_3^{(1)} &= (15/7)/(75/7) = 0.2.
\end{aligned}$$

Scale $\mathbf{y}^{(1)}$ so that the unity component is unity.

$$\mathbf{y}^{(1)} = \begin{pmatrix} 0.30 \\ 0.50 \\ 0.20 \end{pmatrix} \qquad \lambda^{(1)}_{inverse} = 0.3000000 \qquad \mathbf{x}^{(1)} = \begin{pmatrix} 1.000000 \\ 1.666667 \\ 0.666667 \end{pmatrix}.$$

The subsequent iterations will not be presented here, they will be given in Table 1.2. The iterations were continued until $\lambda_{inverse}$ changed less than 0.000001 between iterations.

The final solution for the smallest eigenvalue λ_3 and the corresponding eigenvector $\mathbf{x}_3$ is

$$\lambda_3 = \frac{1}{\lambda_{inverse}} = \frac{1}{0.398568} = 2.508981,$$

and

$$\mathbf{x}_3 = [1.000000 \quad 2.145797 \quad 0.599712]^T.$$

Table 1.2: The Inverse Power Method

k	$\lambda inverse$	x_1	x_2	x_3
0		1.000000	1.000000	1.000000
1	0.3000000	1.000000	1.666667	0.666667
2	0.353333	1.000000	1.981132	0.603774
3	0.382264	1.000000	2.094439	0.597565
4	0.393346	1.000000	2.130396	0.598460
...	.....	.....	.....	.....
12	0.398568	1.000000	2.145796	0.599712
13	0.398568	1.000000	2.145797	0.599712

1.1.1.4 The Shifted Power Method

The eigenvalues of a matrix $\mathbf{A}$ can be shifted by subtracting $s\mathbf{x}$ from both sides of the eigenvalue problem. Then,

$$\mathbf{A}\mathbf{x} - s\mathbf{x} = \lambda\mathbf{x} - s\mathbf{x}, \tag{1.33}$$

which yields

$$(\mathbf{A} - s\mathbf{I})\mathbf{x} = (\lambda - s)\mathbf{x}, \tag{1.34}$$

which can be rewritten as

$$\mathbf{A}_{shifted}\mathbf{x} = \lambda_{shifted}\mathbf{x}, \tag{1.35}$$

where $\mathbf{A}_{shifted} = (\mathbf{A} - s\mathbf{I})$ and $\lambda_{shifted} = \lambda - s$ is the eigenvalue of the shifted matrix. By Theorem 3, the eigenvectors of $\mathbf{A}$ and shifted matrix $\mathbf{A} - s\mathbf{I}$ are the same.

Shifting the eigenvalues of a matrix can be used for the following purposes:

(1) Find the opposite extreme eigenvalue, which is either the smallest or the largest (in absolute) eigenvalue of opposite sign,

(2) Find intermediate eigenvalues,

(3) Accelerate convergence for slowly converging eigenproblems.

(1) *Shifting eigenvalues to find the opposite extreme eigenvalue*

Let us consider an eigenvalue whose eigenvalues are all the same sign, say 1,2,4 and 8. For this matrix 8 is the largest eigenvalue and 1 is the opposite extreme eigenvalue. Firstly, we will solve the equation for the largest eigenvalue $\lambda_{largest} = 8$ by the direct power method. Shifting the eigenvalues by $s = 8$ yields the shifted eigenvalues -7, -6, -4 and 0. Now, we will solve for the largest shifted eigenvalue (in absolute value) of the matrix, $\lambda_{shifted,largest} = -7$ by the power method. Then the smallest eigenvalue of the matrix is

$$\lambda_{smallest} = \lambda_{shifted,largest} + 8 = -7 + 8 = 1.$$

Now, let us consider another matrix whose eigenvalues are -1, 2, 4 and 8. For this matrix, largest eigenvalue is 8 and the opposite extreme value is -1. We will solve for the largest eigenvalue $\lambda_{largest} = 8$ by using power method and shift the eigenvalues by 8. After shifting, the eigenvalues will be -9, -6, -4 and 0. Then, we will solve for the largest eigenvalue of the shifted matrix $\lambda_{shifted,largest} = -9$ by the power method and then $\lambda_{largest,negative} = \lambda_{shifted,largest} = -9 + 8 = -1$.

The above procedure is called the *shifted direct power method* [4]. The procedure is as follows:

(1) Solve for the largest (in absolute value) eigenvalue $\lambda_{largest}$ by direct power method,

(2) Shift the eigenvalues of the matrix $\mathbf{A}$ by $s = \lambda_{largest}$ to obtain the shifted matrix

(3) Solve for the largest (in absolute value) eigenvalue $\lambda_{shifted}$ of the shifted matrix $\mathbf{A}_{shifted}$ by the direct power method.

(4) Calculate the opposite extreme eigenvalue of the matrix $\mathbf{A}$ by $\lambda = \lambda_{shifted} + s$.

Example 3 The shifted direct power method for opposite extreme eigenvalues

Find the opposite extreme eigenvalue of matrix $\mathbf{A}$ given in Example 1 by shifting the eigenvalues by $s = \lambda_{largest} = 13.870584$. The original and the shifted matrices are

$$\mathbf{A} = \begin{pmatrix} 8 & -2 & -2 \\ -2 & 4 & -2 \\ -2 & -2 & 13 \end{pmatrix},$$

$$\mathbf{A}_{shifted} = \begin{pmatrix} (8 - 13.870584) & -2 & -2 \\ -2 & (4 - 13.870584) & -2 \\ -2 & -2 & (13 - 13.870584) \end{pmatrix}$$

$$= \begin{pmatrix} -5.870584 & -2.000000 & -2.000000 \\ -2.000000 & -9.870584 & -2.000000 \\ -2.000000 & -2.000000 & -0.870584 \end{pmatrix}.$$

Take the initial vector $\mathbf{x}^{(0)} = [1.0 \;\; 1.0 \;\; 1.0]$. Scale the second component to unity. Applying the power method to $\mathbf{A}_{shifted}$, we get

$$\mathbf{A}_{shifted}\mathbf{x}^{(0)} = \begin{pmatrix} -5.870584 & -2.000000 & -2.000000 \\ -2.000000 & -9.870584 & -2.000000 \\ -2.000000 & -2.000000 & -0.870584 \end{pmatrix} \begin{pmatrix} 1.0 \\ 1.0 \\ 1.0 \end{pmatrix}$$

$$= \begin{pmatrix} -9.870584 \\ -13.870584 \\ -4.870584 \end{pmatrix} = \mathbf{y}^{(1)}.$$

Scaling the unity component of $\mathbf{y}^{(1)}$ to unity, we get

$$\lambda^{(1)}_{shifted} = -13.870584 \quad \text{and} \quad \mathbf{x}^{(1)} = \begin{pmatrix} 0.711620 \\ 1.000000 \\ 0.351145 \end{pmatrix}.$$

The results of the first iteration were shown above, and the other iterations will not be presented here, but they will be given in the Table 1.3.

The largest eigenvalue (in absolute value) of $\mathbf{A}_{shifted}$ is $\lambda_{shifted,largest} = 11.361604$. Therefore, the opposite extreme eigenvalue of matrix $\mathbf{A}$ will be

$$\lambda = \lambda_{shifted,largest} + 13.870584 = -11.361604 + 13.870584 = 2.508980.$$

Table 1.3: Shifting Eigenvalues to Find the Opposite Extreme Eigenvalue

k	$\lambda_{shifted}$	x_1	x_2	x_3
0		1.000000	1.000000	1.000000
1	-13.870584	0.711620	1.000000	0.351145
2	-11.996114	0.573512	1.000000	0.310846
3	-11.639299	0.514510	1.000000	0.293629
4	-11.486864	0.488187	1.000000	0.285948
...	.....	.....	.....	.....
19	-11.361604	0.466027	1.000000	0.279482
20	-11.361604	0.466027	1.000000	0.279482

(2) *Shifting eigenvalues to find intermediate eigenvalues*

The intermediate eigenvalues λ_{inter} lie between the largest and the smallest eigenvalue. Let us consider a matrix whose eigenvalues are 1, 2, 4 and 8. We will solve for the largest eigenvalue $\lambda_{largest} = 8$ by direct power method and the smallest eigenvalue $\lambda_{smallest} = 1$ by inverse power method. We are left to determine two intermediate eigenvalues λ_{inter} =2 and 4. If λ_{inter} is guessed as $\lambda_{guess} = 5$ and the eigenvalues are shifted by $s = 5$, the eigenvalues of the shifted matrix will be -4, -3, -1 and 3. If we apply the inverse power method to the shifted matrix, $\lambda_{shifted} = -1$ will be found, then, $\lambda = \lambda_{shifted} + s = -1 + 5 = 4$.

The above method is called the *shifted inverse power method* [4]. The procedure is in the following:

(1) Guess a value for λ_{guess} for the intermediate eigenvalue of the shifted matrix

(2) Shift the eigenvalues by $s = \lambda_{guess}$ to obtain the shifted matrix $\mathbf{A}_{shifted}$

(3) Solve for the eigenvalue $\lambda_{shifted,inverse}$ of the inverse shifted matrix $\mathbf{A}^{-1}_{shifted}$ by the inverse power method applied to matrix $\mathbf{A}_{shifted}$

(4) Solve for $\lambda_{shifted} = 1/\lambda_{shifted,inverse}$

(5) Solve for the intermediate eigenvalue $\lambda_{inter} = \lambda_{shifted} + s$.

Example 4 The shifted inverse power method for intermediate eigenvalues

We will try to find an intermediate eigenvalue of $\mathbf{A}$ by guessing its value for $\lambda_{guess} = 10$. The corresponding shifted matrix $\mathbf{A}_{shifted}$ is

$$\mathbf{A}_{shifted} = (\mathbf{A} - \lambda_{guess}\mathbf{I}) = \begin{pmatrix} (8-10.0) & -2 & -2 \\ -2 & (4-10.0) & -2 \\ -2 & -2 & (13-10.0) \end{pmatrix}$$

$$= \begin{pmatrix} -2.0 & -2.0 & -2.0 \\ -2.0 & -6.0 & -2.0 \\ -2.0 & -2.0 & 3.0 \end{pmatrix}.$$

By Doolittle LU method, $\mathbf{L}$ and $\mathbf{U}$ is found as

$$\mathbf{L} = \begin{pmatrix} 1.0 & 0.0 & 0.0 \\ 1.0 & 1.0 & 0.0 \\ 1.0 & 0.0 & 1.0 \end{pmatrix} \quad \text{and} \quad \mathbf{U} = \begin{pmatrix} -2.0 & -2.0 & -2.0 \\ 0.0 & -4.0 & 0.0 \\ 0.0 & 0.0 & 5.0 \end{pmatrix}.$$

Take the initial vector $\mathbf{x}^{(0)} = [1.0 \quad 1.0 \quad 1.0]$. Scale the first component to unity. Solving for $\mathbf{x}'$ by forward substitution using $\mathbf{L}\mathbf{x}' = \mathbf{x}^{(0)}$ we get

$$\begin{pmatrix} 1.0 & 0.0 & 0.0 \\ 1.0 & 1.0 & 0.0 \\ 1.0 & 0.0 & 1.0 \end{pmatrix} \begin{pmatrix} x_1' \\ x_2' \\ x_3' \end{pmatrix} = \begin{pmatrix} 1.0 \\ 1.0 \\ 1.0 \end{pmatrix},$$

which yields

$$\begin{aligned} x_1' &= 1.0, \\ x_2' &= 1.0 - 1.0(1.0) = 0.0, \\ x_3' &= 1.0 - 1.0(1.0) - 1.0(0.0) = 0.0. \end{aligned}$$

Now, we will solve for $\mathbf{y}^{(1)}$ by backward substitution using $\mathbf{U}\mathbf{y}^{(1)} = \mathbf{x}'$

$$\begin{pmatrix} -2.0 & -2.0 & -2.0 \\ 0.0 & -4.0 & 0.0 \\ 0.0 & 0.0 & 5.0 \end{pmatrix} \begin{pmatrix} y_1^{(1)} \\ y_2^{(1)} \\ y_3^{(1)} \end{pmatrix} = \begin{pmatrix} 1.0 \\ 0.0 \\ 0.0 \end{pmatrix},$$

which will give the result

$$\begin{aligned} y_1^{(1)} &= 0.0/(5.0) = 0.0, \\ y_2^{(1)} &= [0.0 - (0.0)(0.0)]/(-4.0) = 0.0, \\ y_3^{(1)} &= [1.0 - (-2.0)(0.0) - (-2.0)(0.0)]/(-2.0) = -0.50. \end{aligned}$$

Scaling the unity component of $\mathbf{y}^{(1)}$ to unity, we get

$$\mathbf{y}^{(1)} = \begin{pmatrix} 0.50 \\ 0.0 \\ 0.0 \end{pmatrix} \qquad \lambda_{shifted,inverse}^{(1)} = -0.50 \quad \text{and} \quad \mathbf{x}^{(1)} = \begin{pmatrix} 0.1 \\ 0.0 \\ 0.0 \end{pmatrix}.$$

The results of the first iteration were shown above, and the other iterations will be given in the Table 1.4.

The largest eigenvalue of $\mathbf{A}_{shifted}^{-1}$ is $\lambda_{shifted,inverse} = -0.724866$, in magnitude. Therefore, the corresponding eigenvalue of matrix $\mathbf{A}_{shifted}$ will be

$$\lambda_{shifted} = \frac{1}{\lambda_{shifted,inverse}} = \frac{1}{-0.724866} = -1.379566.$$

Thus the intermediate eigenvalue of matrix $\mathbf{A}$ is

$$\lambda_{inter} = \lambda_{shifted} + s = -1.379566 + 10.000000 = 8.620434.$$

and the corresponding eigenvector is

$$\mathbf{x} = [1.0 \quad -0.526465 \quad 0.216247]^T.$$

Table 1.4: Shifting Eigenvalues to Find Intermediate Eigenvalue

k	$\lambda_{shifted,inverse}$	x_1	x_2	x_3
0		1.000000	1.000000	1.000000
1	-0.500000	1.000000	0.000000	0.000000
2	-1.996114	1.000000	-0.454545	0.363636
3	-1.639299	1.000000	-0.493827	0.172840
4	-1.486864	1.000000	-0.527463	0.233653
...	.....	.....	.....	.....
14	-1.361604	1.000000	-0.526465	0.216248
15	-1.361604	1.000000	-0.526465	0.216247

(3) *Shifting eigenvalues to accelerate convergence*

The shifting eigenvalue method can be used for accelerating the convergence of power method for a slowly converging problem. When an estimate λ_{est} of an eigenvalue of matrix $\mathbf{A}$ is known, from several initial iterations of the direct power method, the eigenvalues can be shifted by the estimate value in order that the shifted matrix has an eigenvalue near zero. This eigenvalue can be found by inverse power method later. The procedure is as follows [4]:

(1) Make an estimation λ_{est} of the eigenvalue λ after some iteration of direct power method.

(2) Shift the eigenvalues by $s = \lambda_{est}$ to get the shifted matrix $\mathbf{A}_{shifted}$.

(3) Solve for the eigenvalue $\lambda_{shifted,inverse}$ of the inverse matrix $\mathbf{A}_{shifted}^{-1}$ by the inverse power method applied to the shifted matrix $\mathbf{A}_{shifted}$. Let the first guess for $\mathbf{x}$ be the value of $\mathbf{x}$ corresponding to λ_{est}.

(4) Solve for $\lambda_{shifted} = \dfrac{1}{\lambda_{shifted,inverse}}$

(5) Evaluate $\lambda = \lambda_{shifted} + s$.

Example 5 The shifted inverse power method for accelerating convergence

Let us consider the Example 1, which converges so slowly. Convergence can be accelerated, say, after the fifth iteration. In iteration 5, $\lambda^{(5)} = 13.694744$ and $\mathbf{x}^{(5)} = [-0.192991 \quad -0.188895 \quad 1.000000]$. Thus, we will shift the matrix by s=13.694744.

$$\mathbf{A}_{shifted} = (\mathbf{A} - s\mathbf{I}) = \begin{pmatrix} -5.694744 & -2.000000 & -2.000000 \\ -2.000000 & -9.694744 & -2.000000 \\ -2.000000 & -2.000000 & -0.694744 \end{pmatrix},$$

$$\mathbf{L} = \begin{pmatrix} 1.000000 & 0.000000 & 0.000000 \\ 0.351201 & 1.000000 & 0.000000 \\ 0.351201 & 0.144300 & 1.000000 \end{pmatrix},$$

$$\mathbf{U} = \begin{pmatrix} -5.694744 & -2.000000 & -2.000000 \\ 0.000000 & -8.992342 & -1.297598 \\ 0.000000 & 0.000000 & 0.194902 \end{pmatrix}.$$

Now, let $\mathbf{x}^{(0)} = \mathbf{x}^{(5)}$, and continue scaling the third component of $\mathbf{x}$ to unity. Applying the inverse power method to matrix $\mathbf{A}_{shifted}$ yields the results presented in Table 1.5. The eigenvalue $\lambda_{shifted}$ of the shifted matrix $\mathbf{A}_{shifted}$ is

$$\lambda_{shifted} = \frac{1}{\lambda_{shifted,inverse}} = \frac{1}{5.686952} = 0.175841.$$

Thus, the eigenvalue λ of the original matrix $\mathbf{A}$ is

$$\lambda = \lambda_{shifted} + s = 0.175841 + 13.694744 = 13.870585.$$

As a summary, the largest eigenvalue is found by direct power method, the smallest one is found by both the inverse power method and by shifting the eigenvalues by the largest eigenvalue, and the intermediate eigenvalues is found by shifting the eigenvalues. The corresponding eigenvectors are also found during the processes.

Table 1.5: Shifting Eigenvalues to Accelerate Convergence

k	$\lambda_{shifted,inverse}$	x_1	x_2	x_3
0		-0.192991	-0.188895	1.000000
1	5.568216	-0.295286	-0.141881	1.000000
2	5.691139	-0.291674	-0.143554	1.000000
3	5.686807	-0.291799	-0.143496	1.000000
4	5.686957	-0.291794	-0.143498	1.000000
5	5.686952	-0.291794	-0.143498	1.000000
6	5.686952	-0.291794	-0.143498	1.000000

1.1.2 The Direct Method

In the previous sections, the solutions of linear eigenvalue problems are presented, but an equation of the form

$$\mathbf{A}\mathbf{x} = \mathbf{B}(\lambda)\mathbf{x}, \tag{1.36}$$

cannot be solved by the power method. Linear and nonlinear problems can be both solved by direct power method which involves finding the zeros of the characteristic method.

We have stated that for a linear equation of the form

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x}, \tag{1.37}$$

the characteristic equation can be found by

$$\det(\mathbf{A} - \lambda\mathbf{I}) = 0. \tag{1.38}$$

Expanding this equation for a large system may not be easy and may be time consuming.

For a nonlinear eigenproblem, the characteristic equation is obtained from

$$\det[\mathbf{A} - \mathbf{B}(\lambda)] = 0. \tag{1.39}$$

Expanding this equation gives a nonlinear function in terms of λ and the zeros of this function will be solved by Secant Method which will be given in details in Appendix F.

In order to apply the Secant Method. two initial approximations of λ will be used, λ_0 and λ_1, the corresponding values of the determinant will be calculated, and these results will be used to construct a linear relationship between λ and the value of the characteristic determinant. The solution of this linear relationship will be taken as the next approximation to λ, and the iterations will be used till the convergence [4].

The direct method determines only the eigenvalues. The corresponding eigenvectors must be determined by substituting the eigenvalues into the system of equations and solving for the corresponding eigenvectors, or by applying the inverse power method one time.

Example 6 The direct method for a linear eigenproblem

Let us find the largest eigenvalue of the matrix

$$\mathbf{A} = \begin{pmatrix} 8 & -2 & -2 \\ -2 & 4 & -2 \\ -2 & -2 & 13 \end{pmatrix}.$$

The characteristic determinant corresponding to the above matrix is

$$f(\lambda) = \det(\mathbf{A} - \lambda\mathbf{I}) = \begin{pmatrix} (8-\lambda) & -2 & -2 \\ -2 & (4-\lambda) & -2 \\ -2 & -2 & (13-\lambda) \end{pmatrix} = 0.$$

The above equation can be solved by Secant method. Let $\lambda_0 = 15.0$ and $\lambda_1 = 13.0$. Thus,

$$f(\lambda_0) = f(15.0) = \begin{vmatrix} (8-15.0) & -2 & -2 \\ -2 & (4-15.0) & -2 \\ -2 & -2 & (13-15.0) \end{vmatrix} = -90.0,$$

$$f(\lambda_1) = f(13.0) = \begin{vmatrix} (8-13.0) & -2 & -2 \\ -2 & (4-13.0) & -2 \\ -2 & -2 & (13-13.0) \end{vmatrix} = 40.0.$$

By the Secant method,

$$\frac{f(\lambda_1) - f(\lambda_0)}{\lambda_1 - \lambda_0} = slope = \frac{f(\lambda_2) - f(\lambda_1)}{\lambda_2 - \lambda_1}, \tag{1.40}$$

from where $f(\lambda_2) = 0$ is the desired solution. So, the slope is

$$slope = \frac{40.0 - (-90.0)}{13.0 - 15.0} = -65.0.$$

Now, since we know the slope by (1.40) we can find λ_2 by replacing the value of $f(\lambda_2) = 0.0$. Therefore,

$$\lambda_2 = \lambda_1 - \frac{f(\lambda_1)}{slope} = 13.0 - \frac{40}{(-65.0)} = 13.615385.$$

The other steps will not be presented here, they will be given in Table 1.6. The solution is $\lambda = 13.870585$.

Table 1.6: The Direct Method for a Linear Eigenproblem

k	λ_k, deg	$f(\lambda_k)$	$(Slope)_k$
0	15.000000	-90.000000	
1	13.000000	40.000000	-65.000000
2	13.615385	14.157487	-41.994083
3	13.952515	-4.999194	-56.822743
4	13.864536	0.360200	-60.916914
5	13.870449	0.008098	-59.547441
6	13.870585	-0.000014	-59.647887
7	13.870585	0.000000	

Example 7 The direct method for a nonlinear eigenproblem

Consider the nonlinear eigenproblem:

$$\begin{aligned} x_1 + 0.4x_2 &= \sin(\lambda)x_1 \\ 0.2x_1 + x_2 &= \cos(\lambda)x_2. \end{aligned}$$

The characteristic determinant of the above system is

$$f(\lambda) = \det[\mathbf{A} - \mathbf{B}(\lambda)] = \begin{vmatrix} (1-\sin(\lambda)) & 0.4 \\ 0.2 & (1-\cos(\lambda)) \end{vmatrix} = 0.$$

Now we will solve the last equation by Secant method. Let $\lambda_0 = 50.0$ deg and $\lambda_1 = 55.0$ deg. Therefore,

$$f(\lambda_0) = \begin{vmatrix} (1-\sin(50)) & 0.4 \\ 0.2 & (1-\cos(50)) \end{vmatrix} = \begin{vmatrix} 0.233956 & 0.4 \\ 0.2 & 0.357212 \end{vmatrix} = 0.003572,$$

$$f(\lambda_1) = \begin{vmatrix} (1-\sin(55)) & 0.4 \\ 0.2 & (1-\cos(55)) \end{vmatrix} = \begin{vmatrix} 0.180848 & 0.4 \\ 0.2 & 0.426424 \end{vmatrix} = 0.002882.$$

Writing the linear relationships among λ and $f(\lambda)$, we get

$$\frac{f(\lambda_1) - f(\lambda_0)}{\lambda_1 - \lambda_0} = slope = \frac{f(\lambda_2) - f(\lambda_1)}{\lambda_2 - \lambda_1}, \tag{1.41}$$

from where $f(\lambda_2) = 0$ is the desired solution. So, the slope is

$$slope = \frac{(-0.002882) - (-0.003572)}{55.0 - 50.0} = -0.001292.$$

Now, since we know the slope we can find λ_2 from (1.41) by replacing the value of $f(\lambda_2) = 0.0$. Therefore,

$$\lambda_2 = \lambda 1 - \frac{f(\lambda_1)}{slope} = 55.0 - \frac{(-0.002882)}{(-0.001292)} = 52.767276.$$

The other steps will not be presented here, they will be given in Table 1.7. The resulting value for $\lambda = 53.131096$ deg and the tolerance number between iterations is 0.000001.

Table 1.7: The Direct Method for a Nonlinear Eigenproblem

k	λ_k, deg	$f(\lambda_k)$	$(Slope)_k$
0	50.0	0.003572	
1	55.0	-0.002882	-0.001292
2	52.767276	0.000496	-0.001513
3	53.095189	0.000049	-0.001365
4	53.131096	-0.000001	

1.1.3 The QR Method

The direct method and power method find individual eigenvalues; on the other hand, QR method finds all the eigenvalues simultaneously [4].

Triangular matrices have their eigenvalues on the diagonal of the matrix. Consider the upper triangular matrix **U**:

$$\begin{pmatrix} u_{11} & u_{12} & u_{13} & \cdots & u_{1n} \\ 0 & u_{22} & u_{23} & \cdots & u_{2n} \\ 0 & 0 & u_{33} & \cdots & u_{3n} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & u_{nn} \end{pmatrix}$$

The eigenproblem, $(\mathbf{U} - \lambda\mathbf{I})$, is given by

$$(\mathbf{U} - \lambda\mathbf{I}) = \begin{pmatrix} (u_{11} - \lambda) & u_{12} & u_{13} & \cdots & u_{1n} \\ 0 & (u_{22} - \lambda) & u_{23} & \cdots & u_{2n} \\ 0 & 0 & (u_{33} - \lambda) & \cdots & u_{3n} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & (u_{nn} - \lambda) \end{pmatrix} \tag{1.42}$$

The characteristic polynomial, $|\mathbf{U} - \lambda\mathbf{I}|$, is

$$(u_{11} - \lambda)(u_{22} - \lambda)(u_{33} - \lambda)\cdots(u_{nn} - \lambda) = 0. \tag{1.43}$$

The roots of the above equation is

$$\lambda_i = u_{ii}, \qquad \text{i} = 1, 2, \ldots, \text{n}. \tag{1.44}$$

The QR method uses the similarity transformations to transform the matrix $\mathbf{A}$ into triangular form. Now, let us give the definition of the concept *similarity transformation.*

Definition 1 An $n \times n$ matrix $\mathbf{A}$ is called *similar* to an $n \times n$ matrix $\mathbf{B}$ if

$$\mathbf{A} = \mathbf{P}^{-1}\mathbf{B}\mathbf{P},$$

for some nonsingular $n \times n$ matrix $\mathbf{P}$. This transformation which gives $\mathbf{A}$ from $\mathbf{B}$ is called *similarity transformation.*

THEOREM 4 If $\mathbf{A}$ is similar to $\mathbf{B}$, then $\mathbf{A}$ and $\mathbf{B}$ have the same eigenvalues. Furthermore, if $\mathbf{x}$ is an eigenvector of $\mathbf{B}$, then $\mathbf{y} = \mathbf{P}^{-1}\mathbf{x}$ is an eigenvector of $\mathbf{A}$ corresponding to the same eigenvalue.

(You can see the proof of this theorem in Appendix H.)

The Gram-Schmidt Orthogonalization Process (GSOP) (see in Appendix G) starts with matrix $\mathbf{A}$, whose columns are written in the form $[a_1 a_2 \cdots a_n]$ and construct the matrix $\mathbf{Q}$, whose columns defines a set of orthonormal vectors $q_1, q_2, \ldots, q_n$, i.e, the vectors are orthogonal to each other and the norm of the vectors are all equal to 1. The matrix that connects the matrix $\mathbf{A}$ to $\mathbf{Q}$ is the upper triangular matrix $\mathbf{R}$ whose elements are the vector products $r_{ij} = q_i^T a_j$, $(i, j = 1, 2, \ldots, n)$

The result is the QR factorization

$$\mathbf{A} = \mathbf{Q}\mathbf{R}. \tag{1.45}$$

The QR process starts with the GSOP. This process is then reversed to give

$$\mathbf{A}' = \mathbf{R}\mathbf{Q}. \tag{1.46}$$

Matrices $\mathbf{A}$ and $\mathbf{A}'$ can be shown to be similar as follows.

$$\begin{aligned} \mathbf{A} = \mathbf{QR} &\Rightarrow \mathbf{Q}^{-1}\mathbf{A} = \mathbf{Q}^{-1}\mathbf{QR} = \mathbf{IR} = \mathbf{R} \\ &\Rightarrow \mathbf{Q}^{-1}\mathbf{AQ} = \mathbf{RQ} = \mathbf{A}'. \end{aligned} \tag{1.47}$$

The above equation shows that $\mathbf{A}$ and $\mathbf{A}'$ are similar to each other by Definition 1, and thus they have the same eigenvalues by Theorem 3.

The step that will be followed by GSOP is as follows:

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix} = [a_1 \quad a_2 \quad \cdots \quad a_n]. \tag{1.48}$$

Assuming that the column vectors $a_i \quad (i = 1, 2, \ldots, n)$ are linearly independent; therefore, they span the n-dimensional vector space. Thus, any vector can be expressed as a linear combination of these vectors. An orthonormal set of column vectors $q_i \quad (i = 1, 2, \ldots, n)$ can be determined by these column vectors $a_i \quad (i = 1, 2, \ldots, n)$ by the following steps.

Choose q_1 to have the same direction of a_1. Then normalize a_1 to obtain q_1.

$$q_1 = \frac{a_1}{\|a_1\|}, \tag{1.49}$$

where $\|a_1\|$ denotes the magnitude of a_1:

$$\|a_1\| = (a_{11}^2 + a_{21}^2 + \cdots + a_{n1}^2)^{1/2}. \tag{1.50}$$

To determine q_2, first subtract the component of a_2 in the direction of q_1 to determine vector a_2', which is normal to q_1. Thus

$$a_2' = a_2 - (q_1^T a_2) q_1. \tag{1.51}$$

Choose q_2 to have the direction of a_2'. Then normalize a_2' to obtain q_2.

$$q_2 = \frac{a_2'}{\|a_2'\|}. \tag{1.52}$$

To determine q_3, first subtract the components of a_3 in the directions of q_1 and q_2. Thus

$$a_3' = a_3 - (q_1^T a_3) q_1 - (q_2^T a_3) q_2. \tag{1.53}$$

Choose q_3 to have direction of a_3'. Then normalize a_3' to obtain q_3.

$$q_3 = \frac{a_3'}{\|a_3'\|}. \tag{1.54}$$

This process continues until a complete set of n orthonormal unit vectors is obtained. The general expression for a_i' is

$$a_i' = a_i - \sum_{k=1}^{i-1} (q_k^T a_i) q_k \qquad (i = 1, 2, \ldots, n), \tag{1.55}$$

and the general expression for q_i is

$$q_i = \frac{a_i'}{\|a_i'\|} \qquad (i = 1, 2, \ldots, n). \tag{1.56}$$

The matrix $\mathbf{Q}$ is composed of the column vectors $q_i \quad (i = 1, 2, \ldots, n)$. Thus,

$$\mathbf{Q} = [q_1 \quad q_2 \quad \cdots \quad q_n]. \tag{1.57}$$

The upper triangular matrix $\mathbf{R}$ is constructed from the elements computed in the evaluation of $\mathbf{Q}$. The diagonal elements of $\mathbf{R}$ are the magnitudes of the a_i' vectors:

$$r_{ii} = \|a_i'\| \qquad (i = 1, 2, \ldots, n). \tag{1.58}$$

The elements not on the diagonal of $\mathbf{R}$ are the components of a_i vectors which are subtracted from the a_i vectors in the evaluation of a_i'. Then,

$$r_{ij} = q_i^T a_j \qquad (i = 1, 2, \ldots, n, \quad j = i+1, i+2, \ldots, n). \tag{1.59}$$

So, the matrix $\mathbf{R}$ is of the form

$$\mathbf{R} = \begin{pmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ 0 & r_{22} & \cdots & r_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & r_{nn} \end{pmatrix}. \tag{1.60}$$

The first step in the QR process is to set $\mathbf{A}^{(0)} = \mathbf{A}$ and factor $\mathbf{A}^{(0)}$ by GSOP into $\mathbf{Q}^{(0)}$ and $\mathbf{R}^{(0)}$. The next step is to reverse the factors $\mathbf{Q}^{(0)}$ and $\mathbf{R}^{(0)}$ to obtain

$$\mathbf{A}^{(1)} = \mathbf{R}^{(0)} \mathbf{Q}^{(0)}. \tag{1.61}$$

$\mathbf{A}^{(1)}$ is similar to $\mathbf{A}$, so the eigenvalues are preserved. $\mathbf{A}^{(1)}$ is factored by GSOP to obtain $\mathbf{Q}^{(1)}$ and $\mathbf{R}^{(1)}$, and the factors are reversed to obtain $\mathbf{A}^{(2)}$.Thus,

$$\mathbf{A}^{(2)} = \mathbf{R}^{(1)}\mathbf{Q}^{(1)}. \tag{1.62}$$

The process is continued to determine $\mathbf{A}^{(3)}, \mathbf{A}^{(4)}, \ldots, \mathbf{A}^{(n)}$. When $\mathbf{A}^{(n)}$ approaches triangular form, within some tolerance, the eigenvalues of $\mathbf{A}$ are the diagonal elements, and the eigenvectors can be found by the inverse shifted power method. The process is as follows:

$$\begin{aligned} \mathbf{A}^{(k)} &= \mathbf{Q}^{(k)}\mathbf{R}^{(k)} \\ \mathbf{A}^{(k+1)} &= \mathbf{R}^{(k)}\mathbf{Q}^{(k)}. \end{aligned} \tag{1.63}$$

Example 8 The basic QR method

Let us apply the QR method to the matrix $\mathbf{A}$ given by $\mathbf{A} = \begin{pmatrix} 8 & -2 & -2 \\ -2 & 4 & -2 \\ -2 & -2 & 13 \end{pmatrix}$.

The column vectors associated with matrix $\mathbf{A} = [a_1 \quad a_2 \quad a_3]$, are

$$a_1 = \begin{pmatrix} 8 \\ -2 \\ -2 \end{pmatrix} \qquad a_2 = \begin{pmatrix} -2 \\ 4 \\ -2 \end{pmatrix} \qquad a_3 = \begin{pmatrix} -2 \\ -2 \\ 13 \end{pmatrix}.$$

Let q_1 have the direction of a_1, and we will divide a_1 by its magnitude to find q_1. Therefore,

$$\|a_1\| = [8^2 + (-2)^2 + (-2)^2]^{1/2} = 8.485281,$$

and

$$q_1 = [0.942809 \quad -0.235702 \quad -0.235702]^T.$$

Now, for q_2. we will subtract the component of a_2 in the direction of q_1:

$$\begin{aligned} a_2' &= a_2 - (q_1^T a_2)q_1 \\ &= \begin{pmatrix} -2 \\ 4 \\ -2 \end{pmatrix} - [0.942809 \quad -0.235702 \quad -0.235702] \begin{pmatrix} -2 \\ 4 \\ -2 \end{pmatrix} \begin{pmatrix} 0.942809 \\ -0.235702 \\ -0.235702 \end{pmatrix}. \end{aligned}$$

After calculation

$$q_1^T a_2 = -2.357023, \qquad a_2' = \begin{pmatrix} -2 - (-2.222222) \\ 4 - (0.555555) \\ -2 - (0.555555) \end{pmatrix} = \begin{pmatrix} 0.222222 \\ 3.444444 \\ -2.555557 \end{pmatrix}.$$

The magnitude of a_2' is $\|a_2'\| = 4.294700$. Thus,

$$q_2 = \frac{a_2'}{\|a_2'\|} = [0.051743 \quad 0.802022 \quad -0.595049]^T.$$

Furthermore, let us solve q_3. Firstly, we will subtract the components of a_3 in the direction of q_1 and q_2.

$$\begin{aligned} a_3' &= a_3 - (q_1^T a_3)q_1 - (q_2^T a_3)q_2 \\ &= \begin{pmatrix} -2 \\ -2 \\ 13 \end{pmatrix} - [0.942809 \quad -0.235702 \quad -0.235702] \begin{pmatrix} -2 \\ -2 \\ 13 \end{pmatrix} \begin{pmatrix} 0.942809 \\ -0.235702 \\ -0.235702 \end{pmatrix} \\ &- [0.051743 \quad 0.802022 \quad -0.595049] \begin{pmatrix} -2 \\ -2 \\ 13 \end{pmatrix} \begin{pmatrix} 0.051743 \\ 0.802022 \\ -0.595049 \end{pmatrix}. \end{aligned}$$

Performing the calculations, we get

$$q_1^T a_3 = -4.478343, \quad q_2^T a_3 = -9.443165$$

$$\begin{aligned} a_3' &= \begin{pmatrix} -2 - (-4.222222) - (-0.488618) \\ -2 - (1.055554) - (-7.573626) \end{pmatrix} \Big(13 - (1.055554) - (5.619146) \Big) \\ &= \begin{pmatrix} 2.710840 \\ 4.518072 \\ 6.325300 \end{pmatrix}, \end{aligned}$$

and $\|a_3'\| = 8.232319$. Thus

$$q_3 = \frac{a_3'}{\|a_3'\|} = [0.329293 \quad 0.548821 \quad 0.768350]^T.$$

So, matrix $\mathbf{Q}^{(0)} = [q_1 \quad q_2 \quad q_3]$ is found as

$$\mathbf{Q}^{(0)} = \begin{pmatrix} 0.942809 & 0.051743 & 0.329293 \\ -0.235702 & 0.802022 & 0.548821 \\ -0.235702 & -0.595049 & 0.768350 \end{pmatrix}.$$

Matrix $\mathbf{R}$ is constructed from the elements computed in the calculation of $\mathbf{Q}^{(0)}$. Thus,

$$\begin{aligned} r_{11} &= \|a_1'\| = 8.485281 \quad r_{12} = q_1^T a_2 = -2.357023 \quad r_{13} = q_1^T a_3 = -4.478343 \\ r_{22} &= \|a_2'\| = 4.294700 \quad r_{23} = q_2^T a_3 = -9.443165 \quad r_{33} = \|a_3'\| = 8.232319. \end{aligned}$$

In matrix notation, $\mathbf{R}^{(0)}$ is as follows:

$$\mathbf{R} = \begin{pmatrix} 8.485281 & -2.357023 & -4.478343 \\ 0 & 4.294700 & -9.443165 \\ 0 & 0 & 8.232819 \end{pmatrix}.$$

Next step is to evaluate $\mathbf{A}^{(1)} = \mathbf{R}^{(0)}\mathbf{Q}^{(0)}$. Thus,

$$\mathbf{A}^{(1)} = \begin{pmatrix} 9.611111 & 1.213505 & -1.940376 \\ 1.213505 & 9.063588 & -4.898631 \\ -1.940376 & -4.898631 & 6.325301 \end{pmatrix}.$$

The diagonal elements of $\mathbf{A}^{(1)}$ are the first approximations of eigenvalues of $\mathbf{A}$. The successive steps are given in Table 1.8. The final values of matrices $\mathbf{Q}, \mathbf{R}$ and $\mathbf{A}$ are given below:

$$\mathbf{Q}^{(19)} = \begin{pmatrix} 1.000000 & -0.000141 & 0.000000 \\ 0.000141 & 1.000000 & 0.000000 \\ 0.000000 & 0.000000 & 1.000000 \end{pmatrix},$$

$$\mathbf{R}^{(19)} = \begin{pmatrix} 13.870585 & 0.003171 & 0.000000 \\ 0.000000 & 8.260434 & 0.000000 \\ 0.000000 & 0.000000 & 2.508981 \end{pmatrix},$$

$$\mathbf{A}^{(20)} = \begin{pmatrix} 13.870585 & 0.001215 & 0.000000 \\ 0.001215 & 8.620434 & 0.000000 \\ 0.000000 & 0.000000 & 2.508981 \end{pmatrix}.$$

So,

$$\lambda_1 = 13.870585 \quad \lambda_2 = 8.620434 \quad \lambda_3 = 2.508981$$

Table 1.8: The Basic QR Method

k	λ_1	λ_2	λ_3
1	9.611111	9.063588	6.325301
2	10.743882	11.543169	2.712949
3	11.974170	10.508712	2.517118
4	12.929724	9.560916	2.509360
...	.....	.....	.....
19	13.870584	8.620435	2.508981
20	13.870584	8.620434	2.508981

1.1.4 Eigenvectors

In direct method and QR method, we said that we only got the eigenvalues by these methods, however, we could find the corresponding eigenvectors by applying inverse power method one time [4].

Example 9 Eigenvectors

Let us apply this technique to evaluate the eigenvector $\mathbf{x}_1$ corresponding to the eigenvalue $\lambda_1 = 13.870584$ of the matrix

$\mathbf{A} = \begin{pmatrix} 8 & -2 & -2 \\ -2 & 4 & -2 \\ -2 & -2 & 13 \end{pmatrix}$. We will firstly shift the matrix $\mathbf{A}$ by $\lambda = 13.870584$.

$$\mathbf{A}_{shifted} = (\mathbf{A} - s\mathbf{I}) = \begin{pmatrix} -5.870584 & -2.000000 & -2.000000 \\ -2.000000 & -9.870584 & -2.000000 \\ -2.000000 & -2.000000 & -0.870584 \end{pmatrix}.$$

Applying the Doolittle LU method to $\mathbf{A}_{shifted}$ yields $\mathbf{L}$ and $\mathbf{U}$:

$$\mathbf{L} = \begin{pmatrix} 1.000000 & 0.000000 & 0.000000 \\ 0.340682 & 1.000000 & 0.000000 \\ 0.340682 & 0.143498 & 1.000000 \end{pmatrix},$$

$$\mathbf{U} = \begin{pmatrix} -5.870584 & -2.000000 & -2.000000 \\ 0.000000 & -9.189221 & -1.318637 \\ 0.000000 & 0.000000 & 0.000001 \end{pmatrix}.$$

Let us take the initial vector for $\mathbf{x}^{(0)} = [1.0 \quad 1.0 \quad 1.0]$. We will solve for $\mathbf{x}'$ by forward substitution using $\mathbf{L}\mathbf{x}' = \mathbf{x}$.

$$\begin{pmatrix} 1.000000 & 0.000000 & 0.000000 \\ 0.340682 & 1.000000 & 0.000000 \\ 0.340682 & 0.143498 & 1.000000 \end{pmatrix} \begin{pmatrix} x'_1 \\ x'_2 \\ x'_3 \end{pmatrix} = \begin{pmatrix} 1.0 \\ 1.0 \\ 1.0 \end{pmatrix}$$

$$\Rightarrow x'_1 = 1.000000, \qquad x'_2 = 0.659318, \qquad x'_3 = 0.564707.$$

Now, we will solve for $\mathbf{y}$ by backward substitution using $\mathbf{U}\mathbf{y} = \mathbf{x}'$.

$$\mathbf{U} = \begin{pmatrix} -5.870584 & -2.000000 & -2.000000 \\ 0.000000 & -9.189221 & -1.318637 \\ 0.000000 & 0.000000 & 0.000001 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} 1.000000 \\ 0.659318 \\ 0.564707 \end{pmatrix}.$$

The solution with scaling the third component to unity is

$$\mathbf{y} = \begin{pmatrix} -0.132662 \times 10^6 \\ -0.652404 \times 10^6 \\ 0.454642 \times 10^6 \end{pmatrix} \rightarrow 0.454642 \times 10^6 \begin{pmatrix} -0.291794 \\ -0.143498 \\ 1.000000 \end{pmatrix}.$$

Thus, $\mathbf{x} = [-0.291794 \quad -0.143498 \quad 1.000000]^T$.

1.2 Gerschgorin Circles

In the previous sections, we tried to determine the eigenvalues and eigenvectors of a given matrix, but in this section, we will give the location of the eigenvalues in the space. We will use Theorem 5 in order to determine the situation of the eigenvalues [5].

THEOREM 5 The set of eigenvalues of an $n \times n$ matrix $\mathbf{A}$ is contained in the union of the following n disks, D_i, in the real plane (in fact in complex plane)

$$D_i = \left\{ \lambda \in \mathbb{R} : |\lambda - a_{ii}| \leq \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}| \right\}.$$

(You can see the proof of this theorem in Appendix H.)

Example 10 Gerschgorin Circles

Find the location of the eigenvalues of the matrix given as

$$\mathbf{A} = \begin{pmatrix} 8 & -2 & -2 \\ -2 & 4 & -2 \\ -2 & -2 & 13 \end{pmatrix}.$$

$$D_1 = \{\lambda : |\lambda - 8| \leq (|-2| + |-2|) = 4\},$$

$$D_2 = \{\lambda : |\lambda - 4| \leq (|-2| + |-2|) = 4\},$$

$$D_3 = \{\lambda : |\lambda - 13| \leq (|-2| + |-2|) = 4\},$$

from $D_1 : -4 \leq \lambda - 8 \leq 4 \Rightarrow 4 \leq \lambda \leq 12$,
from $D_2 : -4 \leq \lambda - 4 \leq 4 \Rightarrow 0 \leq \lambda \leq 8$,

from $D_3 : -4 \leq \lambda - 13 \leq 4 \Rightarrow 9 \leq \lambda \leq 22$.

In fact, we know that $\lambda_1 = 13.870585, \lambda_2 = 8.620434, \lambda_3 = 2.508981$. So,

$$\lambda_1 \in D_3, \quad \lambda_2 \in D_1, \quad \lambda_3 \in D_2.$$

In Figure 1.1 we can see the location easily.

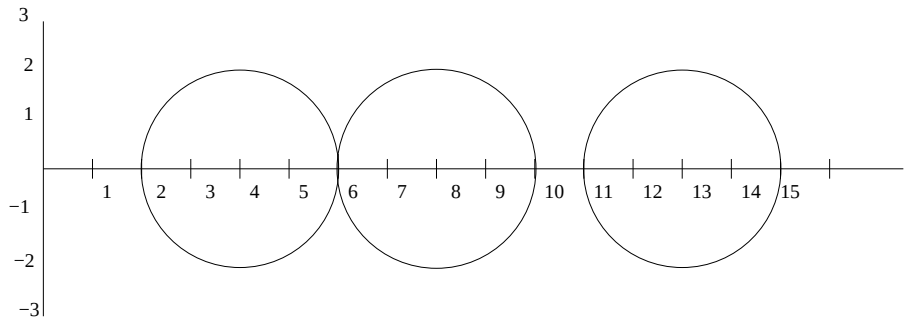

Figure 1.1: Gerschgorin disks for Example 9

There is also another theorem which gives bound to the eigenvalues of a Hermitian matrix, the matrix whose conjugate transpose is itself, that is $\overline{\mathbf{A}^T} = \mathbf{A}^* = \mathbf{A}$. [3]

THEOREM 6 (Cauchy Interlace Theorem) Let $\mathbf{A}$ be an $n \times n$ Hermitian matrix with eigenvalues $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$, and let $\mathbf{H}$ be $m \times m$ principal submatrix of $\mathbf{A}$ (obtained by deleting n-m rows and the corresponding columns from $\mathbf{A}$), with eigenvalues $\mu_1 \leq \mu_2 \leq \cdots \leq \mu_m$. Then, for each $i = 1, 2, \ldots, m$ we have

$$\lambda_i \leq \mu_i \leq \lambda_{i+n-m}.$$

(Proof of this theorem will not be given in this study.)

SECTION 2

HIGH DIMENSIONAL MODEL REPRESENTATION

An enormous knowledge accumulation exists for the solution of matrix eigenvalue equations. The existing routines work well for low or moderate matrix dimensions and there is a strong basis for convergence and error analysis which is mostly based on the norms or similar entities. If the dimension is finite, there are no additive convergence problems for the sums regardless of the magnitude of the dimension. When the matrices under consideration become infinite dimensional, a matter of convergence appears in the sums. Even in the case of finite dimensional systems the finite sums may cause error accumulations when the dimension grows up unboundedly. Hence, the dimensionality plays an important role when the dimension of the matrix under consideration grows up to infinity. This urges scientists try to develop various approximation techniques working well at the infinite dimension limit and construct within an additive expansion such that the first contributive term is a constant followed by N number of univariate terms (N is the number of independent variables), $N(N-1)/2$ bivariate terms and so on. The first attempt for the applications toward this direction was made by Sobol [17]. The purpose of the expansion was some statistical applications. This expansion formula has been revisited, revised, generalized and applied to various problems by Rabitz ([10], [11], [13], [14], [16]). The method has been called High Dimensional Model Representation (HDMR) [2]. It can be given in the general form for a given multivariate function $f(x_1, ..., x_N)$ as

$$f(x_1, \ldots, x_N) = f_0 + \sum_{i=1}^{N} f_i(x_i) + \sum_{\substack{i_1, i_2 = 1 \\ i_1 < i_2}}^{N} f_{i_1 i_2}(x_{i_1}, x_{i_2}) + \cdots + f_{123\ldots N}(x_1, \ldots, x_N), \tag{2.1}$$

where f_0, $f_{i_1}(x_{i_1})$ $(1 \leq i \leq N)$, $f_{i_1 i_2}(x_{i_1}, x_{i_2})$ $(1 \leq i_1 < i_2 \leq N)$ stand for a constant term, totally N univariate functions and totally $N(N-1)/2$ bivariate functions of different independent variables etc., respectively. The highvariate terms containing more than two independent variables are not shown explicitly.

The last term, $f_{123...N}(x_1, ..., x_N)$ is N–variate but of course different than the original function $f(x_1, ..., x_N)$.

These terms are the components of an orthonormal expansion. It is assumed that all these functions are orthogonal to each other and the orthogonality condition is defined by an inner product. Moreover, it is assumed that both the squares of the original function $f(x_1, \ldots, x_N)$ and the component functions are integrable. Square integrals and the inner product is defined over a predetermined interval and for every independent variable a weight function in terms of that independent variable is used. So, for two functions $u(x_1, \ldots, x_N)$ and $v(x_1, \ldots, x_N)$, whose squares are also integrable, inner product is defined as

$$(u, v) \equiv \int_{a_1}^{b_1} dx_1 W_1(x_1) \cdots \int_{a_N}^{b_N} dx_N W_N(x_N) u(x_1, \ldots, x_N) v(x_1, \ldots, x_N). \quad (2.2)$$

The integral of component functions with respect to any one of their arguments under a weight and over an appropriately chosen interval vanishes. That is,

$$\int_{a_{i_s}}^{b_{i_s}} dx_{i_s} W_{i_s}(x_{i_s}) f_{i_1 i_2 \ldots i_m}(x_1, ..., x_m) = 0, \qquad 1 \leq s \leq m \quad (2.3)$$

where the univariate weight functions are normalized as follows

$$\int_{a_{i_s}}^{b_{i_s}} dx_{i_s} W_{i_s}(x_{i_s}) = 1, \qquad 1 \leq i_s \leq N \quad (2.4)$$

This means that we have used a hyperprism and an overall weight function constructed as a product of these univariate weight functions as follows

$$W(x_1, ..., x_N) \equiv \prod_{i=1}^{N} W_i(x_i). \quad (2.5)$$

The choice of hyperprism as the geometry is not strictly required. One could equivalently select some other separable orthogonal geometries. The overall weight function must always be a product of univariate functions each of which depends on a separate independent variable.

In order to find the orthogonal components of $f(x_1, \ldots, x_N)$ some projection operators are used. So, for any $g(x_1, \ldots, x_N)$, whose square is also integrable, if

$$\mathcal{P}_i g(x_1, \ldots, x_N) \equiv \int_{a_i}^{b_i} dx_i W_i(x_i) g(x_1, \ldots, x_N), \quad (2.6)$$

and

$$\mathcal{I} g(x_1, \ldots, x_N) \equiv g(x_1, \ldots, x_N), \quad (2.7)$$

then

$$f(x_1,\ldots,x_N)) = \left\{\prod_{i=1}^{N}(\mathcal{P}_i + [\mathcal{I}-\mathcal{P}_i])\right\} f(x_1,\ldots,x_N) \tag{2.8}$$

can be written. The operator product in (2.8) is rewritten as the following:

$$\prod_{i=1}^{N}(\mathcal{P}_i + [\mathcal{I}-\mathcal{P}_i]) = \prod_{i=1}^{N}\mathcal{P}_i + \sum_{i_1=1}^{N}\left(\prod_{i=1}^{i_1-1}\mathcal{P}_i\right)[\mathcal{I}-\mathcal{P}_{i_1}]\left(\prod_{i=i_1+1}^{N}\mathcal{P}_i\right)$$

$$+\sum_{\substack{i_1,i_2=1\\ i_1<i_2}}^{N}\left(\prod_{i=1}^{i_1-1}\mathcal{P}_i\right)[\mathcal{I}-\mathcal{P}_{i_1}]\left(\prod_{i=i_1+1}^{i_2-1}\mathcal{P}_i\right)[\mathcal{I}-\mathcal{P}_{i_2}]\left(\prod_{i=i_2+1}^{N}\mathcal{P}_i\right)+\cdots \tag{2.9}$$

In this expansion, it can be seen easily that the first term is the constant term, the second one is the sum of the functions in terms of just one independent variable, the third one is the sum of the functions in terms of just two independent variables, and so on. Moreover, any term in (2.9) is orthogonal to other terms. Since, any two terms in (2.9) will not contain the same independent variables, otherwise they will be same term, then integral with respect to that different independent variable will make the whole integral be equal to zero. In addition, this type of decomposition to orthogonal components is unique, so, the following equalities can be written:

$$f_0 = \left\{\prod_{i=1}^{N}\mathcal{P}_i\right\} f(x_1,\ldots,x_N),$$

$$f_i(x_i) = \left\{\left(\prod_{i=1}^{i_1-1}\mathcal{P}_i\right)[\mathcal{I}-\mathcal{P}_{i_1}]\left(\prod_{i=i_1+1}^{N}\mathcal{P}_i\right)\right\} f(x_1,\ldots,x_N),$$

$$f_{i_1i_2}(x_{i_1},x_{i_2}) = \left\{\left(\prod_{i=1}^{i_1-1}\mathcal{P}_i\right)[\mathcal{I}-\mathcal{P}_{i_1}]\left(\prod_{i=i_1+1}^{i_2-1}\mathcal{P}_i\right)\bullet\right.$$
$$\left.\bullet\,[\mathcal{I}-\mathcal{P}_{i_2}]\left(\prod_{i=i_2+1}^{N}\mathcal{P}_i\right)\right\} f(x_1,\ldots,x_N). \tag{2.10}$$

The other terms are also can be derived in the same way, but here we will not consider these terms.

All the observations up to here give some idea about the success degree of the additive decomposition given in (2.1) by observing the attribute of the components to the whole norm. When equation (2.1) is examined carefully, it is seen that that summation consists finitely many terms, actually 2^N terms.

However, if N gets large, the usability and applicability are lost. Due to this fact, it is important to generate a new approximation by cutting off the expansion, that is, considering first few terms and omitting the other elements in the expansion. The importance of this expansion comes from the first term which is the constant term. It is obvious that constant function makes the implementations easier, but in case of necessity, the subsequent univariate component may be taken in order to obtain a more sensitive approximation.

The ratio of the square of the norm of an approximation gotten from a chopping in the right hand side of (2.1) to the square of the norm of the original function can be thought as an additivity measure. In this context, the following definitions can be made:

$$
\begin{aligned}
\sigma_0 &\equiv \frac{\|f_0\|^2}{\|f\|^2} \qquad \text{Constancy,} \\
\sigma_1 &\equiv \sigma_0 + \frac{\left\| \sum_{i=1}^{N} f_i(x_i) \right\|^2}{\|f\|^2} \qquad \text{1. Order Additivity Measure,} \\
\sigma_2 &\equiv \sigma_1 + \frac{\left\| \sum\limits_{\substack{i_1,i_2=1 \\ i_1<i_2}}^{N} f_{i_1,i_2}(x_{i_1}, x_{i_2}) \right\|^2}{\|f\|^2} \qquad \text{2. Order Additivity Measure,} \\
&\vdots
\end{aligned}
\tag{2.11}
$$

SECTION 3

SOLVING EIGENVALUE PROBLEM WITH HDMR

In the first chapter, it was said that an equation of the form

$$\mathbf{A}\mathbf{u} = \lambda\mathbf{u} \tag{3.1}$$

is called matrix eigenvalue problem. In this chapter, we will try to find the necessary equations for determination of the eigenvectors and eigenvalues of $\mathbf{A}$ using HDMR.

3.1 Determination of the HDMR Components

The eigenvector and the eigenvalue of a matrix depend on the entries of that matrix. When the entries of the matrix are changed, its eigenvectors and eigenvalues are also changed. Therefore, we can think the elements of the matrix as independent variables, and the eigenvector and eigenvalues as dependent variables. Although eigenvalue is a dependent variable, in this work it will be taken as a constant variable in order to consider the original problem in linear form.

Let $\mathbf{A}$ be written as

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ \vdots & \vdots & & \vdots \\ a_{N1} & a_{N2} & \cdots & a_{NN} \end{pmatrix} = \begin{pmatrix} x_1 & x_2 & \cdots & x_N \\ x_{N+1} & x_{N+2} & \cdots & x_{2N} \\ \vdots & \vdots & \cdots & \vdots \\ x_{N^2-N+1} & x_{N^2-N+2} & \cdots & x_{N^2} \end{pmatrix}$$

$$\mathbf{A} = x_1\mathbf{A}_1 + x_2\mathbf{A}_2 + \cdots + x_{N^2}\mathbf{A}_{N^2}, \tag{3.2}$$

where

$$\mathbf{A}_i = \begin{pmatrix} 0 & \cdots & 0 & & \cdots & 0 \\ \vdots & & \vdots & & & \vdots \\ 0 & \cdots & 0 & 1 & 0 \cdots & 0 \\ \vdots & & \vdots & & & \vdots \\ 0 & \cdots & 0 & & \cdots & 0 \end{pmatrix}, \tag{3.3}$$

and the only nonzero element of the matrix is in the corresponding position of x_i.

The HDMR expansion of $\mathbf{u}$ is

$$\mathbf{u}(x_1,...,x_{N^2}) = \mathbf{u}_0 + \sum_{i=1}^{N^2} \mathbf{u}_i(x_i) + \sum_{\substack{i_1,i_2=1 \\ i_1<i_2}}^{N^2} \mathbf{u}_{i_1 i_2}(x_{i_1},x_{i_2}) + \sum_{\substack{i_1,i_2,i_3=1 \\ i_1<i_2<i_3}}^{N^2} \mathbf{u}_{i_1 i_2 i_3}(x_{i_1},x_{i_2},x_{i_3}) + \cdots \tag{3.4}$$

If (3.4) is replaced by $\mathbf{u}$ in equation (3.1), then

$$(x_1\mathbf{A}_1 + x_2\mathbf{A}_2 + \cdots + x_{N^2}\mathbf{A}_{N^2}) \left(\mathbf{u}_0 + \sum_{i=1}^{N^2} \mathbf{u}_i(x_i) + \sum_{\substack{i_1,i_2=1 \\ i_1<i_2}}^{N^2} \mathbf{u}_{i_1 i_2}(x_{i_1},x_{i_2}) + \cdots \right)$$

$$= \lambda \left(\mathbf{u}_0 + \sum_{i=1}^{N^2} \mathbf{u}_i(x_i) + \sum_{\substack{i_1,i_2=1 \\ i_1<i_2}}^{N^2} \mathbf{u}_{i_1 i_2}(x_{i_1},x_{i_2}) + \cdots \right) \tag{3.5}$$

is obtained.

This last equation will be used in order to determine HDMR components.

3.1.1 Determination of the Constant Term

We try to find the constant function, so we will start calculation by multiplying both sides of the equation (3.5) by the weight functions and then integrating over the related intervals. The resulting equation will be

$$\left(\sum_{i=1}^{N^2} \int_{a_i}^{b_i} dx_i \, x_i \mathbf{A}_i W_i(x_i) \right) \mathbf{u}_0 + \left(\sum_{i=1}^{N^2} \int_{a_i}^{b_i} dx_i \, x_i \mathbf{A}_i \mathbf{u}_i(x_i) W_i(x_i) \right) = \lambda \, \mathbf{u}_0. \tag{3.6}$$

In equation (3.6), let

$$\int_{a_i}^{b_i} dx_i \, x_i \mathbf{A}_i W_i(x_i) = \mathbf{\Theta}_i \tag{3.7}$$

and

$$\int_{a_i}^{b_i} dx_i \, x_i \mathbf{A}_i \mathbf{u}_i(x_i) W_i(x_i) = \alpha_i. \tag{3.8}$$

Substituting these two values in (3.6) the following equation is obtained

$$\left(\sum_{i=1}^{N^2} \mathbf{\Theta}_i \right) \mathbf{u}_0 + \sum_{i=1}^{N^2} \alpha_i = \lambda \, \mathbf{u}_0. \tag{3.9}$$

In (3.9), if the contribution of the univariate HDMR component is ignored, that is

if the term $\sum_{i=1}^{N^2} \alpha_i$ is not considered, then the Zeroth Order HDMR Approximation will be

$$\left(\sum_{i=1}^{N^2} \mathbf{\Theta}_i \right) \mathbf{u}_0^{(0)} = \lambda^{(0)} \, \mathbf{u}_0^{(0)}. \tag{3.10}$$

This equation is again a matrix eigenvalue problem, and if $\mathbf{\Theta}_i$ is known, it can be solved easily.

3.1.2 Determination of the Univariate HDMR Components

In order to find the univariate HDMR components, at each step one independent variable is excluded, and the integral is taken with respect to the independent variables left. Of course, the corresponding weight functions for each independent variable are multiplied before the integration.

In equation (3.5), multiplying both sides of the equation by the weight functions excluding $W_k(x_k)$ and integrating both sides with respect to the independent variables x_i's over the interval $[a_i, b_i]$, excluding x_k, the following equation yields:

$$\begin{aligned}
&\left(\sum_{\substack{i=1 \\ i \neq k}}^{N^2} \int_{a_i}^{b_i} dx_i \, x_i W_i(x_i) \mathbf{A}_i \right) \mathbf{u}_0 + x_k \mathbf{A}_k \mathbf{u}_0 + \left(\sum_{\substack{i=1 \\ i \neq k}}^{N^2} \int_{a_i}^{b_i} dx_i \, x_i \mathbf{A}_i W_i(x_i) \mathbf{u}_i(x_i) \right) \\
&+ x_k \mathbf{A}_k \, \mathbf{u}_k(x_k) + \left(\sum_{\substack{i=1 \\ i \neq k}}^{N^2} \int_{a_i}^{b_i} dx_i \, x_i \mathbf{A}_i W_i(x_i) \right) \mathbf{u}_k(x_k) \\
&+ \left(\sum_{i=1}^{k-1} \int_{a_i}^{b_i} dx_i \, x_i \mathbf{A}_i W_i(x_i) \mathbf{u}_{ik}(x_i, x_k) \right) + \left(\sum_{i=k+1}^{N^2} \int_{a_i}^{b_i} dx_i \, x_i \mathbf{A}_i W_i(x_i) \mathbf{u}_{ki}(x_i, x_k) \right) \\
&= \lambda(\mathbf{u}_0 + \mathbf{u}_k(x_k)).
\end{aligned} \tag{3.11}$$

Let

$$\int_{a_i}^{b_i} dx_i \, x_i W_i(x_i) \mathbf{A}_i = \mathbf{\Theta}_i \tag{3.12}$$

and

$$\int_{a_i}^{b_i} dx_i \, x_i \mathbf{A}_i W_i(x_i) \mathbf{u}_i(x_i) = \alpha_i. \tag{3.13}$$

If the above variables are replaced in (3.11) and the contribution of the bivariate components, $\sum_{i=1}^{k-1}\int_{a_i}^{b_i} dx_i\, x_i \mathbf{A}_i W_i(x_i)\mathbf{u}_{ik}(x_i,x_k)$ and $\sum_{i=k+1}^{N^2}\int_{a_i}^{b_i} dx_i\, x_i \mathbf{A}_i W_i(x_i)\mathbf{u}_{ki}(x_i,x_k)$, is omitted, then the First Order HDMR approximation will be found as

$$\left(\sum_{\substack{i=1\\ i\neq k}}^{N^2}\mathbf{\Theta}_i\right)\mathbf{u}_0^{(1)} + x_k\mathbf{A}_k\mathbf{u}_0^{(1)} + \sum_{\substack{i=1\\ i\neq k}}^{N^2}\alpha_i + x_k\mathbf{A}_k\mathbf{u}_k^{(1)}(x_k) + \left(\sum_{\substack{i=1\\ i\neq k}}^{N^2}\mathbf{\Theta}_i\right)\mathbf{u}_k^{(1)}(x_k)$$

$$= \lambda^{(1)}(\mathbf{u}_0^{(1)} + \mathbf{u}_k^{(1)}(x_k)). \tag{3.14}$$

If (3.14) is rearranged to find $\mathbf{u}_k(x_k)$ in terms of other terms, we get

$$\left(x_k\mathbf{A}_k + \sum_{\substack{i=1\\ i\neq k}}^{N^2}\mathbf{\Theta}_i - \lambda\mathbf{I}\right)\mathbf{u}_k^{(1)}(x_k) = -\left(x_k\mathbf{A}_k + \sum_{\substack{i=1\\ i\neq k}}^{N^2}\mathbf{\Theta}_i - \lambda\mathbf{I}\right)\mathbf{u}_0^{(1)} - \sum_{\substack{i=1\\ i\neq k}}^{N^2}\alpha_i$$

$$\mathbf{u}_k^{(1)}(x_k) = -\mathbf{I}\mathbf{u}_0^{(1)} - \left(x_k\mathbf{A}_k + \sum_{\substack{i=1\\ i\neq k}}^{N^2}\mathbf{\Theta}_i - \lambda\mathbf{I}\right)^{-1}\sum_{\substack{i=1\\ i\neq k}}^{N^2}\alpha_i. \tag{3.15}$$

In equation (3.15), let

$$\mathcal{A}^{-1} = \left(x_k\mathbf{A}_k + \sum_{\substack{i=1\\ i\neq k}}^{N^2}\mathbf{\Theta}_i - \lambda\mathbf{I}\right)^{-1}. \tag{3.16}$$

It was said that

$$\mathbf{\Theta}_i = \int_{a_i}^{b_i} dx_i\, x_i W_i(x_i)\mathbf{A}_i = \mathbf{A}_i\int_{a_i}^{b_i} dx_i\, x_i W_i(x_i) = \mathbf{A}_i\gamma_i \tag{3.17}$$

where $\gamma_i = \int_{a_i}^{b_i} dx_i\, x_i W_i(x_i)$.

Here, we know that $\gamma_k A_k = \mathbf{\Theta_k}$. So, replacing this equality, (3.15) can be rewritten as

$$\mathbf{u}_k^{(1)}(x_k) = -\mathbf{I}\mathbf{u}_0^{(1)} - \left((x_k-\gamma_k)\mathbf{A}_k + \sum_{i=1}^{N^2}\mathbf{\Theta}_i - \lambda\mathbf{I}\right)^{-1}(\sum_{i=1}^{N^2}\alpha_i - \alpha_k). \tag{3.18}$$

Now, we have the compact form of the univariate HDMR component, but we still do not know the value of the variables α_i. In order to determine these values,

we will use the vanishing property which is that integration of the univariate component with respect to its independent variable is zero. Therefore;

$$\begin{aligned}\mathbf{0}=\int_{a_k}^{b_k} dx_k\, W_k(x_k)\mathbf{u}_k^{(1)}(x_k) \;&=\; -\int_{a_k}^{b_k} dx_k\, W_k(x_k)\mathbf{u}_0^{(1)} \\ &\quad -\int_{a_k}^{b_k} dx_k\, W_k(x_k)\left((x_k-\gamma_k)\mathbf{A}_k+\sum_{i=1}^{N^2}\mathbf{\Theta}_i-\lambda\mathbf{I}\right)^{-1}\times \\ &\quad \times(\sum_{i=1}^{N^2}\alpha_i-\alpha_k). \qquad (3.19)\end{aligned}$$

From equation (3.19) we get

$$\mathbf{u}_0^{(1)}=-\left\{\int_{a_k}^{b_k} dx_k\, W_k(x_k)\left[(x_k-\gamma_k)\mathbf{A}_k+\sum_{i=1}^{N^2}\mathbf{\Theta}_i-\lambda\mathbf{I}\right]^{-1}\right\}(\sum_{i=1}^{N^2}\alpha_i-\alpha_k). \quad (3.20)$$

From this last equation, α_i's in terms of the known variables is found as

$$(\sum_{i=1}^{N^2}\alpha_i-\alpha_k)=-\left\{\int_{a_k}^{b_k} dx_k\, W_k(x_k)\left[(x_k-\gamma_k)\mathbf{A}_k+\sum_{i=1}^{N^2}\mathbf{\Theta}_i-\lambda\mathbf{I}\right]^{-1}\right\}^{-1}\mathbf{u}_0^{(1)} \quad (3.21)$$

Summing the equation (3.21) over k from 1 to N^2, we get

$$\sum_{k=1}^{N^2}\left((\sum_{i=1}^{N^2}\alpha_i-\alpha_k)\right)=-\sum_{k=1}^{N^2}\left(\left\{\int_{a_k}^{b_k} dx_k\, W_k(x_k)\left[(x_k-\gamma_k)\mathbf{A}_k+\sum_{i=1}^{N^2}\mathbf{\Theta}_i-\lambda\mathbf{I}\right]^{-1}\right\}^{-1}\mathbf{u}_0^{(1)}\right) \quad (3.22)$$

$$\sum_{k=1}^{N^2}\left((\sum_{i=1}^{N^2}\alpha_i-\alpha_k)\right)=\sum_{k=1}^{N^2}\sum_{i=1}^{N^2}\alpha_i-\sum_{k=1}^{N^2}\alpha_k=N^2\sum_{i=1}^{N^2}\alpha_i-\sum_{k=1}^{N^2}\alpha_k=(N^2-1)\sum_{i=1}^{N^2}\alpha_i$$

So, replacing the last equation in (3.22) we obtain

$$(N^2-1)\sum_{i=1}^{N^2}\alpha_i=-\sum_{k=1}^{N^2}\left(\left\{\int_{a_k}^{b_k} dx_k\, W_k(x_k)\left[(x_k-\gamma_k)\mathbf{A}_k+\sum_{i=1}^{N^2}\mathbf{\Theta}_i-\lambda\mathbf{I}\right]^{-1}\right\}^{-1}\mathbf{u}_0^{(1)}\right). \quad (3.23)$$

Finally, from equation (3.23), we can easily find the sum of the unknown variables α_i as

$$\sum_{i=1}^{N^2}\alpha_i=-\frac{1}{(N^2-1)}\sum_{k=1}^{N^2}\left(\left\{\int_{a_k}^{b_k} dx_k\, W_k(x_k)\left[(x_k-\gamma_k)\mathbf{A}_k+\sum_{i=1}^{N^2}\mathbf{\Theta}_i-\lambda\mathbf{I}\right]^{-1}\right\}^{-1}\mathbf{u}_0^{(1)}\right). \quad (3.24)$$

Now, if we replace these lastly found value in equation (3.9), we get

$$\left[\sum_{i=1}^{N^2}\mathbf{\Theta}_i-\lambda\mathbf{I}-\frac{1}{(N^2-1)}\sum_{k=1}^{N^2}\left(\left\{\int_{a_k}^{b_k}dx_k\,W_k(x_k)\left[(x_k-\gamma_k)\mathbf{A}_k+\sum_{i=1}^{N^2}\mathbf{\Theta}_i-\lambda\mathbf{I}\right]^{-1}\right\}\right)^{-1}\right]\mathbf{u}_0^{(1)}=0. \tag{3.25}$$

From equation (3.25) let

$$\left\{\int_{a_k}^{b_k}dx_k\,W_k(x_k)\left[(x_k-\gamma_k)\mathbf{A}_k+\sum_{i=1}^{N^2}\mathbf{\Theta}_i-\lambda\mathbf{I}\right]^{-1}\right\}=\int_{a_k}^{b_k}dx_k\,W_k(x_k)\mathcal{A}_k^{-1} \tag{3.26}$$

where

$$\mathcal{A}_k^{-1}=\left[(x_k-\gamma_k)\mathbf{A}_k+\sum_{i=1}^{N^2}\mathbf{\Theta}_i-\lambda\mathbf{I}\right]^{-1} \tag{3.27}$$

After some arrangement of the last eqaution we can find the inverse of the matrix $\mathcal{A}_k$ as

$$\begin{aligned}\mathcal{A}_k^{-1} &= \left[\left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i-\lambda\mathbf{I}\right)\left((x_k-\gamma_k)\left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i-\lambda\mathbf{I}\right)^{-1}\mathbf{A}_k+\mathbf{I}\right)\right]^{-1}\\ &= \left((x_k-\gamma_k)\left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i-\lambda\mathbf{I}\right)^{-1}\mathbf{A}_k+\mathbf{I}\right)^{-1}\left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i-\lambda\mathbf{I}\right)^{-1}\end{aligned} \tag{3.28}$$

Here, it is abviously seen that $\left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i-\lambda\mathbf{I}\right)^{-1}$ is independent of x_k. So from equation (3.25)

$$\int_{a_k}^{b_k}dx_k\,W_k(x_k)\mathcal{A}_k^{-1}=\int_{a_k}^{b_k}dx_k\,W_k(x_k)\left((x_k-\gamma_k)\left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i-\lambda\mathbf{I}\right)^{-1}\mathbf{A}_k+\mathbf{I}\right)^{-1}\times \times\left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i-\lambda\mathbf{I}\right)^{-1} \tag{3.29}$$

Let $(x_k-\gamma_k)=\overline{x}_k$ and $\left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i-\lambda\mathbf{I}\right)^{-1}\mathbf{A}_k=\mathbf{M}$. Since we have used change of variable for x_k, we have to change the integral intervals, briefly we will use the integral interval as $[\overline{a}_k,\overline{b}_k]$. So, replacing these two variables in equation (3.29), the following equation yields:

$$\int_{\overline{a}_k}^{\overline{b}_k}d\overline{x}_k\,W_k(\overline{x}_k)\mathcal{A}_k^{-1}=\left(\int_{\overline{a}_k}^{\overline{b}_k}d\overline{x}_k\,W_k(\overline{x}_k)\left(\mathbf{I}+\overline{x}_k\mathbf{M}\right)^{-1}\right)\left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i-\lambda\mathbf{I}\right)^{-1} \tag{3.30}$$

In equation (3.30) we have an inverse matrix, we will calculate this matrix firstly, but in order to do this procedure we will use the power series expansion. We know that

$$\frac{1}{1-x} = \sum_{k=0}^{\infty} x^k = 1 + x + x^2 + x^3 + \cdots \tag{3.31}$$

By the same anology, we say that

$$(\mathbf{I} + \overline{x}_k\mathbf{M}) = \sum_{l=0}^{\infty}(-1)^l(\overline{x}_k\mathbf{M})^l = I - \overline{x}_k\mathbf{M} + \overline{x}_k{}^2\mathbf{M}^2 - \overline{x}_k^3\mathbf{M}^3 + \cdots \tag{3.32}$$

Now, we will calculate the powers of the matrix $\mathbf{M}$.

$$\mathbf{M}^2 = \left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i - \lambda\mathbf{I}\right)^{-1}\mathbf{A}_k \times \left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i - \lambda\mathbf{I}\right)^{-1}\mathbf{A}_k. \tag{3.33}$$

Say, $\mathbf{A}_k = \mathbf{e}_i\mathbf{e}_j^T$, where $\mathbf{e}_i$'s are the standard unit vectors, which is the vector with nonzero element 1 is in the ith position and all other elements are 0.Then,

$$\begin{aligned}\mathbf{M}^2 &= \left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i - \lambda\mathbf{I}\right)^{-1}\mathbf{e}_i\mathbf{e}_j^T\left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i - \lambda\mathbf{I}\right)^{-1}\mathbf{e}_i\mathbf{e}_j^T \\ &= \sigma\left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i - \lambda\mathbf{I}\right)^{-1}\mathbf{A}_k = \sigma\mathbf{M}\end{aligned} \tag{3.34}$$

where $\sigma = \mathbf{e}_j^T\left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i - \lambda\mathbf{I}\right)^{-1}\mathbf{e}_i$

By induction, we can easily observe that $\mathbf{M}^n = \sigma^{n-1}\mathbf{M}$. So, from equation (3.32)

$$\begin{aligned}(\mathbf{I} + \overline{x}_k\mathbf{M})^{-1} &= \sum_{l=0}^{\infty}(-1)^l\,\overline{x}_k^l\,\mathbf{M}^l = \mathbf{I} + \sum_{l=1}^{\infty}(-1)^l\,\overline{x}_k^l\,\mathbf{M}^l = \mathbf{I} + \left(\sum_{l=1}^{\infty}(-1)^k\,\overline{x}_k^l\,\sigma^{l-1}\right)\mathbf{M} \\ &= \mathbf{I} + \left(\sum_{l=0}^{\infty}(-1)^{k+1}\,\overline{x}_k^{k+1}\right)\mathbf{M} = \mathbf{I} + (-1)\overline{x}_k\left(\sum_{l=0}^{\infty}(-1)^l\,\overline{x}_k^l\,\sigma^l\right)\mathbf{M} \\ &= \mathbf{I} - \frac{\overline{x}_k}{1+\sigma\overline{x}_k}\mathbf{M}\end{aligned} \tag{3.35}$$

Since we have found the inverse of the matrix, we return back to the equation in order to evaluate the integration part. Therefore;

$$\left[\int_{\overline{a}_k}^{\overline{b}_k} d\overline{x}_k\, W_k(\overline{x}_k)\,(\mathbf{I} + \overline{x}_k\mathbf{M})^{-1}\right]\left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i - \lambda\mathbf{I}\right)^{-1} =$$

$$= \left[\int_{\overline{a}_k}^{\overline{b}_k} d\overline{x}_k\, W_k(\overline{x}_k)\left(\mathbf{I} - \frac{\overline{x}_k}{1+\sigma\overline{x}_k}\mathbf{M}\right)\right]\left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i - \lambda\mathbf{I}\right)^{-1}$$

$$= \left[\int_{\overline{a}_k}^{\overline{b}_k} d\overline{x}_k\, W_k(\overline{x}_k)\left(\mathbf{I} - \left(\frac{1}{\sigma} + \frac{1/\sigma}{1+\overline{x}_k\sigma}\right)\mathbf{M}\right)\right]\left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i - \lambda\mathbf{I}\right)^{-1}$$

$$= \left[\mathbf{I} - \left(\frac{1}{\sigma} + \int_{\overline{a}_k}^{\overline{b}_k} d\overline{x}_k\, W_k(\overline{x}_k)\,\frac{1/\sigma}{1+\overline{x}_k\sigma}\right)\mathbf{M}\right]\left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i - \lambda\mathbf{I}\right)^{-1} \quad (3.36)$$

From equation (3.25), replacing the equation (3.36) gives

$$\left[\sum_{i=1}^{N^2}\mathbf{\Theta}_i - \lambda\mathbf{I} - \frac{1}{(N^2-1)}\sum_{k=1}^{N^2}\left\{\left(\mathbf{I} - \left(\frac{1}{\sigma} + \int_{\overline{a}_k}^{\overline{b}_k} d\overline{x}_k\, W_k(\overline{x}_k)\,\frac{1/\sigma}{1+\overline{x}_k\sigma}\right)\mathbf{M}\right)\times \right.\right.$$
$$\left.\left. \times \left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i - \lambda\mathbf{I}\right)^{-1}\right\}^{-1}\right]\mathbf{u}_0^{(1)} = \mathbf{0} \quad (3.37)$$

Let $\left(\frac{1}{\sigma} + \int_{\overline{a}_k}^{\overline{b}_k} d\overline{x}_k\, W_k(\overline{x}_k)\,\frac{1/\sigma}{1+\overline{x}_k\sigma}\right) = z$. So, equation (3.37) can be rewritten as

$$\left[\sum_{i=1}^{N^2}\mathbf{\Theta}_i - \lambda\mathbf{I} - \frac{1}{(N^2-1)}\sum_{k=1}^{N^2}\left\{[\mathbf{I} - z\mathbf{M}]\left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i - \lambda\mathbf{I}\right)^{-1}\right\}^{-1}\right]\mathbf{u}_0^{(1)} = \mathbf{0} \quad (3.38)$$

$$\left\{[\mathbf{I} - z\mathbf{M}]\left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i - \lambda\mathbf{I}\right)^{-1}\right\}^{-1} = \left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i - \lambda\mathbf{I}\right)(\mathbf{I} - z\mathbf{M})^{-1} \quad (3.39)$$

In order to find the inverse matrix $(\mathbf{I} - z\mathbf{M})^{-1}$ we will again use the power series expansion. Therefore;

$$(\mathbf{I} - z\mathbf{M})^{-1} = \left[\mathbf{I} - z\left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i - \lambda\mathbf{I}\right)^{-1}\mathbf{A}_k\right]^{-1}.$$

$$\begin{aligned}(\mathbf{I} - z\mathbf{M})^{-1} &= \sum_{l=0}^{\infty} z^l\mathbf{M}^l \text{ where } M = \left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i - \lambda\mathbf{I}\right)^{-1}\mathbf{A}_k \\ &= \mathbf{I} + \sum_{l=1}^{\infty} z^l\mathbf{M}^l = \mathbf{I} + \left(\sum_{l=1}^{\infty} z^l\sigma^{l-1}\right)\mathbf{M} \\ &= \mathbf{I} + \left(\sum_{l=0}^{\infty} z^{l+1}\sigma^l\right)\mathbf{M} \\ &= \mathbf{I} - \frac{z}{1-z\sigma}\mathbf{M}. \end{aligned} \quad (3.40)$$

Finally; replacing equation (3.40) in (3.37), we will get the resulting equation as

$$\left[\sum_{i=1}^{N^2}\mathbf{\Theta}_i - \lambda\mathbf{I} - \frac{1}{(N^2-1)}\sum_{k=1}^{N^2}\left\{\left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i - \lambda\mathbf{I}\right)\left(\mathbf{I} - \frac{z}{1-z\sigma}\mathbf{M}\right)\right\}\right]\mathbf{u}_0^{(1)} = \mathbf{0} \quad (3.41)$$

If the last equation is arranged, we will see that the last equation is again an eigenvalue equation with coefficient matrix and the weighting matrix given as below.

$$\left[-\lambda\mathbf{I}-\frac{1}{(N^2-1)}\sum_{k=1}^{N^2}\left\{\left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i-\lambda\mathbf{I}\right)\left(\mathbf{I}-\frac{z}{1-z\sigma}\mathbf{M}\right)\right\}\right]\mathbf{u}_0^{(1)}=-\left(\sum_{i=1}^{N^2}\mathbf{\Theta}_i\right)\mathbf{u}_0^{(1)} \tag{3.42}$$

This last equation will be used in order to determine the constant HDMR component in the First Order Approximation of HDMR; but, we see the nonlinear dependence on the eigenvalue λ. So, firstly the first order approximation of the eigenvalue using First Order HDMR Approximation will be determined. We will use the Direct Method, described in Chapter 1.1.2. For this process, we will choose two initial values for eiganvalue λ, and apply the Secant Method to the determinant of the matrix in the left hand side of the equation (3.41). When it converges, the eigenvalue $\lambda_0^{(1)}$, which denotes that the eigenvalue is found by using the First Order HDMR Approximation, will be determined. This completes the approximation of the eiganvalue and the eigenvector using HDMR.

SECTION 4

NUMERICAL ILLUSTRATIONS

4.1 Zeroth Order Approximation Of HDMR

In this section, we will try to solve the eigenvalues and eigenvectors of some matrices by using Zeroth Order HDMR Approximation given in equation (3.10). Moreover, we will calculate an approximation measure $\tilde{\sigma}_0$ and give a sketch of the change of $\tilde{\sigma}_0$ with respect to the change in dimension of the matrix

4.1.1 Matrix with decreasing elements

$$\begin{pmatrix} 1/2 & 1/3 & 1/4 & \cdots & 1/(1+N) \\ 1/3 & 1/4 & 1/5 & \cdots & 1/(2+N) \\ \vdots & \vdots & \vdots & & \vdots \\ 1/(k+1) & 1/(k+2) & 1/(k+3) & \cdots & 1/(k+N) \\ \vdots & \vdots & \vdots & & \vdots \\ 1/(N+1) & 1/(N+2) & 1/(N+3) & \cdots & 1/(N+N) \end{pmatrix} \tag{4.1}$$

In numerical testing, the weight function $W_i(x_i) = 1/4$ and integral interval is $[1, 5]$ for the elements not on the diagonal, and $W_i(x_i) = 2x_i$ and the interval is $[0, 1]$ for the ones on the diagonal. As a sample run, we will show the solution by HDMR for N=3.

```
   *----*     MuPAD 2.5.2 -- The Open Computer Algebra System
  /|   /|
 *----* |     Copyright (c)  1997 - 2002  by SciFace Software
 | *--|-*                   All rights reserved.
 |/   |/
 *----*       Licensed to:   Bilisim Enstitusu

>> N:=3:
>> ai:=1:
```

```
>> bi:=5:
```

"A"

```
+-                -+
|  1/2, 1/3, 1/4   |
|                  |
|  1/3, 1/4, 1/5   |
|                  |
|  1/4, 1/5, 1/6   |
+-                -+
```

```
>> /*****************************************************************/
>> /*****************************************************************/
>> /*****************************************************************/
>> /* DETERMINATION OF EIGENVALUES AND EIGENVECTORS OF A */
>> eigval:=linalg::eigenvalues(A,Multiple);
```

```
                    [[1/2, 1], [1/4, 1], [1/6, 1]]
>> eigvec:=numeric::eigenvectors(A);
```

```
--                             +-               -+                   --
|                              |  1.0, 0.0, 0.0   |                    |
|                              |                  |                    |
|  [0.5, 0.25, 0.1666666667], |  0.0, 1.0, 0.0   |, [0.0, 0.0, 0.0]   |
|                              |                  |                    |
|                              |  0.0, 0.0, 1.0   |                    |
--                             +-               -+                   --
>> /*****************************************************************/
>> /*****************************************************************/
>> /*****************************************************************/
```

"Integral intervals"

```
>> a;
```

```
+-                          -+
| 0, 1, 1, 1, 0, 1, 1, 1, 0 |
+-                          -+
>> b;
```

```
+-                          -+
```

```
                    | 1, 5, 5, 5, 1, 5, 5, 5, 1 |
                    +-                         -+

>> /*****************************************************************/

                  "Matrix of the weight functions "

>> Wt();

                        +-                  -+
                        |  2 x1,  1/4,  1/4  |
                        |                    |
                        |   1/4, 2 x5,  1/4  |
                        |                    |
                        |   1/4,  1/4, 2 x9  |
                        +-                  -+
>> /*****************************************************************/

                    "Sum of all Thetai functions"

>> thetaisum;

            +-                                        -+
            |  0.6666666667,      3.0,          3.0     |
            |                                           |
            |       3.0,     0.6666666667,      3.0     |
            |                                           |
            |       3.0,          3.0,     0.6666666667 |
            +-                                        -+
>>
>> /*****************************************************************/
>> /*****************************************************************/
>> /*****************************************************************/
>> /* DETERMINATION OF ZEROTH ORDER HDMR APPROXIMATION */
>> eigval:=linalg::eigenvalues(thetaisum,Multiple);

                [[-2.333333333, 2], [6.666666667, 1]]
>> eigvec:=numeric::eigenvectors(thetaisum);
```

```
--
|
|
|  [6.666666667, -2.333333333, -2.333333333],
|
|
--
```

$$\begin{bmatrix} 0.5773502692, & 0.0, & 0.8164965809 \\ 0.5773502692, & 0.7071067812, & -0.4082482905 \\ 0.5773502692, & -0.7071067812, & -0.4082482905 \end{bmatrix},$$

$$[2.860321779e\text{-}18,\ 3.909146526e\text{-}19,\ 9.832823112e\text{-}19]$$

```
>> /*************************************************************/
>> /*************************************************************/
>> /*************************************************************/
>> A_thetaisum:=matrix(N,N):
>> A_thetaisum:=A-thetaisum;
```

$$\begin{bmatrix} -0.1666666667, & -2.666666667, & -2.75 \\ -2.666666667, & -0.4166666667, & -2.8 \\ -2.75, & -2.8, & -0.5 \end{bmatrix}$$

```
>> normDifference:=norm(A_thetaisum,Frobenius):
>> print("norm of A - Thetaisum=",normDifference);

                "norm of A - Thetaisum=", 6.743783145
```

```
>> normA:=norm(A,Frobenius):
>> print("norm of the original matrix=",normA);
```

$$\text{"norm of the original matrix="}, \frac{307^{1/2}}{20}$$

```
>> sigma0:=normDifference/normA:
>> print("sigma0=",simplify(sigma0));
```

$$\text{"sigma0="}, 0.4393344069\ 307^{1/2}$$

$$\text{"first eigenvalue="}, 6.666666667$$

"first eigenvector"

$$\begin{bmatrix} 0.5773502692 \\ 0.5773502692 \\ 0.5773502692 \end{bmatrix}$$

$$\text{"second eigenvalue="}, -2.333333333$$

"second eigenvector"

$$\begin{bmatrix} 0.0 \\ 0.7071067812 \\ -0.7071067812 \end{bmatrix}$$

$$\text{"third eigenvalue="}, -2.333333333$$

```
"third eigenvector"

+-                -+
|   0.8164965809   |
|                  |
|  -0.4082482905   |
|                  |
|  -0.4082482905   |
+-                -+
```

Here, we will calculate an approximation measure, in fact additivity measure, $\tilde{\sigma}_0$ which is the ratio of the norm of the difference matrix between the original matrix and the coefficient matrix in the equation found by HDMR, to the norm of the original matrix. In the convergence case, we want $\tilde{\sigma}_0$ to go to 0. When it increases, we say that the solution diverges from the exact solution.

For this example we find the additivity measures as in the following:

When $N = 3$, $\tilde{\sigma}_0 = 12.6499173$.

When $N = 10$, $\tilde{\sigma}_0 = 45.8046644$.

When $N = 50$, $\tilde{\sigma}_0 = 233.0682382$.

As we can see, when the dimension increases, the approximated solution diverges, of course under the integral interval and the weight functions given above. The graphical representation of this result is given in the figure 4.1.

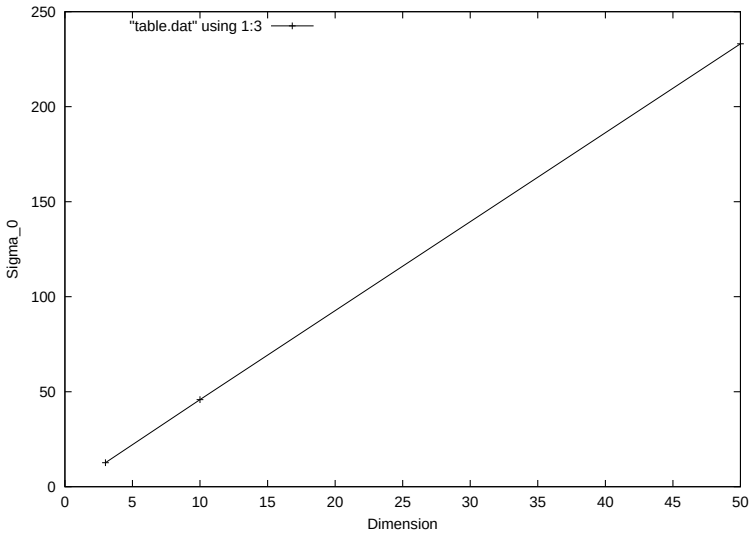

Figure 4.1: Change of $\tilde{\sigma}_0$ with respect to dimension of matrices with decreasing elements

4.1.2 Diagonal Matrix with Increasing Elements

We will not give the output of the program for this example. The values of the additivity measures and the sketch of the relationship between the matrix dimension and $\tilde{\sigma}_0$ will be given.

A is chosen as

$$\begin{pmatrix} 1+1 & 0 & 0 & \cdots & 0 \\ 0 & 2+2 & 0 & \cdots & 0 \\ 0 & 0 & 3+3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & N+N \end{pmatrix} \tag{4.2}$$

When $N = 3$, $\tilde{\sigma}_0 = 0.9087865$.

When $N = 10$, $\tilde{\sigma}_0 = 0.9720109$.

When $N = 50$, $\tilde{\sigma}_0 = 0.9943785$.

When $N = 100$, $\tilde{\sigma}_0 = 0.9983243$.

When a diagonal matrix having increasing elements when the row and column increase and same weight functions for all the elements are used, we observed that $\tilde{\sigma}_0$ is stable aroud 1.0. This means that when the size of the array increase, the approximated solution does not diverge from the exact solution. This situation is shown in figure 4.2.

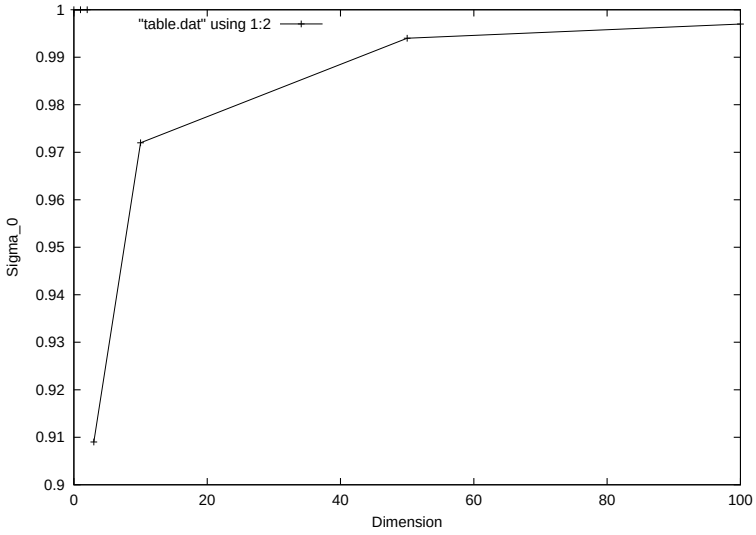

Figure 4.2: Change of $\tilde{\sigma}_0$ with respect to dimension of diagonal matrices with increasing elements

4.1.3 Symmetric Matrix with Decreasing Elements

Now, we will take a similar matrix as in the section 4.1.1.

$$\mathbf{A} = \sum_{i=1}^{N^2} \mathbf{\Theta}_i = \begin{pmatrix} 1 & 1/2 & 1/3 & 1/4 & 1/5 \\ 1/2 & 1/3 & 1/4 & 1/5 & 1/6 \\ 1/3 & 1/4 & 1/5 & 1/6 & 1/7 \\ 1/4 & 1/5 & 1/6 & 1/7 & 1/8 \\ 1/5 & 1/6 & 1/7 & 1/8 & 1/9 \end{pmatrix}, \tag{4.3}$$

and the weight functions as

$$\mathbf{W} = \begin{pmatrix} 1/2 & 1 & 3/2 & 4/2 & 5/2 \\ 1 & 3/2 & 2 & 5/2 & 3 \\ 3/2 & 2 & 5/2 & 3 & 7/2 \\ 2 & 5/2 & 3 & 7/2 & 4 \\ 5/2 & 3 & 7/2 & 4 & 9/2 \end{pmatrix} \tag{4.4}$$

and the integral intervals as

$$b_k = 2/(i+j-1), \quad a_k = 0, \text{ where } x_k = \mathbf{A}_{ij} \tag{4.5}$$

Here the matrix $\mathbf{W}$ is constructed by the weight functions, and $\mathbf{W}_{i,j} = \mathbf{W}_k(x_k)$ where $x_k = \mathbf{A}_{ij}$.

The matrix $\mathbf{A}$ is specially chosen matrix, it can be seen that it is the coefficient matrix in the Zeroth Order Approximation of HDMR. So, we will have the exact solution in the Zeroth Order. Therefore, we have a good condition for obtaining closer solution: If the matrix whose eigenvalues and eigenvectors will be found is almost same as the coefficient matrix in the Zeroth Order Approximation, then almost exact solution will be gotten.

In fact, this case is true for all orders of the approximation. Moreover, we can see the effect of the weight function and the integral interval in this closure.

Now, let us examine same cases. We will take the same matrix $\mathbf{A}$ and we will choose the weight functions and the integral intervals so that some elements of the coefficient matrix in the Zeroth Order Approximation is a perturbation of some elements of this matrix. In this case, closer solutions will be obtained. In order to determine the closureness of the solutions', the additivity measures explained above will be used.

For this example, additivity measures for different matrix dimensions and for different amount of perturbations will be given.

When $N = 3$, $\tilde{\sigma}_0 = 0.61$.

When $N = 10$, $\tilde{\sigma}_0 = 0.65$.

When $N = 50$, $\tilde{\sigma}_0 = 0.95$.

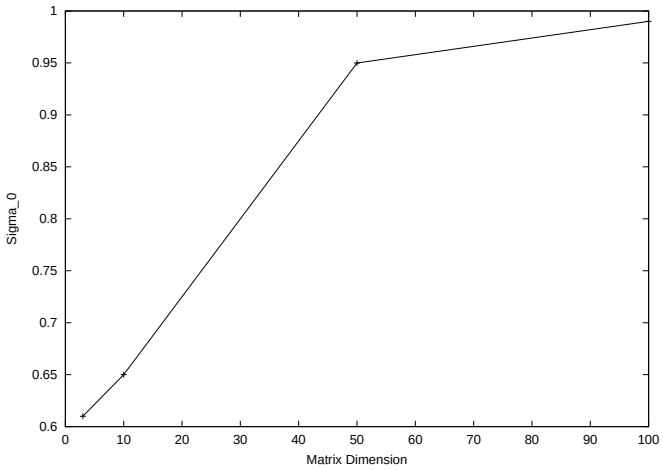

Figure 4.3: Change of $\tilde{\sigma}_0$ versus matrix dimension for perturbed matrix

When $N = 100, \quad \tilde{\sigma}_0 = 0.99.$

The figure 4.4 is for the matrix with different number and different amount of perturbation. The additivity measures for different matrix sizes is given as below.

When $N = 3, \quad \tilde{\sigma}_0 = 0.61.$

When $N = 10, \quad \tilde{\sigma}_0 = 0.76.$

When $N = 50, \quad \tilde{\sigma}_0 = 0.91.$

When $N = 100, \quad \tilde{\sigma}_0 = 0.99.$

Here, the difference between these two plots are the number of the perturbed element and the amount of the perturbation. If the number of the perturbed element is increased and the amount of the perturbation is much, $\tilde{\sigma}_0$ diverges from 0, which means that we obtain less close solution.

As a brief explanation, when we arrange the integral intervals and the weight functions so that the coefficient matrix is a small perturbation of the original matrix, then we will obtain a small deviation of the exact solution.

4.2 First Order Approximation of HDMR

Now, let us examine the solution of the eigenvalue equation using the First Order Approximation of HDMR, given in equation (3.42). As a simple example we will choose the matrix $\mathbf{A}$, the integral intervals and the weight functions as the ones in the previous section. Later we will determine the eigenvalue $\lambda^{(1)}$ after choosing two initial values and applying the secant method, described in section 1.1.2.

In the table 4.1, the exact eigenvalue λ_{exact}, initial values λ_1, λ_2 taken and the eigenvalue found by First Order Approximation $\lambda_0^{(1)}$ are given.

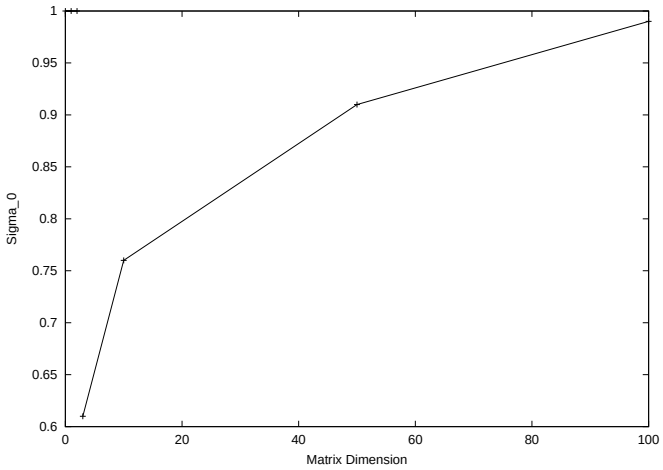

Figure 4.4: Change of $\tilde{\sigma}_0$ versus matrix dimension for perturbed matrix

Another example is for the diagonal matrix in the form

$$\mathbf{A} = \sum_{i=1}^{N^2} \mathbf{\Theta}_i = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 \\ 0 & 0 & 6 & 0 & 0 \\ 0 & 0 & 0 & 8 & 0 \\ 0 & 0 & 0 & 0 & 10 \end{pmatrix}, \tag{4.6}$$

The weight functions are taken as

$$W_i(x_i) = 1, \quad \forall i = 1, ..., 25, \tag{4.7}$$

and the integral intervals are

$$\begin{aligned} &a_k = -1, \quad \text{for k not a diagonal element}'\text{s index} \\ &a_1 = 3/2,\ a_7 = 7/2,\ a_{13} = 11/2,\ a_{19} = 15/2,\ a_{25} = 19/2 \\ &b_k = 1, \quad \text{for k not a diagonal element}'\text{s index} \\ &b_1 = 5/2,\ b_7 = 9/2,\ b_{13} = 13/2,\ b_{19} = 17/2,\ b_{25} = 21/2. \end{aligned} \tag{4.8}$$

The exact eigenvalues, the initial values and the approximated eigenvalues are given for such a matrix in table 4.2.

The deviation between the approximated solution and the exact solution comes from the integration process and the truncation of the HDMR. Also, the initial values play an important role at the convergence to the exact solution. One may get an unexpected value, but which is not wrong, because of the nature of the cenvergence of the Secant method. For example, for initial values 7.5 and 7.8, one expects the resulting value as 8, but the output is 2.

Table 4.1: First order approximation for symmetric matrix

λ_{exact}	λ_1	λ_2	$\lambda_0^{(1)}$
1.56705069109823	1.4	1.5	1.5366032411
0.208534218611013	0.20	0.21	0.2167392042820954
0.0114074916234198	0.01	0.02	0.0119285624783237
0.000305898040151192	0.0003	0.00031	0.00029561918
0.00000328792877217186	0	0.01	0.00000801039561600013

Table 4.2: First order approximation for diagonal matrix

λ_{exact}	λ_1	λ_2	$\lambda_0^{(1)}$
2	0	1	1.99999999801342
4	3	3.5	3.99999999723958
6	5	5.5	6.00000000368036
6	5	5.1	5.99999993942693
8	7	7.2	8.00492624336215
10	9.99	10.1	9.98936998188669
2	7.5	7.8	1.999999998002
6	7	7.5	6.0000000084256
10	9.8	10.3	9.83308280910606

SECTION 5

CONCLUSION

In this work, the solution of the matrix eigenvalue problem, which is of the form

$$\mathbf{A}\mathbf{u} = \lambda\mathbf{u} \tag{5.1}$$

was studied by using HDMR. Necessary equations for Zeroth Order and First Order HDMR Approximations were obtained and the constant term and univariate terms were determined by these approximations.

In the Zeroth Order Approximation, we saw that when the coefficient matrix $\tilde{\mathbf{A}}_0 = \sum_{i=1}^{N^2} \mathbf{\Theta}_i$ is closer to the original matrix A, we get solutions closer to the exact ones. Since the matrix $\tilde{\mathbf{A}}_0$ is constructed by taking the integral of weight functions, carefully chosen weight functions and integral intervals will enable us to get better approximated solutions. However, the complexity of the problem is directly proportional to the success ratio, which shows how close is the approximated solution to the exact solution. In other words, the better the approximated solution, the more operational load. Thus, willing to have better solutions will result in trying to solve a problem having almost the same characteristics of the original problem.

It can be seen easily that the matrix $\tilde{\mathbf{A}}_0$ is independent of the HDMR parameters x_i's. So, during the evaluation of the additivity measures given in page 34, it was seen that due to the lack of obtaining an analytic solution depending on parameters, we could not use the definitions of these measures directly. Therefore, we had to find and define another measure at this point.

Let $\tilde{\mathbf{A}}_0$ be the matrix in the Zeroth Order Approximation, $\tilde{\mathbf{A}}_1$ be the one in First Order Approximation , and so on. Define

$$\begin{aligned} \tilde{\sigma}_0 &= \frac{\|\mathbf{A} - \tilde{\mathbf{A}}_0\|}{\|\mathbf{A}\|}, \\ \tilde{\sigma}_1 &= \frac{\|\mathbf{A} - \tilde{\mathbf{A}}_1\|}{\|\mathbf{A}\|}, \\ \vdots &\quad \vdots \end{aligned} \tag{5.2}$$

Since we are dealing with the eigenvalues and the eigenvectors of $\tilde{\mathbf{A}}_i$'s, using these matrices in the convergence analysis is meaningful. Here, we will expect the value of $\tilde{\sigma}_i$ to go to zero as the approximated solution gets closer to the exact solution. If $\tilde{\sigma}_i$'s increases, we will say that the approximated solution diverges from the exact solution.

In some numerical approximations, we saw that Zeroth Order Approximation is not good for the matrices whose elements go to 0 as its row and column increase. When the size of the array increases, $\tilde{\sigma}_0$ increases unboundedly when the weight function is taken dependent on x_i's for diagonal elements and constant for off diagonal elements.

When we used a diagonal matrix having increasing elements when the row and column increase and same weight functions for all the elements, we observed that $\tilde{\sigma}_0$ is stable aroud 1.0. This means that when the size of the array increase, the approximated solution does not diverge from the exact solution.

In the First Order Approximation, it is observed that when the matrix in consideration is similar to the coefficient matrix in the First Order Approximation equation, then the eigenvalue is close to the exact ones. In order to satisfy this similarity, one may choose the weight functions and the integration intervals so that it gets the resulting coefficient matrix. As another case, the difference between these two matrices also effects the the solution of the problem. If the number of the perturbed elements and the perturbation amount are not much, then closer solutions will be obtained.

SECTION 6

FURTHER WORK

In this work, we deal with the solution of eigenvalue problem by considering eigenvectors as HDMR dependent variables, only. However, eigenvalue is also a dependent variable. As a further study both of them could be considered and HDMR can be applied to a nonlinear problem.

The relation between the HDMR application to the eigenvalue problem of the matrices and multiparametric perturbation expansion can be another interesting problem.

REFERENCES

[1] **Antia, H.M.**, 2000, Numerical Methods for Scientists and Engineers, *Algebraic Eigenvalue Problem*, pp. 502-548, Birkhäuser Verlag, Basel.

[2] **Demiralp, M.**, 2001, Yüksek Boyutlu Model Gösterilim: Değişmezlik ve Toplamsallık Ölçenleri, Çarpımsal Gösterilim ve Çarpımsallıllık Ölçenleri, *12. Ulusal Mekanik Kongresi*, Konya, Türkiye, 10-14 Eylül.

[3] **Greenbaum, A.**, 1997, Iterative Methods for Solving Linear Systems, *Cauchy Interlace Theorem*, pp. 18, SIAM, Philadelphia.

[4] **Hoffman, J.D.**, 2001, Numerical Methods for Engineers and Scientists, Marcel Dekker Inc.

[5] **Kincaid, D. and Cheney, W.**, 1991, Numerical Analysis Mathematics of Scientific Computing, *LU Factorizations*, pp. 129, *Selected Topics in Numerical Linear Algebra,* , pp. 232, *Localizing Eigevalues*, pp. 240, Brooks/Cole Pub. Co., California.

[6] **Koç, C.**, 1996, Topics in Linear Algebra, METU Press, Ankara.

[7] **Koç, C.**, 1996, Basic Linear Algebra, METU Press, Ankara.

[8] **Kreyszig, E.**, 2001, Advanced Engineering Mathematics,*Cramer's Rule*, pp. 347, *Similarity of Matrices Basis of Eigenvectors Diagonalization*, pp. 392-393, John Wiley & Sons Inc., Singapore.

[9] **Li, G., Artamonov, M., Rabitz, H., Wang, S.-W.,Georgopoulos, P.G. and Demiralp, M.** , 2003, High Dimensional Model Representations Generated from Low Order Terms ? Ip-RS-HDMR, *Journal of Computational Chemistry*, **24**, 647-656.

[10] **Li, G., , Rosenthal, C. and Rabitz, H.**, 2001, High Dimensional Model Representations, *Journal of Physical Chemistry*, A, **105**, 7765-7777.

[11] **Li, G., Wang, S-W., Rosenthal, C. and Rabitz, H.**, 2001, High Dimensional Model Representations generated from low dimensional data samples I.mp-Cut-HDMR, *Journal of Mathematical Chemistry*, **30**.

[12] **Mathews, J.H.**, 1992, Numerical Methods for Mathematics, Science, and Engineering, *The Secant Method*, pp. 81, Prentice-Hall International, Englewood Cliffs, New Jersey.

[13] **Rabitz, H. and Alış, Ö.**, 1999, General foundations of High Dimensional Model Representations, *Journal of Mathematical Chemistry*, **25**, 197-233.

[14] **Rabitz, H. and Alış, Ö.F.**, 2001, Efficient Implementation of High Dimensional Model Representations, *Journal of Mathematical Chemistry*, Vol **29**, 127-142.

[15] **Sehmi, N.S.**, 1989, Large Order Structural Eigenanalysis Techniques, *The Gram-Schmidt Orthogonalization Process*, pp. Appendix 2, John Wiley & Sons, New York.

[16] **Shorter, J.A., Rabitz, H. and Ip, P.C.**, 1999, An efficient chemical kinetics solver using High Dimensional Model Representation, *Journal of Physical Chemistry*, A, **103**, 7192-7198.

[17] **Sobol, I.M.**, 1993, Sensitivity estimates for nonlinear mathematical models, *MMCE*, **1**, 407.

[18] **Thomas, G.B. and Finney, R.L.**, 1988, Calculus and Analytic Geometry, *Integrals involving* ax^2+bx+c , pp. 477-478, Addison Wesley, Massachusetts.

APPENDIX A

Cramer's Rule

Cramer's Rule gives solutions of linear systems as quotients of determinants. Cramer's Rule is not practical in computations, but is of theoretical interest in differential equations and other theories that have engineering applications [8].

THEOREM 7 Cramer's Theorem (Solutions of linear systems by determinant) If a linear system of n equations in the same number of unknowns $x_1, \ldots, x_n$

$$\begin{array}{rcl} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n & = & b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{23}x_n & = & b_2 \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots & \cdots & \cdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n & = & b_n \end{array} \tag{A.1}$$

has a nonzero coefficient determinant $D = \det(\mathbf{A})$, the system has precisely one solution. This solution is given by the formulas

$$x_1 = \frac{D_1}{D}, \quad x_2 = \frac{D_2}{D}, \quad \cdots, \quad x_n = \frac{D_n}{D}, \qquad \textit{(Cramers Rule)} \tag{A.2}$$

where D_k is the determinant obtained from D by replacing in D the kth column by the column with the entries $b_1, \ldots, b_n$

PROOF The augmented matrix $\tilde{\mathbf{A}}$, which is obtained by adding the column with the entries $b_1, \ldots, b_n$ to the matrix $\mathbf{A}$, of the system (A.1) is of size $n \times (n+1)$. Hence its rank can be at most n. Now, if $D = \det(\mathbf{A} \neq 0)$, then rank $(\mathbf{A}) = n$. Thus rank $\tilde{\mathbf{A}}$ =rank $\mathbf{A}$. Hence, the system has a unique solution.

Expanding D by its kth column, we obtain,

$$D = a_{1k}C_{1k} + a_{2k}C_{2k} + \cdots + a_{nk}C_{nk}, \tag{A.3}$$

where C_{ik} is the cofactor of the entry a_{ik} in D. $C_{ik} = (-1)^{i+k}M_{ik}$ where M_{ik} is the determinant of order $n-1$, namely, the determinant of the submatrix of $\mathbf{A}$ obtained from $\mathbf{A}$ by deleting the row and the column of the entry a_{ik} (the

ith row and kth column). If we replace the entries in the kth column of D by any other numbers, we obtain a new determinant, say, $\overline{D}$. Clearly, its expansion by the kth column will be of the form (A.3), with $a_{1k}, \ldots, a_{nk}$ replaced by those new numbers and the cofactors C_{ik} as before. In particular, if we choose as new numbers the entries $a_{1l}, \ldots, a_{nl}$ in the lth column of D (where $l \neq k$), then the expansion of the resulting determinant $\overline{D}$ becomes

$$a_{1l}C_{1k} + a_{2l}C_{2k} + \cdots + a_{nl}C_{nk} = 0, \tag{A.4}$$

because $\overline{D}$ has two identical columns and is zero. We now multiply the first equation in (A.1) by C_{1k} on both sides, the second one by $C_{2k}, \cdots$, the last by C_{nk}, and add the resulting equations. This gives

$$C_{1k}(a_{11}x_1 + \cdots + a_{1n}x_n) + \cdots + C_{nk}(a_{n1}x_1 + \cdots + a_{nn}x_n) = b_1C_{1k} + \cdots + b_nC_{nk}. \tag{A.5}$$

Collecting the terms with the same x_j, we can write the left side as

$$x_1(a_{11}C_{1k} + a_{21}C_{2k} + \cdots + a_{n1}C_{nk}) + \cdots + x_n(a_{1n}C_{1k} + a_{2n}C_{2k} + \cdots + a_{nn}C_{nk}).$$

From here, it can be seen that x_k is multiplied by

$$a_{1k}C_{1k} + a_{2k}C_{2k} + \cdots + a_{nk}C_{nk}. \tag{A.6}$$

Equation (A.3) shows that this equals D. Similarly, x_l is multiplied by

$$a_{1l}C_{1k} + a_{2l}C_{2k} + \cdots + a_{nl}C_{nk}. \tag{A.7}$$

Equation (A.4) shows that this is zero when $l \neq k$. Accordingly, the left side of (A.5) equals simply x_kD, so that (A.5) becomes

$$x_kD = b_1C_{1k} + b_2C_{2k} + \cdots + b_nC_{nk}. \tag{A.8}$$

Now, the right side of this is D_k, expanded by its kth column, so that division by D gives Cramer's Rule which is given is (A.2).

□

APPENDIX B

LU Process for Solving Linear Systems

Suppose that $\mathbf{A}$ can be written as a product of a lower triangular matrix $\mathbf{L}$ and an upper triangular matrix $\mathbf{U}$, $\mathbf{A} = \mathbf{LU}$. Then, in order to solve the system of equations $\mathbf{A}x = \mathbf{b}$, it is enough to solve this problem in two stages:

$$\begin{aligned} \mathbf{Lz} &= \mathbf{b} \qquad \text{solve for } \mathbf{z}; \\ \mathbf{Ux} &= \mathbf{z} \qquad \text{solve for } \mathbf{x}. \end{aligned} \tag{B.1}$$

Solving these two equations is simpler than finding the inverse of $\mathbf{A}$.

We begin with an $n \times n$ matrix $\mathbf{A}$ and try to find two matrices

$$\mathbf{L} = \begin{pmatrix} l_{11} & 0 & 0 & \cdots & 0 \\ l_{21} & l_{22} & 0 & \cdots & 0 \\ l_{31} & l_{32} & l_{33} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ l_{n1} & l_{n2} & l_{n3} & \cdots & l_{nn} \end{pmatrix} \qquad \mathbf{U} = \begin{pmatrix} u_{11} & u_{12} & u_{13} & \cdots & u_{1n} \\ 0 & u_{22} & u_{23} & \cdots & u_{2n} \\ 0 & 0 & u_{33} & \cdots & u_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & u_{nn} \end{pmatrix} \tag{B.2}$$

such that $\mathbf{A} = \mathbf{LU}$. When this is possible, it is said that the matrix $\mathbf{A}$ has an *LU-decomposition*.

To derive an algorithm for the LU-factorization of $\mathbf{A}$, we start with the formula for the matrix multiplication:

$$a_{ij} = \sum_{s=1}^{n} l_{is} u_{sj} = \sum_{s=1}^{\min(i,j)} l_{is} u_{sj}. \tag{B.3}$$

Here we have used the fact that $l_{is} = 0$ for $s > i$ and $u_{sj} = 0$ for $s > j$.

Each step in this process determines a new row of $\mathbf{U}$ and one new column of $\mathbf{L}$. At step k, we can assume that the rows $1, 2, \ldots, k-1$ have been computed in $\mathbf{U}$, and the columns $1, 2, \ldots, k-1$ have been computed in $\mathbf{L}$. Putting $i = k = j$ in (B.3) we obtain

$$a_{kk} = \sum_{s=1}^{k-1} l_{ks} u_{sk} + l_{kk} u_{kk}. \tag{B.4}$$

If u_{kk} or l_{kk} has been specified, we use (B.4) to determine the other, with l_{kk} and u_{kk} now known, we use (B.3) to write for the kth row ($i = k$) and the kth column ($j = k$), respectively,

$$a_{kj} = \sum_{s=1}^{k-1} l_{ks}u_{sj} + l_{kk}u_{kj} \quad \text{k} + 1 \leq \text{j} \leq \text{n} \tag{B.5}$$

$$a_{ik} = \sum_{s=1}^{k-1} l_{is}u_{sk} + l_{ik}u_{kk} \quad \text{k} + 1 \leq \text{i} \leq \text{n}. \tag{B.6}$$

If $l_{kk} \neq 0$, equation (B.5) can be used to obtain the elements u_{kj}. Similarly, if $u_{kk} = 0$, equation (B.6) can be used to obtain the elements l_{ik}.

The algorithm based on the preceding analysis is known as *Doolittle's factorization* when $\mathbf{L}$ is unit triangular, i.e., ($l_{ii} = 1$ for $1 \leq i \leq n$) and as *Crout's factorization* when $\mathbf{U}$ is unit upper triangular, i.e, ($u_{ii} = 1$ for $1 \leq i \leq n$). When $\mathbf{U} = \mathbf{L}^T$ so that $l_{ii} = u_{ii}$ for $1 \leq i \leq n$, the algorithm is called *Cholesky's factorization*.

The pseudo code for carrying out Doolittle's factorization is as follows:

input n, a_{ij}
for $k = 1, 2, \ldots, n$ do
  $l_{kk} \longleftarrow 1$
  for $j = k, k+1, \ldots, n$ do
    $u_{kj} \longleftarrow a_{kj} - \sum_{s=1}^{k-1} l_{ks}u_{sj}$
  end
  for $i = k+1, k+2, \ldots, n$ do
    $l_{ik} \longleftarrow \left(a_{ik} - \sum_{s=1}^{k-1} l_{is}u_{sk}\right) \Big/ u_{kk}$
  end
end
output $(l_{ij}), (u_{ij})$

Example 11 Doolittle Factorization

Find the Doolittle factorization of the given matrix

$$\mathbf{A} = \begin{pmatrix} 60 & 30 & 20 \\ 30 & 20 & 15 \\ 20 & 15 & 12 \end{pmatrix}.$$

Doolittle factorization from the algorithm is

$$\mathbf{A} = \begin{pmatrix} 1 & 0 & 0 \\ \frac{1}{2} & 1 & 0 \\ \frac{1}{3} & 1 & 1 \end{pmatrix} \begin{pmatrix} 60 & 30 & 20 \\ 0 & 5 & 5 \\ 0 & 0 & \frac{1}{3} \end{pmatrix} = \mathbf{LU}. \tag{B.7}$$

After finding the LU decomposition of $\mathbf{A}$, we are ready to solve the system $\mathbf{Ax} = \mathbf{b}$, where $\mathbf{A}$ is the coefficient matrix. The following procedure is applied in order to find the solution $\mathbf{x}$ of the system.

$$\begin{aligned} \mathbf{Ax} = \mathbf{LUx} = \mathbf{L}(\mathbf{Ux}) &= \mathbf{Lz} = \mathbf{b} \\ \mathbf{Lz} &= \mathbf{b} \quad \text{solve for } \mathbf{z}; \\ \mathbf{Ux} &= \mathbf{z} \quad \text{solve for } \mathbf{x}. \end{aligned}$$

Example 12 Solution of a system by LU method

Solve the system

$$\begin{pmatrix} 60 & 30 & 20 \\ 30 & 20 & 15 \\ 20 & 15 & 12 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 110 \\ 65 \\ 47 \end{pmatrix} \tag{B.8}$$

by using LU decomposition.

From the previous example we know that the LU decomposition of $\mathbf{A} = \begin{pmatrix} 60 & 30 & 20 \\ 30 & 20 & 15 \\ 20 & 15 & 12 \end{pmatrix}$. So, firstly we will find $\mathbf{z}$ applying the forward substitution to the system

$$\mathbf{Lz} = \begin{pmatrix} 1 & 0 & 0 \\ \frac{1}{2} & 1 & 0 \\ \frac{1}{3} & 1 & 1 \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \\ z_3 \end{pmatrix} = \begin{pmatrix} 110 \\ 65 \\ 47 \end{pmatrix} = \mathbf{b} \Longrightarrow \mathbf{z} = \begin{pmatrix} 110 \\ 10 \\ \frac{1}{3} \end{pmatrix}. \tag{B.9}$$

Now, we will find the solution x by applying backward substitution to the system

$$\mathbf{Ux} = \begin{pmatrix} 60 & 30 & 20 \\ 0 & 5 & 5 \\ 0 & 0 & \frac{1}{3} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 110 \\ 10 \\ \frac{1}{3} \end{pmatrix} = \mathbf{z} \Longrightarrow \mathbf{x} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}. \tag{B.10}$$

APPENDIX C

Secant Method

Secant method is an iterative method used for solving the roots of a function $f = f(x)$ [12]. The Newton-Raphson algorithm, given as

$$x_k = g(x_{k-1}) = x_{k-1} - \frac{f(x_{k-1})}{f'(x_{k-1})}, \quad g(x) = x - \frac{f(x)}{f'(x)}, \tag{C.1}$$

requires the evaluation of two functions per iteration, $f(x_{k-1})$ and $f'(x_{k-1})$. If they are not complicated expressions, the method is desirable. In some cases it may require a considerable amount of effort to use the rules of calculus and derive the formula for $f'(x)$ from $f(x)$. Hence it is desirable to use a method that converges almost as fast as Newton's method yet involves only one evaluations of $f(x)$.

Two initial points $(x_0, f(x_0))$ and $(x_1, f(x_1))$ near the point $(x, 0)$ are needed, as shown in Figure F.1. Define x_2 to be the point of intersection of the line through

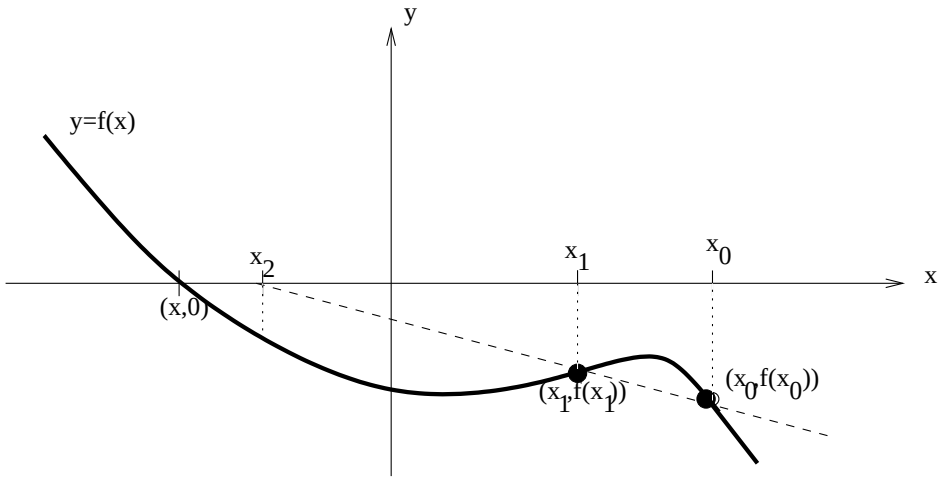

Figure C.1: Geometric construction of x_2 for the secant method

these two points; the Figure F.1 shows that x_2 will be closer to x than to either x_0 or x_1. The equation relating x_2, x_1 and x_0 is found by considering the slope

$$m = \frac{f(x_1) - f(x_0)}{x_1 - x_0} = \frac{0 - f(x_1)}{x_2 - x_1}. \tag{C.2}$$

The values of m in (C.2) are the slopes of the secant line through the first two approximations and the slope of the line through $(x_1, f(x_1))$ and $(x_2, 0)$, respectively. Solve for $x_2 = g(x_1, x_0)$ and get

$$x_2 = g(x_1, x_0) = p_1 - \frac{f(x_1)(x_1 - x_0)}{f(x_1) - f(x_0)}. \tag{C.3}$$

The general term is given by the two-point iteration formula

$$x_{k+1} = g(x_k, x_{k-1}) = x_k - \frac{f(x_k)(x_k - x_{k-1})}{f(x_k) - f(x_{k-1})}. \tag{C.4}$$

Example 13 Secant Method

Start with the points $x_0 = -2.6$ and $x_1 = -2.4$ to find the root $x = -2$ of the polynomial function $f(x) = x^3 - 3x + 2$.

The iteration formula is given as

$$x_{k+1} = g(x_k, x_{k-1}) = x_k - \frac{(x_k^3 - 3x_k + 2)(x_k - x_{k-1})}{x_k^3 - x_{k-1}^3 - 3x_k + 3x_{k-1}}. \tag{C.5}$$

This can be algebraically manipulated to obtain

$$x_{k+1} = g(x_k, x_{k-1}) = \frac{x_k^2 x_{k-1} + x_k x_{k-1}^2 - 2}{x_k^2 + x_k x_{k-1} + x_{k-1}^2 - 3}, \tag{C.6}$$

and the sequence of the iterations is given in the Table F.1.

Table C.1: Convergence of the Secant Method at a Simple Root

k	x_k	$x_{k+1} - x_k$	$e_k = x - x_k$
0	-2.600000000	0.200000000	0.600000000
1	-2.400000000	0.293401015	0.400000000
2	-2.106598985	0.083957573	0.106598985
3	-2.022641412	0.021130314	0.022641412
4	-2.001511098	0.001488561	0.001511098
5	-2.000022537	0.000022515	0.000022537
6	-2.000000022	0.000000022	0.000000022
7	-2.000000000	0.000000000	0.000000000

APPENDIX D

Gram-Schmidt Orthogonalization Process

The Gram-Schmidt Orthogonalization Process (GSOP) is a procedure of constructing a set of orthogonal vectors from any given set of linearly independent vectors ([6], [7], [15]). The process is illustrated in two dimensions by orthogonalizing two independent vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ so that they form an orthonormal pair $\mathbf{y}_1$ and $\mathbf{y}_2$.

To obtain $\mathbf{y}_1$, one simply normalizes the first vector $\mathbf{x}_1$ so that

$$\mathbf{y}_1 = \frac{\mathbf{x}_1}{\|\mathbf{x}_1\|} \quad \text{where} \quad \|\mathbf{x}_1\| = \sqrt{(\mathbf{x}_1^T \mathbf{x}_1)}. \tag{D.1}$$

For $\mathbf{y}_2$, consider the pictorial representation of vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ in Figure G.1: where θ is the angle that separates them. The direction of $\mathbf{y}_2$ has to be

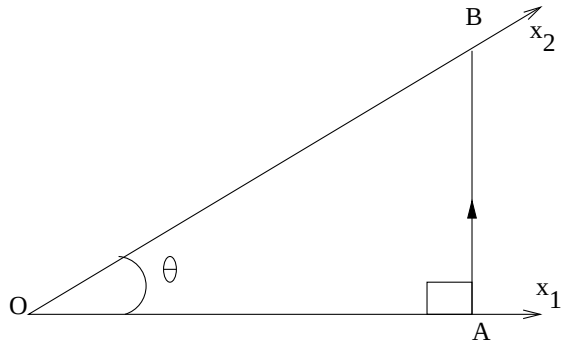

Figure D.1: Orthogonalization of vectors in $\mathbb{R}^2$

parallel to $\vec{AB}$, which in terms of the above diagram is given by

$$\vec{AB} = \vec{OB} - \vec{OA}. \tag{D.2}$$

Now, $\vec{OA}$ is the vector projection of $\mathbf{x}_2$ onto $\mathbf{y}_1$ and can be expressed as

$$\vec{OA} = \|\vec{OB}\| \cos(\theta) \frac{\vec{OA}}{\|\vec{OA}\|} \equiv \|\mathbf{x}_2\| \left(\frac{\mathbf{x}_1^T \mathbf{x}_2}{\|\mathbf{x}_1\| \|\mathbf{x}_2\|} \right) \frac{\mathbf{x}_1}{\|\mathbf{x}_1\|}, \tag{D.3}$$

which simplifies to

$$\vec{OA} = (\mathbf{y}_1^T \mathbf{x}_2) \mathbf{y}_1. \tag{D.4}$$

Substituting for $\vec{OA}$ and $\vec{OB}$ in equation (D.2) gives

$$\vec{AB} \equiv \hat{\mathbf{y}}_2 = \mathbf{x}_2 - (\mathbf{y}_1^T \mathbf{x}_2)\mathbf{y}_1, \tag{D.5}$$

and hence

$$\mathbf{y}_2 = \frac{\hat{\mathbf{y}}_2}{\|\hat{\mathbf{y}}_2\|}, \tag{D.6}$$

from which it is straightforward to show that

$$\mathbf{y}_1^T \mathbf{y}_1 = \mathbf{y}_2^T \mathbf{y}_2 = 1 \quad \text{and} \quad \mathbf{y}_1^T \mathbf{y}_2 = 0. \tag{D.7}$$

In general, if $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m)$ are a set of m linearly independent vectors each of with dimension n, then an orthonormal set $(\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_m)$ can be performed as

$$\begin{aligned} &\text{for} \quad i = 1, 2, \ldots, m \\ &\quad \hat{\mathbf{y}}_i = \mathbf{x}_i - \sum_{j=1}^{i-1} (\mathbf{y}_j^T \mathbf{x}_i)\mathbf{y}_j \qquad (\text{note } \hat{\mathbf{y}}_1 = \mathbf{x}_1) \\ &\quad \mathbf{y}_i = \frac{\hat{\mathbf{y}}_i}{\|\hat{\mathbf{y}}_i\|} \\ &\text{end} \end{aligned} \tag{D.8}$$

In matrix notation the GSOP is often written as

$$\mathbf{X} = \mathbf{YR}, \tag{D.9}$$

where $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m]$, $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_m]$ and $\mathbf{R}$ is an $m \times m$ upper triangular matrix whose elements are given by

$$r(i,i) = \|\hat{\mathbf{y}}_i\|, \;\; i = 1, 2, \ldots, m \quad \text{and} \quad r(i,j) = (\mathbf{y}_j^T \mathbf{x}_i), \;\; j = 1, 2, \ldots, i-1 \tag{D.10}$$

APPENDIX E

PROOFS OF THE THEOREMS

E.1 Proof of Theorem 1

PROOF Suppose that the conclusion is false. Let r be the largest integer such that $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_r\}$ is a linearly independent set. Then, $r < k$ and the set $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{r+1}\}$ is linearly dependent. Therefore, there are scalars c_1, $c_2, \ldots,$ c_{r+1}, not all zero, such that

$$c_1\mathbf{x}_1 + c_2\mathbf{x}_2 + \cdots + c_{r+1}\mathbf{x}_{r+1} = 0. \tag{E.1}$$

Multiplying both sides by $\mathbf{A}$ and using $\mathbf{A}\mathbf{x}_j = \lambda_j\mathbf{x}_j$, we obtain

$$c_1\lambda_1\mathbf{x}_1 + c_2\lambda_2\mathbf{x}_2 + \cdots + c_{r+1}\lambda_{r+1}\mathbf{x}_{r+1} = 0. \tag{E.2}$$

In order to get rid of the last term, we will multiply the first equation by λ_{r+1} and subtract from second one and get

$$c_1(\lambda_1 - \lambda_{r+1})\mathbf{x}_1 + c_2(\lambda_2 - \lambda_{r+1})\mathbf{x}_2 + \cdots + c_n(\lambda_n - \lambda_{r+1})\mathbf{x}_r = 0. \tag{E.3}$$

Here, $c_1(\lambda_1 - \lambda_{r+1}) = 0, \cdots, c_r(\lambda_r - \lambda r + 1) = 0$, since $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ is linearly independent. Hence, $c_1 = c_2 = \cdots = c_r = 0$, since eigenvalues are distinct. But this reduces to $c_{r+1}\mathbf{x}_{r+1} = 0$, hence $c_{r+1} = 0$, since $\mathbf{x}_{r+1} \neq 0$. This contradicts the fact that not all scalars are zero. Hence the conclusion of the theorem must hold.

□

E.2 Proof of Theorem 2

PROOF From Theorem 1, corresponding eigenvectors form a linearly independent set. Since eigenvectors are linearly independent, any vector can be written as a linear combination of this vectors. So eigenvectors form a basis for $\mathbb{R}^n$.

□

E.3 Proof of Theorem 3

PROOF $(1,2)$:Let $\mathbf{Ax} = \lambda\mathbf{x}$ with $\mathbf{x} \neq \mathbf{0}$, since $\mathbf{x}$ is an eigenvector. Then

$$\mathbf{x} = \mathbf{A}^{-1}(\lambda\mathbf{x}) = \lambda\mathbf{A}^{-1}\mathbf{x}. \tag{E.4}$$

Hence

$$\mathbf{A}^{-1}\mathbf{x} = \lambda^{-1}\mathbf{x}, \tag{E.5}$$

where λ^{-1} is the eigenvalue of $\mathbf{A}^{-1}$ and $\mathbf{x}$ the corresponding eigenvector.
$(3,4)$: Let $\mathbf{Ax} = \lambda\mathbf{x}$ with $\mathbf{x} \neq \mathbf{0}$, since $\mathbf{x}$ is an eigenvector. Then

$$\begin{aligned} &\mathbf{Ax} = \lambda\mathbf{x} \\ &\mathbf{Ax} + s\mathbf{x} = \lambda\mathbf{x} + s\mathbf{x} \\ &(\mathbf{A} + s\mathbf{I})x = (\lambda + s)\mathbf{x}. \end{aligned} \tag{E.6}$$

So, $\lambda + s$ is the eigenvalue of the shifted matrix $\mathbf{A} + s\mathbf{I}$ and the corresponding eigenvector is again $\mathbf{x}$.

□

E.4 Proof of Theorem 4

PROOF From

$$\mathbf{Bx} = \lambda\mathbf{x}, \tag{E.7}$$

where λ eigenvalue and $\mathbf{x} \neq 0$ multiplying by $\mathbf{P}^{-1}$, we get

$$\mathbf{P}^{-1}\mathbf{Bx} = \mathbf{P}^{-1}\mathbf{BIx} = \mathbf{P}^{-1}\mathbf{BPP}^{-1}\mathbf{x} = \mathbf{AP}^{-1}\mathbf{x} = \lambda\mathbf{P}^{-1}\mathbf{x}. \tag{E.8}$$

Hence λ is an eigenvalue and $\mathbf{P}^{-1}\mathbf{x}$ is the corresponding eigenvector of $\mathbf{A}$.

□

E.5 Proof of Theorem 5

PROOF Let λ be any eigenvalue of $\mathbf{A}$ and $\mathbf{x} = [x_1 \quad x_2 \quad \cdots \quad x_n]$ the corresponding eigenvector and $\|\mathbf{x}\|_\infty = max\{|a_i| : i = 1, 2, \ldots, n\} = 1$. Let i be the index for which $|a_i| = 1$. Since $(\mathbf{Ax})_i = x_i$ we have

$$\lambda x_i = \sum_{j=1}^{n} a_{ij}x_j. \tag{E.9}$$

Therefore

$$(\lambda - a_{ii})x_i = \sum_{\substack{j=1 \\ j \neq i}}^{n} a_{ij}x_j. \tag{E.10}$$

Taking the absolute values, using the triangular inequality and using $|x_j| \leq |x_i| = 1$, we have

$$|\lambda - a_{ii}| \leq \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}||x_j| \leq \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}|. \tag{E.11}$$

Thus $\lambda \in D_i$.

□

APPENDIX F

MuPAD PROGRAM FOR ZEROTH ORDER CONSTANT HDMR COMPONENT

```
/*************************************************************/
/**** program that computes the constant HDMR component */
N:=10: ai:=1: bi:=5:
A:=matrix(N,N):
for iii from 1 to N do
for jjj from 1 to N do
if (iii=jjj) then A[iii,jjj]:=1/(iii+jjj);
end_if end_for; end_for;
A;
/*************************************************************/
/*definition of standard unit matrices*/
e :=proc(m)
begin
e_ := matrix(N,1);
for i from 1 to N do
     if (i <> m) then e_[i,1] :=0;
     else e_[i,1]:=1;
     end_if; end_for; e_;
end_proc
/*************************************************************/
/*definition of u=[1 1 1 1 ...]^T*/
u:=proc(m)
begin
u_:=matrix(N,1);
for i from 1 to N do
         u_[i,1]:=1;
end_for; u_;
end_proc
/*************************************************************/
/* Identity matrix*/
Iden:=proc()
```

```
begin
Id_:=matrix(N,N);
for i from 1 to N do
for j from 1 to N do
if (i=j) then Id_[i,j]:=1;
end_if; end_for; end_for;
Id_;
end_proc
idenmat:=Iden():
/*****************************************************************/
//Vector for integral intervals [ai,bi]
a:=array(1..N*N):
b:=array(1..N*N):
for i from 1 to N*N do
a[i]:=ai; b[i]:=bi;
end_for;
for k from 0 to N-1 do
a[k*N+k+1]:=0; b[k*N+k+1]:=1;
end_for
print("Integral intervals");
a; b;
/*****************************************************************/
/*Ai */
Ai:=proc(m)
begin
Ai_:=matrix(N,N);
if (m mod N <> 0) then Ai_[ceil(m/N),(m mod N)]:=1;
else Ai_[ceil(m/N),N]:=1; end_if; Ai_;
end_proc;
/*****************************************************************/
/* WEIGHT FUNCTIONS */
//Value of the weight function
W:=proc(a,b)
begin
   W_:=1/(b-a);
end_proc;
//Matrix of the Weight functions
Wt:=proc()
begin
Wt_:=matrix(N,N):
for i from 1 to N do
```

```
for j from 1 to N do
if (i<>j) then Wt_[i,j]:=W(a[(i-1)*N+j],b[(i-1)*N+j]);
else  t:=x.expr2text((i-1)*N+j); Wt_[i,j]:=2*t;
end_if; end_for; end_for; Wt_;
end_proc:
print("Matrix of the weight functions ");
Wt();
/*********************************************************************/
/*********************************************************************/
/* Matrix Thetai=int_{ai}^{bi} Ai xi Wi(xi)*/
thetai:=proc(i1)
begin
thetai_:=matrix(N,N):
ii:=ceil(i1/N):
jj:=i1 mod N:
if (jj=0) then tt:=x.expr2text(i1):
thetai_:=Ai(i1)*numeric::int(Wt()[ii,N]*tt,tt=a[i1]..b[i1]);
else tt:=x.expr2text(i1):
thetai_:=Ai(i1)*numeric::int(Wt()[ii,jj]*tt,tt=a[i1]..b[i1]);
end_if; thetai_;
end_proc
/*********************************************************************/
/* SUM OF ALL THETA_I'S  */
thetaisum:=matrix(N,N):
for i4 from 1 to N*N do
thetaisum:=thetaisum+thetai(i4);
end_for;
print("Sum of all Thetai functions"); thetaisum;
/*********************************************************************/
/* DETERMINATION OF ZEROTH ORDER HDMR APPROXIMATION */
eigval:=linalg::eigenvalues(thetaisum,Multiple);
eigvec:=numeric::eigenvectors(thetaisum);
/*********************************************************************/
A_thetaisum:=matrix(N,N):
A_thetaisum:=A-thetaisum:
normDifference:=norm(A_thetaisum,Frobenius):
print("norm of A - Thetaisum=",normDifference);
normA:=norm(A,Frobenius):
print("norm of the original matrix=",normA);
sigma0:=normDifference/normA:
print("sigma0=",simplify(sigma0));
```

```
print("first eigenvalue=",op(eigvec,[1,1]));
print("first eigenvector"); u001:=matrix(N,1):
u001[1,1]:=op(eigvec,[2,1]): u001[2,1]:=op(eigvec,[2,4]):
u001[3,1]:=op(eigvec,[2,7]): u001;
print("second eigenvalue=",op(eigvec,[1,2]));
print("second eigenvector");  u002:=matrix(N,1):
u002[1,1]:=op(eigvec,[2,2]): u002[2,1]:=op(eigvec,[2,5]):
u002[3,1]:=op(eigvec,[2,8]): u002;
print("third eigenvalue=",op(eigvec,[1,3]));
print("third eigenvector"); u003:=matrix(N,1):
u003[1,1]:=op(eigvec,[2,3]): u003[2,1]:=op(eigvec,[2,6]):
u003[3,1]:=op(eigvec,[2,9]): u003;
quit
```

APPENDIX G

MuPAD PROGRAM FOR FIRST ORDER HDMR APPROXIMATION

```
DIGITS:=15; N:=20; ai:=0:
/* matrix A */
A:=matrix(N,N):
for es from 1 to N do
  A[es,es]:=es+es;
end_for
A;
/* perturbation */
A[1,3]:=A[1,3]+0.001; A[2,17]:=A[2,17]+0.0005; A[3,9]:=A[3,9]+0.0002;
A[4,12]:=A[4,15]+0.003; A[5,18]:=A[5,18]+0.001; A[6,8]:=A[6,8]+0.01;
A[7,7]:=A[7,7]+0.05; A[8,19]:=A[8,19]+0.001; A[9,1]:=A[9,1]+0.002;
A[10,6]:=A[10,6]+0.001; A[11,13]:=A[11,13]+0.001; A[12,8]:=A[12,8]+0.0005;
A[13,20]:=A[13,20]+0.001; A[14,5]:=A[14,5]+0.002;  A[15,10]:=A[15,10]+0.0012;
A[16,2]:=A[16,2]+0.0015; A[17,11]:=A[17,11]+0.0013; A[18,16]:=A[18,16]+0.0025;
A[19,15]:=A[19,15]+0.0016; A[20,4]:=A[20,4]+0.001;
/* initial values for Secant Method */
lambda1:=9.99; lambda2:=10.1;
eps:=0.0000000001;  //for convergence
/*********************************************/
/* definition of standard unit matrices */
e :=proc(m)
begin
        e_ := matrix(N,1);
        for i from 1 to N do
             if (i <> m) then e_[i,1] :=0;
             else  e_[i,1]:=1; end_if; end_for; e_;
end_proc
/*************************************************/
/* definition of u=[1 1 1 1 ...]^T */
u:=proc(m)
begin
```

```
        u_:=matrix(N,1);
        for i from 1 to N do
                u_[i,1]:=1; end_for; u_;
end_proc
/***********************************************/
/* Identity matrix */
Iden:=proc()
begin
        Id_:=matrix(N,N);
        for i from 1 to N do
                for j from 1 to N do
                        if (i=j) then Id_[i,j]:=1;
                        end_if; end_for; end_for; Id_;
end_proc
idenmat:=Iden();
/***********************************************/
/* Vector for integral intervals [ai,bi]  */
a:=array(1..N*N):
b:=array(1..N*N):
for i from 1 to N do
for ii from 1 to N do
if (i=ii) then b[((i-1)*N)+ii]:=(2*(i+ii)+1)/2;
else b[((i-1)*N)+ii]:=1;
end_if; end_for; end_for;
for k from 1 to N do
for kk from 1 to N do
if (k=kk) then a[((k-1)*N)+kk]:=b[((k-1)*N)+kk]-1;
              else a[((k-1)*N)+kk]:=b[((k-1)*N)+kk]-2;
end_if; end_for; end_for;
a; b;
/***********************************************/
/* Matrix Ai */
Ai:=proc(m)
begin
        Ai_:=matrix(N,N);
        if (m mod N <> 0) then Ai_[ceil(m/N),(m mod N)]:=1;
        else Ai_[ceil(m/N),N]:=1; end_if; Ai_;
end_proc;
/***********************************************/
/* WEIGHT FUNCTIONS */
//Value of the weight function
```

```
W:=proc(a,b)
begin
   W_:=1;
end_proc;
//Matrix of the Weight functions
Wt:=proc()
begin
        Wt_:=matrix(N,N):
        for i from 1 to N do
                for j from 1 to N do
Wt_[i,j]:=W(a[(i-1)*N+j],b[(i-1)*N+j]);
                end_for end_for; Wt_;
end_proc:
Wt();
/***********************************************/
/* Matrix Thetai=int_{ai}^{bi} Ai xi Wi(xi)*/
thetai:=proc(i1)
begin
        thetai_:=matrix(N,N):
        ii:=ceil(i1/N):
        jj:=i1 mod N:
        if (jj=0) then tt:=x.expr2text(i1):
                thetai_:=Ai(i1)*numeric::int(Wt()[ii,N]*tt,tt=a[i1]..b[i1]);
        else tt:=x.expr2text(i1):
         thetai_:=Ai(i1)*numeric::int(Wt()[ii,jj]*tt,tt=a[i1]..b[i1]);
        end_if;thetai_;
end_proc
/***********************************************/
/* SUM OF ALL THETA_I'S  */
thetaisum:=matrix(N,N):
for i4 from 1 to N*N do
        thetaisum:=thetaisum+thetai(i4); end_for;
thetaisum;
/***********************************************/
/* DETERMINATION OF ZEROTH ORDER HDMR APPROXIMATION */
eigval:=linalg::eigenvalues(thetaisum,Multiple);
eigvec:=numeric::eigenvectors(thetaisum);
/***********************************************/
/* SUM OF THETA_I'S EXCLUDING THE ARGUMENT GIVEN */
thetaisumxk:=proc(xk)
begin
```

```
        thetaisumxk_:=matrix(N,N): thetaisumxk_:=thetaisum-thetai(xk); thetaisumxk_;
end_proc
/************************************************/
/* Matrix Thetai-lambda*I  */
mat:=proc(lamb2)
begin
mat_:=thetaisum-lamb2*idenmat; mat_;
end_proc
/* Inverse of (Thetai-lambda*I)  */
matinv:=proc(lamb3)
begin
matinv_:=linalg::inverseLU(mat(lamb3)); matinv_;
end_proc
/* Matriix M */
M:=proc(xk,lamb4)
begin
M_:=matrix(N,N): M_:=matinv(lamb4)*Ai(xk); M_;
end_proc
/* Constant sigma */
sigma:=proc(xk,lamb5)
begin
if (xk mod N <> 0) then row:=ceil(xk/N); col:=xk mod N;
else row:=ceil(xk/N); col:=N; end_if;
sigma_:=linalg::transpose(e(col))*matinv(lamb5)*e(row); sigma_;
end_proc
zet:=proc(xk,lamb10)
begin
if (xk mod N <> 0) then row2:=ceil(xk/N); col2:=xk mod N;
else row2:=ceil(xk/N); col2:=N; end_if;
val_:=(1+b[xk]-thetai(xk)[row2,col2])/(1+a[xk]-thetai(xk)[row2,col2]);
zet_:=(1/sigma(xk,lamb10))+Wt()[row2,col2]*(1/(sigma(xk,lamb10)*sigma(xk,lamb10)))
             *ln(abs(val_)); zet_;
end_proc
auxmat:=proc(xk1,lamb11)
begin
auxmat_:=matrix(N,N):
if (xk1 mod N <> 0) then row1:=ceil(xk1/N); col1:=xk1 mod N;
        else row1:=ceil(xk1/N); col1:=N; end_if;
val2:=zet(xk1,lamb11)*sigma(xk1,lamb11);
val3:=1-val2[1,1]; val4:=zet(xk1,lamb11)/(val3);
auxmat_:=mat(lamb11)*(idenmat-val4[1,1]*M(xk1,lamb11)) auxmat_;;
```

```
end_proc
auxmatsum:=proc(lamb7)
begin
auxmatsum_:=matrix(N,N):
for sn from 1 to N*N do
auxmatsum_:=auxmatsum_+auxmat(sn,lamb7);
end_for; auxmatsum_;
end_proc
coeffmat:=proc(lamb9)
begin
coeffmat_:=matrix(N,N):
coeffmat_:=-lamb9*idenmat-(1/(N*N-1))*auxmatsum(lamb9); coeffmat_;
end_proc;
/* Coefficient Matrix */
coeffmat1:=proc(lamb9)
begin
coeffmat1_:=matrix(N,N):
coeffmat1_:=thetaisum-(1/(N*N-1))*auxmatsum(lamb9); coeffmat1_;
end_proc;
/* SECANT METHOD */
it:=0;
repeat
        u01coeffmat1:=coeffmat(lambda1): u01coeffmat2:=coeffmat(lambda2):
        u01coeffmat11:=thetaisum-u01coeffmat1:
        u01coeffmat22:=thetaisum-u01coeffmat2:
        det1:=linalg::det(u01coeffmat11); det2:=linalg::det(u01coeffmat22);
        slope:=(det2-det1)/(lambda2-lambda1); lambda3:=lambda2-(det2/slope);
        print("lambda3",lambda3); dummy:=lambda3-lambda2;
        lambda1:=lambda2; lambda2:=lambda3;
        it:=it+1; print("dummy=",dummy);
until (abs(dummy)<=eps) end_repeat;

print("resulting lambda",lambda3); print("iteration number",it);
lambda3;
/*********************************************/
/* DETERMINATION OF u_0^1 in FIRST ORDER HDMR APPROXIMATION */
eigval:=linalg::eigenvalues(coeffmat1,Multiple);
eigvec:=numeric::eigenvectors(coeffmat1);
/*********************************************/
quit
```

BIOGRAPHY

Esen Akkemik was born in 1978, in Almus, Tokat. She received the high school diploma from Ankara Fethiye Kemal Mumcu Anatolian High School in 1996. She graduated from Middle East Technical University, Arts and Sciences Faculty, Department of Mathematics in 2001. She has worked as a research assistant in İstanbul Technical University. She has been a researcher in The Scientific And Technical Research Council of Turkey (TÜBİTAK), National Electronics and Cryptology Research Institute (UEKAE) since August, 2002.