Approximate Methods of Inverse Preconditioners for Solving the Linear Algebraic Systems

Soran Jalal Abdalla

Submitted to the
Institute of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Science
in
Mathematics

Eastern Mediterranean University
July 2014
Gazimağusa, North Cyprus

Approval of the Institute of Graduate Studies and Research

Prof. Dr. Elvan Yılmaz
Director

I certify that this thesis satisfies the requirements as a thesis for the degree of Master of Science in Mathematics.

Prof. Dr. Nazim Mahmudov
Chair, Department of Mathematics

We certify that we have read this thesis and that in our opinion; it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science in Mathematics.

Asst. Prof. Dr. Suzan Cival Buranay
Supervisor

Examining Committee

1. Prof. Dr. Adıgüzel Dosiyev
2. Assoc. Prof. Dr. Derviş Subaşı
3. Asst. Prof. Dr. Suzan Cival Buranay

ABSTRACT

The efficiency and robustness of iterative methods can be improved using a preconditioner that causes a change in the original matrix implicitly or explicitly. Usually preconditioners are constructed using the structure of the coefficient matrix. Therefore a preconditioner which works well for one class of matrices may fail to give good results for an other class.

The focus of this study is to analyze, the efficiency of approximate inverse preconditioners for solving linear systems that arises from the discretization of the Poisson equation on a rectangle with Dirichlete boundary conditions. To realize this first, geometric construction of second order and a class of third order iterative methods for approximating a simple root of the nonlinear equation $f(x) = 0$ are investigated. Then by the generalization of these methods to Banach spaces, and applying them to the equation $F(N) = N^{-1} - A \equiv 0$, Newton and Chebyshev iterative methods for matrix inversion are studied. These methods are applied to solve linear system of equations obtained from difference analog of Dirichlet problem of Laplace's equation on a rectangle. The research is proceeded with the numerical results achieved and some discussions are made based on these results.

Keywords: Chebyshev's method, approximate inverse preconditioner, finite difference scheme, Laplace equation.

ÖZ

Iteratif yöntemlerin verimlilik ve sağlamlığı kapalı veya açık olarak, orijinal matrisde bir değişime neden olan bir önkoşullandırıcı kullanılarak geliştirilebilir. Genellikle önkoşullandırıcılar katsayı matrisinin yapısı kullanılarak inşa edilir. Bu nedenle bir sınıf matrisler için iyi çalışan bir önkoşullandırıcı başka bir sınıf için iyi sonuçlar vermekte başarısız olabilir.

Bu çalışmanın odak noktası dikdörtgen üzerinde Dirichlet sınır koşullu Poisson denkleminin ayrıştırılması ile oluşan lineer sistemlerin çözümünde yaklaşık ters önkoşullandırıcıların etkinliğini analiz etmektir. Bunu ğerçekleştirmek için önce, $f(x)=0$ doğrusal olmayan denklemin basit bir kökünün yaklaşımında ikinci mertebeden ve üçüncü mertebeden olan bir sınıf iteratif yöntemlerinin geometrik oluşumu incelendi. Daha sonra bu yöntemlerin Banach uzaylarına genişletilmesi ve $F(N)=N^{-1}-A\equiv 0$ denklemine uygulanması ile Newton ve Chebyshev iteratif yöntemleri çalışıldı. Bu yöntemler Laplace denkleminin dikdörtgen üzerinde Dirichlet sınır koşullu probleminin farklar analoğundan elde edilen lineer denklem sistemini çözmek için uygulandı. Araştırma elde edilen sonuçlar ile ilerlendirildi ve bu sonuçlara dayanarak bazı değerlendirmeler yapıldı.

Anahtar kelimeler: Chebyshev yöntemi, yaklaşık ters önkoşollandırıcı, sonlu fark şemaları, Laplace denklemi.

ACKNOWLEDGEMENT

I would like to express my deep sense gratitude to my supervisor, Assist. Prof. Dr. Suzan Cival Buranay for her guidance, patience and understanding.

My sincere thanks are also due to Research Assist. Emine Celiker for having been very helpful on numerous occasions.

My thanks are also due to my entire family for having made all of this possible. I have had immeasurable support from my wife, Shaida, without whose encouragement, understanding and constant help I would not have complete this work.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

In scientific and engineering problems often arise large linear systems

$$Ax = b \tag{1.1}$$

where $x, b \epsilon \mathbb{R}^n$, and $A \epsilon \mathbb{R}^{n \times n}$. Direct solvers like Householder transformation, LU factorization, Gaussian elimination methods are preferred if reliability is important and huge amount of work and storage are needed, causing deficiency in the implementation. Preconditioned Krylov subspace methods are more efficient methods since they use a second matrix named as preconditioner that changes the coefficient matrix implicitly and explicitly causing a more preferable form. An effective preconditioner increases the rate of the used iterative algorithm, considerably. Moreover an iterative method may diverge if a preconditioner is not used. In general three types of preconditioners for a matrix A are constructed:

(1) Implicit methods

(2) Explicit methods

(3) Hybrid methods

(1) Implicit methods: In these methods, approximate inverse is built implicitly, and an approximate decomposition of A is formed. One of these implicit methods is incomplete LU factorization. The idea of an incomplete factorization was first given in [1]-[6]. The generalization of the method to matrices portioned in block matrix form is given in [7]. The incomplete LU factorization preconditioners were developed especially for M-matrices. Therefore the standard ILU factorization faces problems, when the coefficient matrix is indefinite.

Let $A = LU + E,$ and E be the defect matrix.

$$L^{-1}AU^{-1} = I + L^{-1}EU^{-1}$$

The matrices L^{-1} or U^{-1} may have very large norms causing very large perturbations when A is not diagonally dominant. In this case ILU decomposition is ineffective.

(2) Explicit methods: With explicit methods we compute an approximation M or explicit form of the inverse A^{-1} of a given nonsingular matrix A. In these methods to solve the linear system (1.1) the left preconditioning

$$MAx = Mb \tag{1.2}$$

can be used, combined with iteration. Since both M and A are explicitly available, each iteration requires, matrix vector multiplication. It is also possible to use right preconditioning system

$$AMy = b, \tag{1.3}$$

and iterate to get the vector y and then finally calculate

$$x = My. \tag{1.4}$$

Among many preconditioning methods to solve (1.1) a special emphasize is given on preconditioned conjugate gradient methods. The conjugate gradient method is proposed in the years 1940-1950. See Hestenes and Stiefel [8]. Conjugate gradient method minimizes quadratic functions such as $f(x) = \left(\frac{1}{2}x^T Ax\right) - b^T$, when A is positive definite and symmetric, or the residual function $f(x) = (Ax - b)^T(Ax - b)$ in general, and minimization takes place over Krylov spaces. If the matrix A in (1.1) is positive definite and symmetric then the conjugate gradient in preconditioned form can be constructed. It is obtained using M-inner product defined by the preconditioning matrix M which is symmetric, positive definite as

$$\langle x, y\rangle \equiv x^T My$$

and $M^{-1}A$ is self adjoint. Let h denote the pseudoresidual at the k^{th} iteration and $h^k = M^{-1}(Ax^k - b)$. (PCGM) with the next pseudo code presenting a partial description of the computer implementation is given in Owe Axelsson [9]

```
$x := x^0$ ; $r := Ax - b$ :
$h := M^{-1}r$ ;
$d := -h$ ; $\delta_0 := r^T h$ ;
$if\ \delta_0 \le \varepsilon\ \ then\ stop$ ;
$S$:   $h := Ad$ ;                                        (1.5)
$r := \frac{\delta_0}{d^T h}$ ;
$x: = x + \tau d$ ;
$r := r + \tau h$ ;
$h := M^{-1}r$ ;
$\delta_1 := r^T h$ ;
```

```
$if\ \delta_1 \leq \varepsilon\ \ then\ stop$ ;

$\beta := \frac{\delta_1}{\delta_0}$ ;

$\delta_0 := \delta_1$ ;

$d := -h + \beta d$

$Go\ to\ S$
```

An other preconditioned method is given by Lanczos [10], [11] which generates A-orthogonal or M-orthogonal vectors to solve linear system (1.1) when A is positive definite and symmetric matrix. For Lanczos method with A-orthogonal vectors an inner product is selected as

$$\langle x, y \rangle \equiv x^T M y$$

for the symmetric positive definite matrix M, and in the algorithm (1.5) the A-orthogonal vectors d at the $k+1$ iteration are [9]

$$d^{k+1} = M^{-1} A d^k - r_k d^k - S_{k-1} d^{k-1}$$

where

$$r_k = \frac{(M^{-1} A d^k)^T A d^k}{d^{k^T} A d^k} \tag{1.6}$$

$$s_{k-1} = \frac{d^{k^T} A d^k}{d^{k-1^T} A d^{k-1}} \tag{1.7}$$

For the M-orthogonal version of the Lanczos method, the inner product is taken as

$$\langle x, y \rangle = x^T M A^{-1} M y$$

where M is symmetric and nonsingular and the recurrence coefficients (1.6), (1.7) takes the form [9]

$$r_k = \frac{{d^k}^T A d^k}{{d^k}^T M d^k} \quad , \qquad s_{k-1} = \frac{{d^k}^T M d^k}{{d^{k-1}}^T M d^{k-1}}$$

respectively.

The preconditioning of conjugate gradient method and Lanczos method can be implemented implicitly or explicitly based on the construction of M.

(3) Hybrid Methods: These methods are the combination of implicit and explicit methods. The combination of explicit and implicit preconditioners in the block matrix incomplete factorization method is an example for an Hybrid method. Such methods have been studied by Axelsson, Brink Kempler and ll' in [7] for block-tridiagonal M-matrices, and by Conass, Golub and Mevrent [12] and for matrices with a general sparsity pattern by Axelsson [13] and Axelsson and Polman [14]. An other example is incomplete factorization of an explicitly preconditioned matrix. In this method an explicit preconditioner M is calculated first, then an implicit preconditioner C to $I - E$ where $E = I - MA$ which yields the preconditioner $C^{-1}M$, is constructed. This method can be given in the following algorithm [9];

Step1: Compute an explicit preconditioner M, $(MA)_{i,j} = \delta_{i,j}$, $(i,j) \in S$ where S is a subset of the full set $\{(i,j): 1 \leq i \leq n, 1 \leq j \leq n\}$, and $m_{i,j} = 0$ for all index pairs outside the sparsity pattern S.

Step2: Compute an implicit preconditioner to $MA = I - E$. One may use block incomplete factorization of $I - E$

Step3: Solve the system (1.1) or $MAx = \bar{b}$ where $\bar{b} = Mb$, using an iterative method with the preconditioner for $I - E$.

We principally mentioned types of preconditioners. The main purpose of a preconditioning is to accelerate the convergence rate of an iterative method. Therefore in implementation, a preconditioned iteration should require less time than the unpreconditioned iteration. This could be done if the preconditioner is computed easily and the application is practical. However preconditioners generally suffers from some drawback;

1. It is hard to be sure that, a given algorithm will converge in a reasonable time, when faced with a new problem with different coefficient matrix A, satisfying the necessary conditions.
2. A successful preconditioner for one class of problem may be ineffective for an other class.

Therefore the effectiveness of a preconditioner strongly depends on the structure of the coefficient matrix A.

Recently constructing approximations N to the inverse of A are attracting attentions. There has been a lot of interest in such approximate matrix inversions [15]-[18], and in the use of them as inverse preconditioners.

In this project we investigate second and third order convergent approximate methods of inverse preconditioners in solving linear systems arising from difference analog of the Poisson equation on a rectangle with Dirichlet boundary conditions. The study is prepared as follows:

In Chapter 2, geometric construction of Newton method and a class of third order convergent methods are given for nonlinear equation $f(x) = 0$. These methods are extended to Banach spaces and applied to the equation $f(N) = N^{-1} - A \equiv 0$. As a result the Schulz iteration [19] also known as Newton method (NM)

$$N_{m+1} = N_m(2I - AN_m), \qquad m = 0,1,\dots, \tag{1.8}$$

and a third order convergent algorithm of Chebyshev's method (CM) by S. Amat and S. Busquier [20],

$$N_{m+1} = N_m(3I - AN_m(3I - AN_m)), \qquad m = 0,1,\dots \tag{1.9}$$

where N_0 is an initial approximation to A^{-1} are studied. Formulas (1.8), (1.9) are used to construct approximate method of inverse preconditioners in solving (1.1). Computer algorithms of (NM) and (CM) are also provided.

In Chapter 3, 5-point and 9-point difference analog of Dirichlet Poisson equation on a rectangle is given. The structure of the coefficient matrix for the obtained system of finite difference equations using Lexicographical ordering is analyzed.

In Chapter 4, a model problem is taken and the system of finite difference equations obtained from 5-point and 9-point schemes are solved using approximate methods of inverse preconditioners by both Newton and Chebyshev iterations for different mesh steps. Numerical results and discussions are given.

In Chapter 5, concluding remarks are provided based on the numerical results obtained in Chapter 4.

Chapter 2

GEOMETRIC CONSTRUCTION AND FORMULAS FOR METHODS OF APPROXIMATE INVERSE PRECONDITIONERS

2.1 Introduction

For a given nonsingular matrix A, an iterative matrix inversion scheme is a set of instructions for generating a sequence $\{N_m : m = 1,2,\ldots.\}$ converging to A^{-1}. These instructions should provide a way to select the initial approximation N_0 and specify how to improve the approximate inverse from N_m to N_{m+1} for each m. These schemes also need a stopping criterion to determine whether the desired inverse has been obtained. In this Chapter geometric construction of Newton and a class of third order convergent methods: Chebyshev [21], Halley [22] and Super Halley [23] methods are analyzed. Generalization of these methods to Banach Spaces for obtaining iterative matrix inversion algorithms for nonsingular matrices is investigated. Computer algorithms for Newton and Chebyshev iteration as approximate inverse preconditioning methods for solving $Ax = b$ where $x, b \in R^n$ and $A \in R^{n\times n}$ is nonsingular are provided.

2.2 Geometric Construction of Newton and a Class of Third Order Convergent Methods: Real Case

Consider the scalar nonlinear equation

$$f(x) = 0, \tag{2.1}$$

To get an iterate x_n approximating the simple root of (2.1) we use the tangent line

$$y(x) - f(x_n) = f'(x_n)(x - x_n) \tag{2.2}$$

at $(x_n, f(x_n)$. Evaluating (2.2) at the point $(x_{n+1}, 0)$ we get

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \quad , \qquad n = 0,1, \ldots \tag{2.3}$$

which has quadratic convergence at a simple root. There are considerably high number of works studying the convergence, characteristics of this iterative method, see [21], [24]. If the order of convergence is higher than the velocity of convergence is faster, however cost of computation may also be increased. This is the starting point to search high order convergent iterative methods with tolerable cost for computation. The following iterative schemes are constructed by taking tangent curves with quadratic equations, to the graph of f at $(x_n, f(x_n))$, as given in [20].

Consider the equation

$$axy + y + bx + c = 0, \tag{2.4}$$

which defines a hyperbola and take the tangency conditions

$$y(x_n) = f(x_n), \quad y'(x_n) = f'(x_n) \text{ and } y''(x_n) = f''(x_n). \tag{2.5}$$

We take (2.4) in the form

$$a(x - x_n)\big(y - f(x_n)\big) + \big(y - f(x_n)\big) + b(x - x_n) + c = 0. \tag{2.6}$$

From first condition in (2.5) results $c = 0$. Differentiating (2.6), we obtain

$$a\left[\big(y - f(x_n)\big) + (x - x_n)\frac{dy}{dx}\right] + \frac{dy}{dx} + b = 0. \tag{2.7}$$

Appling second condition of (2.5) on (2.7) at $(x_n, f(x_n))$

gives $b = -f(x_n)$. Differentiating (2.7) results in

$$a\left[2\frac{dy}{dx} + (x - x_n)\frac{d^2y}{dx^2}\right] + \frac{d^2y}{dx^2} = 0. \tag{2.8}$$

Imposing the conditions (2.5), on (2.8) the coefficient a satisfies

$$a[2f'(x_n)] + f''(x_n) = 0,$$

and

$$a = -\frac{f''(x_n)}{2f'(x_n)} \;.$$

Substituting the values of the unknown coefficients a, b, c into (2.6) follows

$$-\frac{f''(x_n)}{2f'(x_n)}(x - x_n)\big(y - f(x_n)\big) + y - f(x_n) - f'(x_n)(x - x_n) = 0. \tag{2.9}$$

Let the x-intersection point of the hyperbola (2.9) be $(x_{n+1}, 0)$ then

$$-\frac{f''(x_n)}{2f'(x_n)}(x_{n+1} - x_n)\big(-f(x_n)\big) - f(x_n) - f'(x_n)(x_{n+1} - x_n) = 0,$$

$$(x_{n+1} - x_n)\left[\frac{f''(x_n)}{2f'(x_n)}f(x_n) - f'(x_n)\right] - f(x_n) = 0,$$

solving for x_{n+1} , yields

$$x_{n+1} = x_n + \frac{f(x_n)}{\frac{f''(x_n)}{2f'(x_n)}f(x_n) - f'(x_n)} \;.$$

Let $L_f(x_n) = \frac{f''(x_n)f(x_n)}{f'(x_n)^2}$ which is called the Logarithmic convexity of f at x_n [25],

$$x_{n+1} = x_n + \frac{f(x_n)}{\frac{1}{2}L_f(x_n)f'(x_n) - f'(x_n)} = x_n + \frac{2f(x_n)}{[L_f(x_n) - 2]f'(x_n)},$$

$$x_{n+1} = x_n - \frac{2f(x_n)}{[2 - L_f(x_n)]f'(x_n)} \; , n = 0,1, \ldots \tag{2.10}$$

This iteration is called the Halley method. Geometric construction of Chebyshev algorithm is obtained from a parabola of the form

$$ay^2 + y + bx + c = 0.$$

Consider the parabola

$$a\big(y - f(x_n)\big)^2 + \big(y - f(x_n)\big) + b(x - x_n) + c = 0, \tag{2.11}$$

and apply the conditions (2.5) to get $c = 0$. From differentiating (2.11) one gets

$$2a\big(y - f(x_n)\big)\frac{dy}{dx} + \frac{dy}{dx} + b = 0 \; , \tag{2.12}$$

and imposing $y(x_n) = f(x_n)$ and $y'(x_n) = f'(x_n)$ we obtain $b = -f'(x_n)$.

Similarly differentiating (2.12) follows;

$$2a\left[\frac{dy}{dx}\frac{dy}{dx} + \big(y - f(x_n)\big)\frac{d^2y}{dx^2}\right] + \frac{d^2y}{dx^2} = 0 \; , \tag{2.13}$$

and from (2.5) we get

$$2a\left[\big(f'(x_n)\big)^2\right] + f''(x_n) = 0,$$

$$a = -\frac{1}{2}\left[\frac{f''(x_n)}{\big(f'(x_n)\big)^2}\right].$$

Substituting the results of a, b, c we achieve

$$-\frac{1}{2}\left[\frac{f''(x_n)}{\big(f'(x_n)\big)^2}\right]\big(y - f(x_n)\big)^2 + \big(y - f(x_n)\big) - f'(x_n)(x - x_n) = 0. \tag{2.14}$$

At the point $(x_{n+1}, 0)$ where the parabola and x-axis intercepts equation (2.14) gives

$$-\frac{1}{2}\left[\frac{f''(x_n)}{\left(f'(x_n)\right)^2}\right]\left(-f(x_n)\right)^2 - f(x_n) - f'(x_n)(x_{n+1} - x_n) = 0.$$

Solving for x_{n+1},

$$x_{n+1} = x_n - \left[1 + \frac{1}{2}L_f(x_n)\right]\frac{f(x_n)}{f'(x_n)}, \qquad n = 0,1,\ldots \tag{2.15}$$

thus, the Chebyshev's formula is achieved. Next we consider the hyperbola in the form

$$ay^2 + bxy + y + cx + d = 0. \tag{2.16}$$

To ensure that hyperbola passes through $(x_n, f(x_n))$, (2.16) is taken as

$$\begin{gathered} a\left(y - f(x_n)\right)^2 + b(x - x_n)\left(y - f(x_n)\right) + \left(y - f(x_n)\right) + \\ c(x - x_n) + d = 0, \end{gathered} \tag{2.17}$$

and $d = 0$. Differentiating (2.17) we obtain

$$2a\left(y - f(x_n)\right)\frac{dy}{dx} + b(y - f(x_n)) + b(x - x_n)\frac{dy}{dx} + \frac{dy}{dx} + c = 0. \tag{2.18}$$

Evaluating (2.18) at $(x_n, f(x_n))$ and applying the 1st and 2nd tangency conditions in (2.5) yields

$$c = -f'(x_n).$$

Differentiating (2.18), implicitly gives

$$2a\left[\left(\frac{dy}{dx}\right)^2 + \left(y - f(x_n)\right)\frac{d^2y}{dx^2}\right] + b\frac{dy}{dx} + b\left[\frac{dy}{dx} + (x - x_n)\frac{d^2y}{dx^2}\right] + \frac{d^2y}{dx^2} = 0,$$

and applying the conditions (2.5) we get

$$2a(f'(x))^2 + b[2f'(x_n)] + f''(x_n) = 0.$$

Solving this equation for a, results in

$$a = -\frac{f''(x_n)}{2(f'(x_n))^2} - \frac{b}{f'(x_n)} \quad .$$

Substituting the values of $a, c \; and \; d$ in (2.17) follows;

$$x - x_n = -\big(y - f(x_n)\big)\frac{1 + \left[-\frac{f''(x_n)}{2(f'(x_n))^2} - \frac{b}{f'(x_n)}\right] f(x_n)}{b\big(y - f(x_n)\big) - f'(x_n)} \quad . \tag{2.19}$$

Evaluating (2.19) at $(x_{n+1}, 0)$ becomes

$$x_{n+1} = x_n - \left[1 + \frac{\frac{1}{2} L_f(x_n)}{1 + \frac{b f(x_n)}{f'(x_n)}}\right] \frac{f(x_n)}{f'(x_n)} \quad , \quad n = 0,1, \dots \tag{2.20}$$

where b depends on n, as given in [20]. A class of third order convergent methods, are represented in (2.20). By taking $b = 0$, (2.20) results in (2.15) which is the Chebyshev's method. If $b = -\frac{f''(x_n)}{2f'(x_n)}$, (2.20) yields (2.10) the Halley's method. If $b = -\frac{f''(x_n)}{f'(x_n)}$ then (2.20) implies the Super Halley's method as:

$$x_{n+1} = x_n - \left[1 + \frac{1}{2} \frac{L_f(x_n)}{1 - L_f(x_n)}\right] \frac{f(x_n)}{f'(x_n)} \quad , \quad n = 0,1, \dots \tag{2.21}$$

proposed in [26]. As the limit case when $\lim_{n\to\infty} b = \mp\infty$, equation (2.21) gives the formula (2.3) which is the Newton's method.

2.3 Generalization to Banach Spaces

Definition 1: [9] The real function f form a vector space X to R^+, is called a norm on X, given by $||x||$ for $x \in X$ and satisfies the properties;

i) $||x|| > 0$ and $||x|| = 0 \;\; iff \;\; x = 0$ (2.22)

ii) $\|\alpha x\| = |\alpha|\|x\|$

iii) $\|x + y\| \leq \|x\| + \|y\|$

where α is a scalar and $x, y \in X$ are arbitrary vectors, if $x \in \mathbb{C}^n$ or $x \in R^n$, then

$\|x\|_2 = (\sum_{i=1}^{n}|x_i|^2)^{1/2}$ (Euclidean norm)

$\|x\|_1 = \sum_{i=1}^{n}|x_i|$ (The absolute sum norm) (2.23)

$\|x\|_\infty = max_i|x_i|$ (The maximum norm)

Theorem 1: [9] $\|A\| \coloneqq sup_{x\neq 0}\frac{\|Ax\|}{\|x\|}$ is a matrix norm. (2.24)

Definition 2: [27] The mapping $F: D \subset R^n \to R^m$ is Fréchet (or F-) differentiable at $x \in int(D)$ if there is an $A \in L(R^n, R^m)$ such that

$$\lim_{h\to 0}\frac{\|F(x+h)-F(x)-Ah\|}{\|h\|} = 0. \tag{2.25}$$

The linear operator A is denoted by $F'(x)$, and is called the F-derivative of F at x.

Consider the equation

$$F(x) = 0, \tag{2.26}$$

with a general map $F: X \to Y$. The formula (2.3) and (2.21) can be generalized to Banach spaces as follows: [20]

$$x_{n+1} = x_n - [F'(x_n)]^{-1}\, F(x_n) \tag{2.27}$$

$$x_{n+1} = x_n - \left[I + \frac{1}{2}[I + b\, [F'(x_n)]^{-1}\, F(x_n)]^{-1} L_f(x_n)\right] F'(x_n)^{-1}\, F(x_n) \tag{2.28}$$

$$n = 0,1,\ldots$$

respectively. Here I is identity matrix,

$$L_f(x) = [F'(x)]^{-1}\, F''(x)[F'(x)]^{-1}\, F(x), \tag{2.29}$$

is a linear operator on x for some $x \in X$, $F'(x)$ denote the first order Fréchet derivative of F and $[F'(x)]^{-1}$ is the inverse operator of F', assuming $[F'(x)]^{-1}$ exist. F'' is the second Fréchet derivative of F. The calculation of $F''(x)$ in (2.29) is problematic, for some equations. For example consider the nonlinear system;

$$\begin{gathered} f_1(x_1, x_2, \dots, x_n) = 0 \\ f_2(x_1, x_2, \dots, x_n) = 0 \\ \vdots \\ f_n(x_1, x_2, \dots, x_n) = 0, \end{gathered} \tag{2.30}$$

The first order Fréchet derivative is $J(x_1, x_2, \dots, x_n)$ and it is given as

$$J(x_1, x_2, \dots, x_n) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \frac{\partial f_1}{\partial x_3} & \cdots & \frac{\partial f_1}{\partial x_n} \\ & \vdots & & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \frac{\partial f_n}{\partial x_3} & \cdots & \frac{\partial f_n}{\partial x_n} \end{pmatrix},$$

which is $n \times n$ matrix involving n^2 values. Second order derivative F'' has n^3 values involving,

$$H_i(x) = \begin{pmatrix} \frac{\partial^2 f_i(x)}{\partial {x_1}^2} & \frac{\partial^2 f_i(x)}{\partial x_2 \partial x_1} & \frac{\partial^2 f_i(x)}{\partial x_3 \partial x_1} & \cdots & \frac{\partial^2 f_i(x)}{\partial x_n \partial x_1} \\ & \vdots & & \ddots & \vdots \\ \frac{\partial^2 f_i(x)}{\partial x_1 \partial x_n} & \frac{\partial^2 f_i(x)}{\partial x_2 \partial x_n} & \frac{\partial^2 f_i(x)}{\partial x_3 \partial x_n} & \cdots & \frac{\partial^2 f_i(x)}{\partial {x_n}^2} \end{pmatrix},$$

which is the Hessian's matrix of f_i , i=1,2,3,…,n [27]. To compute $F"(x)$ both high storage capacity and computational effort are required due to the number and the size of the Hessian's matrices. Recently, to solve this problem many authors proposed multi-step methods, which does not require the evaluation of $F''(x)$. Some of these studies are [28]-[31]. The following two-step recurrence formula which has third order convergence rate is given in [31].

$$\begin{aligned} y_{n+1} &= x_n - F'(x_n)^{-1} F(x_n) \\ x_{n+1} &= y_n - F'(x_n)^{-1} F(y_{n+1}), n = 0,1,\dots \end{aligned} \tag{2.31}$$

The two step method (2.31) can also be obtained from (2.28) if b is considered as [20];

$$bF(y_{n+1})[F'(x_n)]^{-1} F(x_n) = \frac{1}{2} L_f(x_n) F(x_n) - F(y_{n+1})$$

2.4 Convergence of Newton and a Class of Third Order Methods

Many authors studied the convergence of Newton method in Banach spaces. A basic work is given by Kantorovich [32], which asserts that Newton iterative method applied to a more general system of nonlinear equations $F(x) = 0$, converges to a solution $x^\star$ near some given point x_0 provided Jacobian of the system satisfies a Lipschitz condition near $x_0 \epsilon$D and its inverse at x_0 satisfies certain boundedness conditions.

Theorm 2: (Kantorovich [32])

Assume for some $x_0 \in D$ that $[F'(x_0)]^{-1}$ exists and that

i) $\|[F'(x_0)]^{-1}\| \leq \beta$

ii) $\|[F'(x_0)]^{-1} F(x_0)\| \leq \eta$

iii) $\|F'(x) - F'(y)\| \leq k\|x - y\|$

for all x and y in D with $h = \beta k\eta \le \frac{1}{2}$. Let $\Omega_* = \{x| \, \|x - x_0\| \le t^*\}$ where

$$t^* = \left(\frac{1 - \sqrt{1 - 2h}}{h}\right)\eta$$

Now if $\Omega_* \subset D$ then the Newton iterations $x_{n+1} = x_n - [F'(x_n)]^{-1} F(x_n)$ are well defined, remains in Ω_* and converges to $x^* \in \Omega_*$ such that $F(x^*) = 0$. In addition

$$\|x^* - x_k\| \le \frac{\eta}{\kappa}(\frac{\left(1 - \sqrt{1 - 2h}\right)^{2^k}}{2k} \;, \;\; k = 0,1, \dots \;\; .$$

The original proof of Kantorovich theorem [32] is long and very complex, therefore many authors [33]-[36] studied to give a nice threatment of this proof. The convergence of iterative methods of third order in (2.28) under Kantorovich conditions and posteriori error estimates are given by S. Amat, and S. Busquier in [37].

Lemma 1: [37] Let $x_0 \in D$ be such that $F'(x_0)$ exist. Assume there exists a real number $c > 0$ such that

$$\left\|[F'(x_0)]^{-1}\left(F''(x) - F''(y)\right)\right\| \le c\|x - y\|,$$

for all x, y in D. Then

$$\left\|[F'(x_0)]^{-1}\left\{F(x) - F(y) - F(y)(x - y) - \tfrac{1}{2}F''(y)(x - y)(x - y)\right\}\right\| \le \tfrac{c}{6}\|x - y\|^3, \tag{2.32}$$

$$\|[F'(x_0)]^{-1}\{F'(x) - F'(y)\}\| \leq \left(\|[F'(x_0)]^{-1}F''(y)\| + \frac{c}{2}\|x-y\|\right)\|x-y\| \tag{2.33}$$

for all x, y in D **.**

Theorem 3: [37] Let us assume $x_0 \in D$ is such that $[F'(x_0)]^{-1}$ exist and for some positive real numbers a, b and c satisfying $a \leq \frac{(b^2+2c)^{\frac{3}{2}} - b(b^2+3c)}{3c^2}$ and that

i) $\|[F'(x_0)]^{-1}F(x_0)\| \leq a$

ii) $\|[F'(x_0)]^{-1}F''(x_0)\| \leq b$

iii) $\|[F'(x_0)]^{-1}\{F''(x) - F''(y)\}\| \leq c\|x-y\|$

for all x, y in D. Besides if $t_{n+1} = t_n - \left(1 + \theta_n + O(\theta_n^2)\right)\frac{f(t_n)}{f'(t_n)}$ where

$\theta_n = \frac{1}{2}\frac{f''(t_n)f(t_n)}{[f'(t_n)]^2}$ then for all $n \geq 0$

$$\|x_{n+1} - x_n\| \leq t_n - t_{n+1} \quad .$$

That is $\{t_n\}$ is a majoring sequence of $\{x_n\}$, $n \geq 0$.

2.5 Approximate Matrix Inversion

If we apply the algorithm (2.27) to the equation $F(N) = N^{-1} - A \equiv 0$, we get the Newton method (NM) in (1.8)

$$N_{m+1} = N_m(2I - AN_m), \qquad m = 0,1 \ldots$$

or

$$N_{m+1} = (2I - N_mA)N_m, \qquad m = 0,1 \ldots \quad .$$

By applying the algorithm (2.28) when $b = 0$, yields the sequence of approximation (1.9), which is Chebyshev method (CM)

$$N_{m+1} = N_m\big(3I - AN_m(3I - AN_m)\big) \text{ or}$$

$$N_{m+1} = \big(3I - N_m A(3I - N_m A)\big)N_m \,, \;\; m = 0,1, \ldots$$

given in [20].

It is also possible to use the Neumann series

$$A^{-1} = N \sum_{k=0}^{\infty} (I - AN)^k$$

which converges when $\rho(I - AN) < 1$. If first two terms are taken we obtain (1.8). if first three terms are taken we get (1.9).

Theorem 4: [31]

Let $A = [a_{ij}]$ be any nonsingular matrix. If N_0 is chosen such that $\|E_0\| \triangleq \|I - AN_0\| < 1$ then the formula (1.8) converges quadratically to A^{-1} .

Proof: The proof of the Theorem 4 is given in **[**31**]** as follows:

Let $E_m = N_m - A^{-1}$ be the error matrix. From (1.8) we get

$$E_{m+1} + A^{-1} = (E_m + A^{-1})\big(2I - A(E_m + A^{-1})\big) \tag{2.34}$$

$$= (E_m + A^{-1})(2I - AE_m - I)$$

$$= 2E_m + 2A^{-1} - E_m A E_m - 2E_m + A^{-1}$$

$$E_{m+1} + A^{-1} = A^{-1} - E_m A E_m$$

$$E_{m+1} = -E_m A E_m \tag{2.35}$$

$$\|E_{m+1}\| \leq \|E_m\| \|AE_m\| \tag{2.36}$$

$$AE_{m+1} = -AE_m AE_m \tag{2.37}$$

$$\|AE_{m+1}\| \leq \|AE_m\|\|AE_m\| = \|AE_m\|^2 \tag{2.38}$$

If $\|AE_0\| = a$ then by (2.38), $\|AE_1\| \leq a^2$, $\|AE_2\| \leq a^4$, …, $\|AE_m\| \leq a^{2m}$ then by (2.36), $\|E_1\| \leq a\|E_0\|$, $\|E_2\| \leq \|E_1\|\|AE_1\| \leq a^2\|E_0\|, \dots, \|AE_m\| \leq a^{2m-1}\|E_0\|$.

Now by induction it can be shown that

If $\|AE_0\| = a < 1$ then $\|E_m\| \to 0$ as $m \to \infty$

therefore $\lim_{m\to\infty} N_m \to A^{-1}$ and by (2.36), $\|E_{m+1}\| \leq \|A\|\|E_m\|^2$ and if $E_m \neq 0$ then $\frac{\|E_{m+1}\|}{\|E_m\|^2} \leq \|A\|$ which gives that order of convergence is at least 2.

Theorem 5: [16]

Let $A = [a_{ij}]$ be a nonsingular matrix and N_0 be an initial approximate inverse taken such that $\|E_0\| \triangleq \|I - AN_0\| < 1$ then the formula (1.9) converges to A^{-1} with third order.

Proof: Proof is given in [16] and mainly based on ideas in [31] and [38].

$E_{k+1} = I - AN_{k+1}$ be the error at k^{th} iteration. Using (1.9)

$$E_{k+1} = I - AN_{k+1} = I - A\big[N_k[3I - AN_k(3I - AN_k)]\big]$$

$$= I - AN_k[3I - 3AN_k + (AN_k)^2]$$

$$= (I - AN_k)^3 = (E_k)^3 \tag{2.39}$$

Since $\|E_0\| < 1$, then from (2.39) we have that

$\|E_{k+1}\| \leq \|E_k\|^3 \leq \cdots \leq \|E_0\|^{3^{k+1}} \to 0$ as $k \to \infty$

i.e. $I - AN_k \to 0$ as $k \to \infty$ and $N_k \to A^{-1}$, as $k \to \infty$. Let $e_k \triangleq N_k - A^{-1}$ then by (1.9) we obtain

$$A^{-1} + e_{k+1} = N_{k+1} = N_k\big(3I - AN_k(3I - AN_k)\big)$$

$$= (A^{-1} + e_k)\big(3I - A(A^{-1} + e_k)(3I - A(A^{-1} + e_k))\big)$$

$$= (A^{-1} + e_k)(I - Ae_k + (e_k)^2)$$

$$= A^{-1} + e_k(Ae_k)^2$$

$$e_{k+1} = e_k(Ae_k)^2$$

$$\text{So } \|e_{k+1}\| \leq \|A\|^2\|e_k\|^3$$

2.6 Methods of Approximate Inverse Preconditioners

Let N_M be the approximate inverse obtained after performing M^{th} iterations by (NM) in (1.8) or by (CM) in (1.9) satisfying $\|I - AN_M\| \leq \varepsilon < 1$ for some desired accuracy ε. N_M can be applied to the system $Ax = b$ in (1.1) as a right preconditioning as

$$AN_M y = b\,. \tag{2.40}$$

We obtain the approximate solution of x as $x \equiv N_M b$ since $AN_M \approx I$. It is also possible to apply N_M as a left precinditioner to the system (1.1) as

$$N_M A x = N_M b, \tag{2.41}$$

In left preconditioned system (2.41) again the approximate solution is $x \equiv N_M b$. Next we show that, if $AN_0 = N_0 A$ then,

$$AN_m = N_m A \text{ for all } m = 0,1,\ldots \tag{2.42}$$

where N_m is the preconditioner in (NM) or in (CM). Let N_m be the approximate inverse at m^{th} iteration in (NM). Assume N_0 is selected such that $AN_0 = N_0A$ then

$$AN_1 = AN_0(2I - AN_0)$$

$$= N_0A(2I - N_0A)$$

$$= N_0(2A - AN_0A)$$

$$= N_0(2I - AN_0)A$$

$$= N_1A$$

so (2.42) is true for $k = 1$. Using mathematical induction let us assume that (2.42) is true for $n = k$ then

$$AN_{k+1} = AN_k(2I - AN_k)$$

$$= N_kA(2I - N_kA)$$

$$= N_k(2A - AN_kA)$$

$$AN_{k+1} = N_k(2I - AN_k)A$$

$$AN_{k+1} = N_{k+1}A$$

By using the same technique (2.42) is verified for Chebyshev method (1.9) in [16]. The computer algorithms for solving the linear system (1.1) using (1.8) and (1.9) are as follows, based on the algorithm for (NM) given in [15].

Algorithm: Newton Method (NM)

```
Step1: Input $(A_{n\times n}, b_{n\times 1}, N_{0_{n\times n}}, \varepsilon)$                (2.43)

Step2: $B \coloneqq AN_0$

Step3: $N_1 \coloneqq N_0(2I - B)$

Step4: Evaluate $p \coloneqq \|I - B\|_\infty$

$m = 1$

Step5: While $p > \varepsilon,$ Do

        $B \coloneqq AN_m$

        $N_{m+1} \coloneqq N_m(2I - B)$

        Evaluate $p \coloneqq \|I - B\|_\infty$

        $m = m + 1$

    End Do

Step6: $x \coloneqq N_m b$
```

Algorithm: Chebyshev Method (CM)

```
Step1: Input $(A_{n\times n}, b_{n\times 1}, N_{0_{n\times n}}, \varepsilon)$                (2.44)

Step2: $B \coloneqq AN_0$

Step3: $N_1 \coloneqq N_0\big(3I - B(3I - B)\big)$

Step4: Evaluate p$\coloneqq \|I - B\|_\infty$
```

```
$m = 1$

Step5: While $p > \varepsilon,$ Do
        $B \coloneqq AN_m$
        $N_{m+1} \coloneqq N_m\big(3I - B(3I - B)\big)$
        Evaluate p$\coloneqq \|I - B\|_\infty$
        $m = m + 1$
    End Do

Step6: x$\coloneqq N_m b$
```

In these algorithms ε is the predescribed accuracy, N_m is the preconditioner of A in the m^{th} Newton and Chebyshev iterations, where N_0 is the initial approximate inverse.

Chapter 3

FINITE DIFFERENCE SCHEMES FOR POISSON'S EQUATION

3.1 Introduction

The construction of difference schemes for the numerical solution of Poisson problem with Dirichlet conditions on the sides of a rectangle is analyzed. Using 5 and 9 point stencils system of difference equations are obtained. The structure of the coefficient matrices arised from the difference equations are investigated.

3.2 The Dirichlet Poisson Problem on Rectangle

Let $R = \{(x, y): 0 < x < a, 0 < y < b\}$ be an open rectangle $\gamma^j,\ j = 1,2,3,4$ be the sides of this rectangle including the vertices. Let the numbering be in counterclockwise direction starting from the side which lies on the x-axis.

The Dirichlet Poisson equation on a rectangle is

$$\Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x, y) \quad on\ R \tag{3.1}$$

$$u = \varphi^m \ on\ \gamma^m,\ \ m = 1,2,3,4$$

3.3 Construction of Difference Schemes

The construction of 5-point and 9-point schemes are given as follows in [39]. Let us draw two systems of parallel lines on the plane:

$$x = x_0 + ih = x_i \tag{3.2}$$

$$y = y_0 + kh = y_k$$

Consider the node (i,k) of the net, and take the four nodes closest to it which are $(i+1,k), (i,k+1), (i-1,k), (i,k-1)$ as shown in the figure below

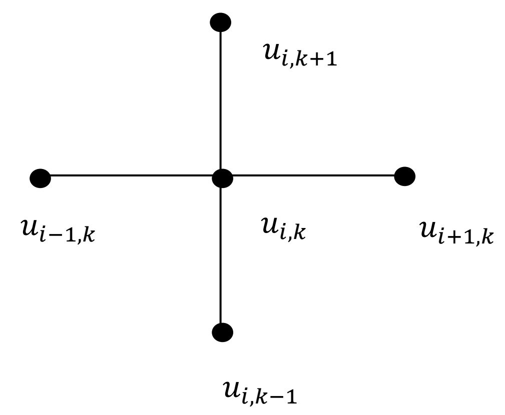


Figure 3.1: 5-Point Stencil.

We aim to find an approximate expression for Δu at the node (i,k). From Taylor's formula the expressions for the neighboring points of u_{ik} are as follows:

$$u_{i+1,k} - u_{i,k} = hu_x + \frac{h^2}{2!}u_{x^2} + \frac{h^3}{3!}u_{x^3} + \frac{h^4}{4!}u_{x^4} + \cdots$$

$$u_{i-1,k} - u_{i,k} = -hu_x + \frac{h^2}{2!}u_{x^2} - \frac{h^3}{3!}u_{x^3} + \frac{h^4}{4!}u_{x^4} + \cdots$$

$$u_{i,k+1} - u_{i,k} = hu_y + \frac{h^2}{2!}u_{y^2} + \frac{h^3}{3!}u_{y^3} + \frac{h^4}{4!}u_{y^4} + \cdots$$

$$u_{i,k-1} - u_{i,k} = -hu_y + \frac{h^2}{2!}u_{y^2} - \frac{h^3}{3!}u_{y^3} + \frac{h^4}{4!}u_{y^4} + \cdots \quad . \tag{3.3}$$

We look for Δu as linear combination of the differences in (3.3). The next expression is obtained for Δu depending on the derivatives by adding the equations in (3.3) term by term.

$$\boxdot\, u_{i,k} = u_{i+1,k} + u_{i,k+1} + u_{i-1,k} + u_{i,k-1} - 4u_{i,k} =$$
$$2\left[\frac{h^2}{2!}\left({u_x}^2 + {u_y}^2\right) + \frac{h^4}{4!}\left({u_x}^4 + {u_y}^4\right) + \frac{h^6}{6!}\left({u_x}^6 + {u_y}^6\right) + \cdots\right] \tag{3.4}$$

which yields

$$\frac{1}{h^2}\boxdot\, u_{i,k} = \Delta u + E_{i,k} \tag{3.5}$$

where

$$E_{i,k} = \frac{2h^2}{4!}\left(u_{x^4} + u_{y^4}\right) + \frac{2h^4}{4!}\left(u_{x^6} + u_{y^6}\right) + \cdots \tag{3.6}$$

is the remainder term. Taking the values of derivatives up to fourth orders, and evaluating the fourth order derivatives at the mean points $E_{i,k}$ becomes an expression of the form

$$E_{i,k} = \frac{4h^2}{4!}cM_4 \tag{3.7}$$

where,

$$M_4 = max\left\{\left|\frac{\partial^4 u}{\partial x^4}\right|, \left|\frac{\partial^4 u}{\partial y^4}\right|\right\} \;\; and \;\; |c| \leq 1.$$

For the Poisson equation (3.1) we get

$$\tfrac{1}{h^2}\boxdot\, u_{i,k} = f_{i,k}\ , \tag{3.8}$$

when the remainder term $E_{i,k}$ in (3.5) is neglected. If $f(x,y)=0$ in (3.8) then we get the difference equation of the Laplace equation as;

$$\frac{1}{h^2}\boxdot u_{i,k}=0, \tag{3.9}$$

Which is an approximation of (3.5).

Assign a square mesh R_h, with step $h=\frac{a}{n_1}=\frac{b}{n_2}$, $n_1\geq 2, n_2\geq 2$ are integers, obtained with the lines in (3.2) as $x=0+ih, y=0+kh,\ \ i=0,1,\dots,n_1$, $k=0,1,\dots,n_2$. ${\gamma_h}^k$ is the set of grids on $\gamma^k, k=1,2,3,4$ and $\gamma_h=\bigcup_{k=1}^{4}{\gamma_h}^k$, $\overline{R_h}=R_h\cup\gamma_h$. The following difference problem is obtained, for (3.1),

$$u_h=B_1u_h-\frac{h^2}{4}f\ \ on\ \ R_h \tag{3.10}$$

$$u_h={\varphi_h}^m\ \ on\ \ {\gamma_h}^m\ ,\ \ m=1,2,3,4, \tag{3.11}$$

where ${\varphi_h}^m$ is the trace of φ^m on ${\gamma_h}^m$ and

$$B_1u(x,y)=\big(u(x+h,y)+u(x-h,y)+u(x,y+h)+u(x,y-h)\big) \tag{3.12}$$

derived from (3.8).

Next we consider a high accurate difference operator. Beside with the values of the function at the nodes of the net $(i,k),(i+1,k),(i,k+1),(i-1,k),(i,k-1)$ which are considered in the formation of $\boxdot u_{i,k}$ we also consider the values of u at the nodes $(i+1,k+1),(i+1,k-1),(i-1,k-1),(i-1,k+1)$ as shown in Figure 3.2 .

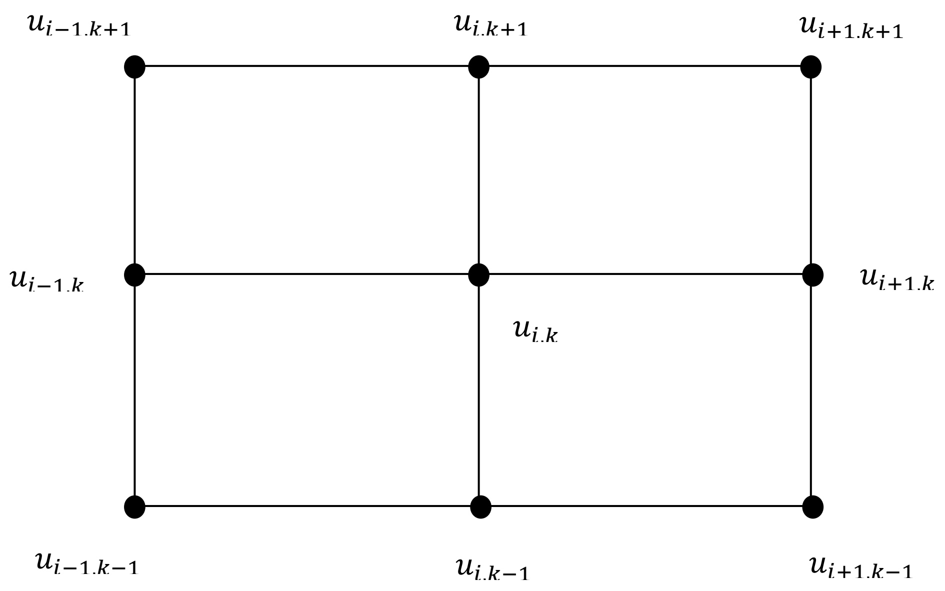


Figure 3.2: 9-Point Stencil.

and expand them near the point $u_{i,k}$ using Taylor's formula,

$$u_{i+1,k+1} - u_{i,k} = h\left(\frac{\partial}{\partial x} + \frac{\partial}{\partial y}\right)u + \frac{h^2}{2!}\left(\frac{\partial}{\partial x} + \frac{\partial}{\partial y}\right)^2 u + \frac{h^3}{3!}\left(\frac{\partial}{\partial x} + \frac{\partial}{\partial y}\right)^3 u + \cdots$$

$$u_{i-1,k+1} - u_{i,k} = h\left(-\frac{\partial}{\partial x} + \frac{\partial}{\partial y}\right)u + \frac{h^2}{2!}\left(-\frac{\partial}{\partial x} + \frac{\partial}{\partial y}\right)^2 u + \frac{h^3}{3!}\left(-\frac{\partial}{\partial x} + \frac{\partial}{\partial y}\right)^3 u + \cdots$$

$$u_{i-1,k-1} - u_{i,k} = h\left(-\frac{\partial}{\partial x} - \frac{\partial}{\partial y}\right)u + \frac{h^2}{2!}\left(-\frac{\partial}{\partial x} - \frac{\partial}{\partial y}\right)^2 u + \frac{h^3}{3!}\left(-\frac{\partial}{\partial x} - \frac{\partial}{\partial y}\right)^3 u + \cdots$$

$$u_{i+1,k-1} - u_{i,k} = h\left(\frac{\partial}{\partial x} - \frac{\partial}{\partial y}\right)u + \frac{h^2}{2!}\left(\frac{\partial}{\partial x} - \frac{\partial}{\partial y}\right)^2 u + \frac{h^3}{3!}\left(\frac{\partial}{\partial x} - \frac{\partial}{\partial y}\right)^3 u + \cdots \tag{3.13}$$

with the above differences we form the sum $\boxplus\, u_{i,k}$ which gives

$$\boxplus\, u_{i,k} = u_{i+1,k+1} + u_{i-1,k+1} + u_{i-1,k-1} + u_{i+1,k-1} - 4u_{i,k}$$

$$= 4\left\{\frac{h^2}{2!}\left(u_{x^2} + u_{y^2}\right) + \frac{h^4}{4!}\left(u_{x^4} + 6u_{x^2y^2} + u_{y^4}\right) + \frac{h^6}{6!}\left(u_{x^6} + 15u_{x^4y^2} + 15u_{x^2y^4} + u_{y^6}\right) + \cdots\right\} \qquad (3.14)$$

Finally we will look for the combination $c_1 \boxdot\, u_{i,k} + c_2 \boxplus\, u_{i,k}$ to get an approximate expression for Δu. There is no way to choose c_1 and c_2 such that the fourth order derivatives will vanish, however by choosing $c_1 = \frac{2}{3h^2}$ and $c_2 = \frac{1}{6h^2}$ the term with the fourth order derivatives form a biharmonic operator

$$\Delta\Delta u = \frac{\partial^4 u}{\partial x^4} + 2\frac{\partial^4 u}{\partial x^2 \partial y^2} + \frac{\partial^4 u}{\partial y^4}$$

which is known since $\Delta u = f(x,y)$ and $\Delta\Delta u = \Delta f(x,y)$. Therefore we get the high accurate scheme

$$\frac{1}{6h^2}\left(4 \boxdot\, u_{i,k} + \boxplus\, u_{i,k}\right) = \Delta u + \frac{2h^2}{4!}\Delta^2 u + \frac{2h^4}{6!}\left(\Delta^3 u + 2\frac{\partial^4}{\partial x^2 \partial y^2}\Delta u\right) + E_{i,k}\ , \qquad (3.15)$$

where

$$E_{i,k} = \frac{2}{3}\frac{h^6}{8!}\left[3\Delta^4 u + 16\frac{\partial^4}{\partial x^2 \partial y^2}\Delta^2 u + 20\frac{\partial^8 u}{\partial x^4 \partial y^4}\right] + \cdots$$

if we ignore the error term $E_{i,k}$, results

$$\frac{1}{6h^2}\left(4 \boxdot\, u_{i,k} + \boxplus\, u_{i,k}\right) = \Delta u + \frac{2h^2}{4!}\Delta^2 u + \frac{2h^4}{6!}\left(\Delta^3 u + 2\frac{\partial^4}{\partial x^2 \partial y^2}\Delta u\right) \qquad (3.16)$$

If $\Delta u = 0$ then (3.16) gives a high accurate scheme for the Laplace equation, as

$$\frac{1}{6h^2}\left(4 \boxdot u_{i,k} + \boxplus u_{i,k}\right) = 0, \tag{3.17}$$

by which the exact equation (3.15) is approximated.

If $\Delta u = f(x,y)$ then (3.16) gives

$$\frac{1}{6h^2}\left(4 \boxdot u_{i,k} + \boxplus u_{i,k}\right) = \left[f_{i,k} + \frac{2h^2}{4!}\Delta f_{i,k} + \frac{2h^4}{6!}\left(\Delta^2 f_{i,k} + 2\frac{\partial^4 f_{i,k}}{\partial x^2 \partial y^2}\right)\right]. \tag{3.18}$$

When the function $f(x,y)$ is given analytically then the implementation of (3.18) is not troubling. However if $f(x,y)$ is given as grid function, then the values on the right side of (3.18) can be approximated using difference schemes of high accuracy. Assuming that $f(x,y)$ is given analytically we derive the following difference problem for the Dirichlet Poisson equation on the rectangle given in (3.1) as follows;

$$u_h = B_2 u_h - \frac{6h^2}{20}\left[f_{i,k} + \frac{2}{4!}h^2\Delta f_{i,k} + \frac{2}{6!}h^4\left(\Delta^2 f_{i,k} + 2\frac{\partial^4 f_{i,k}}{\partial x^2 \partial y^2}\right)\right] \tag{3.19}$$

where

$$u_h = {\varphi_h}^m \quad on \quad {\gamma_h}^m, m = 1,2,3,4\,, \tag{3.20}$$

and

$$B_2 u(x,y) \equiv \frac{\left(u(x+h,y) + u(x,y+h) + u(x-h,y) + u(x,y-h)\right)}{5} + \frac{\left(u(x+h,y+h) + u(x-h,y+h) + u(x-h,y-h) + u(x+h,y-h)\right)}{20} \tag{3.21}$$

defined from (3.18) .

3.4 Lexicographical Ordering for the Poisson Model Problem

Consider the difference problem given in equations (3.10), (3.11) for grid values on the boundary $\gamma_h{}^m$, $m = 1,2,3,4$, $u_{i,k}$ is known for the boundary data (3.11), i.e.

$$u_{i,0} = \varphi_h{}^1(ih, 0) \text{ for } i = 0,1, \dots, n_1$$

$$u_{n_1,k} = \varphi_h{}^2(n_1 h, kh) \text{ for } k = 0,1, \dots, n_2 \tag{3.22}$$

$$u_{i,n_2} = \varphi_h{}^3(ih, n_2 h) \text{ for } i = 0,1, \dots, n_1$$

$$u_{0,k} = \varphi_h{}^4(0, kh) \text{ for } k = 0,1, \dots, n_2$$

The number of unknown $u_{i,k}$ is $(n_1 - 1) \times (n_2 - 1)$ which is the number of inner grid points. The system of equations is obtained by eliminating the boundary values (3.22) which appears in (3.10). We form the commonly used matrix form $Ax = b$ with an $(n_1 - 1)(n_2 - 1) \times (n_1 - 1)(n_2 - 1)$ matrix A and $(n_1 - 1)(n_2 - 1) \times 1$ dimensional vectors x and b by representing the twofold indexed unknown $u_{i,k}$ by a single indexed vector x. This implies that the inner grid points must be enumerated in some way. Figure 3.3 represents the Lexicographical ordering, [40]

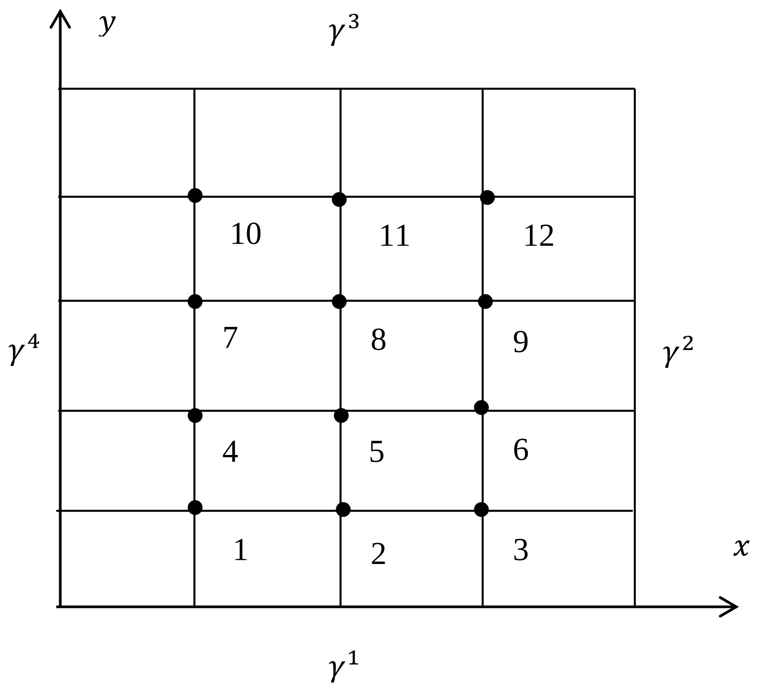


Figure 3.3: Lexicographical Ordering, for the case $n_1 = 4$ and $n_2 = 5$.

The coefficient matrix A obtained for the difference problem (3.10), (3.11) using Lexicographical ordering has the following structure in Figure 3.4 .

$$A = \begin{bmatrix} T & -I & & & & \\ -I & T & -I & & & \\ & -I & T & -I & & \\ & & \ddots & \ddots & \ddots & \\ & & & & \ddots & \\ & & & \ddots & T & -I \\ & & & & -I & T \end{bmatrix} \qquad T = \begin{bmatrix} 4 & -1 & & & & \\ -1 & 4 & -1 & & & \\ & -1 & 4 & -1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & & \ddots & \\ & & & \ddots & 4 & -1 \\ & & & & -1 & 4 \end{bmatrix}$$

Figure 3.4: Structure of the Coefficient Matrix Using 5-point Scheme and Lexicographical Ordering.

Accordingly A takes the form of a block-tridiagonal matrix built from $(n_2 - 1) \times (n_2 - 1)$ blocks T which again are tridiagonal $(n_1 - 1) \times (n_1 - 1)$ matrices. I is the $(n_1 - 1) \times (n_1 - 1)$ identity matrix.

Using the difference problem (3.19), (3.20) for obtaining highly accurate numerical solution of the Dirichlet Poisson equation on the rectangle given in (3.1) and applying the Lexicographical ordering the coefficient matrix A has the structure as given in Figure 3.5 , [40]

$$A=\begin{bmatrix} D & C & & & & \\ C & D & C & & & \\ & C & D & C & & \\ & & \ddots & \ddots & \ddots & \\ & & & & & \ddots & \\ & & & \ddots & & D & C \\ & & & & & C & D \end{bmatrix} \qquad D=\begin{bmatrix} 20 & -4 & & & & \\ -4 & 20 & -4 & & & \\ & -4 & 20 & -4 & & \\ & & \ddots & \ddots & \ddots & \\ & & & & & \ddots & \\ & & & \ddots & & & -4 \\ & & & & & -4 & 20 \end{bmatrix}$$

$$C=\begin{bmatrix} -4 & -1 & & & & \\ -1 & -4 & -1 & & & \\ & -1 & -4 & -1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & & & \ddots & \\ & & & \ddots & & & -1 \\ & & & & & -1 & -4 \end{bmatrix}$$

Figure 3.5: Structure of the Matrix A Using 9-point Scheme and Lexicographical Ordering.

Both D and C are tridiagonal matrices of size $(n_1 - 1) \times (n_1 - 1)$ and A is a block tridiagonal matrix built from $(n_2 - 1) \times (n_2 - 1)$ blocks. The coefficient matrix A obtained both from the 5-point difference and the 9-point difference analog using Lexicographical ordering is diagonally dominant, positive definite and symmetric matrix.

Chapter 4

NUMERICAL RESULTS AND DISCUSSIONS

4.1 Introduction

This chapter accomplishes the study with the numerical solution of the test problem chosen from Laplace’s equation. Second and high order accurate difference schemes are used to get the system of equations for the approximate solutions. The obtained algebraic linear systems are solved by preconditioning them with approximate inverses via (NM) and (CM). The computations are performed in Mathematica and numerical results are displayed with tables and figures.

4.2 Description of the Model Problem

Let R be the rectangle defined as

$R = \{(x, y): 0 < x < 1, 0 < y < 1\}$, consider the problem

$\Delta u = 0 \quad on\ \ R$

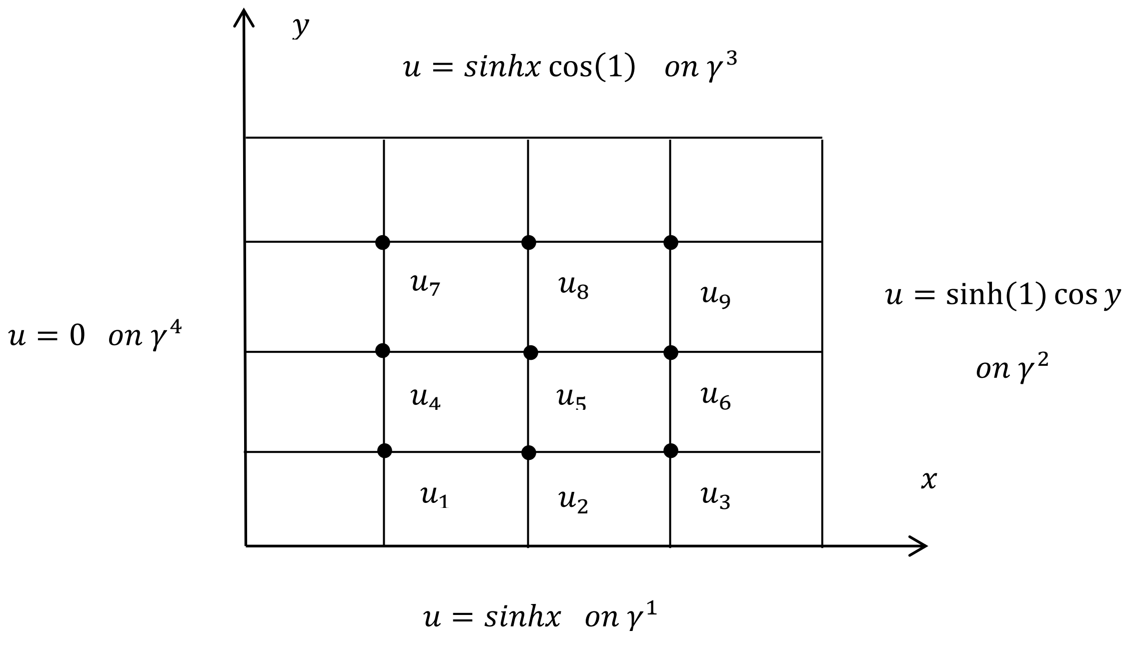
$u = \sinh(x) \quad on\ \ \gamma^1: \{0 \le x \le 1,\ \ y = 0\}$

$u = \sinh(1)\cos(y) \quad on\ \ \gamma^2: \{0 \le y \le 1,\ \ x = 1\}$

$u = \sinh(x)\, cosy(1) \quad on\ \ \gamma^3: \{0 \le x \le 1,\ \ y = 1\}$

$u = 0 \quad on\ \ \gamma^4: \{0 \le y \le 1,\ \ x = 0\}$

The exact solution of this problem is $u(x, y) = \sinh(x)\cos(y)$. This model problem, is represented in Figure 4.1, with mesh step $h = \frac{1}{4}$.



y
$u = sinhx\cos(1) \quad on\,\gamma^3$
u_7
u_8
u_9
$u = \sinh(1)\cos y$
$on\,\gamma^2$
$u = 0 \quad on\,\gamma^4$
u_4
u_5
u_6
u_1
u_2
u_3
x
$u = sinhx \quad on\,\gamma^1$

Figure 4.1: The Model Problem and Representation of Inner Grids for h=1/4.

4.3 The Choice of the Initial Inverse

It is given in Theorem 4 (Theorem 5) in Chapter 2 that if $\|I - AN_0\| < 1$ then (NM) (respectively (CM) converges). Therefore it is important to choose an approximate initial inverse N_0 which satisfies this condition for A obtained from second order (5-point) and high order (9-point) schemes. For the algebraic linear system obtained from 5-point scheme, N_0 is selected as the diagonal matrix

$$N_0 = \left[\frac{1}{4}, \frac{1}{4}, \dots, \frac{1}{4}\right]$$

and for the algebraic linear system arised from 9-point scheme it is selected as

$$N_0 = \left[\frac{1}{20}, \frac{1}{20}, \dots, \frac{1}{20}\right].$$

Table 4.1 represents the initial errors between the identity matrix and AN_0 in second norm for the linear systems obtained using difference schemes with mesh steps $h = \frac{1}{4}, \frac{1}{8}, \frac{1}{16}$.

Table 4.1: Initial Errors in Second Norm for the Linear Systems Obtained from 5-Point and 9-Point Schemes.

	5-point scheme	**9-point scheme**
$\boldsymbol{h}$	$\|\|\boldsymbol{I} - \boldsymbol{AN_0}\|\|_2$	$\|\|\boldsymbol{I} - \boldsymbol{AN_0}\|\|_2$
$\boldsymbol{\frac{1}{4}}$	**0.707107**	**0.665685**
$\boldsymbol{\frac{1}{8}}$	**0.92388**	**0.909814**
$\boldsymbol{\frac{1}{16}}$	**0.980785**	**0.977016**

4.4 Computational Results

The algorithms (2.42) and (2.43) are realized by using Mathematica and sparse matrix computations, due to the property that, the coefficient matrix A has 5-nonzero diagonals, and 9-nonzero diagonals when arised from 5-point and 9-point schemes respectively.

Tables 4.2 - 4.4 represents the CPU-times and the errors in maximum norm per iteration solved by (NM), for 5-point scheme.

Table 4.2: Maximum Errors and CPU-Times by (NM), Using 5-Point Scheme with h=1/4.

Iteration	$\|u - u_h\|_\infty$	CPU time
1	**0.34268**	**4.33681×10^{-19}**
2	**0.171173**	**1.50704×10^{-17}**
3	**0.042542**	**0.016**
4	***0.002345***	**0.032**
5	**0.000324855**	**0.064**

Table 4.3: Maximum Errors and CPU-Times by (NM), Using 5-Point Scheme with h=1/8.

Iteration	$\|u - u_h\|_\infty$	CPU time
1	**0.696826**	**0.032**
2	**0.551503**	**0.046**
3	**0.395184**	**0.14**
4	**0.203264**	**0.249**
5	**0.05734**	**0.515**
6	**0.015625**	**1.123**
7	**0.0000670886**	**1.888**
8	**0.0000933207**	**2.699**

Table 4.4: Maximum Errors and CPU-Times by (NM), Using 5-Point Scheme with h=1/16.

Iteration	$\|u - u_h\|_\infty$	CPU time
1	**0.901475**	**0.016**
2	**0.828801**	**0.124**
3	**0.717372**	**0.453**
4	**0.580328**	**2.776**
5	**0.410804**	**22.792**
6	**0.214423**	**84.366**
7	**0.0614492**	**163.739**
8	**0.00510848**	**229.758**
9	**0.0000150869**	**318.46**

The Tables 4.5 - 4.7 presents the CPU-Time and the errors in maximum norm per iteration solved by (CM), for 5-point scheme.

Table 4.5: Maximum Errors and CPU-Times by (CM), Using 5-Point Scheme with h=1/4.

Iteration	$\|\mathbf{u}-\mathbf{u_h}\|_\infty$	**CPU time**
1	**0.228485**	**1.30104×10^{-18}**
2	**0.0282675**	**0.015**
3	**0.000289966**	**0.016**
4	***0.000334916***	**0.031**

Table 4.6: : Maximum Errors and CPU-Times by (CM), Using 5-Point Scheme with h=1/8.

Iteration	$\|\mathbf{u}-\mathbf{u_h}\|_\infty$	**CPU time**
1	**0.616165**	**0.016**
2	**0.363531**	**0.094**
3	**0.0849632**	**0.655**
4	**0.00109617**	**1.872**
5	**0.0000933188**	**4.212**

Table 4.7: Maximum Errors and CPU-Times by (CM), Using 5-Point Scheme with h=1/16.

Iteration	$\|\mathbf{u}-\mathbf{u_h}\|_\infty$	**CPU time**
1	**0.858971**	**0.047**
2	**0.696446**	**0.499**
3	**0.456363**	**18.642**
4	**0.152793**	**132.492**
5	**0.00657905**	**313.296**
6	**0.0000231964**	**629.152**

The Figures 4.2 – 4.4 compare the convergency of the (NM) and (CM) with respect to the errors in maximum norm per iteration, obtained for the model problem using 5-point scheme.

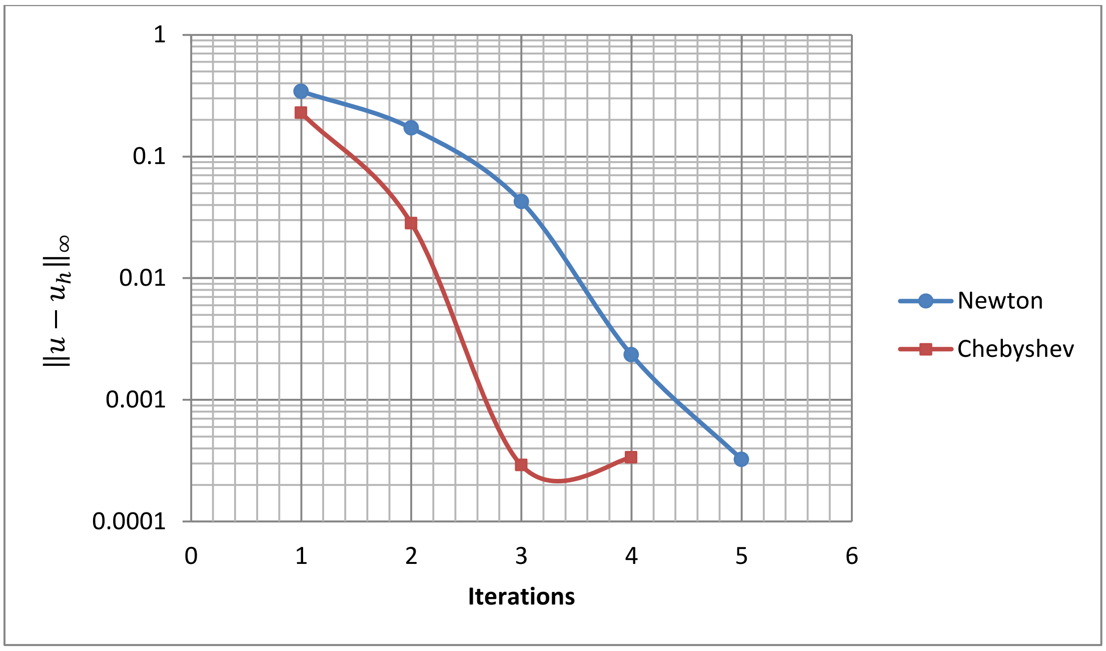


Figure 4.2: Comparison of the Convergency of (NM) and (CM) Using 5-Point Scheme for h=1/4.

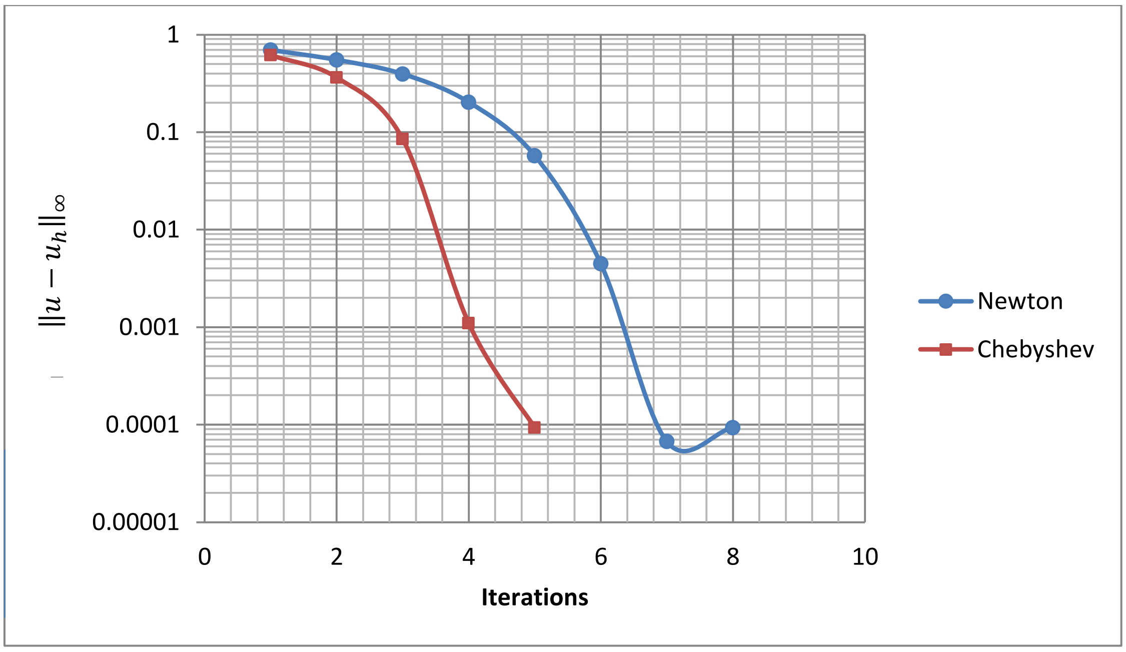


Figure 4.3: Convergency Comparison of (NM) and (CM) Using 5-Point Scheme with h=1/8.

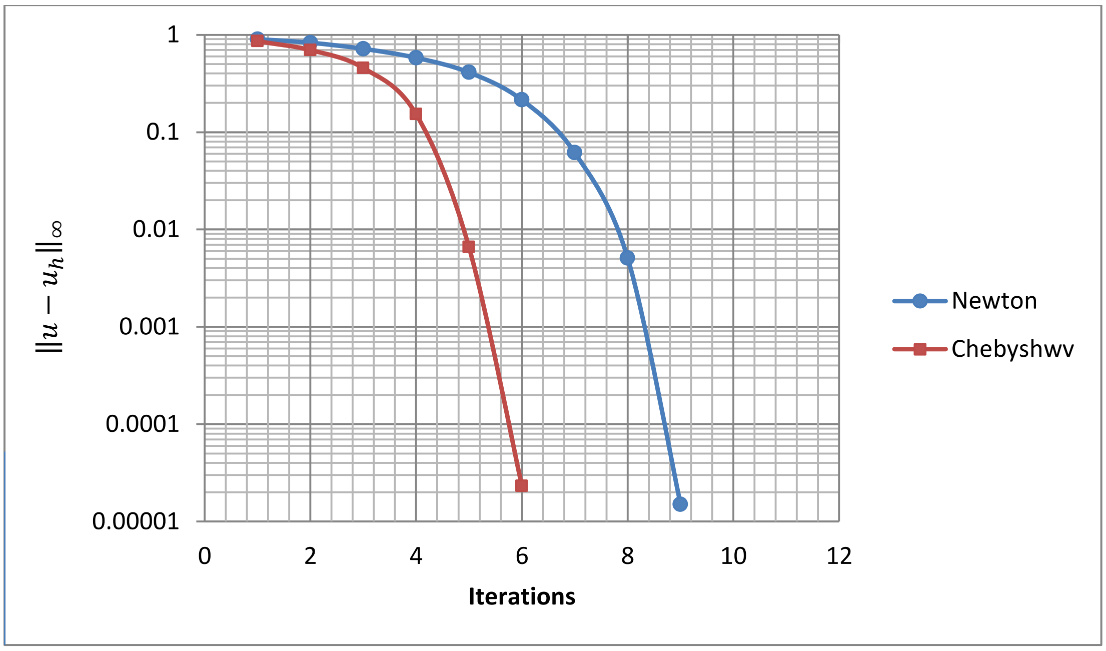


Figure 4.4: Comparison of the Convergency between (NM) and CM) Using 5-Point Scheme with h=1/16.

Tables 4.8 – 4.10 demonstrates the CPU-time and errors in maximum norm per iteration solved by the (NM) using 9-point scheme.

Table 4.8: The Maximum Errors and the CPU-Times by the (NM) Using 9-Point Scheme with h=1/4.

Iteration	$\|\mathbf{u}-\mathbf{u_h}\|_\infty$	**CPU time**
1	**0.297164**	**0.015**
2	**0.131721**	**0.015**
3	**0.0257342**	**0.016**
4	$\mathbf{9.90784\times10^{-4}}$	**0.031**
5	$\mathbf{1.47042\times10^{-6}}$	**0.031**
6	$\mathbf{2.71928\times10^{-9}}$	**0.093**

Table 4.9: The Maximum Errors and the CPU-Times by the (NM) Using 9-Point Scheme with h=1/8.

Iteration	$\|\mathbf{u}-\mathbf{u_h}\|_\infty$	**CPU time**
1	**0.667265**	**0.016**
2	**0.517839**	**0.063**
3	**0.346072**	**0.249**
4	**0.159098**	**1.014**
5	**0.0350901**	**2.512**
6	**0.0017048**	**4.946**
7	$\mathbf{4.02388\times10^{-6}}$	**8.845**
8	$\mathbf{2.72061\times10^{-11}}$	**16.38**

Table 4.10: The Maximum Errors and the CPU-Times by the (NM) Using 9-Point Scheme with h=1/16**.**

Iteration	$\|\mathbf{u}-\mathbf{u_h}\|_\infty$	**CPU time**
1	**0.889034**	**0.047**
2	**0.802113**	**0.281**
3	**0.684669**	**2.324**
4	**0.540570**	**30.296**
5	**0.358293**	**201.663**
6	**0.166407**	**399.861**
7	**0.037548**	**774.201**
8	**0.00191431**	**1442.63**
9	$\mathbf{4.97578\times10^{-6}}$	**2880.17**
10	$\mathbf{3.29532\times10^{-11}}$	**6240.57**
11	$\mathbf{7.1907\times10^{-13}}$	**13778.6**

Tables 4.11 – 4.13 displays the CPU-time and errors in maximum norm per iteration by the (CM) using 9-point scheme.

Table 4.11: The Maximum Errors and CPU-Times by the (CM) Using 9-Point Scheme for h=1/4.

Iteration	$\|\mathbf{u}-\mathbf{u_h}\|_\infty$	**CPU time**
1	**0.194803**	**0.015**
2	**0.0170823**	**0.016**
3	$\mathbf{1.12666\times10^{-5}}$	**0.047**
4	$\mathbf{2.72254\times10^{-9}}$	**0.078**

Table 4.12: The Maximum Errors and CPU-Times by the (CM) Using 9-Point Scheme for h=1/8.

Iteration	$\|u - u_h\|_\infty$	**CPU time**
1	**0.572891**	**0.156**
2	**0.311594**	**0.468**
3	**0.0562883**	**2.511**
4	**0.000341877**	**7.41**
5	3.59302×10^{-11}	**18.626**

Table 4.13: The Maximum Errors and CPU-Times by the (CM) Using 9-Point Scheme for h=1/16.

Iteration	$\|u - u_h\|_\infty$	**CPU time**
1	**0.846079**	**0.312**
2	**0.660144**	**4.711**
3	**0.407870**	**128.498**
4	**0.111965**	**633.879**
5	**0.00258994**	**1784.17**
6	3.20283×10^{-8}	**5025.67**
7	3.59302×10^{-11}	**18868.6**
8	7.1907×10^{-13}	**29674.13**

The Figures 4.5 – 4.7 compare the convergency between the (NM) and (CM) with respect to the errors in maximum norm per iteration, obtained for the solution of the model problem using 9-point scheme.

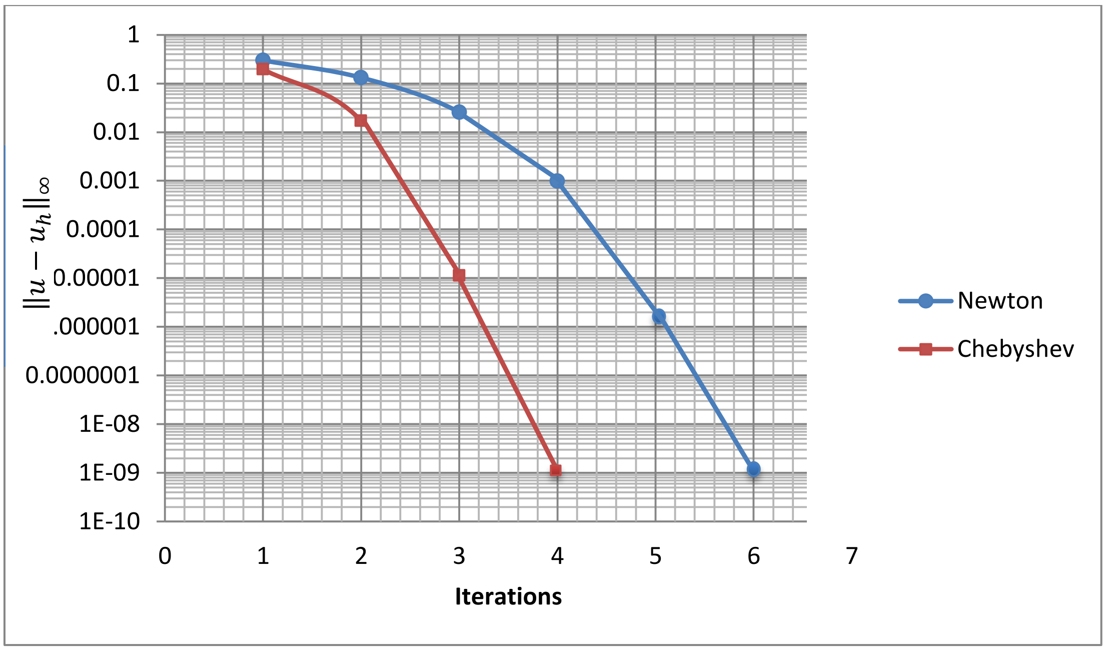


Figure 4.5: Convergency Comparison of (NM) and (CM) Using 9-Point Scheme with h=1/4.

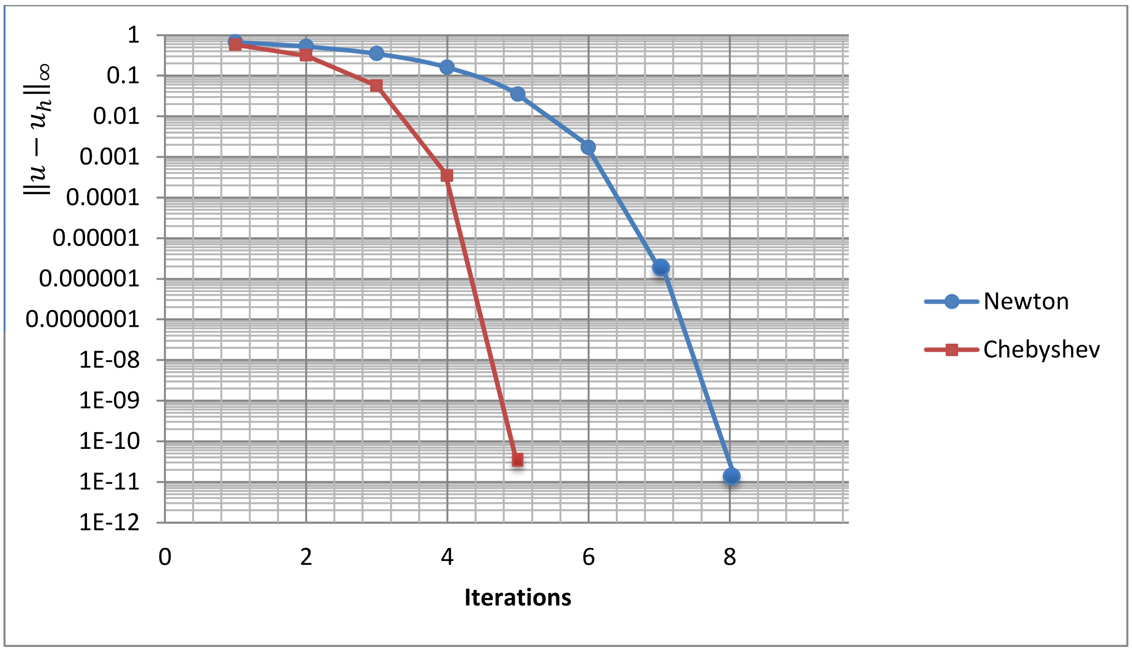


Figure 4.6: Convergency comparison of (NM) and (CM) Using 9-Point Scheme with h=1/8.

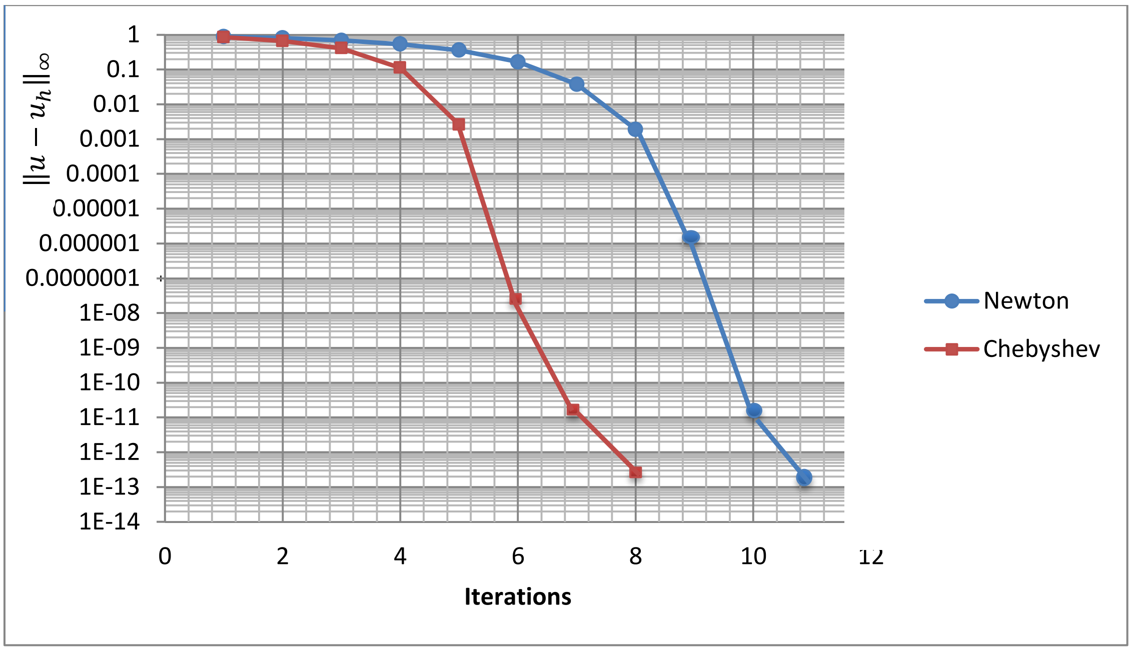


Figure 4.7: Convergency comparison of (NM) and (CM) Using 9-Point Scheme with h=1/16.

4.5 Discussions

The implementation of (NM) for the model problem using 5-point scheme requires at least 5 iterations for $h=\frac{1}{4}$, 7 iterations for $h=\frac{1}{8}$ and 9 iterations for $h=\frac{1}{16}$ in order to have the accuracy $O(h^2)$. However (CM) needs 3 iterations for $h=\frac{1}{4}$, 5 iteration for $h=\frac{1}{8}$ and 6 iterations for $h=\frac{1}{16}$ for the solution of the same system. These results can be observed from Tables 4.2 – 4.7. The comparisons of the CPU-time by both methods with respect to the iteration numbers to get an accuracy of $O(h^2)$ for mesh steps $h=\frac{1}{4},\frac{1}{8}$ and $\frac{1}{16}$ is presented in Table 4.14.

Table 4.14: Minimal Iteration Numbers and CPU-Times by (NM) and (CM), for 5-point scheme.

h	**Iteration by (NM)**	**Iteration by (CM)**	**Total no. of matrix multiplications (CPU-time) (NM)**	**Total no. of matrix multiplications (CPU-time) (CM)**
$\frac{1}{4}$	**5**	**3**	**10 - (0.064)**	**9 - (0.016)**
$\frac{1}{8}$	**7**	**5**	**14 - (1.888)**	**15 - (4.212)**
$\frac{1}{16}$	**9**	**6**	**18 - (318.46)**	**18 - (629.152)**

The minimal iteration numbers required for the approximate inverse preconditioned linear system by (NM) arised from 9-point scheme is 6, 8 and 11 for $h = \frac{1}{4}, \frac{1}{8}$ and $\frac{1}{16}$ respectively, to achieve the accuracy of $O(h^6)$. The minimal iteration numbers for $h = \frac{1}{4}, \frac{1}{8}$ and $\frac{1}{16}$ are 4, 5 and 8 respectively by (CM) in order to get the similar order of accuracy. The comparisons of the total number of matrix multiplications, and required CPU-times for these minimal iterations are given in Table 4.15.

Table 4.15: Minimal Iteration Numbers and CPU-Times by (NM) and (CM) with 9-Point Scheme.

h	**Iteration by (NM)**	**Iteration by (CM)**	**Total no. of matrix multiplications (CPU-time) (NM)**	**Total no. of matrix multiplications (CPU-time) (CM)**
$\frac{1}{4}$	**6**	**4**	**12 - (0.093)**	**12 - (0.078)**
$\frac{1}{8}$	**8**	**5**	**16 - (16.38)**	**15 - (18.377)**
$\frac{1}{16}$	**11**	**8**	**22 - (13778.6)**	**24 - (29674.13)**

From Tables 4.14 and 4.15 one can conclude that (CM) performed less iteartion than (NM) to achieve accuracy of $O(h^2)$ and $O(h^6)$ respectively. These results can also be observed from the Figures 4.2 – 4.7

Chapter 5

CONCLUSION

The effectiveness of approximate inverse preconditioners by Newton's method (NM) and Chebysheve's method (CM) are analyzed for algebraic linear systems of difference equations in solving the Dirichlet type Poisson equation on a rectangle. The cost of forming the initial approximate inverse N_0 is minimized by choosing them as diagonal matrices with main diagonal entries as the reciprocals of the original coefficient matrix entries A, arised from second order (5-point) and high order (9-point) schemes. By this choice of the initial approximate inverse the error in second norm between the identity matrix and AN_0 is obtained to be less than 1. Ofcourse as close as we take the initial approximate inverse to the exact inverse, less number of iterations will be needed. Newton and Chebyshev methods are explicit preconditioning methods, attempting to approximate A^{-1}, which is usually dense, even though the coefficient matrix A is sparse matrix. For this purpose implementation of (NM) method is realized by performing 2 matrix by matrix multiplication and (CM) is applied by performing 3 matrix by matrix multiplication, per iteration.

In this study it is shown that when started with the same initial approximate inverse, (CM) is converging faster than (NM). The CPU-times presented for the realization of these methods also depend on the performance of Mathematica, therefore these values may change if one uses a different programing language.

Finally we like to mention that these explicit approximate inverse preconditioners require several matrix by matrix multiplications at each iteration, and needs the storage of the full approximate inverse matrix. However the computational cost needed for constructing these preconditioners can be tolerated, for time dependent problems when implicit schemes are used, resulting a sequence of algebraic linear systems having same coefficient matrix and different right-hand side.

REFERENCES

[1] Buleer, I. A Numerical Method for the Solution of Two-Dimensional and Three-Dimensional Equations of Diffusion. Math. Sb. **51**, 227-238, (1960) [English Transl: Rep. BNL-TR-551, Brookhaven National Laboratory, Upton, New York, 1973].

[2] Varga, S. Matrix Iterative Analysis. Englewood Cliffs, N. J.: Prentice Hall, (1962).

[3] Oliphant, T. A. An Extrapolation Process for Solving Linear Systems. Quart. Appl. Math. **20**, 257-267, (1962).

[4] Dupont, T. R. P. Kendall and H. H. Rachford, Jr. An Approximate Factorization Procedure for Solving Self-Adjoint Elliptic Difference Equation. SIAM J. Numer. Anal. **5**, 554-573, (1968).

[5] Dupont, T. A Factorization Procedure for the Solving of Elliptic Difference Equations, SIAM J. Numer. Anal. **5**, 753-782, (1968).

[6] Woznicki, Z. Two-Sweep Iterative Method for Solving Large Linear Systems and Their Approximations to the Numerical Solution of Multi-Group Multi-Dimensional Newton Diffusion Equation. Report No. 1447/Cytronet/PM/A Dissertation Institude of Nuclear Research, Warszawa, (1973).

[7] Axelsson, O. S. Brinkkemper, and V. P. ll'in On Some Versions of Incomplete

Blockmatrix Factorization Iterative Methods. Lin. Alg. Appl. **58**, 3-15, (1984).

[8] Hestenes, M. R. and E. Stiefel, Methods of Conjugate Gradients for Solving Linear Systems. J. Res. Nat. Bur. Standards Sect. B. **49**, 409-436, (1952).

[9] Axelsson, O. Iterative Solution Methods, Cambridge University Press, (1996).

[10] Lonczos, C. An Iterative Method for the Solution of the Eigenvalue Problem of Linear Differential and Integral Operations. J. Res. Nat.Bur. Standards **45**, 255-282, (1950).

[11] Lanczos, C. Solutions of Systems of Linear Equations by Minimized Iterations. J. Res. Nat. Bur. Standards Sect. B. **49**, 33-53, (1952).

[12] Concus, C. G. H. Golub, and G. Mevrant Block Preconditioning for the Conjugate Gradient Method, SLAM J. SCI. Stat. Comp. **6**, 152-220, (1985).

[13] Axelsson, O. Incomplete Blockmatrix Factorization Preconditioning Methods. The Ultimate Answer, J. Comp. Appl. Math. **12**, **13**, 3-18, (1985).

[14] Axelsson O. and B. Polman, on Approximate Factorization Methods for Block Matrices Suitable for Vector and Parallel Processors. Lin. Alg. Appl. **77**, 3-26, (1986).

[15] Saberi Nazafi, H.: M. Shams Solery, Computational Algorithms for Computing the Inverse of a Square Matrix, Quasi-Inverse of a Non-Square Matrix and Block Matrices, Applied Mathematics and Computations, **183**, 539-550, (2006).

[16] H.ov-Biao Li, Ting-Zho Huang, Yong Zhang, Xing-Ping Liv, Tong-Xiong Gu, Chebyshev-Type Methods and Preconditioning Techniques, Applied Mathematics and Computation, **218**, 260-270, (2011).

[17] Toutounian, F. Soleymani, F. An Iterative Method for Computing the Approximate Inverse of a Square Matrix and the Moore-Penrose Inverse of a Non-Square Matrix, Applied Mathematic Computation, **224**, 671-680, (2013).

[18] Soleymani, F. On a Fast Iterative Method for Approximate Inverse of Matrices, Commun. Korean Math. Soc **28** No. 2, 407-418, (2013).

[19] Schulz, G. Iterative Berechning der Reziproken Matrix, Z.Angrew. Math. Mech. **13**, 57-57, (1933).

[20] Amat, S. Busquier, S. J. M. Gvtiérro, Geometric Construction of Iterative Functions to Solve Nonlinear Equations, Journal of Computational and Applied Mathematics, **157**, 197-205, (2003).

[21] Traulo, J. F. Iterative Methods for Solution of Equations, Prentice Hall, Englewood. Cliffs, N. J, (1964).

[22] Scavo, T. R. Thoos, J. B. On the Geometry of Halley's Method, Amer. Math. Monthly, **102**, 417426, (1995).

[23] Gutiérrez, J. M. Hermández, M. A. An acceleration of Newton's Method Super-Halley Method, Appl. Math. Compute. **117**, 223-239, (2001).

[24] Phillips, G. M. Taylor, P. J. Theory and Applications of Numerical Analysis, Academy Press, (1980).

[25] Hernández, M. A. Newton-Raphson Method and Convexity, Zb. Rad. Prirod. Mat. Fak. Ser. Mat. **22**(1), 159-166, (1993).

[26] Gutierrez, J. M. Henćrdez, M. A. An Acceleration of Newton's Method Super-Halley Method, Appl. Math. Comput. **117**, 223-293, (2001).

[27] Ortega, J. M. Kheinboldt, W. C. Iterative Solution of Nonlinear Equations in Several Variables, Academic Press, New York, (1970).

[28] Ezquerro, J. A. Gutiérrez, J. M. Hernández, M. A. Salanova, M. A. Resolución de Ecuaciones de Riccati Algebraicas Mediante Procesos iterations de Tencer Order, Proceedings of XW CEDYA-VI CMA, Las Palmas de Gran Canarias, Spain, 1069-1079 (in Spanish), 1999.

[29] Martinez, J. M. Practical Quasi-Newton Methods for Solving Nonlinear Systems, T. Comput. Appl. Math. **124** (1-2) 97-121, (2000).

[30] Hernandez, M. A. Chebyshev's, Approximation Algorithms and Applications, Computers and Mathematics with Applications **41** 433-445, (2001).

[31] Potra, F. A. Pták, V. Nondiscrete Iteration and Iterative Process, in: Research Notes in Mathematics, Vol. **103**, Pitman, Boston, (1984).

[32] Kantorovich, I. V. Functional Analysis and Applied Mathematics, Translated by C. D. Benster, National Bureau of Standards Report. 1509, (1952).

[33] Kantorovich, L. V. and Akilov, G. P. Functional Analysis in Normed Space, Pergamon, New York, (1964).

[34] Ortega, T. M. The Newton-Kantororich Theorem, This MONTHLY, **75** 658-660, (1968).

[35] Dennis, T. E. On the Kantororich Hypothesis for Newton's Method, SLAM J. Numer. Anal. **6** 493-507, (1969).

[36] Rall, L. B. Computational Solutions of Nonlinear Operator Equations. Wiley, New York, (1969).

[37] Amat, S. Busquier, S. Third-Order Iterative Method Under Kantorovich Conditions, J. Math. Anal, Appl. 336 243-261, (2007).

[38] Wu, X. Y. A Note on Computation Algorithm for the Inverse of a Square Matrix, Appl. Math. Comput. **187** (2) 962-964, (2007).

[39] Kantorovich, L. V. and V. I. Krylov, Approximate Methods of Higher Analysis (Noorhoff Leiden) (1988).

[40] Wofgang Hackbush, Iterative Solution of Large Sparse Systems of Equations, Springer Verlas New York Inc, (1994).