

To the Graduate Council:

I am submitting herewith a thesis written by Nian She entitled "Numerical Studies of Algorithms for Solving Some Differential Equations." I have examined the final copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Arts, with a major in Mathematics.

Laurence A. Bales

Laurence A. Bales, Major Professor

We have read this thesis
and recommend its acceptance:

Steven M. Serbin

Oliver Karate

Accepted for the Council:

Cowminkel

Vice Provost
and Dean of The Graduate School

STATEMENT OF PERMISSION TO USE

In presenting this thesis in partial fulfillment of the requirements for a Master's degree at The University of Tennessee, Knoxville, I agree that the Library shall make it available to borrowers under rules of the Library. Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgment of the source is made.

Permission for extensive quotation from or reproduction of this thesis may be granted by my major professor, or in his absence, by the Head of Interlibrary Services when, in the opinion of either, the proposed use of the material is for scholarly purposes. Any copying or use of the material in this thesis for financial gain shall not be allowed without my written permission.

Signature Alan She

Date Aug 14, 85

NUMERICAL STUDIES OF ALGORITHMS FOR
SOLVING SOME DIFFERENTIAL EQUATIONS

A Thesis
Presented for the
Master of Science
Degree
The University of Tennessee, Knoxville

Nian She
August 1985

ABSTRACT

Some iterative methods are studied for solving a stiff system of two ordinary differential equations with smooth and nonsmooth initial data. Some fundamental conditions required by Sammon [6] are verified. If the high dimensional systems are considered, two-term or three-term conjugate gradient methods can be explored. According to our examples, these two methods have no significant difference in the sense of number of iterations. Finally, the Incomplete Cholesky conjugate gradient method is used to compare with the classical conjugate gradient method. In all test problems ICCG was faster than GC .

TABLE OF CONTENTS

CHAPTER		PAGE
1	DISCRETE APPROXIMATION METHOD FOR PARABOLIC PROBLEM WITH NONSMOOTH INITIAL DATA.	1
	1.1. Introduction	1
	1.2. Numerical Results by Using Obrechhoff Method . . .	12
2	TWO-TERM AND THREE-TERM CONJUGATE GRADIENT METHODS . .	22
3	PRECONDITIONED CONJUGATE GRADIENT ALGORITHM.	32
	3.1. Introduction	32
	3.2. Incompletely Choleski Conjugate Gradient Algorithm.	34
	3.3. An Application of Incomplete Choleski Decomposition.	37
	LIST OF REFERENCES	46
	VITA	48

LIST OF TABLES

TABLE		PAGE
1	Errors for Single Step Schemes.	15
2	True Solution $z = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}$	29
3	True Solution $z = \begin{pmatrix} 1 \\ -1 \\ \vdots \\ 1 \\ -1 \end{pmatrix}$	30
4	Incomplete Conjugate Gradient Method.	40
5	Conjugate Gradient Method	41

CHAPTER 1

DISCRETE APPROXIMATION METHOD FOR PARABOLIC PROBLEM
WITH NONSMOOTH INITIAL DATA

1.1. Introduction

In this paper we consider an example which was used by Peter Sammon [5] and is related to the numerical solution of parabolic partial differential equations. This example is a stiff system of two ordinary differential equations and two different starting values are used to roughly simulate the case of smooth and non-smooth initial data for the parabolic initial boundary value problem.

If $M \geq 1$ is given, let $h^2 = 1/M$, $S_h = \mathbb{R}^2$ and $k = 1/(M-1)$. Also, let

$$\theta_h(t) = \begin{pmatrix} \cos(kt) & \sin(kt) \\ -\sin(kt) & \cos(kt) \end{pmatrix}, \quad (1.1)$$

$$L_h(t) = \theta_h(-t) \begin{pmatrix} 1 & 0 \\ 0 & M \end{pmatrix} \theta_h(t) \quad \text{for } 0 \leq t \leq 1,$$

and let $u_h(t)$ be defined by

$$\frac{d}{dt} u_h(t) + L_h(t) u_h(t) = 0, \quad u_h(0) = v_h, \quad (1.2)$$

$$\text{for } 0 \leq t \leq 1$$

Then it is easy to show that

$$u_h(t) = \vartheta_h(-t) \exp\left(-\begin{pmatrix} 1 & k \\ k & M \end{pmatrix} t\right) v_h \quad \text{for } 0 \leq t \leq 1$$

is a solution of (1.2). In fact,

$$\begin{aligned} \frac{d}{dt} u_h(t) &= \left(\frac{d}{dt} \vartheta_h(-t)\right) \exp\left(-\begin{pmatrix} 1 & -k \\ k & M \end{pmatrix} t\right) v_h - \vartheta_h(-t) \begin{pmatrix} 1 & -k \\ k & M \end{pmatrix} \exp\left(-\begin{pmatrix} 1 & -k \\ k & M \end{pmatrix} t\right) v_h \\ &= \left\{ \begin{pmatrix} -k \sin(kt) & -k \cos(kt) \\ k \cos(kt) & -k \sin(kt) \end{pmatrix} - \begin{pmatrix} \cos(kt) & -k \sin(kt) & -k \cos(kt) & -M \sin(kt) \\ \sin(kt) & +k \cos(kt) & -k \sin(kt) & +M \cos(kt) \end{pmatrix} \right\} \exp\left(-\begin{pmatrix} 1 & -k \\ k & M \end{pmatrix} t\right) v_h \\ &= \begin{pmatrix} -\cos(kt) & M \sin(kt) \\ -\sin(kt) & -M \cos(kt) \end{pmatrix} \exp\left(-\begin{pmatrix} 1 & k \\ k & M \end{pmatrix} t\right) v_h \\ &= - \begin{pmatrix} \cos(kt) & -\sin(kt) \\ \sin(kt) & \cos(kt) \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & M \end{pmatrix} \exp\left(-\begin{pmatrix} 1 & -k \\ k & M \end{pmatrix} t\right) v_h \\ &= -\vartheta_h(-t) \begin{pmatrix} 1 & 0 \\ 0 & M \end{pmatrix} \exp\left(-\begin{pmatrix} 1 & -k \\ k & M \end{pmatrix} t\right) v_h \\ &= -\vartheta_h(-t) \begin{pmatrix} 1 & 0 \\ 0 & M \end{pmatrix} \vartheta_h(t) \vartheta_h(-t) \exp\left(-\begin{pmatrix} 1 & k \\ k & M \end{pmatrix} t\right) v_h \quad (\text{since } \vartheta_h(-t) \vartheta_h(t) = I) \end{aligned}$$

$$= -\theta_h(-t) \begin{pmatrix} 1 & 0 \\ 0 & M \end{pmatrix} \theta_h(t) u_h(t) = -L_h(t) u_h(t)$$

Hence $u_h(t) = L_h(-t) \exp\left(-\begin{pmatrix} 1 & k \\ k & M \end{pmatrix} t\right) v_h$ satisfies (1.2).

To show the uniqueness, let $f(t, u_h(t)) = -L_h(t) u_h(t)$;
then

$$\frac{d}{dt} u_h(t) = f(t, u_h(t))$$

It is obvious that f is continuously differentiable. If we can show that f satisfies the Lipschitz condition, the uniqueness follows from the Picard Theorem. Indeed,

$$\begin{aligned} \|f(t, u_h(t)) - f(t, w_h(t))\| &= \| -L_h(t) u_h(t) + L_h(t) w_h(t) \| \\ &\leq \|L_h(t)\| \|u_h(t) - w_h(t)\| \quad . \end{aligned}$$

Claim: $L_h(t)$ is selfadjoint and $\|L_h(t)\| = M$.

Proof: Since $L_h(t)$ is a matrix which contains only real entries, to show the selfadjoint is equivalent to show $L_h(t) = (L_h(t))^T$:

$$(L_h(t))^T = (\theta_h(-t) \begin{pmatrix} 1 & 0 \\ 0 & M \end{pmatrix} \theta_h(t))^T = [\theta_h(t)]^T \begin{pmatrix} 1 & 0 \\ 0 & M \end{pmatrix}^T [\theta_h(-t)]^T$$

$$\begin{aligned}
&= \begin{pmatrix} \cos(kt) & -\sin(kt) \\ \sin(kt) & \cos(kt) \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & M \end{pmatrix} \begin{pmatrix} \cos(kt) & \sin(kt) \\ -\sin(kt) & \cos(kt) \end{pmatrix} \\
&= \theta_h(-t) \begin{pmatrix} 1 & 0 \\ 0 & M \end{pmatrix} \theta_h(t) = L_h(t)
\end{aligned}$$

Since $L_h(t)$ is selfadjoint and finite dimension,

$$|L_h(t)| = \max_i |\lambda_i| \quad (1.4)$$

where λ_i is the i^{th} eigenvalue of $L_h(t)$. It is easy to find all eigenvalues of $L_h(t)$.

Simplifying $|L_h(t) - \lambda I| = 0$ gives

$$\lambda^2 + M - \lambda(1 + M) = 0$$

$$(\lambda - 1)(\lambda - M) = 0$$

Hence $\lambda = 1$ and $\lambda = M$.

By assumption, $M \geq 1$; it follows that $\max_i |\lambda_i| = M$. This completes the claim. Therefore,

$$\|f(t, u_h(t)) - f(t, w_h(t))\| \leq M \|u_h(t) - w_h(t)\|$$

and f satisfies the Lipschitz condition.

It is also easily verified that $\{L_h(t)\}_{0 \leq t \leq 1}$ satisfies the following conditions:

$L_h(t)$ is positive definite for $0 \leq t \leq 1$ and (1.6)
satisfies

- (i) $\|\phi\|^2 \leq C_+(L_h(t)\phi, \phi)$ for $0 \neq \phi \in \mathbb{R}^2$
- (ii) $\sup_{\|\phi\|=1, \phi \in \mathbb{R}^2} L_h^2 \|L_h(t)\phi\| < C_B$ (uniformly in h and t)

$L_h^{(\lambda)}(t) \equiv \left(\frac{d}{dt}\right)^\lambda L_h(t)$ exists for $0 \leq t \leq 1$ and (1.7)
 $\lambda > 0$, and

$$\|L_h^{(\lambda)}(t) T_h(s)\| \leq C_I(\lambda) \text{ for } 0 \leq s, t \leq 1 \text{ and } \lambda > 0$$

where $T_h(t) = L_h^{-1}(t)$.

Note: These are fundamental conditions required by Sammon in [6] of the approximate elliptic differential operators in general equations of the form (1.2). Here we verify the conditions for the specific $L_h(t)$ defined above.

Proof: To show that $L_h(t)$ is positive definite, let $x = (x_1, x_2)^T$ with $\|x\| = 1$. We need only to show that $(L_h(t)x, x) > 0$.

Indeed, since $L_h(t)$ is selfadjoint, we have that

$$\min_i \{\lambda_i\} = \inf_{\|x\|=1} (L_h(t)x, x) \quad i = 1, 2$$

But $\min_i \{\lambda_i\} = 1$ from the previous calculation; hence $L_h(t)$ is positive definite.

To prove (1.6)(i), let us consider $\|\phi\| = 1$ first. Then

$$1 = \inf_{\|\phi\|=1} (L_h(t)\phi, \phi)$$

and

$$\|\phi\|^2 = 1 = \inf_{\|\phi\|=1} (L_h(t)\phi, \phi) \leq (L_h(t)\phi, \phi) . \quad (1.8)$$

In general, for $\|\phi\| \neq 0$, it follows from (1.8) that

$$\begin{aligned} \|\phi\|^2 &= \|\phi\|^2 \left(\frac{\phi}{\|\phi\|}, \frac{\phi}{\|\phi\|} \right) \leq \|\phi\| (L_h(t) \frac{\phi}{\|\phi\|}, \frac{\phi}{\|\phi\|}) \\ &= (L_h(t)\phi, \phi) . \end{aligned}$$

Taking $C_+ = 1$ we obtain that

$$\|\phi\|^2 \leq C_+ (L_h(t)\phi, \phi) .$$

To show (1.8)(ii) above, since $M = \sup_{\|\phi\|=1} \|L_h(t)\phi\|$;

$$\sup_{\|\phi\|=1, \phi \in \mathbb{R}^2} h^2 \|L_h(t)\phi\| = h^2 \sup_{\|\phi\|=1, \phi \in \mathbb{R}^2} \|L_h(t)\phi\| = h^2 M = 1$$

Taking $C_B = 1 + \varepsilon$, where $\varepsilon > 0$ is independent in h and t we obtain

$$\sup_{\|\phi\|=1, \phi \in \mathbb{R}^2} h^2 \|L_h(t)\| < c_B .$$

This completes the proof of (1.6).

To show (1.7) holds, we rewrite $L_h(t)$ as

$$L_h(t) = \begin{pmatrix} \cos^2(kt) + M \sin^2(kt) & (1 - M) \cos(kt) \sin(kt) \\ (1 - M) \cos(kt) \sin(kt) & \sin^2(kt) + M \cos^2(kt) \end{pmatrix} .$$

Then

$$\begin{aligned} L_h'(t) &= \begin{pmatrix} -2k \cos(kt) \sin(kt) + 2kM \sin(kt) \cos(kt) & (1 - M)k \cos(2kt) \\ (1 - M)k \cos(2kt) & 2k \sin(kt) \cos(kt) - 2kM(\sin(kt) \cos(kt)) \end{pmatrix} \\ &= \begin{pmatrix} -k \sin(2kt) + kM \sin(2kt) & (1 - M)k \cos(2kt) \\ (1 - M)k \cos(2kt) & k \sin(2kt) - kM \sin(2kt) \end{pmatrix} \\ &= \begin{pmatrix} (M - 1)k \sin(2kt) & (1 - M)k \cos(2kt) \\ (1 - M)k \cos(2kt) & (1 - M)k \sin(2kt) \end{pmatrix} \\ &= k(1 - M) \begin{pmatrix} -\sin(2kt) & \cos(2kt) \\ \cos(2kt) & \sin(2kt) \end{pmatrix} . \end{aligned}$$

$$L_h''(t) = 2k^2(1 - M) \begin{pmatrix} -\cos(2kt) & -\sin(2kt) \\ -\sin(2kt) & \cos(2kt) \end{pmatrix}$$

$$= 2k^2(1 - M) \begin{pmatrix} -\sin(\frac{\pi}{2} + 2kt) & \cos(\frac{\pi}{2} + 2kt) \\ \cos(\frac{\pi}{2} + 2kt) & \sin(\frac{\pi}{2} + 2kt) \end{pmatrix}$$

$$L_h'''(t) = 2^2k^3(1 - M) \begin{pmatrix} \sin(2kt) & -\cos(2kt) \\ -\cos(2kt) & -\sin(2kt) \end{pmatrix}$$

$$= 2^2k^3(1 - M) \begin{pmatrix} -\sin(\frac{\pi}{2} \cdot 2 + 2kt) & \cos(\frac{\pi}{2} \cdot 2 + 2kt) \\ \cos(\frac{\pi}{2} \cdot 2 + 2kt) & \sin(\frac{\pi}{2} \cdot 2 + 2kt) \end{pmatrix}$$

$$L_h^{(4)}(t) = 2^3k^4 \begin{pmatrix} \cos(2kt) & \sin(2kt) \\ \sin(2kt) & -\cos(2kt) \end{pmatrix}$$

$$= 2^3k^4(1 - M) \begin{pmatrix} \sin(\frac{\pi}{2} \cdot 3 + 2kt) & \cos(\frac{\pi}{2} \cdot 3 + 2kt) \\ \cos(\frac{\pi}{2} \cdot 3 + 2kt) & \sin(\frac{\pi}{2} \cdot 3 + 2kt) \end{pmatrix}$$

⋮

Continuing in this way, we obtain

$$L_h^{(\ell)}(t) = 2^{\ell-1} k^{\ell} (1 - M)$$

$$\begin{pmatrix} -\sin(\frac{\pi}{2}(\ell - 1) + 2kt) & \cos(\frac{\pi}{2}(\ell - 1) + 2kt) \\ \cos(\frac{\pi}{2}(\ell - 1) + 2kt) & \sin(\frac{\pi}{2}(\ell - 1) + 2kt) \end{pmatrix}$$

Hence, by induction, $L_h^{(\ell)}(t)$ exists for all $\ell > 0$ and $0 \leq t \leq 1$.

Let

$$Q = \begin{pmatrix} -\sin\alpha & \cos\alpha \\ \cos\alpha & \sin\alpha \end{pmatrix},$$

where $\alpha = \frac{\pi}{2}(\ell - 1) + 2kt$. Then set $|Q - \lambda I| = 0$ and we obtain

$$-\sin^2\alpha + \lambda \sin\alpha - \lambda \sin\alpha + \lambda^2 - \cos^2\alpha = 0.$$

This implies that $\lambda^2 - 1 = 0$. Hence $\lambda = \pm 1$, and $\|Q\| = 1$.

Thus

$$\|L_h^{(\ell)}(t)\| = 2^{\ell-1} k^{\ell} |1 - M| \|Q\| = 2^{\ell-1} k^{\ell-1} = (2k)^{\ell-1}.$$

On the other hand, the eigenvalues of $T_h = L_h^{-1}(t)$ are 1 and $\frac{1}{M}$ since $M \geq 1$, $\|T_h(s)\| = 1$. Therefore

$$\|L_h^{(\ell)}(t) T_h(s)\| \leq \|L_h^{(\ell)}(t)\| \|T_h(s)\| = (2k)^{\ell-1}, \ell > 0.$$

Taking $G(\ell) = (2k)^{\ell-1}$ we obtain (1.7). This completes the proof.

In the case $\ell = 0$ it can be shown that $\|L_h(t) T_h(s)\|_2$ is uniformly bounded. In fact, for $\ell = 0$,

$$\begin{aligned} L_h(t) T_h(s) &= \theta_h(-t) \begin{pmatrix} 1 & 0 \\ 0 & M \end{pmatrix} \theta_h(t) \theta_h(-s) \begin{pmatrix} 1 & 0 \\ 0 & \frac{1}{M} \end{pmatrix} \theta_h(s) \\ &= \theta_h(-t) \begin{pmatrix} 1 & 0 \\ 0 & M \end{pmatrix} \begin{pmatrix} \cos(kt) & \sin(kt) & \cos(ks) & -\sin(ks) \\ -\sin(kt) & \cos(kt) & \sin(ks) & \cos(ks) \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & \frac{1}{M} \end{pmatrix} \theta_h(s) \\ &= \theta_h(-t) \begin{pmatrix} 1 & 0 \\ 0 & M \end{pmatrix} \begin{pmatrix} \cos k(t-s) & \sin k(t-s) \\ -\sin k(t-s) & \cos k(t-s) \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & \frac{1}{M} \end{pmatrix} \theta_h(s) \\ &= \theta_h(-t) \begin{pmatrix} \cos k(t-s) & \frac{1}{M} \sin k(t-s) \\ -M \sin k(t-s) & \cos k(t-s) \end{pmatrix} \theta_h(s) \\ &= \theta_h(-t) P(t-s) \theta_h(s) \end{aligned}$$

where

$$P(t-s) = \begin{pmatrix} \cos k(t-s) & \frac{1}{M} \sin k(t-s) \\ -M \sin k(t-s) & \cos k(t-s) \end{pmatrix}$$

Hence

$$\|L_h(t) T_h(s)\|_2 \leq \|\theta_h(-t)\|_2 \|P(t-s)\|_2 \|\theta_h(s)\|_2. \quad (1.9)$$

In order to complete the proof, we need the following Lemma.

If A is an $m \times n$ matrix, then

$$\begin{aligned} \text{(i)} \quad & \frac{1}{\sqrt{n}} \|A\|_2 \leq \|A\|_1 \leq \sqrt{m} \|A\|_2 \\ \text{(ii)} \quad & \frac{1}{\sqrt{m}} \|A\|_2 \leq \|A\|_\infty \leq \sqrt{n} \|A\|_2. \end{aligned}$$

Since $\|\theta_h\|_\infty = \|\theta_h\|$, it follows that

$$\|\theta_h(-t)\|_2 \leq \sqrt{2} \|\theta_h\|_\infty \leq 2\sqrt{2}, \quad \|\theta_h(s)\|_2 \leq 2\sqrt{2}.$$

Moreover,

$$\|P(t-s)\|_\infty = \|P(t-s)\|_1 \leq 1 + |M \sin k(t-s)|$$

$$= 1 + \left| M \sin \frac{(t-s)}{(M-1)} \right| = 1 + \left| \frac{\sin \frac{1-s}{M-1}}{\frac{1}{M}} \right|$$

$$\leq 1 + \left| \frac{\sin \frac{t-s}{M-1}}{\frac{1}{M-1}} \right| = 1 + |t-s| \left| \frac{\sin \frac{t-s}{M-1}}{\frac{t-s}{M-1}} \right|$$

$$\leq 1 + 2 \left| \frac{\sin \beta}{\beta} \right| \quad \text{since } 0 \leq t, s \leq 1, \text{ where } \beta = \frac{t-s}{M-1}.$$

But $\frac{\sin \beta}{\beta} \leq 1$ for all β , so

$$\|P(t-s)\|_{\infty} = \|P(t-s)\|_1 \leq 1 + \frac{2\tau M}{M-1}.$$

If $M \geq 2$, then $\|P(t-s)\|_{\infty} = \|P(t-s)\|_1 \leq 1 + \frac{2 \cdot 2}{2-1} = 1 + 4\tau$.

Therefore, $\|L_h(t) T_h(s)\|_2 \leq 1 + 4\tau$.

1.2. Numerical Results by Using Obrechhoff Method

Consider the differential equation

$$y' = A(t)y + f(t) \quad (1.10)$$

where $A(t)$ is a 2×2 matrix and $f(t) \in C'(\mathbb{R}^2)$, then

$$\begin{aligned} y'' &= A'(t)y + A(t)y' + f' \\ &= A'(t)y + A(t)(A(t)y + f(t)) + f'(t) \\ &= ((A(t))^2 + A'(t))y + A(t)f(t) + f'(t) \end{aligned}$$

the general Obrechhoff method is a multistep-multiderivative method given by

$$\sum_{j=0}^m \alpha_j y_{j+n} = \sum_{i=1}^k k^i \sum_{j=0}^m \beta_{ij} y_{m+n}^{(i)}.$$

In this example, we choose a specific one

$$y_{n+1} + q_1 k y'_{n+1} + q_2 k^2 y''_{n+1} = y_n + p_1 k y'_n + p_2 k^2 y''_n \quad (1.11)$$

with coefficients p_1, p_2, q_1, q_2 to be specified. Now, we apply Obrechhoff method to (1.2). Recall from the previous chapter,

$$L_h(t) = \theta_h(-t) \begin{pmatrix} 0 & 1 \\ 0 & M \end{pmatrix} \theta_h(t) \quad \text{for } 0 \leq t \leq 1$$

By simplifying, we obtain

$$L_h(t) = \begin{pmatrix} 1 + (M - 1) \sin^2(kt) & \frac{1 - M}{2} \sin(2kt) \\ \frac{1 - M}{2} \sin(2kt) & 1 + (M - 1) \cos^2(kt) \end{pmatrix}$$

$$L'_h(t) = \begin{pmatrix} (M - 1) k \sin(2kt) & (1 - M) k \cos(2kt) \\ (1 - M) k \cos(2kt) & -(M - 1) k \sin(2kt) \end{pmatrix}$$

Set $y = u_h(t)$ and $A(t) = -L_h(t)$; (1.2) becomes

$$y'(t) = A(t) y(t) \quad (1.12)$$

and (2.2) becomes

$$[I + kq_1 A(t_{n+1}) + k^2 q_2 (A^2(t_{n+1}) + A'(t_{n+1})) y_{n+1}] \quad (1.13)$$

$$= [I + kp_1 A(t_n) + k^2 p_2 (A^2(t_n) + A'(t_n)) y_n]$$

Let

$$\tilde{Q}_{n+1} = I + kq_1 A(t_{n+1}) + k^2 q_2 (A^2(t_{n+1}) + A'(t_{n+1}))$$

and

$$\tilde{P}_n = I + kp_1 A(t_n) + k^2 p_2 (A^2(t_n) + A'(t_n)) .$$

Then (2.4) becomes

$$Q_{n+1} y_{n+1} = P_n y_n$$

$$y_{n+1} = [Q_{n+1}]^{-1} P_n y_n \quad (1.14)$$

The results of applying the Obrechhoff Method are given in Table 1. Such calculations were done in double precisions on an IBM 360/370 system at The University of Tennessee. Two types of initial data were used: $v_h^{(1)} \equiv \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ was to corresponding to smooth initial data and $u_h^{(2)} \equiv \begin{pmatrix} 1 \\ 1 \end{pmatrix}$ was to correspond to less well behaved initial data.

The rate on Table 1 was taken in the following way:

$$\text{Rate} = \frac{\ln(\text{error using } N_1 \text{ stops/error using } N_2 \text{ stops})}{\ln(N_2/N_1)}$$

The coefficients used in Table 1 were chosen as following: Consider $y' = \lambda y$ where λ is a scalar, then $y(t) = y(0) e^{\lambda t}$:

TABLE 1
ERRORS FOR SINGLE STEP SCHEMES

	M	Error for $VH1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$	Rate	Error for $VH2 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$	Rate
$V=1:$ $\frac{p}{q} = \frac{1}{1+x}$	5	0.3945 D-1	0.88	0.7237 D-1	1.32
	10	0.1929 D-1		0.2479 D-1	1.14
	20	0.9446 D-2	0.96	0.1057 D-1	1.07
	30	0.6245 D-2	0.98	0.6713 D-2	1.05
	40	0.4663 D-2	0.99	0.4918 D-2	1.04
	50	0.3721 D-2	0.99	0.3880 D-2	
$V=2:$ $\frac{p}{q} = \frac{1}{1+x+\frac{x^2}{2}}$	5	0.3144 D-2	1.86	0.1363 D-1	2.12
	10	0.6911 D-3		0.2440 D-2	2.05
	20	0.1629 D-3	1.93	0.5271 D-3	2.03
	30	0.7097 D-4	1.97	0.2236 D-3	2.02
	40	0.3952 D-4	1.98	0.1229 D-3	2.02
	50	0.2514 D-4	1.98	0.7762 D-4	
$V=3:$ $\frac{p}{q} = \frac{1+Bx+Bx^2}{(1+x)^2}$ $= \frac{1}{2}(1+\frac{1}{3})$ $p_1 = 2 - 1$ $p_2 = 2 - 2 + \frac{1}{2}$	5	0.8958 D-3	3.37	0.1153 D-1	2.30
	10	0.5843 D-4		0.1770 D-2	2.16
	20	0.5438 D-5	3.18	0.3526 D-3	2.09
	30	0.1467 D-5	3.10	0.1457 D-3	2.06
	40	0.5910 D-6	3.07	0.7904 D-4	2.05
	50	0.2944 D-6	3.05	0.4951 D-4	
$V=4:$ $\frac{p}{q} = \frac{1-\frac{1}{2}x+\frac{1}{12}x^2}{1+\frac{1}{2}x+\frac{1}{12}x^2}$	5	0.2509 D-4	4.47	0.4181 D-3	2.64
	10	0.6664 D-6		0.4915 D-4	2.31
	20	0.2799 D-7	4.24	0.8755 D-5	2.17
	30	0.4863 D-8	4.14	0.3494 D-5	2.12
	40	0.1444 D-8	4.10	0.1864 D-5	2.09
	50	0.5700 D-9	4.08	0.1156 D-5	

$$y''(t) = \lambda y' = \lambda^2 y$$

Substituting y , y' and y'' to (2.2) to obtain

$$(1 + q_1 k \lambda + q_2 k^2 \lambda^2) y_{n+1} = (1 + p_1 k \lambda + p_2 k^2 \lambda^2) y_n$$

$$y_{n+1} = \frac{1 + p_1 k + p_2 k^2 \lambda^2}{1 + q_1 k + q_2 k^2 \lambda^2} y_n ;$$

hence

$$e^x \approx \frac{1 + p_1 x + p_2 x^2}{1 + q_1 x + q_2 x^2} = \frac{p(x)}{q(x)}$$

where p 's and q 's are chosen from the Pade Table except for $v = 3$. In that case, we use a specific approximation to $\frac{p(x)}{q(x)}$, where the quadratic denominator is the square of a linear factor.

From Table 1 we can see that the convergent rate for smooth initial data is optimal in the sense that it is best possible given the degree of approximation of $\frac{p(x)}{q(x)}$ to e^x ; while for nonsmooth initial data, it can not be improved under the conditions listed in (1.6) and (1.7).

Finally, in this chapter, we will point out a computational method of applying Obrechhoff method to compare with the true solution. By a previous claim, we know that

$$y(t) = e_h(-t) \exp\left(-\left(\frac{1}{k} - \frac{k}{M}\right)t\right) v_h \quad \text{for } 0 \leq t \leq 1 \quad (1.15)$$

is the solution of (1.12). In the following, a particular technique is employed to satisfy the computation and also to explain the smoothness of the initial data used in the example.

Let

$$B = \begin{pmatrix} -1 & k \\ -k & -M \end{pmatrix}, \quad v_h = \begin{pmatrix} v_h^1 \\ v_h^2 \end{pmatrix}$$

then the eigenvalues of B are given by

$$\lambda = \frac{-(1+M) \pm \sqrt{(1-M)^2 - 4k^2}}{2}$$

It is obvious that $\lambda_1 \neq \lambda_2$.

Let $u = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}$ be an eigenvector corresponding to λ_1 ; then

$$Bu = \lambda_1 u$$

$$\begin{aligned} e^{Bt}u &= (I + Bt + \frac{B^2 t^2}{2!} + \dots + \frac{B^n t^n}{n!} + \dots)u \\ &= u + But + \frac{B^2 ut^2}{2!} + \dots + \frac{B^n ut^n}{n!} + \dots \\ &= u + \lambda_1 ut + \frac{\lambda_1^2 ut^2}{2!} + \dots + \frac{\lambda_1^n ut^n}{n!} + \dots \\ &= (1 + \lambda_1 t + \frac{\lambda_1^2 t^2}{2!} + \dots + \frac{\lambda_1^n t^n}{n!} + \dots)u \\ &= e^{\lambda_1 t} u \end{aligned}$$

Similarly, let $v = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}$ be an eigenvector corresponding to λ_2 .

We have

$$e^{Bt} v = e^{\lambda_2 t} v$$

since $\lambda_1 \neq \lambda_2$, u and v are linearly independent. So u_h can be represented as the linear combination of u and v , i.e.

$$u_h = \alpha_1 u + \alpha_2 v \quad (1.16)$$

Hence

$$\begin{aligned} e^{Bt} u_h &= e^{Bt} (\alpha_1 u + \alpha_2 v) = \alpha_1 e^{\lambda_1 t} u + \alpha_2 e^{\lambda_2 t} v \\ &= e^{\lambda_1 t} (\alpha_1 u) + e^{\lambda_2 t} (\alpha_2 v) \end{aligned} \quad (1.17)$$

Now the problem reduces to how to choose α_1 and α_2 . Since $(B - \lambda_1 I)u = 0$, we have

$$-(1 + \lambda_1)u_1 + k u_2 = 0 \quad (1.18)$$

$$-k u_1 - (M + \lambda_1)u = 0$$

The equations (1.18) are linearly dependent. We can choose $u_2 = \beta_1$ where β_1 is arbitrary; thus

$$u_1 = \frac{-(M + \lambda_1)\beta_1}{k}$$

Hence

$$u = \beta_1 \begin{pmatrix} \frac{-(M + \lambda_1)}{k} \\ 1 \end{pmatrix} \quad (1.19)$$

Similarly, we can choose $u_2 = \beta_2$ and obtain

$$v = \beta_2 \begin{pmatrix} \frac{-(M + \lambda_2)}{k} \\ 1 \end{pmatrix} \quad (1.20)$$

Substitute (1.19) and (1.20) to (1.16) and choose $\beta_1 = \beta_2 = 1$, and we have

$$v_h = \beta_1 \begin{pmatrix} \frac{-(M + \lambda_1)}{k} \\ 1 \end{pmatrix} + \alpha_2 \begin{pmatrix} \frac{-(M + \lambda_2)}{k} \\ 1 \end{pmatrix} \quad (1.21)$$

If

$$\lambda_1 = \frac{-(1 + M) + \sqrt{(1 - M)^2 - 4k^2}}{2} \approx -1$$

$$\lambda_2 = \frac{-(1 + M) - \sqrt{(1 - M)^2 - 4k^2}}{2} \approx -M$$

then

$$\lambda_2 - \lambda_1 = -\sqrt{(1 - M)^2 - 4k^2} \approx -(M - 1)$$

$$M + \lambda_1 \approx M - 1 = \frac{1}{k}$$

and

$$M + \lambda_2 = \frac{M - 1 - \sqrt{(1 - M)^2 - 4k^2}}{2} \approx \frac{k^2}{M - 1} = k$$

$$v_h \approx \alpha_1 \begin{pmatrix} \frac{1}{k^2} \\ 1 \end{pmatrix} + \alpha_2 \begin{pmatrix} -k^2 \\ 1 \end{pmatrix} \quad (1.22)$$

Therefore, by solving the system of equations:

$$\begin{cases} \frac{-\alpha_1}{k^2} - \alpha_2 k^2 = v_h^1 \\ \alpha_1 + \alpha_2 = v_h^2 \end{cases} \quad (1.23)$$

We obtain α_1 and α_2 approximately. In particular, for

$$v_h = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad \begin{pmatrix} \alpha_1 \\ \alpha_2 \end{pmatrix} \approx \begin{pmatrix} -2k^2 \\ 1 \end{pmatrix}$$

so substituting those to (1.17) we can see that for $u_h = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$

$$e^{\beta t} v_h \approx -k^2 e^{-t} \begin{pmatrix} -\frac{1}{k^2} \\ 1 \end{pmatrix} + k^2 e^{-\frac{1}{k} t} \begin{pmatrix} -1 \\ 1 \end{pmatrix}$$

$$= e^{-t} \begin{pmatrix} 1 \\ -k^2 \end{pmatrix} + e^{-\frac{1}{k}t} \begin{pmatrix} -k^2 \\ k^2 \end{pmatrix} \approx e^{-t} \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad (\text{as } k \rightarrow 0)$$

which involves only the slowly varying components, and hence simulates a smooth problem.

On the other hand, when $v_h = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$

$$e^{\beta t} v_h \approx -2k^2 e^{-t} \begin{pmatrix} -\frac{1}{k^2} \\ 1 \end{pmatrix} + e^{-\frac{1}{k}t} \begin{pmatrix} -1 \\ 1 \end{pmatrix}$$

$$= e^{-t} \begin{pmatrix} 2 \\ -2k^2 \end{pmatrix} + e^{-\frac{1}{k}t} \begin{pmatrix} -1 \\ 1 \end{pmatrix}$$

which involves substantially the rapidly varying component $e^{-\frac{1}{k}t}$ and hence simulates a rough problem.

In this chapter the example used is a 2×2 system; high dimensional systems can also be investigated. In this case, we solve $n \times n$ linear systems and for this, one may use the conjugate gradient method or incomplete Choleski conjugate gradient method that will be discussed in the latter chapters.

CHAPTER 2

TWO-TERM AND THREE-TERM CONJUGATE GRADIENT METHODS

In this chapter, we compare the computational results of the classical two-term conjugate gradient method with the three-term conjugate gradient method. We are interested in investigating the number of iterations of both methods. From the following examples, we did not find any difference of the number of the iteration between the two-term and three-term conjugate gradient methods.

We start the discussion by briefly reviewing the classical conjugate gradient method. Consider the linear equation

$$Au = b \tag{2.1}$$

where $A \in \mathbb{R}^N \times \mathbb{R}^N$ and A is symmetric positive definite. If $\bar{u} = A^{-1}b$ is the solution of (3.1), the idea to find $\bar{u}$ would be avoid computing A^{-1} and develop an iteration method which will converge in a finite number of iterations.

Define

$$F(u) = \frac{1}{2} u^T Au - b^T u \tag{2.2}$$

then

$$\begin{aligned}
 F(u) - F(\bar{u}) &= \frac{1}{2} u^T A u - b^T u - \frac{1}{2} \bar{u}^T A u + b^T u \\
 &= \frac{1}{2} u^T A u - (A\bar{u})^T u - \frac{1}{2} \bar{u}^T A \bar{u} + (A\bar{u})^T \bar{u} ,
 \end{aligned}$$

because $A\bar{u} = b$ implies that $(A\bar{u})^T u = b^T u$.

Hence

$$\begin{aligned}
 F(u) - F(\bar{u}) &= \frac{1}{2} u^T A u - \frac{1}{2} (A\bar{u})^T u - \frac{1}{2} \bar{u}^T A u + \frac{1}{2} \bar{u}^T A \bar{u} \\
 &= \frac{1}{2} (u - \bar{u})^T A u - \frac{1}{2} (A\bar{u})^T (u - \bar{u}) \\
 &= \frac{1}{2} [(u - \bar{u})^T A (u - \bar{u})] \\
 F(u) - F(\bar{u}) &= \frac{1}{2} (u - \bar{u})^T A (u - \bar{u}) \tag{2.3}
 \end{aligned}$$

Therefore, to find the solution $\bar{u}$ is equivalent to minimize $F(u)$.

Since A is positive definite, it is obvious that $F(u)$ attains minimum where $u = \bar{u}$.

The first such iteration method is called Steepest Descent.

Since

$$\begin{aligned}
 F(u) &= \frac{1}{2} \sum_{i,j=1}^n a_{ij} u_i u_j + \sum_{i=1}^n b_i u_i \\
 \frac{\partial F(u)}{\partial u_i} &= \sum_{j=1}^n a_{ij} u_j - b_i
 \end{aligned}$$

Thus

$$\text{grad } F(u) = Au - b$$

Let

$$r = b - Au$$

Steepest descent suggests that iterations move in the direction of

$$r = \text{grad } F(u) \quad .$$

Set

$$u_{n+1} = u_n + \lambda_n r_n \quad .$$

We choose λ_n to minimize $F(u_{n+1})$. We know from calculus that $F(u_{n+1})$ can be minimized by setting $\left. \frac{\partial F}{\partial \lambda} \right|_{\lambda=\lambda_n} = 0$, hence it is easy to show that

$$\lambda_n = \frac{(v_n, v_n)}{(v_n, Ar_u)} \quad . \quad (2.4)$$

The classical conjugate gradient method comes from the modification of Steepest Descent by modifying the choice of directions. The Algorithm is listed in the following.

Two-Term Conjugate Gradient Algorithm

u_0 is arbitrary

$$r_0 = b - Au_0$$

$$p_0 = r_0$$

$$\delta = (r_0, r_0)$$

for $n \geq 0$

$$\lambda_n = \frac{\delta_n}{(Ap_n, p_n)}$$

$$u_{n+1} = u_n + \lambda_n p_n$$

$$r_{n+1} = r_n - \lambda_n Ap_n$$

$$\delta_{n+1} = (r_{n+1}, r_{n+1})$$

$$\beta_n = \frac{\delta_{n+1}}{\delta_n}$$

$$p_{n+1} = r_{n+1} + \beta_n p_n$$

It can be shown that $r_0, r_1, \dots$ and the direction vectors $p_0, p_1, \dots$ satisfy the relations

$$(r_i, r_j) = 0 \quad \text{for } i \neq j$$

$$(P_i, AP_j) = 0 \quad \text{for } i \neq j \quad (2.5)$$

$$(r_i, AP_j) = 0 \quad \text{for } i \neq j \text{ and } i \neq j + 1$$

Thus the residual vectors $r_0, r_1, \dots$ are mutually orthogonal and the direction vectors $P_0, P_1, \dots$ are mutually A -conjugate. From the first relation in (3.5), it follows that $r_s = 0$ for some $s \leq N$. Thus the two-term conjugate gradient method converges, in the absence of rounding errors, in at most N iterations. It also can be shown that the error vector $\epsilon_n = u_n - \bar{u}$ associated with the conjugate gradient method satisfies

$$\|\epsilon_{n+1}\|_2 < \|\epsilon_n\|_2 \quad \text{whenever } \epsilon_n \neq 0. \quad (2.6)$$

Three-Term Conjugate Gradient Algorithm

u_0 is arbitrary

$$r_0 = b - Au_0$$

$$\gamma_1 = \frac{(r_0, r_0)}{(r_0, Ar_0)}$$

$$u_1 = u_0 + \gamma_1 r_0$$

$$r_1 = r_0 - \gamma_1 A r_0$$

$$\rho_1 = 1$$

For $n \geq 1$

$$\gamma_{n+1} = \frac{r_n, r_n}{(r_n, A r_n)}$$

$$\rho_{n+1} = \left(1 - \frac{\gamma_{n+1}}{\gamma_n} \frac{r_n, r_n}{(r_{n-1}, r_{n-1})} \frac{1}{\rho_n}\right)^{-1}$$

$$u_{n+1} = \rho_{n+1} \{ \gamma_{n+1} r_n + u_n \} + (1 - \rho_{n+1}) u_{n-1}$$

$$r_{n+1} = \rho_{n+1} \{ -\gamma_{n+1} A r_n + r_n \} + (1 - \rho_{n+1}) r_{n-1}$$

This algorithm may be regarded as a polynomial acceleration procedure based on the RF method. The details can be found in Hageman and Young [6].

The examples we used to test these two algorithm are the ill-conditional tridiagonal matrix

$$A = \begin{pmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & 0 & -1 & 2 & \\ & & & \ddots & -1 & 2 & -1 \\ & & & & -1 & 2 & \\ & & 0 & & & -1 & 2 \end{pmatrix} \quad \text{where } A \in \mathbb{R}^N \times \mathbb{R}^N$$

The eigenvalues of A are given by

$$\lambda_k = 4 \sin^2 \left[\frac{1}{2} \left(\frac{k\pi}{N+1} \right) \right], \quad k = 1, 2, \dots, N \quad (2.7)$$

The condition number $K(A)$ is approximately evaluated as

$$K(A) \approx \frac{\max_i \lambda_i}{\min_i \lambda_i} \approx \frac{4}{4 \left(\frac{1}{2} \frac{\pi}{N+1} \right)^2} =$$

$$\frac{16(N+1)^2}{4(\pi)^2} = \frac{4(N+1)^2}{\pi^2}$$

If $N = 100$, $K(A) \approx 4134$.

The computational results are shown in Tables 2-3. The true solutions of $Ax = b$ for different b are chosen as $(1, 1, \dots, 1)^T$, $(-1, -1, \dots, -1)^T$, $(-1, 1, -1, 1, \dots)^T$ and $(1, -1, 1, -1, \dots)^T$ respectively. Also, those solutions are set in odd and even order. From those tables we find that the number of iterations

TABLE 2

$$\text{TRUE SOLUTION } z = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}$$

	Number of Unknowns	Number of Iterations	CPU Time	Tolerance
Two-Term Conjugate Gradient Method	100	50	0.82	10^{-9}
	200	100	3.15	
	500	250	19.37	
	1000	500	77.36	
	2000	1000	308.79	
	5000	2500	1939.53	
Three-Term Conjugate Gradient Method	100	50	1.08	10^{-9}
	200	100	4.13	
	500	250	25.48	
	1000	500	101.28	
	2000	1000	403.66	
	5000			

TABLE 3

$$\text{TRUE SOLUTION } z = \begin{pmatrix} 1 \\ -1 \\ \vdots \\ 1 \\ -1 \end{pmatrix}$$

	Number of Unknowns	Number of Iterations	CPU Time	Tolerance
Two-Term	100	50	0.81	10^{-9}
Conjugate Gradient	200	100	3.14	
Method	500	250	19.38	
	1000	500	77.40	
	2000	1000	310.63	
	5000	2500	1948.45	
Three-Term	100	50	1.08	10^{-9}
Conjugate Gradient	200	100	4.13	
Method	500	250	25.48	
	1000	500	101.36	
	2000	1000	405.65	
	5000	2500	2547.79	

of those two methods is the same and they all take exactly $N/2$ steps to converge.

We know that the conjugate gradient method works best when the condition number of A is not too large. This has led to many effects aimed at preconditioning A in order to reduce its condition number before the conjugate gradient method is applied. We will discuss such a method in the next chapter.

CHAPTER 3

PRECONDITIONED CONJUGATE GRADIENT ALGORITHM

3.1. Introduction

A preconditioned conjugate gradient algorithm described here was given by Meijerink and Van der Vorst [4]. Let A be a self-adjoint positive definite matrix, and A_0 be a self-adjoint positive definite preconditioning matrix of A , i.e., A_0 satisfies the estimate

$$\lambda_0(A_0\phi, \phi) \leq (A\phi, \phi) \leq \lambda_1(A_0\phi, \phi) \text{ for all } \phi \in \mathbb{R}^N. \quad (3.1)$$

Consider the equation

$$Ax = b \text{ where } A \text{ is an } N \times N \text{ matrix and } x, b \in \mathbb{R}^N \quad (3.2)$$

The algorithm is derived from the classical conjugate gradient algorithm applied to approximately solve $A_0^{-1}Ax = A_0^{-1}b$ in the inner product $(A_0\cdot, \cdot)$. It is easy to see that the preconditioned conjugate gradient algorithm will converge faster.

Let ϵ_n and $\tilde{\epsilon}_n$ be the errors of CG and Chebyshev acceleration methods respectively, for solving (4.2), where $\bar{r} = (M(A) - M(A))/ (M(A) + M(A))$ with $M(A)$ and $m(A)$ the extreme eigenvalues of A . Clearly, the convergent rate improves as $M(A)/m(A) \rightarrow 1$. Since we intend to apply CG to the system $A_0^{-1}Ax = A_0^{-1}b$, the convergence will be enhanced by selecting A_0 in such a way that $M(A_0^{-1}A)/$

$M(A_0^{-1}A)$ is as close to unity as possible. This is equivalent to saying λ_1/λ_0 in (4.1) should be as close to unit as possible. For example, if we could select $A_0^{-1} = A^{-1}$, then $\lambda_1/\lambda_0 = 1$, and we would obtain convergence in one iteration. However, this is not practical. On the other hand, if $A_0^{-1} = I$, the algorithm is exactly the same as the ordinary conjugate gradient method applied to (4.2). A preconditioned conjugate gradient algorithm is given in the following:

Preconditioned Conjugate Gradient Algorithm

x^0 is arbitrary

$$s^0 = b - Ax^0$$

$$d^0 = A_0^{-1}(b - Ax^0)$$

$$\delta_0 = (A_0^{-1}s^0, s^0)$$

for $n > 0$

$$\alpha_n = \delta_n / (Ad^n, s^n)$$

$$x^{n+1} = x^n + \alpha_n d^n$$

$$s^{n+1} = s^n - \alpha_n Ad^n$$

$$\delta_{n+1} = (A_0^{-1} s^{n+1}, s^{n+1})$$

$$\beta_{n+1} = \delta_{n+1} / \delta_n$$

$$d^{n+1} = A_0^{-1} s^{n+1} + \beta_{n+1} d^n$$

The iterations stop when $\delta_n \leq \gamma \delta_0$, where γ is a given small number.

From this algorithm we can see that the crucial problem is how to choose A_0 as the preconditioning matrix of A . The rest of this chapter will describe a method of choosing A_0 .

The method we will describe is called the Incomplete Choleski Conjugate Gradient Method, which was developed by J.A. Meijerink and H.A. Van der Bost in 1976.

3.2 Incompletely Choleski Conjugate Gradient Algorithm

The Incompletely Choleski Conjugate Gradient Method is based on the following idea: If K is an arbitrary nonsingular matrix, then $A = K - R$ represents splitting of the matrix A and an iterated method for $Ax = b$ is given by

$$Kx_{n+1} = (K - A)x_n + b = Rx_n + b \quad (3.4)$$

or
$$x_{n+1} = x_n + K^{-1}(b - Ax_n) = x_n + \Delta x_n \quad (3.5)$$

The more K resembles A , the faster the method will converge.

The algorithm will be constructed in the following way:

Let $L = (\ell_{ij})$ be a lower triangular matrix, so

$$\ell_{ij} = 0 \quad \text{if } i < j.$$

Let $P_n \equiv \{(i, j) \mid i \neq j, 1 \leq i \leq n, 1 \leq j \leq n\}$

be a set of places (i, j) where the zero of L will occur.

Note that P_n contains all pairs of indices of off-diagonal matrix entries.

Theorem 3.1. If A is a symmetric M-matrix, there exists for each $P \subset P_n$ having the property that $(i, j) \in P$ implies $(j, i) \in P$, a uniquely defined lower triangular matrix L and a symmetric nonnegative matrix R , with $\ell_{i,j} = 0$ if $(i, j) \in P$ and $r_{ij} = 0$ if $(i, j) \notin P$, such that the splitting $A = LL^T - R$ is a regular splitting.

The proof of this theorem can be found in [4] and the algorithm based on this theorem is given in the following:

Incomplete Choleski Conjugate Gradient Algorithm

x_0 is arbitrary

$$r_0 = b - Ax_0$$

$$P_0 = (LL^T)^{-1}r_0$$

$$\delta_0 = (r_0, (LL^T)^{-1}r_0)$$

for $n \geq 0$

$$\alpha_n = \delta_n / P_n, AP_n)$$

$$x_{n+1} = x_n + \alpha_n P_n$$

$$r_{n+1} = r_n - \alpha_n AP_n$$

$$\delta_{n+1} = (r_{n+1}, (LL^T)^{-1}r_{n+1})$$

$$\beta_n = \delta_{n+1} / \delta_n$$

$$P_{n+1} = (LL^T)^{-1}r_{n+1} + \beta_n P_n$$

3.3. An Application of Incomplete Choleski Decomposition

Consider the $n^2 \times n^2$ matrix

$$A = \begin{pmatrix} \begin{array}{ccc|ccc} 4 & -1 & 0 & & & \\ -1 & 4 & -1 & & & \\ 0 & -1 & 4 & & & \\ & & & \ddots & & \\ & & & & -1 & \\ & & & & -1 & 4 \end{array} & & I \\ \hline & & & \begin{array}{ccc|ccc} 4 & -1 & 0 & & & \\ -1 & 4 & -1 & & & \\ 0 & -1 & 4 & & & \\ & & & \ddots & & \\ & & & & -1 & \\ & & & & -1 & 4 \end{array} & \\ \hline -I & & & & & \end{pmatrix}$$

where each submatrix is $n \times n$.

It is obvious that A is symmetric and positive definite.

Places of zero of entries are given by

$$P^* = \{(i, j) \mid |i-j| \neq 0, 1, n\} \quad (3.6)$$

where n is the half bandwidth of the matrix.

The elements of the diagonal of A are denoted by a_i . The upper-diagonal elements are denoted by b_i and the elements of the m th upper-diagonal are denoted by c_i , where i is the index of the row of A in which the respective elements occur. Theorem 3.1 guarantees the existence of incomplete Choleski decompositions for A .

In the following computation, it is convenient to write the LL^T decomposition in the form: LDL^T , where D is a diagonal matrix. If the elements of D are denoted by d_i and the elements of L are denoted analogous to A by a_i , b_i and c_i , then it is easy to verify that the following recurrence relations hold for these elements:

$$\tilde{b}_i = b_i ,$$

$$\tilde{c}_i = c_i$$

$$\tilde{a}_i = \frac{1}{d_i} = a_i - \tilde{b}_{i-1}^2 \tilde{d}_{i-1} - \tilde{c}_{i-m}^2 \tilde{d}_{i-m} \quad (3.7)$$

$$i = 1, 2, \dots, n^2$$

where the elements that are not defined should be replaced by zero.

In our example, $a_i = 4$, $b_i = b_i = -1$ if $i \neq kn$ for $k = 1, 2, \dots, n-1$ and $b_i = b_i = 0$ if $i = kn$, and $c_i = c_i = -1$ for $i = 1, 2, \dots, n$. Hence (4.7) reduces to

$$\tilde{b}_i = \begin{cases} -1 & \text{if } i \neq kn \\ 0 & \end{cases} \quad \begin{matrix} \text{for } k = 1, 2, \dots, n-1 \text{ and} \\ i = 1, 2, \dots, n \end{matrix}$$

$$\tilde{c}_i = -1$$

$$\tilde{a}_i = \frac{1}{\tilde{d}_i} \begin{cases} 4 - \tilde{d}_{i-1} - \tilde{d}_{i-m} & \text{if } i-1 \neq kn \text{ for} \\ & k = 1, 2, \dots, n-1 \\ 4 - \tilde{d}_{i-m} & \text{if } i-1 \neq kn \text{ and} \\ & i = 1, 2, \dots, n^2 \end{cases}$$

The results are shown in Table 1 (page 15), and Tables 2-5 below. We compare the Incomplete Choleski Conjugate Gradient method with the Classical Conjugate Gradient method. In the following discussion, we denote Incomplete Choleski Conjugate Gradient Method as ICCG and Conjugate Gradient Method as CG. The true solution of $Ax = b$ is $x = (1, 1, \dots, 1)^T$. From Table 4, Table 5, and Figure 1-4, we can see that the number of iterations i.e., increasing almost linearly when the size of the matrix increases, but ICCG used approximately only half the iterations that CG used. The results of CPU time of comparing both ICCG and CG are also shown on Tables 4-5.

TABLE 4
INCOMPLETE CONJUGATE GRADIENT METHOD

Number of Unknowns	Number of Iterations	CPU Time	Tolerance
10^2	10	0.31	10^{-7}
22^2	18	2.22	
32^2	26	6.48	
44^2	34	15.68	
71^2	52	61.26	
100^2	72	167.17	
141^2	100	458.10	
171^2	120	804.79	
10^2	12	0.36	10^{-9}
22^2	22	2.65	
32^2	31	7.68	
44^2	42	19.22	
71^2	65	76.35	
100^2	89	205.92	
141^2	122	558.15	
171^2	147	938.79	
10^2	14	0.40	10^{-11}
22^2	26	3.08	
32^2	35	8.57	
44^2	47	21.42	
71^2	73	85.33	
100^2	101	232.78	
141^2	140	640.21	
171^2	168	1122.64	
10^2	16	0.44	10^{-13}
22^2	28	3.30	
32^2	39	9.47	
44^2	52	23.55	
71^2	83	96.86	
100^2	114	261.98	
141^2	172	780.17	
171^2	208	1388.66	

TABLE 5
CONJUGATE GRADIENT METHOD

Number of Unknowns	Number of Iterations	CPU Time	Tolerance
10^2	14	0.28	10^{-7}
22^2	37	2.96	
32^2	55	9.18	
44^2	75	23.45	
71^2	120	97.10	
100^2	168	264.39	
141^2	237	751.68	
171^2	287	1343.68	
10^2	14	0.28	10^{-9}
22^2	41	3.27	
32^2	61	10.16	
44^2	85	26.54	
71^2	140	113.11	
100^2	197	315.05	
141^2	278	881.79	
171^2	337	1577.27	
10^2	14	0.28	10^{-11}
22^2	45	3.59	
32^2	67	11.17	
44^2	93	29.04	
71^2	152	122.98	
100^2	215	343.76	
141^2	303	963.63	
171^2	368	1720.23	
10^2	14	0.28	10^{-13}
22^2	48	3.84	
32^2	72	12.05	
44^2	101	31.60	
71^2	165	133.71	
100^2	231	369.24	
141^2	327	1038.34	
171^2	398	1860.36	

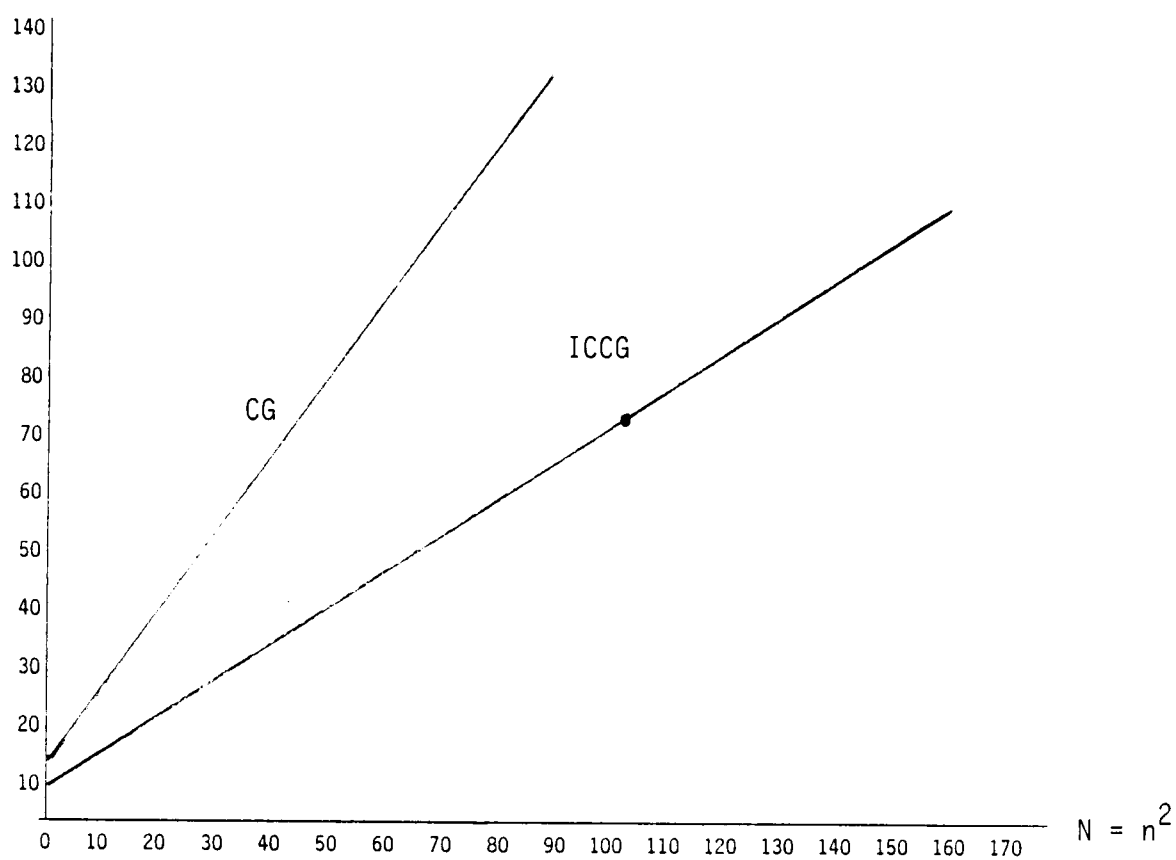


Figure 1. CG and ICCG with tolerance 10^{-7}

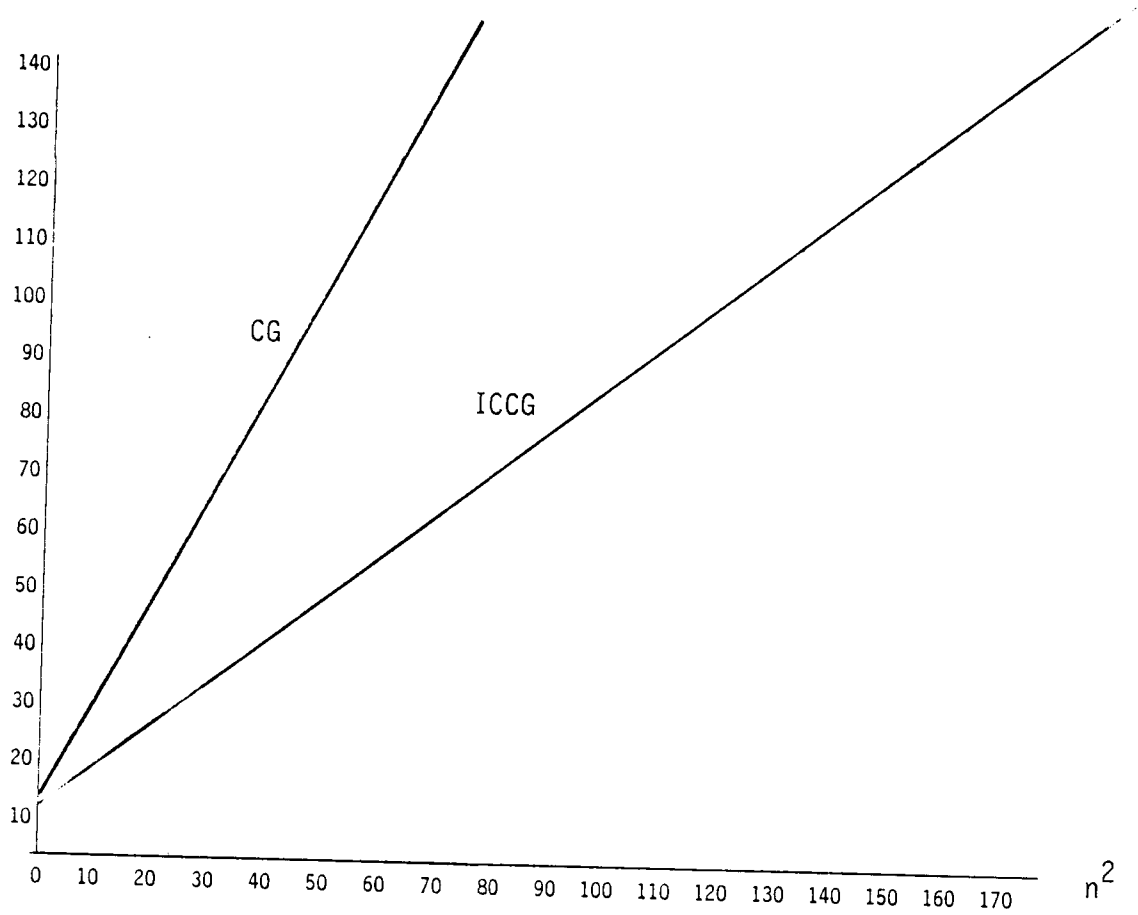


Figure 2. CG and ICCG with tolerance 10^{-9}

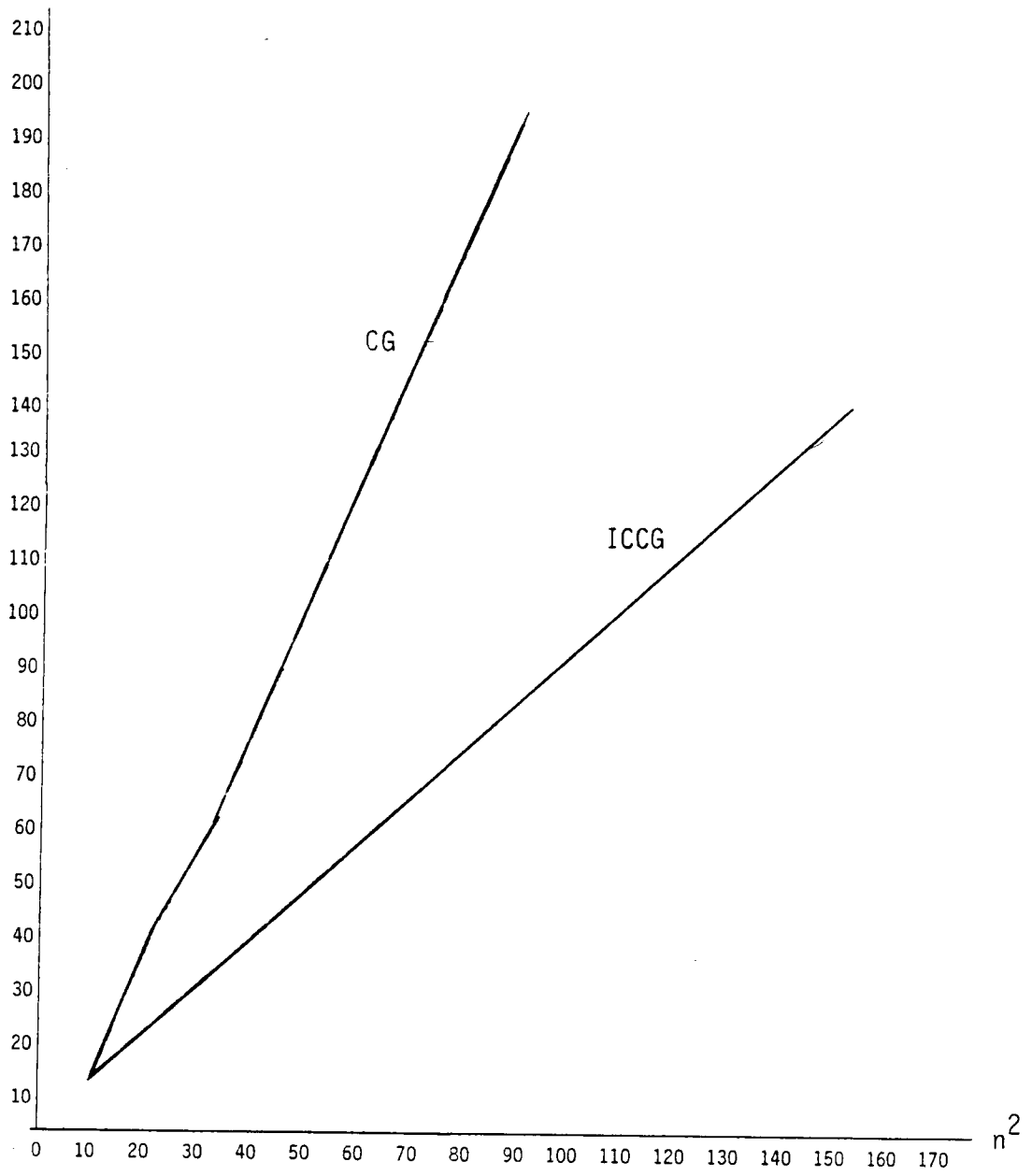


Figure 3. ICCG with tolerance 10^{-11}

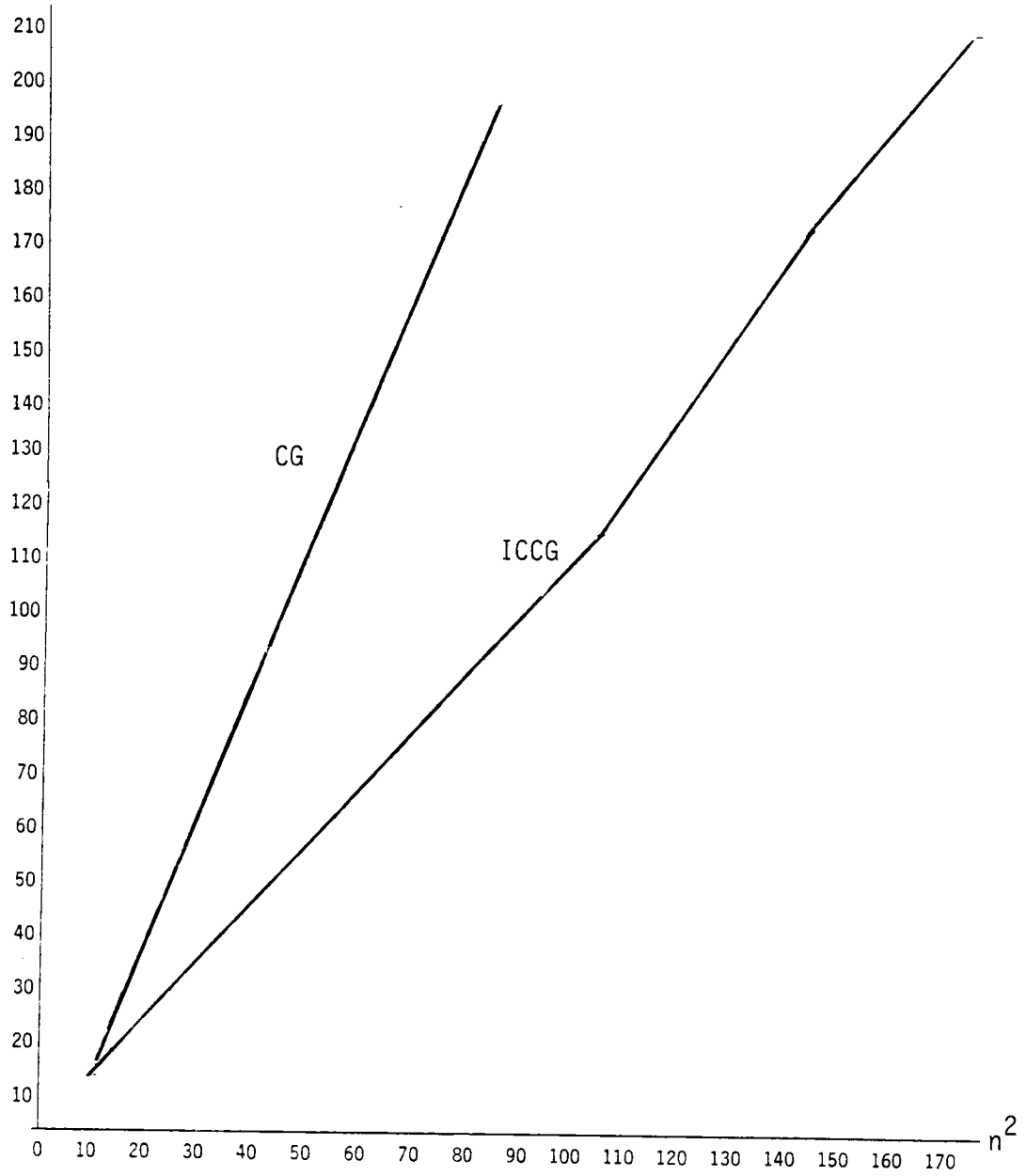


Figure 4. CG and ICCG with tolerance 10^{-13}

LIST OF REFERENCES

LIST OF REFERENCES

- Hageman, L.A. and D.M. Yong. Applied Iterative Methods. Academic Press, New York, 1981.
- Hutson, P. Application of Functional Analysis and Operator Theory. Academic Press, New York, 1980.
- Kershaw, D. The incomplete Cholesky-Conjugate Gradient Method for the Iterative Solution of System of Linear Equations. Journal of Computational Physics 26, 1978.
- Meijerink, J.A. and H.A. van der Vorst. An Iterative Solution Method for Linear Systems of which the Coefficient Matrix is a Symmetric M-Matrix. Math. Comp. January (1977).
- Miller, R. and A.N. Michel. Ordinary Differential Equations. Academic Press, New York, 1982.
- Sammon, P. Fully Discrete Approximation Methods for Parabolic Problems with Nonsmooth Initial Data. SIAM J. Numer. Anal. June 1983.

VITA

Nian She was born in Wuhan, Hubei, China, on January 13, 1957. He attended elementary school in that city and was graduated from the 62nd High School in July of 1975. The following September, he entered Hunan University in China and graduate in August of 1978. He studied in Civil Engineering in that university. In the Fall of 1978 he worked as an assistant engineer in No. 1 Niddling factory in Wuhan.

After two years of working in the factory, he came to the United States in the spring of 1980. He studied English for three months at the English Language Institute at The University of Tennessee, Knoxville. In the fall of 1980 he entered Knoxville College and studied toward a Bachelor's degree in Mathematics. This degree was awarded in June of 1982 along with the Mathematics and Physics Faculty Award, as well as the David Mazkewitsch Memorial Award.

In the fall of 1982 he accepted a teaching assistantship at The University of Tennessee, Knoxville and began work toward a Master's degree in Mathematics. He has been working on Numerical Analysis and will receive his Master's degree in August of 1985. In September 1985 he will begin work on his Ph.D. in Management Science.