THE NUMERICAL TREATMENT OF ILL-POSED PROBLEMS USING
THE METHOD OF CONJUGATE GRADIENTS

By

Sally A. Ross


A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE

in

THE FACULTY OF GRADUATE STUDIES
INSTITUTE OF APPLIED MATHEMATICS

We accept this thesis as conforming
to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA

September 1988

© Sally A. Ross, 1988
In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Department of Mathematics

The University of British Columbia
1956 Main Mall
Vancouver, Canada
V6T 1Y3

Date 20 Sept. 88
Abstract

This thesis examines the use of the method of Conjugate Gradients as an iterative method to be applied to linear system of equations that are ill-conditioned. An overview of the particular problems associated with the solution of ill-conditioned linear systems is given and the method of Conjugate Gradients described. It is shown that, at each iteration, the method of Conjugate Gradients weights the contribution from the singular vectors using a polynomial that is characterized as the solution to a weighted least squares problem. The form of the polynomials is shown to approximate a series of interpolating polynomials that constitute an efficient filtering technique required for computing solutions to ill-conditioned systems. The performance of the method of Conjugate Gradients is shown to compare favourably with other accepted methods for ill-conditioned linear systems. An application from the field of image processing is given and the efficient computation of a smooth reconstructed image from a defocussed picture is demonstrated.
Table of Contents

Abstract ii

List of Tables vi

List of Figures vii

Acknowledgement viii

1 Introduction 1
  1.1 Sensitivity of Linear Systems ................................. 7

2 The Numerical Treatment of Integral Equations 12
  2.1 The Continuous Problem ...................................... 13
  2.2 The Discrete Problem .......................................... 17
  2.3 Regularization Techniques .................................... 23
    2.3.1 Regularization of the Problem ............................ 24
    2.3.2 Regularization of the Solution : Iterative Filtering ..... 27

3 The Method of Conjugate Gradients 30
  3.1 The Method of Conjugate Directions .......................... 31
  3.2 The Method of Conjugate Gradients ........................... 36
    3.2.1 The Conjugate Gradients Algorithm ....................... 36
    3.2.2 Properties of the Method of Conjugate Gradients ........ 39
    3.2.3 Conjugate Gradients as an Iterative Method ............ 42
Bibliography 106

Appendices 111

A Appendix 111

A.1 Kronecker Products ................................. 111
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Comparison between the SVE and the SVD</td>
<td>19</td>
</tr>
<tr>
<td>4.2</td>
<td>Coefficients for the first 3 polynomials for the Trapezoidal example</td>
<td>60</td>
</tr>
<tr>
<td>4.3</td>
<td>Coefficients for the first 4 polynomials for the Phillips' example</td>
<td>62</td>
</tr>
<tr>
<td>4.4</td>
<td>Comparison of accuracy and work for the Hilbert example.</td>
<td>75</td>
</tr>
<tr>
<td>4.5</td>
<td>Comparison of accuracy and work for the Trapezoidal example.</td>
<td>77</td>
</tr>
<tr>
<td>4.6</td>
<td>Comparison of accuracy and work for the Simpson's example.</td>
<td>77</td>
</tr>
<tr>
<td>4.7</td>
<td>Comparison of accuracy and work for the Phillips' example.</td>
<td>78</td>
</tr>
<tr>
<td>4.8</td>
<td>Conjugate Gradient analysis for the Hilbert example</td>
<td>82</td>
</tr>
<tr>
<td>4.9</td>
<td>Conjugate Gradient analysis for the Trapezoidal example</td>
<td>83</td>
</tr>
<tr>
<td>4.10</td>
<td>Conjugate Gradient analysis for the Simpson's example</td>
<td>84</td>
</tr>
<tr>
<td>4.11</td>
<td>Conjugate Gradients analysis for the Phillips' example</td>
<td>85</td>
</tr>
<tr>
<td>5.12</td>
<td>Analysis from the Conjugate Gradients algorithm for the Cross example</td>
<td>101</td>
</tr>
<tr>
<td>5.13</td>
<td>Analysis from the Conjugate Gradients algorithm for the L example</td>
<td>102</td>
</tr>
<tr>
<td>5.14</td>
<td>Analysis from the Conjugate Gradients algorithm for the Box example</td>
<td>102</td>
</tr>
</tbody>
</table>
List of Figures

4.1 Examples of filtering polynomials ........................................... 61
4.2 Examples of the filter values for the Hilbert example .................... 70
4.3 Examples of the filter values for the Trapezoidal example ................. 71
4.4 The singular value spectra for the 4 test examples .......................... 76
4.5 Chosen solution for the Simpson and the Phillips' test cases ............... 79

5.6 The digitized sharp test images ............................................... 91
5.7 The digitized defocused test images .......................................... 93
5.8 Numerical representation of the degraded test images ....................... 94
5.9 The error for each of the test images ....................................... 95
5.10 The singular values of the reconstruction matrix ............................ 96
5.11 Numerical results from iterations for the Cross ............................ 97
5.12 Visual representation of iterates for the Cross ............................. 98
5.13 Visual representation of iterates for the L ................................ 99
5.14 Visual representation of iterates for the Box ............................... 100
Acknowledgement

I am indebted to my supervisor, Dr James Varah, for his constant patience and expert guidance regarding the preparation of this thesis and throughout my studies in Vancouver. Special thanks go to my family in Belgium, for always being so supportive, but especially for paying the telephone bill. I would like to acknowledge, with particular gratitude, the Gibbins family for giving me a stress-free home to enable this to be completed. This thesis may have been abandoned had it not been for the support and encouragement given by Steve Adams, I am very grateful to him for so many reasons.
Chapter 1

Introduction

The computation of solutions to problems for which the exact solution does not depend continuously on the supplied data is challenging; such problems are said to be ill-posed and encountered frequently in various application areas. Solutions are difficult to find since small changes in the data may alter the solution considerably. The data for many physical problems may be generated as the results gathered from practical experimentation and small errors are inherent in the problem. Any numerical computation may introduce further small errors through the effects of roundoff. These small errors, which are effectively due to "noise" in the system must not be reflected as large oscillations in the solution. It is of great practical importance to be able to efficiently compute a meaningful solution to such problems. This thesis examines the use of the method of Conjugate Gradients for computing the numerical solution of linear systems of equations which arise from ill-posed problems.

One source of ill-posed problems is that of the solution to linear integral equations of the first kind

\[ \int K(s,t)f(s)ds = g(t); \]  

such equations arise in many applications, e.g., chemistry and image processing. The operator, \( K(\cdot, \cdot) \), has the characteristic that it smooths the function, \( f(\cdot) \), in the mapping \( f(\cdot) \rightarrow g(\cdot) \). The function, \( g \), will always be smooth, even in cases where \( f \) is not continuous. A solution will not exist for arbitrary \( g \). The stronger the effect of the smoothing is, the more difficult it will be to evaluate the solution. The numerical treatment of (1.1)
is often achieved by first discretizing the equation to form a linear system,

$$Ax = b$$  \hspace{1cm} (1.2)

where $A \in \mathbb{R}^{n \times n}$ and $b \in \mathbb{R}^n$. The ill-posed property of the problem is reflected in the inherent ill-conditioning of the coefficient matrix $A$. The concern of this thesis is the numerical solution of ill-conditioned linear systems (1.2); the theory for the solution of the continuous problem (1.1) is used for understanding and motivation.

The problem itself is difficult to solve. Exact solutions may be impossible to find. A solution (if it exists) may be non-unique and highly unstable, e.g., in the field of image processing many very different original pictures may be indistinguishable when the image is "blurred" in some way. The strategy is instead to calculate an approximate solution that will be acceptable. It is therefore necessary to state conditions under which a solution will be deemed acceptable; such as some degree of smoothness in the solution. This will then ensure that the effect of the small errors, to cause the calculation of misleading results, is minimized. An acceptable solution is therefore one that is stable and that will give a small residual error.

Once discretized, the vector $b$ may not lie in the range of $A$. The first step is to require that the problem (1.2) be stated instead as the minimization

$$\min_x ||Ax - b||^2_2$$ \hspace{1cm} (1.3)

which is the usual linear least squares problem. The problem (1.3) will have a solution but the ill-conditioning of $A$ means that two estimates that are equally accurate in terms of the solution to (1.2) may be very different from each other. The need to be able to compute a stable solution of (1.3) has motivated the strategy of including in the problem a restriction on the class of allowable solutions.

Motivation from the continuous problem, [6], demonstrates that the difficulties stem from the inclusion, as components of the solution, of vectors that are highly oscillatory.
The "noise" in the system has an exaggerated effect on the contribution of these vectors so that they are given inaccurate weightings that may be disproportionately high. An approximation is formed so that the contributions from these unstable components is reduced, or excluded. The class of allowable estimates is thus restricted to be functions that are smooth but will (hopefully) still be relatively accurate.

The motivation leads to the technique of Regularization. Regularization is used in general to mean the restatement of an ill-posed problem as a well-posed one that is made close to the original by the use of a small parameter. The parameter controls the extent of the regularization. The idea of regularization was first introduced by Tikhonov [37]. The original problem (1.3) is restated as a constrained least squares problem to restrict the solution to be smooth whilst minimizing the residual error. Understanding of the continuous problem demonstrates that the idea of regularization can be seen to be equivalent to including in (1.1) a small contribution from the function \( f \) to the right hand side of the equation and thus converting the equation to one of the second kind. The solution of second kind integral equations is a well-posed problem [5, 9]. The ill-conditioning of the coefficient matrix is now no cause of difficulty and the corresponding discrete problem may be solved by any direct method.

In practical applications the system of linear equations generated is often very large and sparse. The use of direct solution methods is restricted. Iterative techniques can be used successfully by setting the parameter to zero and regularizing the solution by terminating the iterative process before the oscillatory components of the solution have converged. The procedure is known as an Iterative Filtering technique and was first studied by Landweber [24]: a series of less and less regularized solutions is generated by filtering the contributions of the unstable vectors. The extent of the regularization is controlled by the choice of the iterative method and the number of iterations performed.
Various iterative methods have been studied, [17, 36]. Each method generates an associated filtering function. The purpose of this thesis is to examine the filtering properties of the iterative method of Conjugate Gradients.

The method of Conjugate Gradients is an efficient numerical iterative algorithm for computing the solution of a linear system of equations, (1.2). At each step the majority of the work effort is found in the formation of a single matrix-vector product with the original coefficient matrix; for this reason the scheme is particularly suited to large sparse systems of linear equations. The original presentation of the method of Conjugate Gradients [22], was as a minimization process that terminated in a finite number of steps; thus as a direct numerical method. It was soon discovered that although in a practical environment the process did not terminate as predicted, a good estimate of the solution is achieved after only a few steps. The basic algorithm may be simply modified to give an efficient solution method for the linear least squares problem (1.3). In [25] it was first shown that the performance of the method could be further improved by the notion of preconditioning. The method of Conjugate Gradients, particularly when preconditioned, was seen to be a very powerful and economical iterative procedure.

The method of Conjugate Gradients has been applied to ill-conditioned systems of equations [35] and [8] with little explanation as to the reasons for its success. Recently, Vogel [42] demonstrated that the method of Conjugate Gradients does have the properties of a favourable iterative filtering method. This thesis extends the ideas in [42] by examining the results of the filtering process in more detail. A number of discretized integral equations are provided as numerical examples, including an application from the field of picture processing; these are more general than the illustrations given in [42]. The method of Conjugate Gradients is shown to be an economical method to compute a smooth, accurate solution when the system of equations is ill-conditioned. The method is successful without the need to apply a preconditioner; in fact any preconditioning may
Chapter 1. Introduction

inhibit the ability of the method to calculate a smooth approximating.

Although the integral equation (1.1) is a source of the features of the problem, it is the solution of the resulting linear system which is the topic here. The material in later chapters requires the understanding of the sensitivity of linear systems to small errors in the data. As an introduction the basic notion of conditioning is detailed in §1.1 at the end of this Chapter.

The numerical treatment of integral equations is the subject of Chapter 2. The continuous equation (1.1) is examined in §2.1 where Fredholm integral equations of the first kind are shown to be a source of ill-posed problems. The analogies between the continuous and discretized problem are presented in §2.2. With this motivation the technique of regularization is discussed in §2.3. The original method of regularization, presented in §2.3.1, is shown to be inappropriate for solution by iterative procedures. §2.3.2 describes the approach to the problem by using the iteration count as the parameter to control the extent of the regularization.

A description of the method of Conjugate Gradients is presented as Chapter 3. The method of Conjugate Gradients is shown to belong to the general class of Conjugate Directions schemes in §3.1, and the theory on which the development of the algorithm is based is discussed. The specific case of the method of Conjugate Gradients is the subject of §3.2. The algorithm is detailed in §3.2.1 and the properties required in later chapters are discussed in §3.2.2. The alteration in the manner in which the algorithm is considered, (from direct to iterative) is discussed in §3.2.3 and the evaluation of the convergence rate presented in §3.2.4. The improvement of the method by preconditioning techniques is given in §3.3. Preconditioning is motivated by the idea of a Generalized Conjugate Gradients method in §3.3.1 and the practical implementation detailed in §3.3.2. Some of the common choices for the preconditioner are described in §3.3.3. §3.4 shows that the method of Conjugate Gradients can be successfully used to solve the linear least squares
Chapter 1. Introduction

problem (1.3); an efficient algorithm is given in §3.4.1 and the preconditioned version presented in §3.4.2.

The use of the method of Conjugate Gradients as an iterative filtering technique is examined in Chapter 4. The features that are required by any iterative method to act as an effective filter are discussed in §4.1. The relations that arise in the algorithm are given and the polynomial that forms the filtering function developed in §4.2. In §4.3 the polynomial is shown to approximate the Lagrangian interpolant through the singular values and thus be an effective filter; a number of numerical illustrations are provided. The polynomial may also be viewed as an expansion of the approximation of the inverse of the coefficient matrix; this approach, to the explanation of the properties, is examined in §4.4. §4.5 discusses the practical application of Conjugate Gradients as an iterative filter. Results are compared with other recognised solution methods for ill-conditioned systems. The method of Conjugate Gradients is shown to be a competitive alternative.

An application from the field of Picture Reconstruction is examined in Chapter 5. A description of the problem and its mathematical formulation is detailed in §5.1. The mathematical formation of general problems from image processing is discussed in §5.1.1 and the specific example used for the numerical tests derived in §5.1.2. The results of the numerical tests are presented in §5.2.

Chapter 6 is a discussion of the findings of this thesis.

As a preliminary to the problems that are the topic of the later chapters the following section defines what is meant by a linear system to be described as ill-conditioned. Results concerning the Kronecker product of matrices, summarized for use in Chapter 5, are detailed in the Appendix.
Chapter 1. Introduction

1.1 Sensitivity of Linear Systems

An important question in the numerical computation of the solution to linear systems of equations is the effect on the solution of small changes in either the elements of the coefficient matrix, or in the data forming the right hand side. Numerical systems such as (1.2) frequently contain small errors which arise either in the way the data is collected or in the numerical calculations performed in the solution process. These errors mean that the original problem is rarely represented exactly in its numerical form and the resulting system is some perturbation of the original problem.

There are two considerations:

• How the exact solution, $\tilde{x}$, changes when the system is perturbed;
• How the computed solution, $\hat{x}$, differs from the exact solution.

The first issue concerns the sensitivity of the system and is dependent only on the problem itself, but the error in the calculated solution is a feature of the chosen solution method.

If a system is non-singular, so that $A^{-1}$ can be formed, there is a unique solution to (1.2): $x^* = A^{-1}b$ for each vector $b$. However the matrix may be “almost” singular in the sense that small perturbations of the elements will result in a matrix that is rank deficient. In this case, although (mathematically) the system has a unique solution, computationally it has the features of a singular system, so that a solution (if one exists) may not be unique. The smaller the alterations to the matrix $A$ that produce a singular matrix $A + \delta A$ are, the greater the problems in computing the solution to (1.2).

A measure of the extent of the near-singularity of $A$, and so the relative magnitude of the difficulties, is given by the condition number of the matrix

$$\kappa(A) = \frac{\sigma_1}{\sigma_n} = \|A\|_2\|A^{-1}\|_2.$$  (1.4)
where $\sigma_1 \geq \ldots \geq \sigma_n > 0$ denote the singular values of the matrix $A$. (Singular values are the square roots of the eigenvalues of $A^T A$.) If $A + \delta A$ is singular then the 2-norm of the smallest such perturbation matrix is equal to the magnitude of the smallest singular value, i.e., $\|\delta A\|_2 = \sigma_n$ [16], so that the shortest (2-norm) distance from $A$ to the set of singular matrices is given by $\sigma_n$. When this distance is close to zero the resulting condition number (1.4) may be very large. Matrices for which the condition number is high are known as ill-conditioned.

The effect on the solution of small variations in the data is illustrated by considering a perturbation of the right hand side to $\tilde{b} = b + \delta b$. The solution $\tilde{x} = x + \delta x$ is such that

$$\frac{\|\delta x\|}{\|x\|} \leq \|A\|\|A^{-1}\| \frac{\|\delta b\|}{\|b\|}.$$  

(See [16, 26] for details and a similar bound for a perturbation of the matrix $A$). This means that perturbations of the data in (1.2) are reflected in the solution by a factor of the condition number. Systems that involve a poorly conditioned coefficient matrix may give rise to unstable solutions, i.e., solutions which may show a finite change based on an infinitesimal variation of the data.

When a matrix is ill-conditioned it effectively has a finite "near" null-space, i.e., there are non-zero vectors $w$ such that $Aw \sim 0$. If $\tilde{x}$ solves (1.2) then $\tilde{x} + w$ is also an acceptable solution since the residual $\|A(\tilde{x} + w) - b\|_2$ is small. However $\|w\|_2$ need not be small, so the two solutions may be quite different from each other. The vectors $w$ that form this near-null space are combinations of the singular vectors that correspond to the small singular values. They are often violently oscillatory [6]. The inclusion of these vectors as components of the solution is the cause of the instability.

In numerical calculations it is not usually possible to obtain the exact error $\|x^* - u\|_2$ of an estimate $u$ to the solution of (1.2). One measure of the accuracy of $u$ is the norm
of the residual $\|r\|_2 = \|Au - b\|_2$. The residual and the exact error are related as

$$\frac{\|x^* - u\|_2}{\|x^*\|_2} \leq \kappa(A) \frac{\|r\|_2}{\|b\|_2}$$

(1.6)

If the condition number $\kappa(A)$ is small, this is a reasonable bound and the residual is a reliable quantity to test the accuracy of an estimate. However, in the ill-conditioned case, although a large residual is an indication of a poor estimate, a small residual can be misleading.

In some cases the particular vector $b$ for the system gives rise to a solution which lies far from the "near" null-space. The matrix is still ill-conditioned but the system itself is less susceptible to errors since the contribution from the oscillatory singular vectors is such that roundoff effects will still be relatively non-destructive. (Equally, a system with the same coefficient matrix may have a solution that is highly oscillatory and will be extremely sensitive to errors even of the same size as the machine precision.) These are special cases; it is generally true that the higher the condition number the greater the difficulties in evaluating an appropriate solution.

In summary: the consequence of a matrix being ill-conditioned is that the solution is effectively non-unique in that there are many near-solutions. These near solutions are the exact solutions to slightly perturbed problems, but it is not necessarily true that they are close to the actual solution. In addition it is difficult to determine the accuracy of any chosen estimate. Conditioning is a feature of the matrix and not the method of solution. However an algorithm must successfully handle the difficulties (of computing a stable solution) that arise when dealing with poorly conditioned systems.

The numerical solution of (1.2) introduces errors which effectively result in perturbed problems, since all numbers are represented and computations performed using a finite word length. When the matrix is ill-conditioned, if the data is altered either by the numerical calculations (or in the way that the problem is initialized) any attempt at
solving the system may result in the wrong solution being selected. Additionally, since the measure of accuracy of the solution is given by the residual, it is often difficult to distinguish between good and bad solutions. Therefore there are two problems:

- To select a solution to the problem so that small numerical perturbations do not result in large oscillatory components being exaggerated in the solution.
- To be confident that the solution obtained is reasonable.

Problems can arise when standard solution methods are applied to ill-conditioned systems. Numerical methods for linear systems effectively form the solution \( x^* = A^{-1}b \). When \( A \) is ill-conditioned it may be very close to a matrix which is not invertible and so this method of solution (by effectively forming the inverse) is not advised. Perturbation of the elements of \( A \) or \( b \) even by an amount equivalent to that of the machine precision may adversely affect the exact solution \( x^* \). This indicates the need to alter the statement of the problem (1.2) before it is solved so as to ensure that some form of pseudo-inverse of \( A \) is used instead.

In many physical applications the systems are very large and sparse, with the solution approximated by an iterative method. When the coefficient matrix is ill-conditioned, slow convergence rates and instability in the solution process can cause problems, but a more pertinent issue is whether the solution obtained is the one sought (since the residual, which is the usual indicator of convergence, is not reliable). Some modifications to the solution strategy are needed to allow successful resolution of this issue.

Insight into the exact nature of the difficulties and guidance on how to overcome them can be gained by considering the continuous analogy to the problem (1.2); that of Fredholm integral equations of the first kind. The solution to integral equations has long caused difficulties due to the ill-posed nature of many of the problems. The theory is the
subject of many dedicated texts, for example [34] and [9] and their numerical treatment is equally well documented, [5, 13, 1].

The next chapter begins by considering the continuous case of Fredholm Integral equations of the first kind. Their discretization results in an ill-conditioned linear system. The similarities between the continuous and discrete cases are examined in §2.2 and the modified solution strategy motivated. The methods used to approach ill-posed problems is the subject of sections §2.3.1 and §2.3.2.

(It is noted that the terms ill-posed and ill-conditioned are not strictly interchangable. Ill-conditioned is a term to describe a particular feature of the coefficient matrix whereas ill-posed is used to describe a quality of the entire problem. A problem may be ill-posed because: There is no solution; There are many solutions; The solution is unstable. Linear systems that arise from ill-posed problems are known to have ill-conditioned coefficient matrices. While it is possible for a matrix $A$ to be ill-conditioned and the problem well-posed (in that the particular right-hand-side gives a straightforward stable solution) this is a special case. Throughout this thesis it is assumed that ill-conditioned implies ill-posed and the terms used as synonyms.)
Chapter 2

The Numerical Treatment of Integral Equations

The solution of problems formed by discretizations of Fredholm integral equations of the first kind

\[ \int_{a}^{b} K(s,t)f(s)ds = g(t), \quad c \leq t \leq d \quad (2.7) \]

are known to require special care. Equations of this form are distinguished from other integral equations because the unknown function \( f(s) \) appears only under the integral sign and that the interval \( t \in [c,d] \) need not necessarily coincide with the range of integration. We will consider only linear equations where the kernel, \( K(s,t) \), is square integrable (or \( L_2 \)). An \( L_2 \) kernel is one such that

\[ \int_{a}^{b} \int_{a}^{b} K(s,t)^2 dsdt < \infty \quad (2.8) \]

The solution of (2.7) is difficult to calculate because in general it is not continuously dependent of the data. Small alterations in the data may lead to large oscillations in the solution; thus there may be many "near" solutions. The problem is therefore ill-posed in that a solution (if it exists) may be non-unique or highly unstable.

Numerical solutions of (2.7) can be obtained by first discretizing the problem to obtain an algebraic system of linear equations of the form (1.2). It is expected that the solution to the discrete problem will in some way reflect that of the continuous one. It is possible therefore to understand the difficulties mentioned in §1.1 in a continuous setting.

In §2.1 the continuous problem is detailed as motivation to the solution of the discrete counterpart in §2.2 The motivation from the theory of integral equations suggests a
modification to the solution strategy for ill-conditioned systems. The original problem is replaced by a well-conditioned problem that is made close to the original by the presence of a small parameter; this technique is known as Regularization. §2.3 examines the technique of Regularization. §2.3.1 discusses the creation of a well-conditioned equivalent system by a constrained minimization problem and §2.3.2 describes how Regularization may be achieved by iterative methods.

2.1 The Continuous Problem

Fredholm equations of the first kind, (2.7), frequently result from physical problems (e.g., chemical analysis, image reconstruction, geophysics) [1, 13, 31, 27, 37]. In these situations the function \( f \) corresponds to some actual physical phenomenon that is required to be calculated while \( g \) is the set of data, recorded by some apparatus, on which the estimation of \( f \) is based. The measuring instruments will have some degree of smoothing effect on the data which are represented in the operator \( K \). There are experimental limitations on the accuracy of the data acquisition, and thus some loss in the details of \( f \). In particular high frequency components become exaggerated. With these problems it is important to avoid treating small oscillations in \( g \), that are actually due to the noise in the system, as if they were caused by large oscillations in \( f \). The solution method becomes important as, during its course, any numerical computation is liable to further complicate the situation by introducing additional errors that resemble noise.

There are two important difficulties associated with the solution of (2.7). An equation of the form (2.7) is not guaranteed to have a solution for arbitrary kernel, \( K(s,t) \), or data \( g(t) \), and a solution (if it exists) need not be unique. A computationally more troublesome feature is that the problem is not well-posed, in that the solution does not depend continuously on the data. This means that small perturbations of the function
Chapter 2. The Numerical Treatment of Integral Equations

$g(t)$ can be reflected as arbitrarily large changes in the solution $f(s)$.

The problem of existence and uniqueness can be illustrated by simple example. With

$$K(s, t) = \cos(s) \cos(t) \text{ and } 0 \leq s \leq 2\pi \quad (2.9)$$

any given $f(s)$ will produce a function $g(t)$ that is a multiple of $\cos(s)$. This means that if an arbitrary $g(t)$ is not of this form it will not be possible to solve for $f(s)$. In addition for any solution $f(s)$ that does exist $f(s) + a \sin(s)$, for any value of $a$, is also a solution.

An illustration of the ill-posed property of the system is as follows. The Riemann-Lebesgue lemma [5, 34] states that:

*For any square-integrable kernel, $K(s, t)$,

$$\int_a^b K(s, t) \sin(ns) ds \to 0 \text{ as } n \to \infty \quad (2.10)$$

This means that if $h(s)$ is a solution to (2.7) then $\tilde{h}(s) = h(s) + C \sin(ns)$ for any constant $C$ is also an approximate solution since (for large $n$),

$$\int_a^b K(s, t) |h(s) + C \sin(ns)| ds \to g(t) \text{ as } n \to \infty. \quad (2.11)$$

Large values of $n = N$ that give rise to a slight perturbation of the data

$$\int_a^b K(s, t) [h(s) + C \sin(Ns)] ds \quad = \quad \int_a^b K(s, t) h(s) ds + C \int_a^b K(s, t) \sin(Ns) ds$$

$$= \quad g(t) + \epsilon, \text{ where } \epsilon \ll 1$$

$$= \quad \bar{g}(t)$$

correspond to a solution

$$\tilde{h}(s) = h(s) + C \sin(Ns)$$

$$\sim h(s) + C$$

where $C$ may be arbitrarily large.
In order to solve (2.7) it is necessary to resolve the problem of how to choose, from the many solutions that are available, a solution that is reasonable. A solution will be accepted as reasonable if it is stable and will give a small residual, i.e.,

\[ \| \int K(s,t)f(s)ds - g(t) \|_2 \sim 0. \]  \hspace{1cm} (2.12)

The class of allowable solutions must in some way be restricted. One approach is to require that the solution \( f(s) \) be smooth. Whether this smooth choice is the "best" is difficult to assess; it is, however, the exact solution to some small perturbation of the equation (2.7) and will have guaranteed stability, and so is a reasonable solution.

The singular value expansion, (SVE), of the system (2.7) can be used to understand the problem of selecting a reasonable solution, \([6, 19]\), and to give an analysis of the existence of a smooth solution.

For any square integrable kernel \( K(s,t) \) there exist two sets of orthogonal functions \( \{\Phi_i(t)\} \) and \( \{\Psi_i(s)\} \) known as the left and right singular functions respectively. Associated with these functions are scalars \( \mu_i \) such that

\[ \int K(s,t)\Psi_i(s)ds = \mu_i\Phi_i(t) \]  \hspace{1cm} (2.13)

The \( \mu_i \) are the singular values of the kernel \( K(s,t) \) and since

\[ K(s,t) = \sum_{i=1}^{\infty} \mu_i\Psi_i(s)\Phi_i(t), \]  \hspace{1cm} (2.14)

in order for the kernel to be \( L_2 \) the sum (2.14) must converge and therefore zero is the only limit point of the singular values, so that, \( \mu_i \to 0 \) as \( i \to \infty \). From this

\[ g(t) = \sum_{i=1}^{\infty} \eta_i\Phi_i(t) \]  \hspace{1cm} (2.15)

where \( \eta_i = (\Phi_i,g) \) (where \( \langle \cdot, \cdot \rangle \) denotes the usual generalized inner product).

The solution can then be expanded as the infinite sum

\[ f(s) = \sum_{i=1}^{\infty} \eta_i\Psi_i(s) \]  \hspace{1cm} (2.16)
This again illustrates the ill-posed property of the problem. The limit point of $\mu_i$ is zero so the convergence of (2.16) is not guaranteed. The singular functions $\Psi_i$ associated with the small singular values tend to be highly oscillatory [6], and so highly sensitive to roundoff effects when forming the expansion (2.16). It is the inclusion of these as components of the solution that causes the instability. A condition under which convergence, and with it, stability, is assured, is given by Picard's Condition [34, 39]:

The existence of a smooth solution $f(x) \in L_2$ to equation (2.7) where $K(s,t) \in L_2$ is guaranteed provided

$$\sum_{i=1}^{\infty} \left| \frac{\eta_i}{\mu_i} \right|^2 < \infty$$

(2.17)

where $\eta_i = (\Phi_i, g)$, and $\mu_i$ are the components of the singular value expansion of $K(s,t)$.

For a reasonable solution to exist then, the $\eta_i$ must decrease to zero at a faster rate than the decay of the scalars $\mu_i$ [1]. This will ensure the convergence of (2.17). The decay of the singular values is a function of the smoothing property of the kernel. The smoother the kernel is, the faster the singular values will decay. This increases the ill-conditioning and places more severe restrictions on an admissible solution.

The integral equation (2.7) can be treated numerically by first discretizing it using some quadrature rule to produce a linear system (1.2). A set of $n$ linear equations are thus created:

$$\sum_{j=1}^{n} w_i K(s_i, t_j) f(s_i) = g(t_i) \text{ for } i = 1, \ldots, n$$

(2.18)

where $\{w_i\}$ are the quadrature weights at the nodes $s_i \in [c, d]$. The points $t_i$ are chosen from $[a, b]$. This gives the linear system (1.2) formed by

$$A = (a_{ij}) \text{ where } a_{ij} = w_j K(s_i, t_j)$$

(2.19)

$$x_i = f(s_i) \text{ and } b_i = g(t_i)$$

(2.20)

In the continuous problem infinitesimal changes in $g(t)$ or $K(s,t)$ may produce a finite
change in \( f(s) \). This is reflected in the discrete version by the inherent ill-conditioning of the matrix \( A \) (in the sense detailed in §1.1).

### 2.2 The Discrete Problem

The linear system

\[
Ax = b \tag{2.21}
\]

with \( A \in \mathbb{R}^{n \times n} \) and \( b \in \mathbb{R}^{n} \) that is equivalent to the continuous problem (2.7) is created by discretization of the integral equation by a chosen quadrature method. (The linear system has the properties discussed in §1.1.) The purpose here is to investigate the quality of a numerical solver for general ill-posed problems, so the exact solution sought is taken to be that of the discrete problem (2.21). How accurately the solution to the discrete problem \( x \) matches the continuous function \( f(s) \) is a separate consideration that involves the truncation error introduced by the discretization. Any quadrature method with reasonably small truncation error will be acceptable and it will result in an ill-conditioned system of equations. Different discretizations do not lead to significant differences in the underlying ill-conditioning of the matrix, only in the accuracy with which the discrete solution resembles its continuous counterpart.

The use of a quadrature method to discretize the problem may result in a system such that the data vector \( b \) is not in the range of the coefficient matrix, i.e., \( b \notin R(A) \). In this case it is usual to solve the problem by computing the solution of minimum residual norm, or the Linear Least Squares solution.

\[
\min_{x} \|Ax - b\|_{2}^{2}. \tag{2.22}
\]

The singular value decomposition of the matrix \( A \) can be used to demonstrate the similarities between the two problems (2.21) and (2.7). The continuous theory can then
be used to motivate an appropriate solution strategy for all numerical ill-conditioned problems.

The existence of singular values and their associated singular vectors in the continuous problem is reflected in the discrete problem in the decomposition of $A$ as $U \Sigma V^T$, where

$$
\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_n)
$$

$$
U = [u_1, \ldots, u_n]
$$

$$
V = [v_1, \ldots, v_n]
$$

and $u_i, v_i$ are the left and right singular vectors of $A$, respectively. (To avoid scaling difficulties it is assumed that the matrix $A$ is scaled so that $\sigma_1 = 1$.) This is the standard singular value decomposition, (SVD) (see, e.g., [16, 15]) of a matrix. The vectors $u_i$ and $v_i$ are discrete approximations to the continuous functions $\Phi_i$ and $\Psi_i$. The singular values $\{\sigma_i\}$ correspond to the constants $\mu_i$ of the continuous problem and therefore will decay to zero. The matrix $A$ is thus inherently ill-conditioned as defined in §1.1. The solution $x$ is the finite equivalent of the sum (2.16)

$$
x = \sum_{i=1}^n \frac{u_i^T b}{\sigma_i} v_i
$$

(2.23)

In forming the sum (2.23) numerically, the small magnitude of the singular values may cause roundoff effects to associate incorrect weights with the "later" singular vectors. This means that the troublesome effects of these vectors as components of the solution become increased. The properties of the singular value expansion of (2.7) and the singular value decomposition (2.23) of (2.21) are summarized in Table (2.1) to emphasize the correspondence and similarities between the two.

With the strong connection between the analysis of the two problems and the similarity in the difficulties associated with their solution, it is natural to assume a discrete
Chapter 2. The Numerical Treatment of Integral Equations

Table 2.1: Table emphasizing the relation between the SVE and the SVD analogue of the Picard Condition given in (2.17). This condition can be used to specify the existence of a smooth solution to the discretized system of equations.

The Picard Condition (2.17) requires that the ratio \( \frac{\eta_i}{\mu_i} \) decay to zero as the index, \( i \), increases. This is assured by the relative decay of \( \eta_i \) and \( \mu_i \), so \( \frac{\eta_i}{\mu_i} \to 0 \) as \( i \to \infty \). Once discretized the summations, shown in the table, become finite. In practice the discrete quantities \( \gamma_i = u_i^Tb \) do not decrease to a value below the noise level of the system (unlike the continuous equivalent \( \eta_i = (\Phi_i, g) \)). The singular values \( \sigma_i \), however, continue to decay to zero. The criterion under which the solution to a system (2.21) will be deemed reasonable can therefore be given as a version of Picard's Condition in discrete form [40]. It is required that the solution depend continuously on the data: i.e., that it be insensitive to noise or small perturbations of the system. A reasonable solution is one such that the residual is small for an approximation that is smooth, which mathematically can be stated as:

\[
\|Ax - b\|_2^2 = O(\epsilon) \quad \text{and} \quad \|x\|_2^2 = O(1)
\]  

(2.24)

for noise level \( \epsilon \). Expanding this in terms of the SVD gives

\[
\|Ax - b\|_2^2 = \sum_{i=1}^{n}(\sigma_i v_i^T x - \gamma_i)^2
\]  

(2.25)
Chapter 2. The Numerical Treatment of Integral Equations

\[ \|x\|_2^2 = \sum_{i=1}^{n}(v_i^Tx)^2. \]  

(2.26)

Thus if \(x\) is to be a reasonable solution given that for some value \(m\) the scalars \(\gamma_i\) have a magnitude approximately that of the noise of the system, i.e., \(|\gamma_i| \sim \epsilon\) for \(i \geq m\), then the previous \(\gamma_i\) must be of the order of the associated singular values, i.e.,

\[ |\gamma_i| = O(\sigma_i) \text{ for } i \leq m. \]  

(2.27)

The condition above (2.27) requires that the rate of decrease of \(\gamma_i\) to a magnitude of the order of the noise level of the system (i.e., \(\gamma_i \to \epsilon\) for \(i \leq m\)) be faster than that for the singular values \(\sigma_i\). This is a discrete form of Picard's Condition and its use assures the existence of a computable smooth solution. (A solution may exist even if this does not hold but it is not guaranteed to be smooth, and some other approach is necessary.)

The condition (2.27) can be used as a strategy to compute the solution to (2.21) by employing a truncated form of the SVD. Analysing the quantities \(\gamma_i = u_i^Tb\) and \(\sigma_i\) for \(i = 1, \ldots, n\) (which are readily accessible once the SVD has been computed) even in cases where the noise level \(\epsilon\) is not precisely known, a value \(m\) can be selected with

\[ |\gamma_i| \sim \epsilon, \quad m + 1 \leq i \leq n \]  

(2.28)

Then the ratios

\[ \left| \frac{\gamma_i}{\sigma_i} \right|, \quad i = 1, \ldots, m \]  

(2.29)

are all of "reasonable" magnitude, whereas

\[ \left| \frac{\gamma_i}{\sigma_i} \right|, \quad i = m + 1, \ldots, n \]  

(2.30)

grow rapidly in size. In forming the full solution (2.23) large errors are accumulated as the summation proceeds. The division by small numbers (2.30) causes large factors to
be associated with the (oscillatory) singular vectors. A smooth solution is guaranteed if the expansion is truncated before the destructive components are included

\[ x^m = \sum_{i=1}^{m} \frac{u_i^T b}{\sigma_i} v_i. \]  

(2.31)

This is the method known as the Truncated SVD see for example, [8, 38].

If the exact solution to the system (2.21) can be expressed in terms of the first few singular vectors only, this method of truncating the SVD expansion is extremely accurate and the ill-conditioning of the coefficient matrix is of no consequence. If the noise level is unreasonably high, or the kernel particularly smooth so that the singular values decrease very rapidly, the value of \( m \) may be extremely low. In this case, although the computed solution is smooth, the residual may be unacceptably large.

When the magnitude of \( m \) is reasonably large the calculation of a smooth, relatively accurate solution (in terms of a small residual) is easily obtained. This will give a solution to a slightly perturbed system

\[ \hat{A}x = b \]  

(2.32)

where \( \hat{A} = U\hat{\Sigma}V^T \) and \( \hat{\Sigma} = \text{diag}(\sigma_1, \ldots, \sigma_m, 0, \ldots, 0) \). The cost of forming the solution in this way is \( n^3 \) operations for the decomposition plus \( mn^2 \) operations to perform the summation. The smoothness property is guaranteed by selecting \( m \) using the condition (2.27).

The expansion of the solution is assumed to be in the singular vectors of the coefficient matrix. The success of this approach is based on how well the solution can be expressed in terms of the first few singular vectors. Other sets of orthogonal vectors may be used if the solution is known to be expressable in some other form; a truncated form of the QR method is simple to apply, see for example [39],[8]. (The QR method is described in most linear algebra texts, e.g., [15, 16].)
Here the discrete Picard Condition, (2.27), is used both as an indication of the existence of a stable solution and as the strategy for creating it. An alternative strategy can be derived by using the Picard Condition only to show existence of a reasonable solution. The smooth solution may be calculated by restricting the class of admissible solutions by means of a constrained minimization problem. The stability is guaranteed by either minimizing the residual whilst restricting the solution to be smooth:

$$\min_x ||Ax - b||_2^2 \text{ such that } ||x||_2^2 \leq \varepsilon$$  \hspace{1cm} (2.33)

or by minimizing the instability whilst admitting as possible solutions those that result in a small residual:

$$\min_x ||x||_2^2 \text{ such that } ||Ax - b||_2^2 \leq \delta$$  \hspace{1cm} (2.34)

for small values $\varepsilon$ and $\delta$. These two problems (2.33) and (2.34) can be restated as one;

$$\min_x (||Ax - b||_2^2 + \lambda ||x||_2^2)$$  \hspace{1cm} (2.35)

where $\lambda$ is a small parameter.

In this way the ill-conditioned problem is transformed into a well-conditioned form, and can be solved by standard methods. The analogy with the continuous theory is again evident since (2.35) is, in some sense, the discrete equivalent of the Fredholm integral equation of the second kind

$$\check{g}(t) = \lambda \check{f}(t) + \int_a^b \check{K}(s,t) \check{f}(s)ds$$  \hspace{1cm} (2.36)

where

$$\check{g}(t) = K^*g(t) = \int_a^b K(s,t)g(s)ds$$  \hspace{1cm} (2.37)

$$\check{K}(s,t) = KK^*(s,t) = \int_a^b \check{K}(s,t)K(s,t)ds$$  \hspace{1cm} (2.38)
In cases where the integral equation is of the second kind (2.36) the problem is well-posed see [5, 6] and [9]. The corresponding algebraic system is of the form \((I - \lambda A)x = b\) and unless \(\lambda\) is very large the near singularity of \(A\) is not a source of difficulty.

The solution of (2.21) by solving instead (2.35), or (2.32), are examples of the use of the technique of Regularization. The following section outlines the method of Regularization and discusses some of the ways in which it is implemented.

§2.3.1 examines the numerical solution of (2.35) and §2.3.2 presents an alternative approach to Regularization which is motivated by the method of truncating the SVD (2.31).

2.3 Regularization Techniques

The term Regularization is used in a broad sense to mean the replacement of the ill-conditioned problem by a well-conditioned system that is made close to the original by means of a small parameter. There are two basic approaches:

- To alter the specification of the problem by means of a constraint and then apply standard solution methods to the new well-conditioned system (2.35). The parameter is used to control the contribution of the constraint.

- To solve the original least squares problem (2.22) using a modified iterative method to give a sequence of less and less regularized estimates. The regularization parameter is provided by the iteration count.

Both methods try to select a smooth solution that gives a small residual.

The parameter is required to be small so that the modified system resembles the original problem. However, the larger the value is taken for the parameter, the smoother the computed solution will be. Thus the parameter represents some trade-off between
accuracy (in terms of the residual) and smoothness. The choice of the value of the parameter for optimum results is non-trivial, see for example [43]. A straight-forward approach is to make the choice interactively by solving the system for various values of the parameter and selecting the "best" solution.

The first approach is presented in §2.3.1 and the iterative approach to solving the original, minimum norm, problem is dealt with in §2.3.2.

### 2.3.1 Regularization of the Problem

The original method of regularization was introduced by Tikhonov, [37], and Phillips, [28]. In its general form a smoothing operator is included within the constraint and is stated as:

$$\min_{x}(\|Ax - b\|_2 + \lambda\|Lx\|_2), \quad (2.39)$$

where $\lambda$ is a small parameter and $L$ some chosen operator. $L$ is usually some discrete approximation to a differential operator so that for example, $Lx = x, \bar{x}, \ddot{x}$. The above system (2.39) is well conditioned and will have a unique solution provided the intersection of the null-space of $A$ and $L$ is trivial, i.e., $N(K) \cap N(L) = \emptyset$.

**Regularization in Standard Form.**

With the operator $L$ taken as the identity so that $Lx = x$ the problem (2.39) is often referred to as being in standard form. The solution can then be performed using the associated normal equations

$$(A^TA + \lambda I)x = A^Tb. \quad (2.40)$$

If the SVD of $A$ is obtained (2.40) can then be written as

$$V(\bar{\Sigma}^2)V^T x = V \Sigma U^T b \quad (2.41)$$
where $U, V$ and $\Sigma$ are the matrices as from the SVD and $\tilde{\Sigma}$ is a diagonal matrix with non-zero entries as $\{\sigma_i + \lambda\}$. The solution is the expressable as the sum

$$x_\lambda = \sum_{i=1}^{n} \frac{\sigma_i}{\sigma_i^2 + \lambda}(u_i^T b)v_i.$$  

(2.42)

The expansion (2.42) is given in the same set of vectors as the truncated form of the SVD given above (2.31). The difference is that all the singular vectors contribute to the solution but the contributions are damped (from the exact solution (2.23)) by a factor

$$\frac{\sigma_i^2}{\sigma_i^2 + \lambda}.$$  

(2.43)

This results in a smoother transition of the weights than in the truncated form of the SVD (2.31).

Once the singular value decomposition has been performed the solution can be efficiently obtained a number of times for different values of the parameter $\lambda$, and the best one selected.

Regularization in this standard form performs similarly to the method of truncating the singular value decomposition see [39].

The Method of Tikhonov Regularization

The choice above of the operator $L$ as the identity is not necessarily optimal. The stability of $x$ can be further ensured by the inclusion of the smoothing operator as part of the constraint. The associated normal quations are then

$$(A^T A + \lambda L^T L)x = A^T b$$  

(2.44)

An expansion solely in terms of the singular vector of the matrix $A$ is no longer possible. Various suggestions have been given to produce an effective solution method to avoid the formation of the factors $A^T A$ and $L^T L$. 
[8, 16] and [38] demonstrate the use of the generalized singular value decomposition of the matrix pair $(A, L)$ so that

\[
A = UD_dW^{-1} \quad \text{(2.45)}
\]
\[
L = VD_eW^{-1} \quad \text{(2.46)}
\]

where $U$ and $V$ are orthogonal, $D_d$ and $D_e$ diagonal and $W$ causes the diagonalization of $(A, L)$. This gives the solution as

\[
x_\lambda = \sum_{i=1}^{n} \left( \frac{u_i^T b}{d_i + \lambda e_i/d_i} \right) w_i. \quad \text{(2.47)}
\]

In [27], a bidiagonalization algorithm is presented as a more efficient method of performing the solution to (2.39).

**Use of Iterative Methods**

The methods above are effectively direct methods for evaluating a solution to the well-conditioned problem (2.39). Solution of (2.39) may also be performed using iterative techniques. The normal equations may be written in factored form as:

\[
\begin{pmatrix}
A^T & \lambda L^T \\
\lambda L & \lambda L
\end{pmatrix}
\begin{pmatrix}
A \\
\lambda L
\end{pmatrix} = A^T b. \quad \text{(2.48)}
\]

Most iterative techniques can then be applied to this factored form of the equation, see [8, 41, 45]. The performance of the iterative scheme is still very sensitive to the condition number of $A$. This is in contrast to the direct approaches above where the ill-conditioning of the coefficient matrix is no longer a source of trouble. The performance of the iterative technique may be improved by some preconditioning of the equations before they are solved. This may be done by the introduction of a scaling matrix to produce a coefficient matrix with an improved singular value spectrum. However, this has the effect
of accelerating the computation of an accurate solution and in so doing may "miss" the smooth approximation: it is not obvious whether the use of preconditioning is advised for any particular ill-conditioned system [8].

The use of iterative techniques for the regularized problem is also inhibited by the presence of the parameter. For each new value of the parameter the entire solution process must be completed and thus is wasteful in computational effort. The use of an iterative method as a solution strategy for the problem (2.39) is thus limited to very large, sparse systems, where the use of direct methods is prohibitive.

2.3.2 Regularization of the Solution: Iterative Filtering

In §2.3.1 the solution of ill-conditioned problems is achieved by constraining the equations by solving instead the equation (2.39) for small positive values of $\lambda$, i.e., $\lambda > 0$. The use of iterative solution techniques is limited. If the parameter $\lambda$ is set to zero (thus requiring the solution of the original Least Squares problem) iterative techniques may be used successfully. The regularization is achieved by terminating the iteration before the components corresponding to the small singular values have converged. The effect is to discriminate against the unstable part of the solution by filtering the contributions of the singular vectors. The truncated form of the SVD in §2.2 is, in some sense, an example of this Iterative Filtering technique. As the iteration proceeds a sequence of less and less regularized estimates of the solution is produced. The extent of the regularization is therefore controlled by the choice of iterative method and the (hopefully small) number of iterations performed.

In the truncated form of the SVD the singular vectors $v_i$ appear as components of $x^m$ either with their full weighting, as in $x^*(2.23)$ (for $i \leq m$) or not at all (for $i > m$). In
this way the truncated SVD expansion can be expressed as

\[ x^m = \sum_{i=1}^{n} H(i) \frac{u_i^T b}{\sigma_i} v_i \]  

(2.49)

where the weight function \( H(k) \) is,

\[ H(k) = \begin{cases} 1 & k = 1, \ldots, m \\ 0 & k = m + 1, \ldots, n \end{cases} \]

Another strategy is to allow all the singular vectors to contribute by a weight function \( F(\sigma) \) that discriminates against the singular vectors associated with the small singular values. The function \( F(\sigma) \) is such that

\[ F(\sigma) \begin{cases} \sim 1 & \text{for } \sigma \text{ "large"} \\ \sim 0 & \text{for } \sigma \text{ "small"} \end{cases} \]

and has a smooth transition for the intervening values of \( \sigma \).

A general iterative technique is given by Strand [36],

\[ x_k = x_{k-1} + M(A^TA)A^T(b - Ax_{k-1}) \]  

(2.50)

where \( M(\sigma^2) \) is some polynomial. The initial estimate \( x_0 = 0 \) is assumed but is not restrictive.

The solution to (2.50) can be written in terms of the singular value decomposition as:

\[ x_k = \sum_{i=1}^{n} \mathcal{F}_k(\sigma_i^2) \frac{u_i}{\sigma_i} v_i, \]  

(2.51)

where

\[ \mathcal{F}_k(\sigma^2) = 1 - (1 - \sigma^2 M(\sigma^2))^k \]  

(2.52)

(2.51) can then be compared with (2.23). The effect is to damp the component along the singular vectors \( v_i \) by an amount \( \mathcal{F}_k(\sigma_i^2) \). \( \mathcal{F}(\cdot) \) is the Filtering Function. Different choices for the filtering function result in different solution schemes.
A special case of (2.50) is given by Landweber, [24], where a simple iterative strategy is applied to the normal equations so that

$$x_k = x_{k-1} + A^T(b - Ax_{k-1})$$

(2.53)

with $A$ normalized so that $\sigma_1 = 1$. (2.53) implicitly takes $\mathcal{M}(\sigma^2) = 1$. This scheme proceeds relatively slowly since after $k$ iterations only the components corresponding to singular values such that $\sigma_i > 1/\sqrt{k}$ have converged.

[17] gives a study of various forms of (2.50) where $T$ is taken as some continuous approximation to the Heaviside step function:

$$H(z) = \begin{cases} 
0 & z = 0 \\
1 & \text{otherwise}
\end{cases}$$

A series of possibilities using varying degrees of the best least squares fits and Chebyshev polynomials as the approximating function are investigated.

This thesis investigates the “filtering” properties of the iterative method of Conjugate Gradients and thus assesses the success with which it can be used as a method for the solution of ill-posed problems. The next chapter details the method of Conjugate Gradients before examining its use as a regularization procedure in Chapter 4.
Chapter 3

The Method of Conjugate Gradients

The method of Conjugate Gradients is an efficient algorithm for computing the solution to a system of linear equations

\[ Ax = b \]  \hspace{1cm} (3.54)

where \( b \in \mathbb{R}^n \) and \( A \in \mathbb{R}^{n \times n} \) is assumed to be symmetric positive definite.

The method was developed independently by Hestenes and Stiefel [22]. The algorithm is based on the process of locating the minimum of a quadratic function over decreasing dimensional planes and can be easily implemented as a repeated series of simple steps. Each iteration requires the formation of just one matrix-vector product and provides an improved estimate of the required solution. Mathematically, the solution \( x^* = A^{-1}b \) will be evaluated in at most \( n \) repetitions and so is a direct solver. Unfortunately, computationally the effects of numerical roundoff are unfavourable. Although ineffective as a direct scheme (as it was originally proposed) a good estimate of the solution is obtained early in the process, regardless of roundoff errors and hence the method of Conjugate Gradients can be used as a successful iterative procedure. Considerations of the convergence rate led to further development of the Preconditioned Conjugate Gradients. This chapter gives an outline of the method of Conjugate Gradients.

The method of Conjugate Gradients belongs to the general class of Conjugate Directions methods, which is motivated by the theory of function minimization. §3.1 uses the minimization process to develop the method of Conjugate Directions. The development begins with the method of Steepest Descents. An examination of the drawbacks of this
simplest method lead to the Conjugate Directions algorithm.

The Conjugate Gradients algorithm itself is discussed in §3.2. §3.2.1 derives the scheme as a special instance of the Conjugate Directions algorithm and §3.2.2 outlines some of the features of the method which will be required in Chapter 4. The effects of roundoff errors and the use of the method as an iterative scheme, are discussed in §3.2.3. Finally, an estimate of the convergence rate of the scheme is mentioned in §3.2.4 and used to motivate the idea of a preconditioned algorithm.

The preconditioned algorithm is discussed in more detail in §3.3. §3.3.1 presents Conjugate Gradients as a method based on a splitting of the coefficient matrix, which leads to a discussion of the role of the preconditioner. Various choices for the preconditioner are mentioned in §3.3.3.

Finally, §3.4 extends the algorithm to cases for which the coefficient matrix is not symmetric positive definite. An algorithm for the Linear Least Squares problem is discussed in §3.4.1 and the preconditioned algorithm is presented in §3.4.2.

3.1 The Method of Conjugate Directions

The solution of the linear system (3.54) and the Conjugate Direction algorithm itself can be motivated by considering the minimization of the quadratic function \( \phi(x) \), defined by

\[
\phi(x) = \frac{1}{2}x^TAx - x^Tb. 
\]

Necessary conditions for the existence of a local minimum for \( \phi (3.55) \) at \( x^* \) are given by:

1. \( \nabla \phi(x^*) = 0 \)
2. \( \nabla^2 \phi(x^*) \) is positive definite.

Hence with \( A = \nabla^2 \phi \) symmetric positive definite, locating the minimum of (3.55) is thus equivalent to solving \( \nabla \phi(x) = Ax - b = 0. \) The problem of locating the minimum
of (3.55) is equivalent to the solution of the linear system (3.54). One of the simplest iterative methods for calculating the location of the minimum $x^*$ of $\phi$ is the method of steepest descents. At the point $x_k$, $\phi$ decreases most rapidly along the direction of the negative gradient of $\phi$

$$- \nabla \phi(x_k) = b - Ax_k.$$  

(3.56)

The vector $r_k = b - Ax_k$ is the residual of the system at $x_k$. In the method of steepest descents the next estimate $x_{k+1}$ is chosen as the point which minimizes $\phi$ along $r_k$, and is determined by choosing a step length parameter $\alpha_k$ which minimizes $\phi(x_k + \alpha_k r_k)$. This gives a step length of

$$\alpha_k = \frac{r_k^T r_k}{r_k^T A r_k}. \tag{3.57}$$

The speed at which this strategy converges to $x^*$ can be prohibitively slow. The search directions are along the inward normals of the $(n - 1)$-dimensional ellipsoids, centred at $x^*$ which form the level surfaces of $\phi$. The singular values of $A$ are precisely the length of the semi-axes of these ellipsoids. For ill-conditioned problems where the ratio of largest to smallest singular value is large, the level surface ellipsoids are very elongated and relatively flat valleys with steep sides. The method of steepest descents requires that at each iteration the new search direction be perpendicular to the last by making a right-angled turn at the point where the search direction becomes tangent to the contour lines. This forces the minimizer to be located by a series of many short steps that proceed back and forth across the valley floor. (The system does not have to be very ill-conditioned before this can be destructive. The presence of one singular value that is much smaller than the rest can produce the elongation and force slow progress.)

This slow progress can be overcome by searching instead along a set of directions that do not correspond to the negative gradient of $\phi$ at each point. One strategy, used by the method of Conjugate Directions, is suggested by the following result [21]:


If \( \hat{x} \) is the unique minimizer of \( \phi \) on the plane defined by the set of \( k \) mutually conjugate vectors \( \pi_k = \{p_1, \ldots, p_k\} \) then the minimum of \( \phi \) is located at \( x^* \) which is contained within the \( (n - k) \)-plane \( \pi_{n-k} \) defined to lie through \( \hat{x} \) and conjugate to \( \pi_k \).

This result means that the dimension of the space, in which \( x^* \) is known to lie, can be reduced.

The notion of conjugacy is an extension to the idea of orthogonality. Two vectors \( u \) and \( v \) are \( G \)-conjugate if the vector \( u \) is orthogonal to the vector \( Gv \), i.e., \( u^T G v = 0 \). (This leads to the term \( G \)-orthogonal sometimes being used.) Unless otherwise specified, in the case where the matrix \( G \) is the coefficient matrix for the system (3.54), the condition \( A \)-conjugate will simply be stated as conjugate.

The minimization of \( \phi \), by the method of Conjugate Directions, is performed by systematically reducing the dimension of the search space as follows:

- Select a point \( x_0 \) and a line \( L_1 \) through \( x_0 \) in the direction \( p_1 \);
- Find \( x_1 \) that minimizes \( \phi \) on \( L_1 \) exactly;
- Construct the \( (n - 1) \)-plane \( \pi_{n-1} \) through \( x_1 \) and conjugate to \( p_1 \).

By the above result, \( \pi_{n-1} \) contains \( x^* \), the location of the minimum of \( \phi \) (thus the dimensionality of the problem has been reduced by one).

This process is repeated starting with the point \( x_1 \) and confining the search to the \( (n - 1) \)-plane \( \pi_{n-1} \).

- Select a line \( L_2 \) in \( \pi_{n-1} \) through \( x_1 \) in the direction of \( p_2 \in \pi_{n-1} \);
- Find \( x_2 \) that minimises \( \phi \) on \( L_2 \) exactly;
- Construct the \( (n - 2) \)-plane \( \pi_{n-2} \) through \( x_2 \) and conjugate to \( p_2 \).
Now $x^*$ is known to be contained in $\pi_{n-2}$.

In this way an iterative procedure is created. At each step the dimension of the search space is reduced by one.

At the $n$-th step the search space $\pi_1$ is a line $L_n$. The location of minimum of $\phi$ on $L_n$, is exactly $x^*$ the location of the global minimum, as required. Hence the termination of the procedure is guaranteed within $n$ steps. The termination of the minimization process within $n$ steps is independent of the initial point $x_0$. For convenience it is usual to choose $x_0 = 0$ as the starting point, but this is not necessary. (If an improved estimate of the solution is available this should be substituted.)

Constructing the planes $\pi_k$ as described above does not produce an effective numerical algorithm. However, the requirement that the next search direction $p_{k+1}$ be in the $(n-k)$-plane $\pi_{n-k}$ (which contains $x^*$) is equivalent to chosing $p_{k+1}$ to be conjugate to all the previous search directions $\{p_1, \ldots, p_k\}$ [21]. This suggests the Conjugate Directions method.

The iterative step for the method of Conjugate Directions can be defined as:

- Select $p_k$ to be conjugate to $\{p_1, \ldots, p_{k-1}\}$;
- Find $x_k$ that minimizes $\phi$ on the line $x_{k-1} + \alpha_k p_k$ exactly.

Each value of $\alpha_k$ is found, as in the method of steepest descents, by differentiating $\phi(x_{k-1} + \alpha_k p_k)$ with respect to $\alpha_k$ and equating the result to zero. This yields the step length as:

$$\alpha_k = \frac{p_k^T r_{k-1}}{p_k^T A p_k},$$

which is always well-defined since $A$ is assumed to be a symmetric positive definite matrix, so by definition $(u, Au) \neq 0$ for any vector $u$.

As presented, there is no explicit specification for the choice of the search directions $p_k$, but reducing the magnitude of $\phi(x_k)$ at each step imposes a restriction on $p_k$. The
value of $\phi$ at each step is given by

$$
\phi(x_k) = \phi(x_{k-1} + \alpha_k p_k)
= \phi(x_{k-1}) + \alpha_k p_k^T (Ax_{k-1} - b) + 1/2\alpha_k^2 p_k^T A p_k
$$

then with the value of $\alpha_k$ as given in (3.58), this gives

$$
\phi(x_k) = \phi(x_{k-1}) - 1/2 \frac{(p_k^T r_{k-1})^2}{p_k^T A p_k}
$$

(3.59)

Thus to ensure a reduction in $\phi$, the direction $p_k$ must be chosen such that it is not orthogonal to the current residual, i.e., $p_k^T r_{k-1} \neq 0$.

A practical algorithm for the method of Conjugate Directions is given as follows:

**Conjugate Directions Algorithm**

Set $x_0 = 0$ and $r_0 = b - Ax_0$

For $i = 1, 2, \ldots$ do while $r_k \neq 0$

choose a direction $p_k$ such that $p_k^T r_{k-1} \neq 0$

$$
\alpha_k = \frac{p_k^T r_{k-1}}{p_k^T A p_k}
$$

(3.60)

$$
x_k = x_{k-1} + \alpha_k p_k
$$

(3.61)

The steps above form the basis for the taxonomy of algorithms known as Conjugate Direction Methods. This general class of algorithm does not place any requirement on the search directions, other than requiring that they not be orthogonal to the residual at each step and that together they form a mutually conjugate set. Specific Conjugate Direction schemes are distinguished by the method that is used to evaluate the search directions. The next section details the method of Conjugate Gradients in which each search direction is formed as an orthogonal projection of the residual.
3.2 The Method of Conjugate Gradients

The general method of Conjugate Directions described in the previous section allows some choice over the method of selecting and computing the search directions. Various choices lead to specific methods. The original method of Conjugate Gradients was suggested by Hestenes and Stiefel. They independently formulated a scheme in which the search directions are formed by $A$-orthogonalization of the residuals. A full description of this scheme, its properties and variations, is given in [22].

§3.2.1 presents the algorithm itself, while some of its properties are mentioned in §3.2.2. The use of the algorithm as an iterative scheme is outlined in §3.2.3 and its convergence rate is examined in §3.2.4.

3.2.1 The Conjugate Gradients Algorithm

The method of Conjugate Directions specifies only the space from which search directions must be selected. Each new search direction is to be conjugate to all previous search directions, which is equivalent to requiring that $p_{k+1} \in \text{span}\{A_p, \ldots A_p\}^\perp$ [16],[21] (where $S^\perp$ is the orthogonal complement of the set $S$.)

Recall that at any point the direction in which the function shows most rapid decrease is that of the negative gradient at that point. The conjugacy requirement for the direction vectors in the the Conjugate Directions method was introduced to reduce the problem of the slow convergence of the method of steepest descents. However it could be advantageous to use as much of the information from the residual direction as possible. One way to achieve this is to make the search direction the orthogonal projection of the residual onto the space containing the admissable search directions. (This is in one sense making the search direction as close as possible to the direction of negative gradient.)
Chapter 3. The Method of Conjugate Gradients

This causes any new search direction to be conjugate to the residual

$$\tau_k^T A p_{k+1} = 0$$

(3.62)

and gives the method of Conjugate Gradients its name.

The method of Conjugate Gradients selects the first search direction as the negative gradient and thereafter each new search direction $p_{k+1}$ is the orthogonal projection of the corresponding residual $r_k$ onto $\text{span}\{A p_1, \ldots, A p_n\}^\perp$.

The Conjugate Gradients algorithm is simply a modification of the general Conjugate Directions algorithm which reflects the above considerations.

- Set $x_0 = 0$;
- Compute $p_1$ to be the vector of steepest descent;
- Obtain the minimum $x_1$ of $\phi$ on the line through $x_0$ in the direction of $p_1$. Compute $p_2$ to be the orthogonal projection of $r_1$ on to $\text{span}\{A p_1\}^\perp$;
- Obtain $x_2 = \min_{\alpha} \phi(x_1 + \alpha p_2)$;
- Iterate until at the $n$-th step, $x_n$ will be the required minimum $x^*$.

The precise calculation of each direction vector is possible from the orthogonality and conjugacy requirements. From (3.62) each new search direction is contained in the space described by the last search direction and the residual so that $p_k \in \text{span}\{r_{k-1}, p_{k-1}\}$ [16], so without loss of generality $p_k$ can be represented as $p_k = r_{k-1} + \beta_k p_{k-1}$, for some $\beta_k$. It is required that each new direction be conjugate to any previous search direction so that

$$p_{k-1}^T A p_k = 0$$

$$= p_{k-1}^T A (r_{k-1} + \beta_k p_{k-1}).$$
Therefore

\[ \beta_k = -\frac{p_{k-1}^T A r_{k-1}}{p_{k-1}^T A p_{k-1}}. \]  

(3.63)

This formula for \( \beta_k \) can be simplified by using the expression for the step length (3.58) and the recurrence relation for the residual

\[ r_k = r_{k-1} - \alpha_k A p_k, \]

(3.64)

and the fact that each search vector must be orthogonal to all future residual directions. Together these give

\[ \|r_{k-1}\|^2 = (r_{k-2} - \alpha_{k-1} A p_{k-1})^T r_{k-1} \]

(3.65)

\[ = r_{k-2}^T r_{k-1} - \alpha_{k-1} p_{k-1}^T A r_{k-1} \]

(3.66)

\[ = -\alpha_{k-1} p_{k-1}^T A r_{k-1} \]

(3.67)

And

\[ p_k^T r_{k-1} = (r_{k-1} + \beta_k p_{k-1})^T r_{k-1} \]

(3.68)

\[ = (r_{k-1}^T r_{k-1}). \]

(3.69)

Then by substituting (3.69) into (3.58) gives (3.71), and substituting (3.67) into (3.63) with (3.71) gives (3.71) The expressions (3.58) and (3.63) can then be rewritten as:

\[ \alpha_k = \frac{r_{k-1}^T r_{k-1}}{p_k^T A p_k} \]

(3.70)

and \( \beta_k = \frac{\|r_{k-1}\|^2}{\|r_{k-2}\|^2}. \)

(3.71)

The steps for the Conjugate Gradients algorithm can now be presented in a form that is practical for programming purposes.
Conjugate Gradients Algorithm.

Set $x_0 = 0$, $r_0 = b$ then for $k = 1, 2, 3, \ldots, n$

\begin{align*}
\beta_k &= \frac{r_{k-1}^2 r_{k-1}}{r_{k-2}^2 r_{k-2}} \quad (\beta_1 = 0) \\
p_k &= r_{k-1} + \beta_k p_{k-1} \\
\alpha_k &= \frac{r_{k-1}^2 r_{k-1}}{p_k^T A p_k} \\
x_k &= x_{k-1} + \alpha_k p_k \\
r_k &= r_{k-1} - \alpha_k A p_k.
\end{align*}

The formulae for $\alpha, \beta$ and $r$ can be written in various forms (see [22] and [29]) but the forms given above are preferred for implementation [29].

The algorithm above appears as it was first presented. Despite its mathematical beauty and computational simplicity the algorithm was not popular, as in practice it proved to be uncompetitive with other available schemes. The next section discusses the source of the problems and shows that when the method is regarded as an iterative scheme (rather than as a direct scheme) it can be considered as a viable solution method.

### 3.2.2 Properties of the Method of Conjugate Gradients

Theoretically the Conjugate Gradients method presented above possesses favourable qualities. It involves at most (in exact arithmetic) $n$ repetitions of a series of simple steps. The coefficient matrix $A$ is not involved in any decomposition process and is required only for the single matrix-vector product performed in the calculation. Any special properties of the matrix $A$ (e.g., banded structure) can be fully exploited reducing the computational effort per iteration, and the storage requirements. This is of particular importance in cases where $A$ is large and sparse. Practically, though, the algorithm's performance falls short of these high expectations.
The Residuals and Search Directions

The residuals $r_k$ are the search directions used by the method of steepest descent, but the Conjugate Gradients algorithm uses the search directions as the conjugate vectors $p_i$ to produce an improved performance. In the algorithm (3.77) the particular conjugate vectors, formed by orthogonal projections of the residuals, create a set of relations between the residuals and the search directions. By definition the direction vectors $\{p_1, \ldots, p_n\}$ form a mutually conjugate set, and the residuals $\{r_0, \ldots, r_n\}$ are mutually orthogonal. In addition $p_i^T r_j = 0$, for $i < j$. We have also that $\text{span}\{p_1, \ldots, p_k\} = \text{span}\{r_0, \ldots, r_{k-1}\} = \text{span}\{b, Ab, \ldots, A^{k-1}b\}$. So in summary:

\[
(r_i, r_j) = 0 \quad i \neq j
\]
\[
(p_i, Ap_j) = 0 \quad i \neq j
\]
\[
(p_i, r_j) = 0 \quad i < j
\]
\[
(p_i, r_j) = \|r_i\|^2 \quad i \geq j.
\]

The Parameters $\alpha$ and $\beta$

The two parameters $\alpha$ and $\beta$ used in the method of Conjugate Gradients are calculated by known formulae (3.58) and (3.63). No a priori knowledge of the system, or other user interaction is required. This is in contrast to many other methods (e.g. SOR and Chebychev iteration) which require parameters to be estimated in order to attain optimal efficiency.

The parameter $\beta_k$ is the square of the ratio of the last two residual $L_2$ norms. If at each step the error $e_k$ is defined as

\[
e_k = x^* - x_k \quad (3.78)
\]
and the inner product \( E(e_k) = (e_k, Ae_k) \), then

\[
E(e_k) = (x^*-x)^T A(x^*-x_k) \\
= ||x^*-x_k||_A \\
= x^T A x^* - 2\phi(x_k)
\]

and the minimization of (3.55) is equivalent to the minimization of the \( A \)-norm of the error. This means that the method of Conjugate Gradients ensures the reduction in magnitude of \( ||x^*-x_k||_A \) at each step and therefore guarantees that an improved estimate to \( x^* \) is obtained at each iteration. The reduction in length of the error vector does not guarantee a monotonic decrease in the magnitude of the residuals, so no bound can be predicted for the \( \beta_k \)'s.

The step lengths \( \alpha_k \) are such that their reciprocal is bounded between the largest and smallest singular values of the matrix \( A \) [22]. With the singular-values ordered \( \sigma_1 \geq \ldots \geq \sigma_n \),

\[
\frac{1}{\sigma_1} < \alpha_k < \frac{1}{\sigma_n}.
\]  

(3.82)

For well conditioned systems a reasonable indication of the sizes of the \( \alpha_k \) can be estimated but when \( \frac{\sigma_1}{\sigma_n} \) is large the \( \alpha_k \)'s are effectively unbounded. However, since it is assumed that the matrix \( A \) is symmetric positive definite, each step length is positive, i.e., \( \alpha_k > 0 \) for \( k = 1, \ldots, n \).

**Finite Termination**

The termination of the method within a finite number of steps is guaranteed using exact arithmetic. The direction vectors \( \{p_1, \ldots, p_k\} \) form a set of \( k \) linearly independent vectors, so the full set of \( n \) direction vectors is therefore \( \text{span}\{p_1, \ldots, p_n\} = \mathbb{R}^n \). An algorithm that minimizes \( \phi \) over \( \text{span}\{p_1, \ldots, p_n\} \) must (in exact arithmetic) locate the minimum
In addition, if $A$ is such that it possesses only $t$ distinct, non-zero, singular values then the solution can be expressed as a combination of $t$ independent vectors and the method will locate the minimum in at most $t$ iterations [22].

### 3.2.3 Conjugate Gradients as an Iterative Method

Following the publication of the original paper detailing the Conjugate Gradients algorithm very little interest was shown in the technique. This neglect stemmed from the fact that in a practical situation its major feature, the finite termination property, failed to hold. The algorithm was not competitive with such schemes as Gaussian Elimination with regard to both accuracy and efficiency, and it fell into disuse. However the method's other properties were promising and this led to a later re-evaluation of the scheme [29].

In [29] extensive tests are performed which demonstrate the effects caused by roundoff errors. Errors due to roundoff are propagated in the formation of the vector products $(r_{k-1}, r_k)$ and $(Ap_{k-1}, p_k)$ [29] so that the orthogonality conditions

\[
(r_i, r_j) = 0 \text{ and } (Ap_i, p_j) = 0 \quad \text{for } i \neq j
\]

no longer hold, and the guarantee of termination within $n$ steps is lost. The propagation effects result in the accumulation of the errors, so that the component of $r_k$ in the most recent direction is accurate but earlier components are less so. This means that the method of Conjugate Gradients is useful as a direct method only for systems of very small order.

One way to minimize the effects of roundoff errors is to restart the process once a reasonable estimate has been obtained [22]. The guarantee of the monotonic decrease in the length of the error $\|x^* - x_k\|_2$ (from (3.79)) justifies the use of any calculated estimate as an improved starting point. However, too many restarts would destroy the advantages of the process over that of steepest descents, and is not in general recommended.
Even when roundoff affects the performance of the algorithm a reasonably accurate estimate of \( x^* \) is obtained long before the \( n \)th step [22]. Reid found (for relatively well-conditioned systems) that the guaranteed reduction of \( \| e_k \|_2 \) held despite the effects of roundoff. This observation led Reid [29] to propose the scheme's successful implementation as an iterative process to generate approximate solutions, and indeed (3.77) is an efficient and competitive procedure compared with other iterative schemes such as Chebyshev or Jacobi iteration. Using economical procedures to perform vector products makes the method of Conjugate Gradients particularly effective for solutions of large sparse systems of linear equations.

Implementing the algorithm as an iterative scheme is particularly economical as it requires \( 5n + 2 \) operations and one matrix-vector product (which will involve a maximum of \( n^2 \) multiplications) per iteration. The product \( A r_k \) can be stored each time to avoid unnecessary multiplications. The storage requirement for the scheme is therefore 4 \( n \)-vectors (\( x, r, p \) and \( A p \)). In the case where \( A \) has a favourable sparsity structure the operation count can be reduced considerably.

In the evaluation of the Conjugate Gradients algorithm as an iterative scheme the critical issue is the number of steps required to reduce the error by a given amount, and thus the computational effort needed to obtain a reasonable estimate. An examination of the rate at which the algorithm (3.77) converges follows in the next section.

### 3.2.4 Convergence Rate

A measure of the efficiency of an iterative solution method is given by its convergence rate. At each step the Conjugate Gradients algorithm produces a new estimate to the solution \( x^* \) such that the error functional (3.79) is minimized. Various attempts have been made to specify a bound for this error and so give an estimate of the rate of convergence
see [12] and references therein. In [10] it is shown that

\[ \|x^* - x_k\|_A \leq Q_k \|x^* - x_0\|_A \] (3.84)

where \( Q_k \) depends only on the singular values of the matrix \( A \). For a general matrix \( A \) it is not possible to specify \( Q_k \) exactly, but different bounds can be placed on the convergence rate by obtaining a value for \( Q_k \) given some information about the singular value spectrum.

A basic bound can be generated [10] assuming that an interval \([a, b]\) is known to contain all the singular values of \( A \). Then

\[ Q_k \leq \left( \frac{1 - \sqrt{\kappa}}{1 + \sqrt{\kappa}} \right)^{2k} \] (3.85)

This is a reasonable bound for matrices where \( \kappa \) (the condition number of \( A \)) is small, but severe ill-conditioning could lead to impractically slow convergence. In cases where more detailed information on the exact structure of the spectrum is available (e.g., clustering of the singular values) this bound can be improved see for example [4] and [30].

The convergence rate of an iterative scheme can often be improved by preconditioning the coefficient matrix. Preconditioning refers to the strategy of transforming the matrix in question to reduce its condition number, or to cluster or otherwise redistribute its eigenvalues. The technique is well understood and widely used for such methods as SSOR [3].

The method of Conjugate Gradients is also well suited to the idea of preconditioning. In exact arithmetic the number of steps required to converge to the solution \( x^* \) is equal to the number of distinct singular values of the coefficient matrix, so the performance of the Conjugate Gradients Algorithm takes account of the full singular value spectrum and not just those on the periphery. Any clustering of the singular values, which would effectively reduce the number of distinct singular values, would therefore have a favourable effect.
on the efficiency of the scheme. An implementation of the Preconditioned Conjugate Gradients algorithm is given in the next section.

3.3 Preconditioned Conjugate Gradients

Recognising the method as an iterative process permits the loss of orthogonality conditions (3.83) to be tolerated, but the question of convergence rate is then much more important. The error bound (3.85) suggests an unfavourable result for poorly conditioned systems. The notion of preconditioning the method of Conjugate Gradients is thus well motivated.

There are two approaches to the idea: The process can be regarded as

• Using Conjugate Gradients as an acceleration to another iterative process.

• Preconditioning the system for Conjugate Gradients iteration.

Theoretically the views are equivalent but, as is shown below, while the first gives a clearer understanding of the notion the second is used for practical implementation. §3.3.1 derives the preconditioned Conjugate Gradients method as a special case of a generalized two step iterative scheme based on a splitting. §3.3.2 gives a practical preconditioned algorithm and discusses the issues involved in the choice of the preconditioner. Some common preconditioners are mentioned in §3.3.3.

3.3.1 Generalized Conjugate Gradients

A generalized two step iterative scheme based on the splitting of the matrix $A = M - N$, where $M$ is symmetric positive definite, can be formed by:

Set $x_{-1} = x_0 = 0$ and $r_0 = b$

For $k = 1, 2, \ldots$ or until $\|r_k\| < TOL$
Chapter 3. The Method of Conjugate Gradients

Solve $Mz_{k-1} = b - (M - N)x_{k-1} = r_{k-1}$

Choose parameters $\omega_k$ and $\nu_{k-1}$ and set

$x_k = x_{k-2} + \omega_k(\nu_{k-1}z_{k-1} + x_{k-1} - x_{k-2}).$

Various iterative methods can be described in this way, e.g., the Jacobi, Gauss-Seidel, and SOR schemes, [16]. Combining the scheme above into one equation gives

$$Mz_k = Mz_{k-2} - \omega_k(\nu_{k-1}(M - N)z_{k-1} + M(z_{k-2} - z_{k-1}))$$ (3.86)

If the parameters $\nu_{k-1}$ and $\omega_k$ are computed so that $z_i^T M z_j = 0$ for $i \neq j$ then $\exists k \leq n$ such that $z_k = 0$, and so $x_k = x^*$. This description is equivalent to a generalized two step Conjugate Gradients method and will therefore converge in no more than $n$ steps. The required values of the parameters $\nu_{k-1}$ and $\omega_k$ are:

$$\nu_{k-1} = \frac{z_{k-1}^T M z_{k-1}}{z_{k-1}^T (M - N) z_{k-1}}$$

and

$$\omega_k = (1 - \frac{\nu_{k-1} z_{k-1}^T M z_{k-1}}{\nu_{k-2} z_{k-2}^T M z_{k-2} \omega_{k-1}})^{-1}$$

Since $M$ is symmetric positive definite, it can be represented as $M = CC^T$ for some non-singular matrix $C$. Choosing a splitting $A = M - N$ is thus equivalent to choosing a scaling matrix $C$. The system $Ax = b$ then becomes the transformed system

$$\bar{A}\bar{x} = \bar{b}$$ (3.87)

with

$$\bar{A} = C^{-1}AC^{-T}, \bar{x} = C^{-T}x, \bar{b} = C^{-1}b$$ (3.88)

(note that $\bar{A}$ is still symmetric positive definite). The Conjugate Gradients algorithm (3.77) can be applied to this preconditioned system.
3.3.2 Practical Implementation of Preconditioned Conjugate Gradients

Applying the Conjugate Gradients algorithm to the preconditioned problem (3.87) gives:

Set $C x_0 = 0$ and $Q^{-1} r_0 = b$ then for $k = 1, 2, 3, \ldots$

\[
\begin{align*}
\beta_k & = \frac{r^T_{k-1} C^{-1} C^{-1} r_{k-1}}{r^T_{k-2} C^{-1} C^{-1} r_{k-2}} \quad (\beta_1 = 0) \\
p_k & = C^{-1} r_{k-1} + \beta_k p_{k-1} \\
\alpha_k & = \frac{r^T_{k-1} C^{-1} C^{-1} r_{k-1}}{p_k^T C C^{-1} A C^{-1} p_k} \\
x_k & = x_{k-1} + \alpha_k p_k \\
C^{-1} r_k & = C^{-1} r_{k-1} - \alpha_k C^{-1} A C^{-1} p_k.
\end{align*}
\]

Computing $A$ is not advised as this would increase the expense and introduce unnecessary roundoff error. Generally, even if $A$ has any sparsity structure (e.g. a banded matrix) $A$ will not, and will require increased storage. The major part of the work per iteration is in performing matrix-vector products. It is possible in such cases that any advantage of routines written to perform the matrix-vector products $Av$ efficiently will be redundant and the overall work per iteration for the preconditioned system may increase substantially. The additional cost of applying the preconditioner must be sufficiently small so as to not negate the savings produced by the increased convergence rate.

A practical scheme is achieved by simplifying each line in the algorithm and setting $M = C^2$. The extra matrix-vector products are reduced by setting $z = M^{-1} r$, which then requires the solution of the system $M z_k = r_k$ once per iteration. The algorithm below is produced.

**Preconditioned Conjugate Gradients Algorithm.**

Set $x_0 = 0$ and $r_0 = b$ then for $k = 1, 2, 3, \ldots$ and $\|r_k\| < TOL$

Solve $M z_k = r_k$
Chapter 3. The Method of Conjugate Gradients

\[ \beta_k = \frac{z_{k-1}^T r_{k-1}}{z_{k-2}^T r_{k-2}} \quad (\beta_1 = 0) \]

\[ p_k = z_{k-1} + \beta_k p_{k-1} \]

\[ \alpha_k = \frac{z_{k-1}^T r_{k-1}}{p_k^T A p_k} \]

\[ x_k = x_{k-1} + \alpha_k p_k \]

\[ r_k = r_{k-1} - \alpha_k A p_k. \]

By preconditioning in this way it is possible to work with only the original matrix, and the sequence of updates is directly for \( x \) (and not \( \bar{x} \)). The algorithm is still applied to a symmetric positive definite system so all the properties of the original scheme hold, but with an improved convergence rate. In the case where \( M = I \) (the identity matrix) the above algorithm reduces to that of the original scheme (3.77).

The algorithm requires one solution of a linear system plus \( 5n + 2 \) operations and one matrix-vector product (with a maximum of \( n^2 \) operations) per iteration. Storage is required for the vectors \( x, r, p, Ap \) and \( z \) (although \( z \) can share the storage with \( Ap \)). Routines may be written so the it is not necessary to store \( A \) or \( M \) in full matrix format.

It is not at all obvious which choice of the preconditioning matrix \( C \) would give the optimum performance. The matrix \( C \) is selected so that \( \kappa(A) < \kappa(A) \), and any redistribution of the eigenvalues so as to cluster them would also have favourable effects. However, the matrix \( C \) is never required in the algorithm as it only appears in the form \( C^2 = M \). As explained above, §3.3.1, preconditioning is equivalent to accelerating an iterative method formed by splitting the original coefficient matrix as \( A = M - N \). The matrix \( M \) of the splitting and the matrix \( M \) that appears in the algorithm are in fact the same matrices. The question of how best to choose \( C \) is thus replaced by the problem of choosing \( M \) in the splitting \( A = M - N \).

In one sense it is possible to view the above algorithm as a direct method (applied
to a modified system $Mz = r$) imbedded within an iterative scheme. Different choices for $M$ will alter the computational effort involved for the direct method. There are two opposing considerations governing the choice of the preconditioner $M$.

- One philosophy is that $M$ should be as close to the identity matrix as possible, ensuring the solution of $Mz = r$ requires minimal extra work.

- Alternatively, since the rate of convergence is related to the condition number of $M^{-1}A$ (from the underlying iterative scheme), the ideal choice for $M$ would be $A$ itself. This choice would reduce the Conjugate Gradient algorithm to that of the method chosen to solve $Mz = r$, since at the first step $r = b$ so the solution $x^*$ would be achieved immediately (but at the cost of solving the full system $Az = b$). Thus $M$ should instead be selected to be as close, in some sense, to $A$ as possible. From the splitting $A = M - N$,

$$
\bar{A} = I + C^{-1}NC^{-1},
$$

so that the number of distinct eigenvalues of $\bar{A}$ is equal to the number for $N$. For the number of iterations to be as few as possible, $N$ should be like the null matrix and thus $M$ should reflect $A$.

Obviously the optimum choice involves some compromise. A short discussion detailing some of the common choices appears in the following section.

### 3.3.3 Common preconditioners

A good preconditioning matrix $M$ is one that achieves a favourable compromise between the reduction in number of iterations needed and the increase in the cost per iteration. A successful choice is somewhat problem dependent since the method of Conjugate Gradients is able to make effective use of symmetries and other structural properties inherent
Chapter 3. The Method of Conjugate Gradients

in the system. Many preconditioners have been proposed (see for example [25],[11], [12], [32],[44]). A comparison of the various choices is beyond the scope of this thesis, but for completeness some are discussed below.

Since Preconditioned Conjugate Gradients has its basis in a splitting $A = M - N$ this can provide some guidance on devising rapidly converging schemes. Any of the classical methods can be used.

- **Jacobi method**

  Diagonal preconditioning is the simplest iterative method. $M$ is taken as the diagonal of $A$ so the solution of $Mz = r$ at each iteration is just a simple division process. It requires no extra storage and an increase of $n$ operations per iteration. This type of preconditioner is equivalent to accelerating the Jacobi iterative scheme by the method of Conjugate Gradients.

- **SSOR iteration**

  The method of SSOR can be accelerated by Conjugate Gradients in a similar manner. With $A = D + L + U$ the preconditioner becomes $M = (D + \omega L)D^{-1}(D + \omega L)$. This needs no extra storage since the solution $Mz = r$ is performed directly using the required elements of $A$.

- **The method of ICCG**

  Decompositions of $A$ may not retain the sparsity structure in the factors and so there may be a radical increase in storage requirement. It is preferable to keep any banded structures that $A$ may possess. The Incomplete Cholesky factorization was proposed as a preconditioner by Meijerink and Van der Vorst [25]. The strategy is to perform the Cholesky decomposition of $A$ but to retain as zeros preassigned elements of $L$. The problem of fill-in is thus confined to some known area (e.g.,
within the bandwidth of $A$). The decomposition is then a splitting $A = LL^T - R$
so that the preconditioner is the product of simple triangular matrices $M = LL^T$. Various forms of the incomplete factorization arise from different choices of the elements required to be zero.

The preconditioning matrix can also be based on the polynomial expansion of the inverse of $A$. This technique is particularly applicable to parallel computers see [32] and [44].

In this preconditioned version Conjugate Gradients is one of the most practical iterative solvers. The method is also of historical interest. Although direct and iterative methods generally form two distinct classes of solution schemes, the method of Conjugate Gradients falls into both and emphasizes the link between the two. It was conceived as a direct scheme but was soon rejected. Its properties, once re-evaluated, proved its worth as an iterative approach. The preconditioned version in one sense combines direct and iterative schemes in one algorithm.

### 3.4 Conjugate Gradients for Linear Least Squares

The derivation of the algorithm, and the discussion of its properties, is valid only for a symmetric positive definite coefficient matrix. A modification to the original algorithm to deal with the more general case is described here.

§3.4.1 presents an algorithm to solve the Linear Least Squares problem while §3.4.2 discusses the preconditioned version.

#### 3.4.1 Linear Least Square Algorithm

For any matrix $A$ the product $A^TA$ is symmetric positive definite (This is valid since $(A^TAv, v) \neq 0$ and $(A^TAv, v) = (Av, Av) \geq 0$ so that $(A^TAv, v) > 0$). Hence in the
general case where \( A \) is not symmetric positive definite (or is overdetermined) the method of Conjugate Gradients can be applied (with \( A^T A \) as the coefficient matrix) to the Normal equations \( A^T A x = A^T b \) and be used to solve the Linear Least Squares problem.

In the case of the Normal equations the error functional (3.79) is replaced by

\[
E(x^* - x_k) = (x^* - x_k)^T A^T A (x^* - x_k) = r_k^T r_k
\]

so that the algorithm ensures the monotonic reduction in length of the residual rather than the error vector.

When working with the Normal Equations, formation of the product \( A^T A \) is not advised as it introduces roundoff errors, and increases the ill-conditioning of the system, since \( \kappa(A^T A) = \kappa(A)^2 \). The resulting, more efficient algorithm, while algebraically equivalent to applying the procedure directly to the normal equations, (with the coefficient matrix \( A^T A \) and right-hand side \( A^T b \)) is computationally superior.

**Conjugate Gradients Least Squares Algorithm**

Set \( x_0 = 0 \) and \( r_0 = b \) then for \( k = 1, 2, 3, \ldots \) while \( \|r_k\| \leq TOL \)

\[
\beta_k = \frac{r_k^T A^T r_{k-1}}{r_{k-2}^T A^T r_{k-2}} \quad (\beta_1 = 0)
\]

\[
p_k = A^T r_{k-1} + \beta_k p_{k-1}
\]

\[
\alpha_k = \frac{r_{k-1}^T A^T r_{k-1}}{p_k^T A^T A p_k}
\]

\[
x_k = x_{k-1} + \alpha_k p_k
\]

\[
r_k = r_{k-1} - \alpha_k A p_k.
\]

One additional matrix-vector product is introduced, (so that solution requires \( 5n + 2 \) operations, and two matrix-vector products per iteration) but the scheme updates \( x \) directly and retains the advantages of the basic algorithm. It is possible for the vector
Chapter 3. The Method of Conjugate Gradients

$A^{T}r$ to share storage with $Ap$, so that space is still required only for the 4 vectors $x, r, p, Ap$.

The use of the method of Conjugate Gradients for finding the solution of minimum norm is studied in [7] and [33].

3.4.2 The Preconditioned Algorithm for Least Squares

The idea of preconditioning can also be applied to the least squares problem.

**Preconditioned Conjugate Gradients Least Squares**

Set $x_0 = 0$ and $r_0 = b$ then for $k = 1, 2, 3, \ldots$ while $\|r_k\| \leq TOL$

\[
\begin{align*}
\text{Solve } M s_{k-1} &= A^{T} r_{k-1} \\
\beta_k &= \frac{s_{k-1}^{T} A A^{T} s_{k-1}}{s_{k-2}^{T} A A^{T} s_{k-2}} \quad \text{($\beta_1 = 0$)} \\
p_k &= s_{k-1} + \beta_k p_{k-1} \\
\text{Solve } M q_k &= p_k \\
\alpha_k &= \frac{s_{k-1}^{T} A A^{T} s_{k-1}}{q_k^{T} A A^{T} q_k} \\
x_k &= x_{k-1} + \alpha_k M^{-1} p_k \\
r_k &= r_{k-1} - \alpha_k A q_k.
\end{align*}
\]

Notice the algorithm requires the solution of two linear systems per iteration. For this the question of balancing the increased cost against the reduction in number of iterations to produce a decrease in the overall cost is particularly critical.

Comparisons of some preconditioners for the least squares problem are given in [33] and [7].
Chapter 4

The Filtering Properties of the Method of Conjugate Gradients

The solution of ill-conditioned systems, that are large and sparse, by the technique of Tikhonov Regularization in §2.3.1 is prohibitive; direct methods are costly and iterative methods are ineffective due to the small parameter. Iterative methods can be used efficiently by achieving the regularization via an iterative filtering technique, §2.3.2. This chapter examines the use of the method of Conjugate Gradients as the iterative method to be applied.

The first section describes the properties of a “favourable” filtering method, extending the ideas from §2.3.2. §4.2 describes the filtering function associated with the method of Conjugate Gradients using the relations from the algorithm (3.77). The similarity of the polynomial filtering function to a step function, is investigated in §4.3. The polynomial at each iteration is characterized as the solution to a weighted least squares problem and is shown to approximate the Lagrange interpolant through the singular values. A detailed examination of the filtering effects is given for the polynomial formed at the second iteration; the theory is extended qualitatively for later iterations and numerical evidence is presented, using four integral equations as test cases. In §4.4 the polynomial, at each iteration, is viewed as the expansion of an approximation to the inverse of $A$; this gives an alternative approach to the explanation of the filtering properties. The practical implementation is discussed in §4.5 and numerical tests show that the method of Conjugate Gradients gives comparable results to the method of truncating the SVD (2.31).
4.1 Iterative Filtering

In Chapter 2 it was shown that the objective when computing the solution to a system of equations that are ill-conditioned is to find a smooth approximation that is close to the exact solution. In many practical situations the system is very large and often sparse. In these cases iterative solution methods are preferred. The approximation may be achieved by the technique of iterative filtering as in §2.3.2. In Chapter 2 some of the common iterative methods that have been used as iterative filters were mentioned. For an effective filtering method the convergence of the components of the solution corresponding to the "larger" singular values should be rapid as these form the smooth solution. As the contributions from the "later" oscillatory singular vectors converge the solution becomes unstable. It is valuable for the rate of convergence of these vectors to be slower, so that they can be damped. A favourable iterative method will give a series of estimates that, when expressed in terms of the singular vectors of the system

\[ x_k = \sum_{i=1}^{n} F(\sigma_i; k) \frac{u_i^T b}{\sigma_i} v_i, \] (4.92)

the function, \( F(\sigma; k) \) filters the contribution of the singular vectors by damping those corresponding to the small singular values. Qualitatively \( F(\sigma; k) \) should have the property that, for the ordered singular values \( \sigma_1 > \ldots > \sigma_{p(k)} > \ldots \sigma_q > \ldots > \sigma_n \)

\[ F(\sigma; k) = \begin{cases} 
1 & \text{if } p(k) \geq i \geq 1 \quad \text{(i.e., for } \sigma_i \text{ "large")} \\
0 & \text{if } n \geq i > q \quad \text{(i.e., for } \sigma_i \text{ "small")} \\
s(\sigma) & \text{if } q \geq i > p(k) \quad \text{(i.e., otherwise)}
\end{cases} \]

where \( s(\sigma) \) is some non-decreasing function that has values between one and zero, i.e.,

\[ s'(\cdot) \leq 0, \quad 0 < s(\sigma) < 1. \] (4.93)
The function $\mathcal{F}(\sigma; k)$, is known as the filtering function associated with the chosen iterative method. As the iteration process proceeds (so the value of $k$ increases) the value of $p$ and $q$ should increase monotonically so that the components of each of the singular vectors converge in order, (i.e., those corresponding to the larger singular values first).

The success of a particular iterative method thus depends on finding a set of polynomials $\mathcal{F}(\sigma; k)$ that give good approximations to the Heaviside step function. The filtering properties of the method of Conjugate Gradients are examined in the next section, by considering the polynomials inherent in the iterative process.

### 4.2 The Filtering Function

In order to examine the feasibility of the method of Conjugate Gradients as an iterative filter it is necessary to express the iterates in terms of the singular vectors as in (4.92). From the algorithm (3.77) the method of Conjugate Gradients gives a series of estimates $x_k$ to the solution of (2.21), formed as

\[
\begin{align*}
x_1 &= c_1^{(1)} b \\
x_2 &= c_2^{(2)} Ab + c_1^{(2)} b \\
x_3 &= c_3^{(3)} A^2 b + c_2^{(3)} Ab + c_1^{(3)} b \\
&\vdots
\end{align*}
\]

(4.94)

where the $c_i^{(j)}$ are constants. This means that the iterates are expressible in terms of a polynomial as,

\[
x_k = P_k(A)b
\]

(4.95)

where $P_m(A)$ is a polynomial of degree $(m - 1)$. Each iterate can then be expressed as an expansion in terms of the singular value decomposition of the coefficient matrix. This
is

\[ x_k = \sum_{i=1}^{n} \sigma_i \mathcal{P}_k(\sigma_i) \frac{u_i^T b}{\sigma_i} v_i \]  

(4.96)

so that with \( G_k(\sigma) = \sigma \mathcal{P}_k(\sigma) \)

\[ x_k = \sum_{i=1}^{n} G_k(\sigma_i) \frac{u_i^T b}{\sigma_i} v_i. \]  

(4.97)

The polynomial \( G_k(\sigma) \) is the filtering function at the \( k \)th iteration associated with the method of Conjugate Gradients. \( (G_m(\sigma) \) is a polynomial of degree \( m \).) It is the properties of these polynomials that are considered in order to assess the viability of the use of the method of Conjugate Gradients with ill-conditioned linear systems of equations.

The algorithm for the method of Conjugate Gradients, (3.77), is based on the recurrence relations for the search directions, the iterates, and the residuals,

\[ p_k = r_{k-1} + \beta_k p_{k-1} \]
\[ x_k = x_{k-1} + \alpha_k p_k \]
\[ r_k = r_{k-1} - \alpha_k A p_k \]  

(4.98)

It is therefore also possible to express the search directions in terms of a polynomial;

\[ p_k = \mathcal{D}_k(A)b \]  

(4.99)

From the relations, (4.98), the two polynomials \( \mathcal{P}_k \) and \( \mathcal{D}_k \) are formed by the following recurrence formulae

\[ \mathcal{D}_k(\sigma) = (1 - \sigma \mathcal{P}_{k-1}(\sigma)) + \beta_k \mathcal{D}_{k-1}(\sigma) \]  

(4.100)

\[ \mathcal{P}_k(\sigma) = \mathcal{P}_{k-1}(\sigma) + \alpha_k \mathcal{D}_k(\sigma) \]  

(4.101)

where \( \mathcal{D}_0 = 1 \) and \( \mathcal{P}_0 = 0 \). By rearranging (4.101) and substituting the result into (4.100), together the relations give the three term recurrence relation for the polynomial.
filtering function, $G_k(\sigma)$

$$G_{k+1}(\sigma) = \alpha_{k+1}\sigma(1 - \alpha_{k+1}G_k(\sigma)) + (1 + \frac{\alpha_{k+1}\beta_{k+1}}{\alpha_k})G_k(\sigma)$$

$$- \frac{\alpha_{k+1}\beta_{k+1}}{\alpha_k}G_{k-1}(\sigma) \quad (4.102)$$

with $G_0 = 0$ and $G_1 = \alpha_k\sigma$. The polynomials can thus be expanded in full to give expressions, in terms of the parameters in the algorithm, for the coefficients in (4.95). The first four polynomials are given as illustrations:

$$G_1(\sigma) = \alpha_1(\sigma) \quad (4.103)$$

$$G_2(\sigma) = -(\alpha_1\alpha_2)(\sigma^2) + (\alpha_1 + \alpha_2\beta_2 + \alpha_2)(\sigma) \quad (4.104)$$

$$G_3(\sigma) = (\alpha_1\alpha_2\alpha_3)(\sigma^3) - (\alpha_1\alpha_2 + \alpha_1\alpha_3\beta_3$$

$$+ \alpha_1\alpha_3 + \alpha_2\alpha_3\beta_2 + \alpha_2\alpha_3)(\sigma^2)$$

$$+ (\alpha_1 + \alpha_2\beta_2 + \alpha_2 + \alpha_3\beta_2\beta_3 + \alpha_3\beta_3 + \alpha_3)(\sigma) \quad (4.105)$$

$$G_4(\sigma) = -(\alpha_1\alpha_2\alpha_3\alpha_4)(\sigma^4) + (\alpha_1\alpha_2\alpha_3 + \alpha_1\alpha_2\alpha_4$$

$$+ \alpha_1\alpha_2\alpha_4 + \alpha_1\alpha_3\alpha_4\beta_3 + \alpha_1\alpha_3\alpha_4 + \alpha_2\alpha_3\alpha_4\beta_2 + \alpha_2\alpha_3\alpha_4)(\sigma^3)$$

$$-(\alpha_1\alpha_2 + \alpha_1\alpha_3\beta_3 + \alpha_1\alpha_3 + \alpha_1\alpha_4\beta_3\beta_4$$

$$+ \alpha_1\alpha_4\beta_4 + \alpha_1\alpha_4 + \alpha_2\alpha_3\beta_2 + \alpha_2\alpha_3$$

$$+ \alpha_2\alpha_4\beta_4 + \alpha_2\alpha_4\beta_2 + \alpha_2\alpha_4\beta_4 + \alpha_2\alpha_4$$

$$+ \alpha_3\alpha_4\beta_3\beta_3 + \alpha_3\alpha_4\beta_3 + \alpha_3\alpha_4)(\sigma^2)$$

$$+ (\alpha_1 + \alpha_2\beta_2 + \alpha_2 + \alpha_3\beta_2\beta_3 + \alpha_3\beta_3$$

$$+ \alpha_3 + \alpha_4\beta_2\beta_3\beta_4 + \alpha_4\beta_3\beta_4 + \alpha_4\beta_4 + \alpha_4)(\sigma) \quad (4.106)$$

Without loss of generality the theory in this chapter is developed in terms of the original algorithm (3.77). In the case of the solution to the linear least squares problem
the polynomial, $P_k(\cdot)$ is in terms of the coefficient matrix $A^T A$. The coefficients $c_{i}^{(j)}$ in the expressions (4.103)–(4.106), etc., have the same form so application to the least squares case is achieved simply by replacing "$\sigma$" by "$\sigma^2$" so that $x_k = P_k(\sigma^2)b$.

From the algorithm (3.77), the parameters $\alpha_i$ and $\beta_i$ are all positive. This shows that each polynomial $G_k$ is composed of coefficients $c_i^{(k)}$ for $i = 1, \ldots, k$ from (4.95), that alternate in sign,

$$\text{sgn}(c_i^{(k)}) = \begin{cases} +1 & \text{"odd"} \\ -1 & \text{"even"} \end{cases}$$

and with $G_k(0) = 0$ for all $k$.

It is clear that

- $G_1(\sigma)$ is a straight line with positive gradient.
- $G_2(\sigma)$ is a quadratic that is concave down.

and that in general

- $G_k(\sigma)$ is a polynomial of degree $k$ that has positive gradient as it passes through the origin.

In general (and particularly in the case of an ill-conditioned system) little can be said about the magnitude of the coefficients in each polynomial, as this was discussed in the last chapter. In Tables 4.2 and 4.3 the values of the coefficients for the polynomials from two practical problems are given as illustration.

The two test problems are formed by the discretizations of two integral equations. The first example (which will be referred to as the "Trapezoidal" example) is generated from an example given in [17]. The trapezoidal rule, with a step length of $1/99$, was used to discretize the integral equation (1.1) where

$$K(s,t) = \sqrt{s^2 + t^2}$$
Chapter 4. The Filtering Properties of the Method of Conjugate Gradients

\[ G_1(x) = 1.5x \]
\[ G_2(x) = -163.0x^2 + 109.4x \]
\[ G_3(x) = 3,664,574.9x^3 - 2,458,562.6x^2 + 2,258.76x \]

Table 4.2: The first 3 polynomials from Conjugate Gradients for the Trapezoid example

\[ g(t) = 1/3((1 + t^2)^{3/2} - t^3) \]

for \(0 \leq s, t \leq 1\), which has the exact solution

\[ f(t) = t \quad (4.107) \]

Table 4.2 shows the values for the coefficients of the polynomials for the first 3 iterations for the Trapezoidal example. The second test case was generated using an example studied by Phillips in [28]. The trapezoidal rule was used, with a step length of \(2/33\) to discretize the integral equation given by

\[ K(s, t) = 1 + \cos\left(\frac{\pi(s - t)}{3}\right), \quad |s|, |t| \leq 3 \]
\[ g(t) = (6 - |t|)(1 + 1/2 \cos\left(\frac{\pi t}{3}\right)) - \frac{9\pi}{2} \sin\left(\frac{\pi |t|}{3}\right) \]

which gives a solution of

\[ f(t) = 1 + \cos\left(\frac{\pi t}{3}\right), \quad |t| \leq 3 \quad (4.108) \]

Table 4.3 shows the values of the coefficients of the polynomials from the first 5 iterations for the Phillips example. Both tables illustrate how the coefficients in the polynomials may rise rapidly in magnitude.

Numerical examination, by plotting the polynomials, demonstrates that when considered over all real numbers \(\mathbb{R}\), operator \(G_k(x)\) may be a wildly oscillating function. (Fig. 4.1 shows the actual shape of the early polynomials given in the Tables 4.2 and 4.3.)
Figure 4.1: Actual polynomials for iterations 2 and 3 for the Trapezoidal test case and for iterations 4 and 5 for the Phillips' test case.
Table 4.3: The polynomials from the first 4 iterations for the Phillips' example

\[
\begin{align*}
G_1(x) &= 0.03x \\
G_2(x) &= -0.0016x^2 + 0.08x \\
G_3(x) &= 0.0002x^3 - 0.014x^2 + 0.2x \\
G_4(x) &= -0.0003x^4 + 0.018x^3 - 0.31x^2 + 1.5x \\
G_5(x) &= 0.02x^5 - 1.2x^4 + 21.5x^3 - 101.6x^2 + 69.8x
\end{align*}
\]

However, the values of $G_k(x)$ in (4.95) are required only at the singular values of the matrix $A$; the algorithm uses these implicitly. It should be noted that explicit formation of the polynomial is never required. The oscillating behaviour is interesting however since it implies that as roundoff error accumulates, resulting in the use of values close to the true singular values, the effects may be unpredictable. This suggests the importance of performing only a small number of iterations of the method.

It is possible to give a clearer idea of the evaluation of the polynomial at the singular values by examining the operator in more detail. In §4.3.1 the polynomial is shown to be characterized as the solution to a weighted least squares problem. The theory is further developed by a detailed algebraic examination of $G_2(\sigma)$ in §4.3.2. In §4.3.3 the observations are generalized for the later polynomials and numerical illustrations are presented.

4.3 Characterization of the Operator

This section shows that the method of Conjugate Gradients has filtering properties that give results comparable with the method of truncating the SVD.
4.3.1 Precise: Minimization of the Deviation from Zero

The polynomial operator \( \mathcal{P}_k(\sigma) \) is precisely the polynomial of degree \( k \) that minimizes the \( A \)-norm of the error §3.2.4 since

\[
\min_{x_k} \| x - x_k \|_A = \min_{\mathcal{P}_k} \| x - \mathcal{P}_k(A)b \|_A = \min_{\mathcal{P}_k} ((x - \mathcal{P}_k(A)b)^T A(x - \mathcal{P}_k(A)b)) = \min_{\mathcal{P}_k} b^T(\mathcal{P}_k(A)A\mathcal{P}_k(A) - 2\mathcal{P}_k(A))b \]

\[
= \min_{\mathcal{P}_k} (\sum_{i=1}^n \gamma_i u_i^T)(\sum_{j=1}^n u_j(\sigma_j \mathcal{P}_k(\sigma_j)^2 - 2\mathcal{P}_k(\sigma_j))u_j^T)(\sum_{m=1}^n \gamma_m u_m)
\]

where \( u_i \) are the singular vectors of the matrix \( A \) and \( \sigma_i \) the associated singular values and \( \gamma_i = u_i^T b \). Differentiating the last expression and using the orthogonal properties of \( u_i \) gives

\[
\sum_{i=1}^n (\sigma_i \mathcal{P}_k(\sigma_i) - 1)\mathcal{P}_k'(\sigma_i)|\gamma_i|^2 = 0. \tag{4.109}
\]

(4.109) is equivalent to the minimization

\[
\min_{\mathcal{P}_k} \sum_{i=1}^n (\sigma_i \mathcal{P}_k(\sigma_i) - 1)^2 \frac{\gamma_i^2}{\sigma_i} \tag{4.110}
\]

The minimization (4.110) is that of a discrete weighted least squares problem with the weights \( \{ \frac{\gamma_i^2}{\sigma_i} \} \). The polynomial,

\[
\mathcal{R}_k(x) = x\mathcal{P}_k(x) - 1 = \mathcal{G}_k(x) - 1, \tag{4.111}
\]

is therefore given exactly as the (continuous) polynomial that minimizes the deviation from zero at the (discrete set of) singular values. This is a unique polynomial and gives the exact characterization of the filtering polynomial \( \mathcal{G}_k(x) \); this is the basis on which the evaluation of the filtering properties is made. However, a precise evaluation of \( \mathcal{G}_k(\sigma_i) \) is required to understand the filtering effects; this is not obvious from the
minimization (4.110). The weights depend on the data of the given problem and therefore generalizations are very difficult.

It is possible to give a straightforward qualitative observation. From the theory in §2.2 the existence of a smooth solution requires that the weights \( \{ \frac{\sigma_i^2}{\sigma_i} \} \) decrease in magnitude as \( i \) increases. The least squares fit for the filtering polynomial, \( G_k(x) \), will therefore be such that the values corresponding to \( \sigma_i \) where \( i \) is small are more closely fitted to one than those for \( i \) large. This means that generally \( G_k(\sigma_i) \sim 1 \) for \( i \) small and since the polynomial is required to go through zero, \( G_k(\sigma_i) \sim 0 \) for \( i \) large. Thus qualitatively the polynomial \( G_k(\sigma) = \sigma P_k(\sigma) \) is an approximation to the Heaviside function

\[
H(x) = \begin{cases} 
1 & x > 0 \\
0 & x = 0 
\end{cases}
\]

by a polynomial of degree \( k \). This is observed by Vogel [42].

The similarity of the polynomial to the step function is necessary for \( G_k(x) \) to be an effective filter, however it is not sufficient. Progress toward a more precise evaluation of \( G_k(x) \) may be achieved by studying a "simpler" polynomial that somehow approximates the polynomial from (4.110). A more quantitative explanation is a non-trivial problem, due to the dependence of the weights on the data, and although theoretically it is possible to find \( G_k(\sigma_i) \) exactly via the minimization (4.110) this is not practical. The following section begins by showing that \( G_2(x) \) may be regarded as an approximation to an interpolating polynomial through the first two singular values. With this simplification it is possible to gain insight into the filtering properties.

### 4.3.2 Approximate: Interpolation

The precise algebraic representation of the first few constants \( \alpha_1, \alpha_2 \) and \( \beta_2 \) is fairly easy.

\[
\alpha_1 = \frac{b^T b}{b^T A b} \quad (4.112)
\]
\[
\alpha_2 = \frac{(b^T b)(b^T A^2 b) - (b^T A b)^2}{(b^T A^3 b)(b^T A b) - (b^T A^2 b)^2} b^T A b
\]
\[
\beta_2 = \frac{(b^T b)(b^T A^2 b) - (b^T A b)^2}{(b^T A b)^2}
\]

To simplify the algebra it is assumed that the linear system has a diagonal coefficient matrix. The above can then be simplified using

\[ b^T b = \sum_{i=1}^{n} b_i^2 = b \]
\[ b^T A b = \sum_{i=1}^{n} \sigma_i b_i^2 = s \]
\[ b^T A^2 b = \sum_{i=1}^{n} \sigma_i^2 b_i^2 = d \]
\[ b^T A^3 b = \sum_{i=1}^{n} \sigma_i^3 b_i^2 = t \]

The polynomial given at the second iteration (4.104) can therefore be expressed using the symbols in (4.115) as,

\[ G_2(\sigma) = \frac{bt - sd}{ts - d^2}\sigma - \left( \frac{bd - s^2}{ts - d^2} \right) \sigma^2 \]  

(4.116)

(4.116) may be expanded using the terms,

\[ (bt - sd) = (\sigma_1 + \sigma_2)(\sigma_1 - \sigma_2)^2 b_1^2 b_2^2 + (\sigma_1 + \sigma_3)(\sigma_1 - \sigma_3)^2 b_1^2 b_3^2 + \ldots \]  

(4.117)
\[ (bd - s^2) = (\sigma_1 - \sigma_2)^2 b_1^2 b_2^2 + (\sigma_1 - \sigma_3)^2 b_1^2 b_3^2 + \ldots \]  

(4.118)
\[ (ts - d^2) = \sigma_1 \sigma_2 (\sigma_1 - \sigma_2)^2 b_1^2 b_2^2 + \sigma_1 \sigma_3 (\sigma_1 - \sigma_3)^2 b_1^2 b_3^2 + \ldots \]  

(4.119)

where the ellipsis indicate that the terms continue to be included in the summation for all \( i, j \) such that \( j > i \). Substituting these into the polynomial (4.116) gives \( G_2(x) \) as

\[ G_2(x) = \frac{(\sigma_1 + \sigma_2)(\sigma_1 - \sigma_2)^2 b_1^2 b_2^2 + \ldots}{\sigma_1 \sigma_2 (\sigma_1 - \sigma_2)^2 b_1^2 b_2^2 + \ldots} x - \frac{(\sigma_1 - \sigma_2)^2 b_1^2 b_2^2 + \ldots}{\sigma_1 \sigma_2 (\sigma_1 - \sigma_2)^2 b_1^2 b_2^2 + \ldots} x^2 \]

(4.120)

With the assumption that \( \sigma_2 \) is distinct from \( \sigma_1 \) (4.120) may be rewritten as
Chapter 4. The Filtering Properties of the Method of Conjugate Gradients

\[ G_2(x) = \frac{\sigma_1 + \sigma_2 + \frac{(\sigma_1 + \sigma_3)(\sigma_1 - \sigma_3)^2b_1^2b_3^2 + \ldots}{(\sigma_1 - \sigma_2)^2b_1^2b_2^2 + \ldots}}{\sigma_1\sigma_2 + \frac{\sigma_1\sigma_3(\sigma_1 - \sigma_3)^2b_1^2b_2^2 + \ldots}{(\sigma_1 - \sigma_2)^2b_1^2b_2^2 + \ldots}} x + \frac{1 + \frac{(\sigma_1 - \sigma_3)^2b_1^2b_3^2 + \ldots}{(\sigma_1 - \sigma_2)^2b_1^2b_2^2 + \ldots}}{\sigma_1\sigma_2 + \frac{\sigma_1\sigma_3(\sigma_1 - \sigma_3)^2b_1^2b_2^2 + \ldots}{(\sigma_1 - \sigma_2)^2b_1^2b_2^2 + \ldots}} x^2. \]  

(4.121)

With the expression for \( G_2(x) \) in this form (4.121) the “shape” of this polynomial can be examined by considering the magnitude of the three quotient terms,

\[ \frac{(\sigma_1 + \sigma_3)(\sigma_1 - \sigma_3)^2b_1^2b_3^2 + \ldots}{(\sigma_1 - \sigma_2)^2b_1^2b_2^2 + \ldots} \]  

(4.122)

\[ \frac{\sigma_1\sigma_3(\sigma_1 - \sigma_3)^2b_1^2b_2^2 + \ldots}{(\sigma_1 - \sigma_2)^2b_1^2b_2^2 + \ldots} \]  

(4.123)

\[ \frac{(\sigma_1 - \sigma_3)^2b_1^2b_3^2 + \ldots}{(\sigma_1 - \sigma_2)^2b_1^2b_2^2 + \ldots} \]  

(4.124)

Recall that the system under consideration is diagonal so that the singular vectors are just the unit vectors, thus the quantities \( \gamma_i = u_i^Tb \) are equivalent to \( b_i \). In §2.2 the discrete Picard condition gives assumptions that are made concerning the relative values of the \( \gamma_i \) and the \( \sigma_i \) in order for a smooth solution to exist. It is therefore valid to assume that the \( b_i \) will decay to the noise level of the system at least as fast as the decay of the singular values \( \sigma_i \). The singular values will then continue to decay to zero.

In an “ideal” case, with little or no noise, the \( b_i \) will decay very rapidly. Thus the three quotient terms (4.122), (4.123), (4.124) will all be negligible. The polynomial then reduces to

\[ G_2(x) \sim \frac{\sigma_1 + \sigma_2}{\sigma_1\sigma_2} x - \frac{1}{\sigma_1\sigma_2} x^2 \]  

(4.125)
This gives clear insight into the behaviour of $G_k$ as the righthand side of the expression (4.125) is exactly the unique Lagrange interpolating polynomial (quadratic) through the points $(\sigma_1, 1), (\sigma_2, 1), (0, 0)$, which is denoted as $L_2[(\sigma_1, 1), (\sigma_2, 1), (0, 0)]$.

This is a representation of the continuous polynomial which obviously may have values very different from one at all continuous points between $\sigma_1$ and $\sigma_2$. This explains the behaviour shown in Fig. 4.1. However, the Lagrangian interpolant, when considered evaluated only at the singular values, is a good approximation to the step function,

$$H_k(\sigma) = \begin{cases} 1 & \sigma = \sigma_1, \sigma_2 \\ 0 & \sigma = \sigma_3 \ldots \sigma_n, \end{cases}$$

and thus the method of Conjugate Gradients shows favourable filtering properties.

The similarity of the polynomial $G_2(\sigma)$ to the Lagrange interpolant means that after two iterations, the method of Conjugate Gradients gives similar results to regularization in standard form (2.39), i.e., when the magnitude of the regularization parameter is small relative the first two singular values and large relative to the rest. This procedure is also comparable to truncating the SVD expansion with the first two singular vectors. From the point of view of computational effort both the truncated SVD and Regularization require $O(n^3)$ operations and are unsuitable for large sparse systems. This result (4.125) suggests the the method of Conjugate Gradients (which requires $O(n^2)$ operations per iteration) may be a viable alternative.

It is interesting to note, by considering again (4.121), that if the singular values are clustered in some way, e.g., if the first two singular values are very close, so that $\sigma_2 \sim \sigma_1$, then the expansion reduces instead to approximate the interpolating polynomial through $(\sigma_1, 1), (\sigma_3, 1)$ and $(0, 0)$.

Although this gives insight into the nature of the filtering properties of the method of Conjugate Gradients in general the polynomial, $G_2$, is only an approximation to the
Chapter 4. The Filtering Properties of the Method of Conjugate Gradients

$L_2[(\sigma_1, 1), (\sigma_2, 1), (0, 0)]$. The manner in which $G_2$ deviates from the Lagrangian interpolant will depend on the magnitude of the three quotient terms, (4.122)–(4.124). To be able to reliably predict how close $G_2$ will be represented by $L_2$ requires re-consideration of the difficulties associated with ill-posed problems.

A system with unreasonably high noise level will give significant magnitudes to (4.122)–(4.124) and therefore $G_2$ may deviate substantially from the interpolant. However this corresponds to a solution that is very difficult to compute, as it is highly unstable. This can be seen from consideration of the weighted least squares fit (4.110). Recall that the polynomial is formed so that the singular values are close to unity whilst requiring that the polynomial goes through the origin. In a system where the weights $\frac{\gamma_i}{\sigma_i}$ are high for $i$ small and negligible for $i$ large the fit will be accurate for the larger singular values and thus generate a smooth approximation. If all the weights are of a similar magnitude, or they are unreasonably high for the later singular vectors, (as is the case for "noisy" systems) the fit will be very poor as too much emphasis will be given to the unstable vectors. This demonstrates that the precise nature of $G_2$ is highly problem dependent. However, in cases where the exact solution is reasonably smooth the polynomial $G_2$ will be well represented by $L_2[(\sigma_1, 1), (\sigma_2, 1), (0, 0)]$.

4.3.3 Extension to Later Iterates

Numerically it is shown that the interpretation given in the last section holds more generally: i.e., at the third iteration, $G_3(x)$ is an approximation to the interpolating polynomial through the first three singular values,

$$G_3(x) \sim L_3[(\sigma_1, 1), (\sigma_2, 1), (\sigma_3, 1), (0, 0)].$$  (4.126)
Chapter 4. The Filtering Properties of the Method of Conjugate Gradients

In general, at the $k$th iteration $G_k$ is an approximation to the Interpolant through $k$ singular values,

$$G_k(x) \sim L_k[(\sigma_1, 1), \ldots, (\sigma_k, 1), (0, 0)] \quad (4.127)$$

In Fig. 4.2 the first few iterates are shown where the plots are evaluated at the singular values for another test problem. The close resemblance to the step function is clear. The test problem was generated so that the coefficient matrix was that of the Hilbert matrix with entries

$$(a)_{ij} = \frac{1}{i + j - 1} \quad \text{for} \quad 1 \leq i, j \leq 99 \quad (4.128)$$

This is a well known ill-conditioned matrix. The vector $b$ was generated so that the exact solution was given by $x_i = 1/i$; a random noise factor was then added.

Each successive iterative gives an approximation to the interpolating polynomial through all previous singular values plus one more. Once a singular vector is included in the solution it will remain so for all following iterations. As the noise level is approached the $G_k$ will deviate more from the $L_k$ and the weight for the next singular vector may be too small, or too large.

In Fig. 4.3 the polynomials are plotted at the singular values for the first few iterations in the Trapezoidal example detailed in §4.2. The deviation from the step function is evident and corresponds to the stage at which the noise level has been reached.

This analysis shows that the method of Conjugate Gradients continues to act like the method of truncating the SVD, but that there is also some damping effect as in the method of regularization in standard form. More accurately, the method of Conjugate Gradients gives results that are similar to performing the technique of regularization in its standard form a number of times with successively smaller values for the parameter; each iterate will correspond to using a new value for the parameter.
Figure 4.2: The polynomials from iterations 2, 3, 4 and 5 evaluated at the singular values for the Hilbert matrix
Figure 4.3: The polynomials from iterations 3, 5, 7 and 9 evaluated at the singular values for the Trapezoidal example.
4.4 Approximation to the Inverse

The theory in the last section may be reinforced by considering the polynomial $P_k(A)$ as an approximation to the inverse of the coefficient matrix.

The method of Conjugate Gradients solves the linear system $Ax = b$ so that $x = A^{-1}b$. If the process were to continue for the full $n$ iterations then $x^* = P_n(A)b$. So that in the absence of roundoff error the polynomial is an expansion of the inverse,

$$A^{-1} = P_n(A)$$

This means that each singular value satisfies the identity

$$\frac{1}{\sigma_i} x^* = P_n(\sigma_i) x^*$$

and therefore that at the solution $x^*$

$$\sigma_i P_n(\sigma_i) = 1.$$ (4.131)

(4.131) shows that the expansion of the inverse is exactly the interpolating polynomial forced through unity at each of the singular values $L_n[(\sigma_1, 1), \ldots, (\sigma_n, 1), (0, 0)]$. It is interesting to note that although the $\alpha_i$ and the $\beta_i$ that contribute to the formation of the polynomials depend on the right hand side vector $b$, this dependence is lost and the final polynomial is unique for each matrix.

Again qualitatively it can be said that at each stage of the algorithm the polynomial, $P_k(A)$, is some approximation to the inverse of $A$. This also follows from (4.110) that can be written equivalently as

$$\min_{P_k} \|AP_k(A) - I\|_2^2$$ (4.132)

The notion of $P_k(A)$ as the approximation of the inverse of $A$ can be emphasized by extending a point made by Hestenes in [21].
The algorithm (3.77) can be rewritten in a matrix form as

\[ x_0 = 0, r_0 = b \quad k = 1, 2, \ldots \]

\[ E_k = \frac{p_k p_k^T}{p_k^T A p_k} \]

\[ D_k = I - AE_k \]

\[ p_k = D_k^T r_{k-1} \]

\[ x_k = x_{k-1} + E_k r_{k-1} \]

\[ r_k = D_k r_{k-1} \quad (4.133) \]

where \( D_k \) and \( E_k \) are matrices. From the above the iterate \( x_k \) can be given as the matrix expansion,

\[ x_k = E_1 D_0 r_0 + E_2 D_1 D_0 r_0 + \ldots + E_k D_{k-1} \ldots D_2 D_0 r_0 \]

\[ = (E_1 + E_2 + \ldots + E_k) r_1 \]

\[ = B_k b \]

where the matrix \( B_k \) is formed as the sum of outer products \( \sum_{i=1}^{k} \frac{p_i p_i^T}{p_i^T A p_i} \), and is the matrix representation of \( P_k(A) \). Therefore recalling the conjugacy property of the direction vectors from the algorithm (3.77), the operator given by \( B_k A \) is equivalent to the identity for all previous directions, and maps all future ones to the null-vector, i.e.,

\[ B_k A p_j = p_j \quad j \leq k \quad (4.134) \]

\[ B_k A p_j = 0 \quad j > k. \quad (4.135) \]

This relates to the idea that once a singular vector is included in the estimate it remains included for all following iterations, and that one more is included at each iteration. The above conditions give that

\[ (B_k A - I)p_j = 0 \quad j \leq k \quad (4.136) \]
Chapter 4. The Filtering Properties of the Method of Conjugate Gradients

$B_k$ is thus an approximation to the inverse of $A$ on the $span\{p_1, \ldots, p_k\}$ which is equivalent to $span\{b, \ldots, A^{k-1}b\}$. This means that as the iterations progress the identity

$$\sigma_j \mathcal{P}_k(\sigma_j) = 1 \quad (4.137)$$

holds for successively more singular values. (4.137) may be interpreted as a polynomial that is interpolated through the singular values at unity.

4.5 Numerical Tests

The performance of the method of Conjugate Gradients was tested numerically using a number of test problems. The same test problems were solved using the method of truncating the SVD (2.31) and by Landweber's iteration (2.53). The results are discussed below.

In every case the method of Conjugate Gradients shows a significant improvement, in terms of work for a given accuracy, over the method of Landweber. The performance of the method is therefore compared to the method of truncating the SVD. For the tests using the method of truncating the SVD, the truncation was performed at the point where the quantity $\gamma_i$ had reached the noise level, thus using the criterion in [40] to select the regularization parameter. The test cases used are those used earlier in §4.2 and §4.3.3 and detailed again below for clarity. A fourth test case was generated by discretizing the integral equation

$$K(s,t) = e^{st}$$

$$g(t) = \frac{e^{t+1} - 1}{t + 1}$$

for $0 \leq s, t \leq 1$ which gives a solution of

$$f(t) = e^t \quad (4.138)$$
Table 4.4: Comparison of accuracy and work for the Hilbert example.

<table>
<thead>
<tr>
<th>method</th>
<th>error</th>
<th>itn.</th>
</tr>
</thead>
<tbody>
<tr>
<td>CG</td>
<td>1.977 x 10^{-5}</td>
<td>10</td>
</tr>
<tr>
<td>TSVD</td>
<td>1.700 x 10^{-5}</td>
<td>7</td>
</tr>
<tr>
<td>LAND</td>
<td>1.130 x 10^{-2}</td>
<td>200 (+)</td>
</tr>
</tbody>
</table>

where Simpson's rule was used with a step length of 1/99. This equation was used in [6].

The singular value spectrum for each of the test cases is given in Fig. 4.4; this shows the clustering in each case and thus gives an indication of the number of stable singular vectors that should be included in an appropriate approximation.

Table 4.4, which shows the results for the Hilbert case, with coefficient matrix

\[
(a)_{ij} = \frac{1}{i + j - 1}
\]  (4.139)

and solution \( x = 1/i \), the method of Conjugate Gradients compares well, with regard to accuracy, with the truncated SVD. The matrix has four large singular values with the others clustered close to zero. The progress toward the chosen estimate is equivalent to that of the truncated SVD for the first four iterations and then begins to slow. This is expected from the spectrum of the singular values. The computational effort required for this application of the method of Conjugate Gradients is substantially less than that required for the same accuracy to be achieved by the method of truncating the SVD, (approximately 1/10 the number of operations); this is significant when considering larger systems.

For the tests performed with the examples set up as the Trapezoidal, Table 4.5 with kernel

\[
K(s,t) = \sqrt{s^2 + t^2} \quad 0 \leq s, t \leq 1
\]  (4.140)

and solution \( f(t) = t \), and the Simpson's case (see (4.138)) in Table 4.6 the accuracy
Figure 4.4: The singular values for the test examples. From left to right and top to bottom they show the Hilbert, Trapezoidal, Phillips and Simpson test cases.
Chapter 4. The Filtering Properties of the Method of Conjugate Gradients

<table>
<thead>
<tr>
<th>method</th>
<th>error</th>
<th>itn.</th>
</tr>
</thead>
<tbody>
<tr>
<td>CG</td>
<td>$4.942 \times 10^{-1}$</td>
<td>5</td>
</tr>
<tr>
<td>TSVD</td>
<td>$4.684 \times 10^{-1}$</td>
<td>16</td>
</tr>
<tr>
<td>LAND</td>
<td>$5.176 \times 10^{-1}$</td>
<td>200(+)</td>
</tr>
</tbody>
</table>

Table 4.5: Comparison of accuracy and work for the Trapezoidal example.

<table>
<thead>
<tr>
<th>method</th>
<th>error</th>
<th>itn.</th>
</tr>
</thead>
<tbody>
<tr>
<td>CG</td>
<td>5.894</td>
<td>6</td>
</tr>
<tr>
<td>TSVD</td>
<td>5.892</td>
<td>6</td>
</tr>
<tr>
<td>LAND</td>
<td>5.901</td>
<td>200(+)</td>
</tr>
</tbody>
</table>

Table 4.6: Comparison of accuracy and work for the Simpson's example.

obtained is again favourable. In both cases the convergence is slow but the method of Conjugate Gradients is shown to be an efficient alternative to the method of truncating the SVD. The slow convergence is emphasized since for the Simpson's example another 10 iterations were required to gain the extra accuracy in the third decimal.

The Phillips' example with kernel

$$K(s, t) = 1 + \cos\left(\frac{s - t}{3}\right), \quad |s|, |t| \leq 3$$

(4.141)

and solution $f(t) = 1 + \cos\left(\frac{\pi t}{3}\right)$ is not as ill-conditioned as the other cases. Many of the components give no contribution to the solution; of the first 19 singular vectors only 9 have a value for $\gamma_i$ larger than zero. Table 4.7 shows that the accuracy of the method of Conjugate Gradients is again competitive with the method of truncating the SVD, and whilst the number of iterations is "larger" it is noted that this still corresponds to a much smaller amount of computational effort than the $O(n^3) \sim 10^6$ operations required by the truncated SVD.

In the test cases the solution is chosen with reference to the actual error which is
available as the exact solution to the integral equations. In Fig 4.5 the chosen solutions are shown for the test example of Simpson and Phillips'. The solution to the Phillips' example shows almost perfect resemblance to the exact solution. For the Simpsons example the oscillation is interesting. It is believed that this is accounted for by an extra singularity in the discretized system and that the approximation is accurate for the discretized system. (It should be noted that the error measure is used a comparison to the continuous solution.) The results from the truncated SVD are comparable and it is noted that if an averaging is performed over neighbouring points an accurate representation of $e^t$ (the continuous solution) is achieved.

When the exact error is not known the question of when to terminate the iterative process is non-trivial. In some cases the convergence may be very slow and the calculation of many iterations is not justified with regard to the work required. The residual does not give an accurate indication of convergence rate or of oscillations in the solution.

The number of iterations is certainly small and so one acceptable approach is simply to study consecutive iterates and terminate the iteration as the oscillatory components appear. In cases where a priori knowledge is available on the singular value spectrum or of the noise in the system a similar criterion for the method of truncating the SVD [40] may be successfully used as the number of iterations for the method of Conjugate Gradients. Generally this knowledge is not available and although all the analysis in this thesis has been achieved via the singular value decomposition, in practice all that is

Table 4.7: Comparison of accuracy and work for the Phillips' example.

<table>
<thead>
<tr>
<th>method</th>
<th>error</th>
<th>itn.</th>
</tr>
</thead>
<tbody>
<tr>
<td>CG</td>
<td>$1.789 \times 10^{-2}$</td>
<td>14</td>
</tr>
<tr>
<td>TSVD</td>
<td>$1.947 \times 10^{-2}$</td>
<td>9 (non-zero)</td>
</tr>
<tr>
<td>LAND</td>
<td>$1.920 \times 10^{-1}$</td>
<td>200 (+)</td>
</tr>
</tbody>
</table>
Figure 4.5: The chosen solution for the Simpson and the Phillips' test examples. The Simpson example is the one above.
available are the $\alpha$ and $\beta$ from the algorithm.

It is suggested that an unreasonable change in the magnitude of the parameters $\alpha$ or $\beta$ from one iteration to the next may be an indication that the process should be terminated. A significant alteration in the size of one or more of the parameters $\alpha_i$ or $\beta_i$ will be an indication that the $i$'th singular value shows a sharp reduction in magnitude from the previous one; this requires that the constants $u_i^Tb$ also reduce rapidly and may suggest that the noise level has been reached. The large value of the parameter will have the effect of placing unnecessary emphasis on the weighting of one of the singular vectors and the polynomial will deviate from the Lagrange interpolant. Tables 4.8–4.11 show results for the four test cases and demonstrate the observations above. The results show numerically that there is a correspondence in the sharp increase and the convenient termination point.

In Table 4.8 $\alpha_{11}$ shows an increase of $10^8$ from $\alpha_{10}$ and $\beta_{12}$ is an increase of $10^{12}$ from $\beta_{11}$ corresponding to the minimum error at iteration 11. There are also increases of $10^5$ and $10^9$ between $\alpha_{6,7}$ and $\beta_{5,6}$ that indicate a reasonably accurate solution. In Table 4.9 the error continues to improve but very slowly. $\alpha_9$ shows an increase of $10^8$ from $\alpha_8$ and there is an increase of $10^{10}$ between $\beta_{6,7}$ and may be used to indicate that sufficient accuracy has been achieved by iteration 8. (Other large increases, consider for example $\beta_{4,5}$ and $\beta_{14,15}$, correspond to relatively good improvements in the solution during the slow convergence.) In Table 4.10 this example again has an error that is monotonically decreasing. However $\alpha_8$ is $10^{12}$ times the size of $\alpha_7$ and corresponds to a convenient termination at iteration 8. In this example the magnitudes of the parameters vary dramatically, compare for example $\alpha_{4,5}, \beta_{5,6}, \beta_{8,9}$ and $\beta_{12,13}$, and is believed to be an indication of the prohibitively slow convergence; suggesting that additional work to improve the accuracy is not recommended. The Phillips' test case is less clear. Table 4.11 shows that no such sharp rise is apparent within the first 15 iterations. The largest
increase is however at the convenient termination point of iteration 15; \( \alpha_{15} \) is \( 10^3 \) times \( \alpha_{14} \). Note also that \( \beta_9 \) is an increase of \( 10^5 \) from \( \beta_8 \) and corresponds to a significant improvement in the solution.

The numerical evidence demonstrates the method of Conjugate Gradients may be used successfully as a solution method for ill-conditioned systems. The results support the theory that each iteration of Conjugate Gradients has a similar effect to increasing the number of singular vectors included in the estimate by one. This means that the results are comparable with those from the truncated form of the SVD and to regularization in its standard form. As with all ill-conditioned systems the performance of any solution method is somewhat problem dependent as it relies heavily on the relative decay of the quantities \( \gamma_i \) and the singular values \( \sigma_i \); this is also the case with the method of Conjugate Gradients.
Chapter 4. The Filtering Properties of the Method of Conjugate Gradients

Table 4.8: The parameters from the first 15 iterations by Conjugate Gradients for the Hilbert example. The "*" indicates points at which $\alpha$ or $\beta$ show significant increases in magnitude.
Chapter 4. The Filtering Properties of the Method of Conjugate Gradients

<table>
<thead>
<tr>
<th>Itn</th>
<th>Resid</th>
<th>$P_{\text{norm}}$</th>
<th>$\alpha$</th>
<th>$\beta$</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$1.622 \times 10^{-1}$</td>
<td>$3.622 \times 10^0$</td>
<td>$1.512 \times 10^0$</td>
<td>$0.000 \times 10^0$</td>
<td>$1.763 \times 10^0$</td>
</tr>
<tr>
<td>2</td>
<td>$8.778 \times 10^{-4}$</td>
<td>$1.562 \times 10^{-2}$</td>
<td>$1.079 \times 10^2$</td>
<td>$1.860 \times 10^{-5}$</td>
<td>$5.164 \times 10^{-1}$</td>
</tr>
<tr>
<td>3</td>
<td>$7.698 \times 10^{-5}$</td>
<td>$5.832 \times 10^{-6}$</td>
<td>$2.248 \times 10^4$</td>
<td>$1.394 \times 10^{-7}$</td>
<td>$4.993 \times 10^{-1}$</td>
</tr>
<tr>
<td>4</td>
<td>$7.680 \times 10^{-5}$</td>
<td>$8.495 \times 10^{-8}$</td>
<td>$3.845 \times 10^3$</td>
<td><em>$2.121 \times 10^{-4}$</em></td>
<td>$4.993 \times 10^{-1}$</td>
</tr>
<tr>
<td>5</td>
<td>$7.017 \times 10^{-6}$</td>
<td>$2.150 \times 10^{-4}$</td>
<td>$3.203 \times 10^2$</td>
<td>$2.531 \times 10^3$</td>
<td>$4.942 \times 10^{-1}$</td>
</tr>
<tr>
<td>6</td>
<td>$6.894 \times 10^{-6}$</td>
<td>$1.676 \times 10^{-9}$</td>
<td>$6.076 \times 10^5$</td>
<td><em>$1.538 \times 10^{-7}$</em></td>
<td>$4.942 \times 10^{-1}$</td>
</tr>
<tr>
<td>7</td>
<td>$2.840 \times 10^{-6}$</td>
<td>$9.051 \times 10^{-6}$</td>
<td>$2.603 \times 10^3$</td>
<td>$5.399 \times 10^3$</td>
<td>$4.933 \times 10^{-1}$</td>
</tr>
<tr>
<td>8</td>
<td>$2.840 \times 10^{-6}$</td>
<td>$4.108 \times 10^{-9}$</td>
<td><em>$1.524 \times 10^0$</em></td>
<td>$3.706 \times 10^{-4}$</td>
<td>$4.933 \times 10^{-1}$</td>
</tr>
<tr>
<td>9</td>
<td>$8.919 \times 10^{-7}$</td>
<td>$2.140 \times 10^{-10}$</td>
<td>$1.625 \times 10^8$</td>
<td>$7.961 \times 10^{-3}$</td>
<td>$4.917 \times 10^{-1}$</td>
</tr>
<tr>
<td>10</td>
<td>$8.918 \times 10^{-7}$</td>
<td>$2.370 \times 10^{-11}$</td>
<td>$3.361 \times 10^5$</td>
<td>$1.240 \times 10^{-2}$</td>
<td>$4.917 \times 10^{-1}$</td>
</tr>
<tr>
<td>11</td>
<td>$8.918 \times 10^{-7}$</td>
<td>$3.450 \times 10^{-9}$</td>
<td>$1.693 \times 10^6$</td>
<td>$1.450 \times 10^2$</td>
<td>$4.917 \times 10^{-1}$</td>
</tr>
<tr>
<td>12</td>
<td>$8.918 \times 10^{-7}$</td>
<td>$4.223 \times 10^{-10}$</td>
<td>$1.016 \times 10^3$</td>
<td>$1.191 \times 10^{-1}$</td>
<td>$4.917 \times 10^{-1}$</td>
</tr>
<tr>
<td>13</td>
<td>$3.624 \times 10^{-7}$</td>
<td>$3.612 \times 10^{-9}$</td>
<td>$8.126 \times 10^6$</td>
<td>$8.526 \times 10^0$</td>
<td>$4.904 \times 10^{-1}$</td>
</tr>
<tr>
<td>14</td>
<td>$3.624 \times 10^{-7}$</td>
<td>$3.931 \times 10^{-12}$</td>
<td>$4.497 \times 10^4$</td>
<td>$1.837 \times 10^{-4}$</td>
<td>$4.904 \times 10^{-1}$</td>
</tr>
<tr>
<td>15</td>
<td>$1.589 \times 10^{-7}$</td>
<td>$1.170 \times 10^{-7}$</td>
<td>$2.376 \times 10^5$</td>
<td>$2.975 \times 10^4$</td>
<td>$4.891 \times 10^{-1}$</td>
</tr>
</tbody>
</table>

Table 4.9: The parameters from the first 15 iterations of Conjugate Gradients for the Trapezoidal example. The "*" indicates points at which $\alpha$ or $\beta$ show significant increases in magnitude.
### Table 4.10: The parameters from the first 15 iterations of Conjugate Gradients for the Simpson's example. The "*" indicates points at which $\alpha$ or $\beta$ show significant increases in magnitude.

<table>
<thead>
<tr>
<th>Itn</th>
<th>Resid</th>
<th>$P_{\text{norm}}$</th>
<th>$\alpha$</th>
<th>$\beta$</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$2.101 \times 10^{-1}$</td>
<td>$3.429 \times 10^1$</td>
<td>$4.871 \times 10^{-1}$</td>
<td>$0.000 \times 10^0$</td>
<td>$6.182 \times 10^0$</td>
</tr>
<tr>
<td>2</td>
<td>$6.221 \times 10^{-4}$</td>
<td>$2.375 \times 10^{-2}$</td>
<td>$7.824 \times 10^1$</td>
<td>$4.797 \times 10^{-7}$</td>
<td>$5.896 \times 10^0$</td>
</tr>
<tr>
<td>3</td>
<td>$4.615 \times 10^{-6}$</td>
<td>$2.381 \times 10^{-6}$</td>
<td>$6.827 \times 10^4$</td>
<td>$1.005 \times 10^{-8}$</td>
<td>$5.894 \times 10^0$</td>
</tr>
<tr>
<td>4</td>
<td>$4.588 \times 10^{-6}$</td>
<td>$7.441 \times 10^{-7}$</td>
<td>$*4.871 \times 10^{-1}$</td>
<td>$8.965 \times 10^{-2}$</td>
<td>$5.894 \times 10^0$</td>
</tr>
<tr>
<td>5</td>
<td>$1.143 \times 10^{-6}$</td>
<td>$3.790 \times 10^{-10}$</td>
<td>$1.375 \times 10^8$</td>
<td>$*2.826 \times 10^{-7}$</td>
<td>$5.894 \times 10^0$</td>
</tr>
<tr>
<td>6</td>
<td>$1.684 \times 10^{-7}$</td>
<td>$4.048 \times 10^{-5}$</td>
<td>$8.330 \times 10^1$</td>
<td>$1.068 \times 10^5$</td>
<td>$5.894 \times 10^0$</td>
</tr>
<tr>
<td>7</td>
<td>$1.684 \times 10^{-7}$</td>
<td>$2.535 \times 10^{-11}$</td>
<td>$*4.872 \times 10^{-1}$</td>
<td>$4.170 \times 10^{-6}$</td>
<td>$5.894 \times 10^0$</td>
</tr>
<tr>
<td>8</td>
<td>$8.363 \times 10^{-8}$</td>
<td>$2.223 \times 10^{-13}$</td>
<td>$4.323 \times 10^{11}$</td>
<td>$*7.725 \times 10^{-5}$</td>
<td>$5.892 \times 10^0$</td>
</tr>
<tr>
<td>9</td>
<td>$8.254 \times 10^{-8}$</td>
<td>$3.585 \times 10^{-7}$</td>
<td>$2.276 \times 10^3$</td>
<td>$1.613 \times 10^6$</td>
<td>$5.892 \times 10^0$</td>
</tr>
<tr>
<td>10</td>
<td>$8.246 \times 10^{-8}$</td>
<td>$1.060 \times 10^{-4}$</td>
<td>$5.732 \times 10^{-1}$</td>
<td>$2.956 \times 10^2$</td>
<td>$5.892 \times 10^0$</td>
</tr>
<tr>
<td>11</td>
<td>$1.266 \times 10^{-9}$</td>
<td>$1.155 \times 10^{-5}$</td>
<td>$2.647 \times 10^8$</td>
<td>$1.090 \times 10^{-1}$</td>
<td>$5.892 \times 10^0$</td>
</tr>
<tr>
<td>12</td>
<td>$1.266 \times 10^{-9}$</td>
<td>$2.160 \times 10^{-17}$</td>
<td>$1.304 \times 10^1$</td>
<td>$*1.817 \times 10^{-16}$</td>
<td>$5.892 \times 10^0$</td>
</tr>
<tr>
<td>13</td>
<td>$1.266 \times 10^{-9}$</td>
<td>$5.673 \times 10^{-16}$</td>
<td>$5.627 \times 10^8$</td>
<td>$2.577 \times 10^1$</td>
<td>$5.892 \times 10^0$</td>
</tr>
<tr>
<td>14</td>
<td>$1.266 \times 10^{-9}$</td>
<td>$4.080 \times 10^{-12}$</td>
<td>$3.604 \times 10^8$</td>
<td>$7.193 \times 10^3$</td>
<td>$5.892 \times 10^0$</td>
</tr>
<tr>
<td>15</td>
<td>$1.266 \times 10^{-9}$</td>
<td>$2.286 \times 10^{-10}$</td>
<td>$1.500 \times 10^9$</td>
<td>$5.601 \times 10^1$</td>
<td>$5.892 \times 10^0$</td>
</tr>
<tr>
<td>Itn</td>
<td>Resid</td>
<td>Pnorm</td>
<td>$\alpha$</td>
<td>$\beta$</td>
<td>Error</td>
</tr>
<tr>
<td>-----</td>
<td>----------</td>
<td>----------</td>
<td>------------</td>
<td>-----------</td>
<td>----------</td>
</tr>
<tr>
<td>1</td>
<td>$1.015 \times 10^1$</td>
<td>$2.377 \times 10^2$</td>
<td>$3.197 \times 10^{-2}$</td>
<td>$0.000 \times 10^0$</td>
<td>$3.026 \times 10^0$</td>
</tr>
<tr>
<td>2</td>
<td>$3.791 \times 10^0$</td>
<td>$4.234 \times 10^1$</td>
<td>$5.095 \times 10^{-2}$</td>
<td>$3.079 \times 10^{-2}$</td>
<td>$1.729 \times 10^0$</td>
</tr>
<tr>
<td>3</td>
<td>$6.926 \times 10^{-1}$</td>
<td>$9.891 \times 10^0$</td>
<td>$1.498 \times 10^{-1}$</td>
<td>$5.331 \times 10^{-2}$</td>
<td>$8.011 \times 10^{-1}$</td>
</tr>
<tr>
<td>4</td>
<td>$2.443 \times 10^{-2}$</td>
<td>$6.243 \times 10^{-1}$</td>
<td>$1.235 \times 10^0$</td>
<td>$4.185 \times 10^{-3}$</td>
<td>$2.149 \times 10^{-1}$</td>
</tr>
<tr>
<td>5</td>
<td>$3.065 \times 10^{-3}$</td>
<td>$2.932 \times 10^{-3}$</td>
<td>$6.833 \times 10^1$</td>
<td>$2.215 \times 10^{-5}$</td>
<td>$6.895 \times 10^{-2}$</td>
</tr>
<tr>
<td>6</td>
<td>$2.010 \times 10^{-3}$</td>
<td>$2.425 \times 10^{-4}$</td>
<td>$9.159 \times 10^1$</td>
<td>$6.795 \times 10^{-3}$</td>
<td>$5.929 \times 10^{-2}$</td>
</tr>
<tr>
<td>7</td>
<td>$2.009 \times 10^{-3}$</td>
<td>$6.862 \times 10^{-4}$</td>
<td>$3.120 \times 10^{-2}$</td>
<td>$2.377 \times 10^0$</td>
<td>$5.928 \times 10^{-2}$</td>
</tr>
<tr>
<td>8</td>
<td>$1.846 \times 10^{-3}$</td>
<td>$9.125 \times 10^{-5}$</td>
<td>$8.874 \times 10^1$</td>
<td>$6.311 \times 10^{-2}$</td>
<td>$5.234 \times 10^{-2}$</td>
</tr>
<tr>
<td>9</td>
<td>$6.451 \times 10^{-4}$</td>
<td>$1.388 \times 10^{-1}$</td>
<td>$2.773 \times 10^{-1}$</td>
<td>$1.520 \times 10^3$</td>
<td>$2.907 \times 10^{-2}$</td>
</tr>
<tr>
<td>10</td>
<td>$4.158 \times 10^{-4}$</td>
<td>$7.698 \times 10^{-2}$</td>
<td>$4.066 \times 10^{-2}$</td>
<td>$5.544 \times 10^{-1}$</td>
<td>$2.866 \times 10^{-2}$</td>
</tr>
<tr>
<td>11</td>
<td>$4.157 \times 10^{-4}$</td>
<td>$1.523 \times 10^{-5}$</td>
<td>$2.128 \times 10^{-1}$</td>
<td>$3.739 \times 10^{-5}$</td>
<td>$2.865 \times 10^{-2}$</td>
</tr>
<tr>
<td>12</td>
<td>$4.156 \times 10^{-4}$</td>
<td>$1.032 \times 10^{-5}$</td>
<td>$1.984 \times 10^0$</td>
<td>$3.497 \times 10^{-1}$</td>
<td>$2.864 \times 10^{-2}$</td>
</tr>
<tr>
<td>13</td>
<td>$3.156 \times 10^{-4}$</td>
<td>$2.369 \times 10^{-4}$</td>
<td>$4.138 \times 10^1$</td>
<td>$2.258 \times 10^1$</td>
<td>$2.149 \times 10^{-2}$</td>
</tr>
<tr>
<td>14</td>
<td>$2.131 \times 10^{-4}$</td>
<td>$1.077 \times 10^{-2}$</td>
<td>$6.751 \times 10^{-1}$</td>
<td>$4.546 \times 10^1$</td>
<td>$1.789 \times 10^{-2}$</td>
</tr>
<tr>
<td>15</td>
<td>$1.841 \times 10^{-4}$</td>
<td>$6.265 \times 10^{-6}$</td>
<td>$4.341 \times 10^2$</td>
<td>$3.308 \times 10^{-4}$</td>
<td>$1.640 \times 10^{-2}$</td>
</tr>
</tbody>
</table>

Table 4.11: The parameters from the first 15 iterations of Conjugate Gradients for the Phillips’ example. The "*" indicates points at which $\alpha$ or $\beta$ show significant increases in magnitude.
Chapter 5

Numerical Application

This chapter comprises a numerical example to demonstrate the practical use of the method of Conjugate Gradients to calculate a smooth approximation to a linear system of equations that is severely ill-conditioned. The problem is that of retrieving an in focus picture from a blurred image. The method of Conjugate Gradients is applied and a smooth approximate solution is computed with minimal work.

The problem is described in §5.1. §5.1.1 discusses the general mathematical interpretation of the problem and §5.1.2 details the problem used for the illustration. The results of a number of test examples are given in §5.2, where both the numerical and visual representation are discussed.

5.1 A Description of the Problem

A full description of the theory of image processing and the mathematical treatment of the problems is available in many texts, see for example [31] and the references therein. This section gives a brief introduction to the ideas of reconstructing an image from a degraded picture and details the particular numerical example used in this thesis.

5.1.1 The Mathematical Reconstruction of Degraded Pictures

The reconstruction of a sharp image from an out of focus picture has important practical use in photography. An image may be blurred due to movement, improper focussing through the lens or inappropriate lighting. The retrieval of the associated in focus image
is a difficult problem. Many original pictures that are quite different can produce a very similar result when blurred: the problem is therefore ill-posed. Reconstruction of the exact picture from a general image is not possible. A calculated estimate must however somehow smooth the error caused by the blurring.

The problem can be modeled mathematically. It is assumed that the original picture is a black and white photograph with a continuous gradation for the grey scale. The variation in the grey level, that results in the visualized image, can be represented as a two-dimensional function of position, \( f(s,t) \), which is defined over the finite region of the picture. The affect which causes the degradation of the image may be represented by an operating function \( K(x,y,s,t) \), known as a point-spread function.

The degraded image \( g(x,y) \) is given by the two-dimensional integral equation

\[
    g(x,y) = \int \int K(x,y,s,t)f(s,t)\,ds\,dt \tag{5.142}
\]

(this is the two dimensional equivalent of (2.7)). In many examples the point-spread function is separable, i.e., the effect in the two space dimensions is independent and the kernel is expressable as the product,

\[
    K(x,y,s,t) = K_1(x,s)K_2(y,t). \tag{5.143}
\]

The action of the blurring may depend on position, i.e., be spatially variant (e.g., with motion or lighting the effects may be much worse over one area of the image); these cases occur frequently and are severely ill-posed. One common example of a spatially varying point spread function that is separable is that of a Gaussian blurr, \[2\].

\[
    K(x,y,s,t) = \exp\{-p_x(x-s)^2 - p_y(y-t)^2\}. \tag{5.144}
\]

where the functions \( p_x \) and \( p_y \) vary linearly from the centre of the image. The more the variation in the functions \( p_x \) and \( p_y \) is, the more severe the degradation will be; this will result in an increase in the ill-conditioning of the problem.
In order to obtain numerical solutions to problems of the form (5.142) the equation may be first discretized. The image may be digitized into a two-dimensional array by dividing the area into a grid of equally sized picture elements, or pixels. Each pixel is assigned a numerical value that represents the average grey level over its area. The theory of digitization is very complex [31] but a straightforward approach is to divide the grey scale into a number of equal length intervals, $I_j$, and assign one numerical value (often the integer, $j$, is used) to represent the "colour" of each interval. The two dimensional array that forms the digitized picture, $F$, can be simply transformed into a vector by stacking each of the rows of the array end on end, (see §A.1), thus an original $n \times n$ image is converted into an $n^2$-vector.

The discretization of the point spread function is performed by some quadrature rule. In the case where the operator is separable as in (5.143) the discretization results in a system of the form

$$(K_1 \otimes K_2)\overline{f} = \overline{g}$$

(5.145)

where $A \otimes B$ is the Kronecker product of two matrices [23]. The matrices $K_1$ and $K_2$ are $n \times n$ arrays and the discrete images $\overline{f}$ and $\overline{g}$ are $n^2$ vectors. In the example of the Gaussian blurr (5.144) the entries in the matrices $K_1$ and $K_2$ are given by

$$(K_1)_{ij} = \exp\{-w_i(i-j)^2\}$$

(5.146)

$$(K_2)_{ij} = \exp\{-w_i(i-j)^2\}$$

(5.147)

where $w_i$ is the weighting function for the quadrature rule.

The discretization of problems from image reconstruction frequently give rise to systems (5.145) that are large and sparse. The point spread function generally acts only locally, that is the "blurring" of any particular point affects only the neighbouring pixels and so many of the elements of $K_1$ and $K_2$ are zero. The number of pixels used in any problem may be substantial since the finer the digitization is, the more accurately
the picture can be represented. The ill-posed nature of the reconstruction problem is reflected in the discretized system by the ill-conditioning of the resulting product of the matrices $K_1$ and $K_2$.

### 5.1.2 The Test Example

Using the properties summarized in the appendix §A.1 it is possible to transform the equation (5.145) into an $(n^2 \times n^2)$ linear system; this may be achieved in various ways. The example used in the tests is based on an example in [14]. The system is created using the singular value decomposition of $K_1$ as,

$$ K_1 = U_1 \Sigma_1 V_1^T $$

where $\Sigma_1 = diag(\sigma_{11}, \sigma_{12}, \ldots, \sigma_{1n})$. Then, using property (3) in §A.1, the coefficient matrix is expanded as

$$ (U_1 \Sigma_1 V_1^T) \otimes K_2 = (U_1 \otimes I)(\Sigma_1 \otimes K_2)(V_1^T \otimes I) $$

(5.149)

The problem of finding the solution of minimum norm is therefore reduced, by using property (4), to

$$ \min_{f_I} \| (\Sigma_1 \otimes K_2) f_I - g_I \|_2 $$

where

$$ f_I = (V_1^T \otimes I) \bar{f} $$

$$ g_I = (U_1^T \otimes I) \bar{g} $$

From the definition of a Kronecker product the coefficient matrix $K_I = (\Sigma_1 \otimes K_2)$ is the block diagonal array

$$ K_I = \begin{pmatrix} \sigma_{11} K_2 \\ \sigma_{12} K_2 \\ \vdots \\ \sigma_{1n} K_2 \end{pmatrix} $$
The vectors, $f_I$ and $g_I$, are just formed as the reduction of the matrices $F_I = V_1^T F$ and $G_I = U_1^T G$, (using property (2)).

The condition number of the coefficient matrix, $K_I$, formed above is the product of the condition numbers of $K_1$ and $K_2$. The matrices $K_1$ and $K_2$ are totally positive [23], and thus their Kronecker product is also. The singular vectors $v_i$ therefore have $(i - 1)$ sign changes and so an increase in the contributions from the singular vectors corresponding to even relatively large singular values may cause instability in the solution. The original $(n \times n)$ image has therefore resulted in an $(n^2 \times n^2)$ system that is sparse and extremely ill-conditioned.

The numerical example used for the tests is a modification of the Gaussian blurring function given in [14]. The elements of the matrices $K_1$ and $K_2$ are given as

\[
(K_1)_{ij} = \left( \frac{\pi}{c_1 \gamma_i} \right)^{1/2} \exp(-c_1 \gamma_i (i - j)^2) \quad (5.150)
\]

\[
(K_2)_{ij} = \left( \frac{\pi}{c_2 \gamma_i} \right)^{1/2} \exp(-c_2 \gamma_i (i - j)^2) \quad (5.151)
\]

with $\gamma_i = 2 - \frac{3.92}{n} \frac{n}{2} + 0.5 - i$. \quad (5.152)

The constants are set such that $c_1 = 1/2$ and $c_2 = 1/4$ (this corresponds to a moderately severe blurring effect, [2, 14]). The three test images are taken as simple $15 \times 15$ digitized arrays that corresponded to the sharp images shown in Fig.5.6. The digitization is set to have 9 grey levels so that values below zero are represented as white up to values above 8 which are represented as black, which allows a visual (if crude) representation of the solution to be given.

### 5.2 Discussion of Results

The system is very large and sparse. The use of the truncated form of the singular value decomposition (2.31) on this system would be extremely inefficient as it would require
Figure 5.6: The original sharp test images of the Cross, L, and Box. For the Cross and the L each test is a binary image composed of 8's and 0's; the Box has an additional middle grey level of 4. The Legend shows the grey scale used for the digitization.
$O(n^8)$ operations (where $n$ is the number of pixels in one row). With efficient routines for the multiplication of banded matrices the use of the method of Conjugate Gradients has a work requirement of $O(n^3)$ operations.

It is not possible to retrieve the exact images in Fig.5.6 from the degraded images, shown in Fig.5.7, due to the severe ill-posed quality of the problem. (Comparison of Fig.5.7 with Fig.5.6 demonstrates the extent of the retrieval problem.) The requirement for the solution to be smooth, as given by the discrete Picard Condition (2.27), can be interpreted visually. An approximate solution will be satisfactory provided it gives a reasonable distinction between the area containing the object and the background. Gradual fading from white to black over the boundaries and into the centre of the image will be acceptable. This corresponds to a smoothing of the effect of the defocussing.

The numerical representation of the images that form matrix $\overline{G}$ are given in Fig. 5.8, they demonstrate the severe degradation that has occurred.

In the test examples the exact error is available. The graphical representation of the exact error for each of the tests is shown in Fig.5.9. The results show that, in each case, there is a marked minimum of this error that corresponds to the best approximation to the exact image. This occurs very quickly; in the region of 4 or 5 iterations for a system of 225 equations! Examination of the singular values (see Fig.5.10) of the coefficient matrix indicates that there are four high values clustered away from the others. In terms of the discrete Picard Condition (2.27) the quantities $(u_i^Tg_i)$ must decrease very rapidly in order to be assured of a smooth solution. For most examples, therefore, the best smooth approximation will include contributions from only the first four singular vectors. This explains the rapid "convergence" (to a smooth solution) for the method of Conjugate Gradients. The value of the error seems quite substantial in each case. However it is acceptable since it accounts for the fading across the boundaries.

The numerical representation of the results of the iteration process is given in Fig.5.11.
Figure 5.7: The result of defocussing the test images of the Cross, L and Box. The legend shows the grey scale used for the digitization.
### Chapter 5. Numerical Application

| CROSS test image | 341 9 430 1 587 9 381 3 1225 8 1402 2 1317 8 899 0 557 7 354 5 388 5 418 3 428 3 387 6 338 1 | 900 0 213 2 209 2 284 1 824 7 758 1 729 0 530 8 258 3 113 5 97 2 108 0 111 1 103 1 87 7 | 116 5 148 0 213 8 338 8 491 3 572 8 541 5 402 9 217 2 125 7 125 6 141 5 145 5 135 1 114 9 | 258 0 310 2 304 4 428 5 486 2 493 0 423 0 143 8 352 6 256 9 736 2 277 7 316 4 755 8 302 2 257 0 | 465 0 562 4 615 5 621 9 578 7 499 3 418 9 387 3 360 3 419 7 512 7 586 1 803 2 559 9 476 2 | 530 0 623 4 672 3 655 1 575 4 467 4 360 2 374 4 375 6 481 2 570 3 652 4 671 4 623 2 530 0 | 454 0 535 0 570 8 561 4 492 2 398 1 324 2 298 7 321 8 395 7 499 5 559 8 578 3 534 9 454 9 | 314 7 371 0 404 3 404 1 360 8 313 6 261 1 231 5 233 4 275 7 238 8 397 0 398 5 380 8 311 4 | 110 2 133 7 162 3 203 2 244 4 257 9 235 6 183 9 124 1 104 2 118 0 135 0 139 8 129 8 105 9 | 14 9 23 7 53 0 116 8 205 1 256 5 247 8 178 9 80 3 25 6 16 2 17 4 17 9 18 8 14 1 | 2 3 10 4 45 3 123 6 237 6 305 1 297 0 212 1 88 5 17 7 2 1 1 1 1 1 1 1 | 1 5 11 6 56 0 183 0 301 8 388 0 377 9 289 7 111 8 21 3 1 0 0 0 0 0 0 0 0 | 1.5 15 5 75 7 230 7 408 9 525 7 512 0 385 4 151 5 28 8 2 3 0 1 0 1 0 0 0 0 0 | 2 7 21 5 104 2 303 3 561 9 722 3 703 8 502 0 208 2 39 6 3 0 0 0 0 0 0 | 20.4 55.8 201 2 547 1 994 6 1270 5 1235 3 833 7 373 3 82.6 22.8 20 0 18.9 16.1 |
| L test image | 31.1 18.8 21.1 20.6 13.3 3.9 0.2 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 | 112 0 87.4 76.0 74.2 47.9 14.1 0.8 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 | 272 0 163.8 184.7 180.2 118.3 34.3 1.9 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 | 466 5 281.0 316.7 309.1 199.5 58.9 3.2 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 | 586 4 356.2 404.9 385.1 255.0 75.3 4.1 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 | 813 0 389.2 418.1 408.1 282.1 77.5 4.3 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 | 71.0 343.8 387.5 378.2 244.5 73.1 5.1 1.1 1.2 1.4 1.7 1.8 1.0 0.8 2.0 | 555 3 332 8 375 1 368 6 240.0 78.4 13.8 8.7 9.9 11.7 13.8 12.8 8.3 6.3 15.2 | 603 3 358.3 401.5 395.4 270.7 115.8 50.7 41.3 47.5 55.8 65.1 60.1 38.8 30.1 72.2 | 717 0 411.7 483.7 482.7 332.6 203.0 137.2 117.2 133.6 158.6 184.9 170.6 112.9 89.5 204.9 | 818 0 457.3 514.7 520.3 413.1 301.1 239.8 208.1 237.1 278.6 325.3 300.0 198.5 150.4 360.2 | 801 3 441.3 496.8 505.5 418.6 332.3 277.3 240.8 277.0 325.7 380.0 350.5 231.9 175.7 420.7 | 640 2 356.0 397 4 402.9 338.9 257.5 233.3 202.6 233.3 274 3 320.0 295.2 195.3 148.0 354.3 | 407 2 222.9 304 0 356.9 283.0 233.8 169.9 175.8 205.3 195.7 127.7 118.7 118.7 118.7 118.7 | 202 1 110.6 124.4 127.1 108.3 87.7 74.5 64.6 74.5 87.6 102.2 94.3 82.4 47.3 113.2 |

**Figure 5.8:** The numerical representation of the defocussed test images of the Cross,L and Box. These matrices form the right hand side of the linear system and show the detail the is not available once they are digitized.
Figure 5.9: The measure of the exact error against the iteration number for the test images of the Cross, L and Box.
Figure 5.10: The spectrum of singular values for the coefficient matrix in each of the reconstructions.

for the cross; this illustrates the actual numerical effects and gives more detail than is available after digitization. The digitized equivalent images are shown in Fig.5.12 using the 9 grey levels. The images at various iterations are shown for the 'L' Fig.5.13 and for the Box Fig.5.14 In each case good estimates of the original images are retrieved from very degraded pictures. Even using the chosen, elementary, digitization the results are quite dramatic. The results confirm that the method of Conjugate Gradients will compute a smooth approximation to the exact solution. The work requirements to perform the retrieval are extremely favourable. It is interesting to note that in the case of the Box the iterated estimates retain the symmetry of the problem.

The elements $\alpha_i$ and $\beta_i$ together with the $L_2$-norms of the direction and residual are shown for each case in Tables 5.12, 5.13 and 5.14. As anticipated the residual is unreliable as an indication of the accuracy of the estimates. The magnitude of $\alpha_i$ and $\beta_i$ do demonstrate the rapid increase at the termination point. For the “Cross” there is an increase in $\beta$ by $10^2$ from iteration 3 to iteration 4; the best solution is at iteration
### Chapter 5. Numerical Application

<table>
<thead>
<tr>
<th>Iteration number</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iteration 1</td>
<td>1.1</td>
<td>1.1</td>
<td>1.1</td>
<td>1.1</td>
</tr>
<tr>
<td>Iteration 2</td>
<td>1.1</td>
<td>1.1</td>
<td>1.1</td>
<td>1.1</td>
</tr>
<tr>
<td>Iteration 3</td>
<td>1.1</td>
<td>1.1</td>
<td>1.1</td>
<td>1.1</td>
</tr>
<tr>
<td>Iteration 4</td>
<td>1.1</td>
<td>1.1</td>
<td>1.1</td>
<td>1.1</td>
</tr>
</tbody>
</table>

Figure 5.11: The exact numerical results from iterations 1, 2, 3, and 4 from the method of Conjugate Gradients for the Cross test image.
Figure 5.12: The visual representation for the results of iterations 1, 2, 3 and 4 for the Cross test image. Comparison with the numerical results in the previous figure shows the digitization effects.
Figure 5.13: The visual representation for the results of iterations 1, 3, 4 and 5 for the L test image. The same grey scale is used for the digitization as before.
Figure 5.14: The visual representation for the results of iterations 2, 3, 4 and 5 for the Box test image. The same grey scale is used for the digitization as before.
Table 5.12: The analysis of the parameters, direction norm and residual for the first 10 iterations of Conjugate Gradients for the Cross test example.

3. In the case of the “L” such an increase does not occur until iteration 7; however the error does not increase as rapidly as for the “Cross”. The “Box” is more troublesome as it is not until iteration number 9 that any substantial increase is demonstrated; \(10^2\) for \(\alpha\). In each case fluctuation in the magnitude of both \(\alpha\) and \(\beta\) does occur after four iterations. The large magnitude of the singular values of the coefficient matrix accounts for the small values for the \(\alpha\)’s.

5.2.1 Termination

If the iteration process is continued too far the picture becomes unstable. This is the expected feature of solutions to ill-posed problems. The question arises as to when to terminate the process for practical cases where the exact error is not available. An exact solution to the question (as with other solution methods for ill-posed problems) is not possible; however, there are various guidelines that may be considered.

1. Since each iteration is a relatively uncostly process one option is simply to view each of the images. This is the simplest, possibly least effective method, and involves no
### Table 5.13: The analysis of the parameters, direction norm and residual for the first 10 iterations of Conjugate Gradients for the L test example.

<table>
<thead>
<tr>
<th>Itn</th>
<th>Pnorm</th>
<th>$\alpha$</th>
<th>$\beta$</th>
<th>Residual</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$4.969 \times 10^5$</td>
<td>$1.828 \times 10^{-5}$</td>
<td>0</td>
<td>$3.038 \times 10^3$</td>
</tr>
<tr>
<td>2</td>
<td>$2.496 \times 10^5$</td>
<td>$1.190 \times 10^{-4}$</td>
<td>$2.087 \times 10^{-1}$</td>
<td>$1.760 \times 10^3$</td>
</tr>
<tr>
<td>3</td>
<td>$7.280 \times 10^4$</td>
<td>$3.057 \times 10^{-5}$</td>
<td>$9.248 \times 10^{-2}$</td>
<td>$1.718 \times 10^3$</td>
</tr>
<tr>
<td>4</td>
<td>$9.610 \times 10^4$</td>
<td>$2.382 \times 10^{-4}$</td>
<td>$9.449 \times 10^{-1}$</td>
<td>$1.371 \times 10^3$</td>
</tr>
<tr>
<td>5</td>
<td>$4.718 \times 10^4$</td>
<td>$5.249 \times 10^{-4}$</td>
<td>$3.044 \times 10^{-1}$</td>
<td>$1.077 \times 10^3$</td>
</tr>
<tr>
<td>6</td>
<td>$6.886 \times 10^4$</td>
<td>$3.678 \times 10^{-5}$</td>
<td>1.184</td>
<td>$1.049 \times 10^3$</td>
</tr>
<tr>
<td>7</td>
<td>$9.709 \times 10^4$</td>
<td>$9.987 \times 10^{-5}$</td>
<td>1.249</td>
<td>$9.475 \times 10^2$</td>
</tr>
<tr>
<td>8</td>
<td>$1.665 \times 10^4$</td>
<td>$4.228 \times 10^{-4}$</td>
<td>$9.486 \times 10^{-2}$</td>
<td>$9.036 \times 10^2$</td>
</tr>
<tr>
<td>9</td>
<td>$2.788 \times 10^4$</td>
<td>$6.556 \times 10^{-5}$</td>
<td>1.363</td>
<td>$8.941 \times 10^2$</td>
</tr>
<tr>
<td>10</td>
<td>$8.449 \times 10^4$</td>
<td>$1.280 \times 10^{-4}$</td>
<td>2.867</td>
<td>$8.386 \times 10^2$</td>
</tr>
</tbody>
</table>

### Table 5.14: The analysis of the parameters, direction norm and residual for the first 10 iterations of Conjugate Gradients for the Box test example.

<table>
<thead>
<tr>
<th>Itn</th>
<th>Pnorm</th>
<th>$\alpha$</th>
<th>$\beta$</th>
<th>Residual</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$1.460 \times 10^5$</td>
<td>$2.439 \times 10^{-5}$</td>
<td>0</td>
<td>$1.517 \times 10^3$</td>
</tr>
<tr>
<td>2</td>
<td>$1.358 \times 10^5$</td>
<td>$9.372 \times 10^{-5}$</td>
<td>$5.560 \times 10^{-1}$</td>
<td>$1.092 \times 10^3$</td>
</tr>
<tr>
<td>3</td>
<td>$3.222 \times 10^4$</td>
<td>$8.436 \times 10^{-4}$</td>
<td>$7.814 \times 10^{-2}$</td>
<td>$6.413 \times 10^2$</td>
</tr>
<tr>
<td>4</td>
<td>$3.346 \times 10^4$</td>
<td>$2.860 \times 10^{-4}$</td>
<td>$6.841 \times 10^{-1}$</td>
<td>$4.797 \times 10^2$</td>
</tr>
<tr>
<td>5</td>
<td>$1.493 \times 10^4$</td>
<td>$1.987 \times 10^{-4}$</td>
<td>$2.454 \times 10^{-1}$</td>
<td>$4.463 \times 10^2$</td>
</tr>
<tr>
<td>6</td>
<td>$5.567 \times 10^3$</td>
<td>$8.169 \times 10^{-4}$</td>
<td>$1.618 \times 10^{-1}$</td>
<td>$4.227 \times 10^2$</td>
</tr>
<tr>
<td>7</td>
<td>$1.249 \times 10^4$</td>
<td>$7.085 \times 10^{-4}$</td>
<td>1.875</td>
<td>$3.811 \times 10^2$</td>
</tr>
<tr>
<td>8</td>
<td>$3.220 \times 10^3$</td>
<td>$2.332 \times 10^{-3}$</td>
<td>$1.477 \times 10^{-1}$</td>
<td>$3.592 \times 10^2$</td>
</tr>
<tr>
<td>9</td>
<td>$1.022 \times 10^4$</td>
<td>$1.246 \times 10^{-3}$</td>
<td>2.857</td>
<td>$3.228 \times 10^2$</td>
</tr>
<tr>
<td>10</td>
<td>$2.388 \times 10^3$</td>
<td>$3.817 \times 10^{-5}$</td>
<td>$1.571 \times 10^{-1}$</td>
<td>$3.226 \times 10^2$</td>
</tr>
</tbody>
</table>
mathematics at all. The decision, as to when to stop, is then at the discretion of the user. It will certainly be only a few iterations and since it was seen that after reaching the minimum the exact error will increase monotonically there is not a high risk of excessive work being performed using this method.

2. The magnitude of the constants from the algorithm may be used. In the previous chapter it was suggested that consideration of the values of the parameters \( \alpha_i \) and \( \beta_i \) which are readily available from the algorithm may give a useful guide on the behaviour of the error. An apparent rapid increase in the size of either the \( \alpha_i \) or \( \beta_i \) suggests that a convenient termination point has been achieved.

3. The singular value decomposition of the matrices \( K_1 \) and \( K_2 \) can be achieved in only \( O(n^3) \) operations. It is possible then to obtain the singular values of the coefficient matrix \( K_L \) and its decay can be used to consider the point at which a truncated SVD may be terminated and thus a convenient number of iterations to be performed by the method of Conjugate Gradients.
Chapter 6

Discussion

This thesis has demonstrated that the method of Conjugate Gradients may be used for the efficient computation of a smooth approximation to the solution of large sparse ill-conditioned linear system of equations. Clear reasoning has been given for the successful filtering, of the oscillatory components, at each iteration and the method of Conjugate Gradients has been shown to give similar results to the method of truncating the SVD with significantly less computational effort. The work forms a basis for a more quantitative study for specific applications. Various points are raised:

- The decision as to when to terminate the iterative process is a complex problem. In cases where a priori knowledge is available regarding the singular value spectrum of the system a similar criterion over the choice of regularization parameter to that of truncating the SVD [40] is recommended. In most practical applications all that is available are the parameters $\alpha_i$ and $\beta_i$ from the algorithm. It is felt that these may be used in forming a termination criterion but since they are dependent on the noise in the system any generalization is not possible.

- It has been shown that the polynomial at the $k$th iteration is the solution to a weighted least squares problem. The weights are specific to the problem, making exact predictions as to the characteristics of the polynomial very difficult. Insight into the behaviour of the polynomial at the $k$th iteration was made possible by demonstrating that it approximates the interpolating polynomial through zero and
the first \( k \) singular values with the value one as the ordinate. With this simplification a qualitative discussion of the filtering properties was presented. The similarity of the interpolating polynomial (when considering evaluation only at the singular values) to the step function means that the filtering properties of the method of Conjugate Gradients are favourable for use as an iterative filtering technique. A more precise description of the polynomial is highly complex due to the dependence of the coefficients on the "noise" in the data. Further study into the weighted least squares problem (4.110) would enable a more precise evaluation of the features and may give more guidance on how to set a termination criterion.

- Although preconditioning is seen to improve the performance of the method of Conjugate Gradients (see for example [25]) it is not obvious that this is the case for ill-conditioned systems. In many cases the number of iterations required by the basic algorithm to achieve the best smooth solution is acceptably small, and the additional cost that will be incurred by use of a preconditioner is not justified. Preconditioning has the effect of increasing the rate of convergence to an approximation that has a required accuracy in terms of the residual. In ill-conditioned cases, where the residual is a poor estimate of accuracy, this acceleration may cause the algorithm to continue past the best smooth solution without ever computing it, so that the slower convergence without preconditioning is preferable. Systems for which the noise level is very low are effectively well-conditioned and preconditioning, using the algorithm detailed in §3.4.2, may be recommended to improve a prohibitively slow iteration process. Further work is recommended to assess the possibility of effective application of the preconditioned Conjugate Gradients method to ill-conditioned linear systems.
Bibliography


Bibliography


Appendix A

Appendix

This appendix details the particular results required from the theory of Kronecker products for the application in Chapter 5.

### A.1 Kronecker Products

The formulation of the mathematical image problem used in Chapter 5 requires the use of the Kronecker product of matrices. The following results are summarized from [23].

1. The Kronecker (tensor) product of two $(n \times n)$ matrices results in an $(n^2 \times n^2)$ matrix and is performed as follows

\[
A \otimes B = \begin{pmatrix}
a_{11}B & a_{12}B & \ldots & a_{1n}B \\
a_{21}B & a_{22}B & \ldots & a_{2n}B \\
\vdots & \vdots & \ddots & \vdots \\
a_{n1}B & \ldots & \ldots & a_{nn}B
\end{pmatrix}
\]

2. The result of multiplying the Kronecker product of two $(n \times n)$ matrixes by an $(n^2)$ vector is the $(n \times n)$ matrix given by

\[
(A \otimes B)x = AXB
\]

3. The Kronecker product of two matrix-products is equivalent to the matrix product of two Kronecker products performed in order, i.e.,

\[
(AB) \otimes (CD) = (A \otimes C)(B \otimes D)
\]
4. The transpose of a Kronecker product is equivalent to the Kronecker product of the transpose of both matrices, i.e.,

\[(A \otimes B)^T = A^T \otimes B^T\]  \hspace{1cm} (A.155)

5. If the singular values of the matrix \(A\) are \(\mu_s\) for \(s = 1, 2, \ldots, n\) and the singular values of the matrix \(B\) are \(\lambda_r\) for \(r = 1, 2, \ldots, n\), then the singular values of the matrix that is the Kronecker product of \(A\) and \(B\) \((A \otimes B)\) are the \(n^2\) values \(\lambda_r \mu_s\).

6. The expansion of an \((n \times n)\) matrix into an \(n^2\)-vector is achieved by stacking the rows of the matrix end on end.

\[
F = \begin{pmatrix}
  f_1^T \\
  f_2^T \\
  \vdots \\
  f_n^T
\end{pmatrix}
\]

\[
f = \begin{pmatrix}
  f_1 \\
  f_2 \\
  \vdots \\
  f_n
\end{pmatrix}
\]