Chapter 1

Introduction

1.1 Systems of Linear Equations

The main content of the course will be concerned with solving problems of the form

$$A\mathbf{x} = \mathbf{b}$$

where A is an $m \times n$ matrix of rank r. The most common sources of such problems are:

- Solution of problems arising in physical modelling
- Data analysis

Of course, we cannot solve such problems generally, because the nature of the solution is influenced greatly by the relationship between m, n, and r. For example, if $m = n = r$, then there is a unique solution. On the other hand, if $m < n$ and $m = r$, then there are infinitely many solutions, and often we need to find the unique solution that satisfies certain constraints. This is the basis for linear programming. Finally, if $m > n$, there may not be a solution, in which case we need to solve a least squares problem to find the vector $\mathbf{x}$ that comes as close as possible, in some sense, to solving the problem.

1.2 Numerical Solution of Differential Equations

Physical phenomena such as fluid flow, heat diffusion and electromagnetism can often be modelled as differential equations. For example, the fundamental laws governing electromagnetic phenomena can be summarized by

Maxwell's equations, which can be written in differential form as

$$\begin{aligned} \nabla \times \mathbf{E} &= -\frac{\partial \mathbf{B}}{\partial t} \\ \nabla \times \mathbf{H} &= \mathbf{J} + \frac{\nabla \mathbf{D}}{\nabla t} \\ \nabla \cdot \mathbf{D} &= \rho \\ \nabla \cdot \mathbf{B} &= 0 \end{aligned}$$

Unfortunately, such equations are often impossible to solve analytically, either due to the complexity of the equations themselves or of the shape of the domain. Thus, such systems are often solved approximately by discretizing the domain and approximating the differential operators with finite differences, which gives rise to systems of linear equations.
As an example, suppose that we wish to solve the differential equation

$$-y'' + \sigma y' = f, \quad 0 < x < 1,$$

with boundary conditions

$$y(0) = \alpha, \quad y(1) = \beta.$$

Let

$$h = \frac{1}{N+1}, \quad x_i = ih, \quad i = 0, 1, \ldots, N+1.$$

Then

$$-y''(x_i) \sim \frac{-y_{i-1} + 2y_i - y_{i+1}}{h^2},$$

and for the first derivative, we use a *centered difference* approximation,

$$y'(x_i) \sim \frac{y_{i+1} - y_{i-1}}{2h}.$$

Alternatively, we can use the *forward difference*

$$y'(x_i) \sim \frac{y_{i+1} - y_i}{h}.$$

In other words, the difference approximation becomes

$$\frac{-y_{i-1} + 2y_i - y_{i+1}}{h^2} + \sigma \left(\frac{y_{i+1} - y_{i-1}}{2h} \right) = f_i, \quad i = 1, 2, \ldots, N$$

or

$$-\left(1 + \frac{\sigma h}{2}\right) y_{i-1} + 2y_i - \left(1 - \frac{\sigma h}{2}\right) y_{i+1} = h^2 f_i \equiv g_i$$

where $f_i \equiv f(x_i)$. We therefore have a system of linear equations for the points y_i. This system can be described using matrix notation as

$$\begin{bmatrix} a & b & & & \\ c & \ddots & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & b \\ & & & c & a \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ \vdots \\ y_N \end{bmatrix} = \mathbf{g}$$

or

$$T\mathbf{y} = \mathbf{g},$$

where the matrix T is *tridiagonal*. While systems involving tridiagonal matrices are easy to solve in general, this particular system is even simpler because the entries of T are constant along each diagonal. We call such a matrix a *Toeplitz matrix*. When $\sigma = 0$, then $T = T^T$ and it is also positive definite.

Sometimes it is useful to reorder the variables in order to facilitate the application of certain algorithms. For example, notice that in our discretization, the value of the solution at odd-numbered nodes (y_{2i+1}) depend only on the values at even-numbered nodes (y_{2i} and y_{2i+2}), and vice versa. Matrices with this property are said to possess a *red-black ordering*, or property (A). Thus, if we reorder the notes so that the odd-numbered nodes come first, followed by the even ones, we obtain a system with the following form:

$$\left[\begin{array}{cccc|cccc} \times & & & & \times & & & \\ & \times & & & \times & \times & & \\ & & \times & & & \times & \times & \\ & & & \times & & & \times & \times \\ \hline \times & \times & & & \times & & & \\ & \times & \times & & & \times & & \\ & & \times & \times & & & \times & \\ & & & \times & & & & \times \end{array}\right] \begin{pmatrix} y_1 \\ y_3 \\ y_5 \\ y_7 \\ y_2 \\ y_4 \\ y_6 \\ y_8 \end{pmatrix} = \begin{pmatrix} g_1 \\ g_3 \\ g_5 \\ g_7 \\ g_2 \\ g_4 \\ g_6 \\ g_8 \end{pmatrix}$$

Having set up such a linear system, two major issues arise:

1. How accurately and efficiently can the linear system be solved? Such questions are the major focus of this course. Such issues are common to all problems involving linear systems and will be discussed in the subsequent sections (in the context of data fitting).

2. How well does the discretized solution (of the linear system) approximate the continuous solution of the differential equation, i.e. what can we say about the error

$$\max_i |y_i - y(x_i)|?$$

This is a major topic in Numerical Solution of Differential Equations (237B and C).

1.3 Data Analysis

Problems that involve analyzing large amounts of data frequently require linear algebra techniques as well. For example, search engines often need to rank their search results in order of importance. Such rankings can be modelled as the stationary distribution of a Markov chain with a large number of states (in the order of billions), which is actually an eigenvalue problem in disguise. Linear algebra problems also arise from data mining and statistical analysis.

We shall consider a simple problem which will illustrate many of the issues associated with numerical linear algebra. Suppose that we are given a set of observations $\{y_j, x_j\}_{j=1}^n$. We would like to fit this data by a model. For instance, we might write

$$p_\ell(x) = b_0 + b_1 x + \cdots + b_\ell x^\ell.$$

Now, let's choose the $\ell + 1$ parameters $\{b_j\}_{j=0}^\ell$ to minimize the modeling error

$$\phi(\mathbf{b}) = \sum_{j=1}^N (y_j - p_\ell(x_j))^2$$

over all coefficient vectors $\mathbf{b}$. We introduce the following notation:

$$X = \begin{bmatrix} 1 & x_1 & \cdots & x_1^\ell \\ 1 & x_2 & \cdots & x_2^\ell \\ \vdots & \vdots & & \vdots \\ 1 & x_N & \cdots & x_N^\ell \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix}$$

and

$$\|x\|_2 = \left(\sum_{i=1}^N |x_i|^2 \right)^{1/2}.$$

Then

$$\phi(\mathbf{b}) = (\mathbf{y} - X\mathbf{b})^T (\mathbf{y} - X\mathbf{b})$$

so

$$\begin{aligned} \phi(\mathbf{b}) &= \mathbf{y}^T\mathbf{y} - \mathbf{y}^T X\mathbf{b} - \mathbf{b}^T X^T\mathbf{y} + \mathbf{b}^T X^T X\mathbf{b} \\ &= \mathbf{y}^T\mathbf{y} - 2\mathbf{b}^T X^T\mathbf{y} + \mathbf{b}^T X^T X\mathbf{b} \end{aligned}$$

Differentiating with respect to each b_i, we obtain

$$\frac{\partial\phi(\mathbf{b})}{\partial b_i} = -2X^T\mathbf{y} + 2X^T X\mathbf{b}.$$

Setting $\nabla\phi = 0$ to obtain the minimizing value of $\mathbf{b}$, we have

$$X^T X\mathbf{b} = X^T\mathbf{y}.$$

This system of linear equations is known as the *normal equations*.

If we write

$$\mathbf{x}^r = \begin{bmatrix} x_1^r \\ x_2^r \\ \vdots \\ x_N^r \end{bmatrix}$$

then $X = (\mathbf{x}^0, \mathbf{x}^1, \ldots, \mathbf{x}^\ell)$ and for $i, j = 0, 1, \ldots, \ell$ we have

$$\begin{aligned} [X^T X]_{ij} &= (\mathbf{x}^i)^T\mathbf{x}^j \\ &= \sum_{r=1}^{N} x_r^i x_r^j \\ &= \sum_{r=1}^{N} x_r^{i+j} \\ &\equiv \mu_{i+j} \end{aligned}$$

and therefore

$$X^T X = \begin{bmatrix} \mu_0 & \mu_1 & \mu_2 & \cdots & \mu_\ell \\ \mu_1 & \mu_2 & & & \mu_{\ell+1} \\ \mu_2 & & & & \\ \vdots & & & & \\ \mu_\ell & \mu_{\ell+1} & & & \mu_{2\ell} \end{bmatrix}.$$

Matrices of this form (where the coefficients are constant along the anti-diagonals) arise quite often and are known as *Hankel matrices*[1]. It has $(2\ell + 1)$ parameters, namely the values μ_j for $j = 0, 1, \ldots, 2\ell$.

Does the solution $\tilde{\mathbf{b}}$ to the normal equations yield a minimum value for $\phi(\mathbf{b})$? Suppose $\hat{\mathbf{b}} = \tilde{\mathbf{b}} + \delta$. Then

$$\begin{aligned}
\phi(\hat{\mathbf{b}}) &= (\mathbf{y} - X\hat{\mathbf{b}})^T(\mathbf{y} - X\hat{\mathbf{b}}) \\
&= (\mathbf{y} - X\tilde{\mathbf{b}} - X\delta)^T(\mathbf{y} - X\tilde{\mathbf{b}} - X\delta) \\
&= (\mathbf{y} - X\tilde{\mathbf{b}})^T(\mathbf{y} - X\tilde{\mathbf{b}}) - \delta^T X^T(\mathbf{y} - X\tilde{\mathbf{b}}) - (\mathbf{y} - X\tilde{\mathbf{b}})X\delta + \delta^T X^T X\delta \\
&= (\mathbf{y} - X\tilde{\mathbf{b}})^T(\mathbf{y} - X\tilde{\mathbf{b}}) + \delta^T X^T X\delta
\end{aligned}$$

Since $\delta^T X^T X\delta = \|X\delta\|_2^2 \geq 0$, it follows that

$$\phi(\hat{\mathbf{b}}) \geq (\mathbf{y} - X\tilde{\mathbf{b}})^T(\mathbf{y} - X\tilde{\mathbf{b}}) = \phi(\tilde{\mathbf{b}})$$

and therefore $\tilde{\mathbf{b}}$ is a vector that minimizes ϕ. Is the solution $\tilde{\mathbf{b}}$ unique? If X has full column rank, then $\delta^T X^T X\delta > 0$ for all nonzero δ and therefore we must have $\delta = 0$ for $\phi(\hat{\mathbf{b}})$ to be a minimum. Otherwise, $\hat{\mathbf{b}}$ is a solution whenever $X\delta = 0$, so the solution is not unique. In such cases, a unique solution $\tilde{\mathbf{b}}$ is often determined by requiring that a solution must be of minimum length; i.e.

$$\tilde{\mathbf{b}}^T\tilde{\mathbf{b}} \leq \hat{\mathbf{b}}^T\hat{\mathbf{b}}$$

for any solution $\hat{\mathbf{b}}$. Later we shall discuss how such a solution can be found.

Another way of solving the least squares problem is to use the residual vector, which is given by

$$\mathbf{r} = \mathbf{y} - X\mathbf{b}. \tag{1.1}$$

Using the normal equations, we can conclude that

$$X^T\mathbf{r} = X^T\mathbf{y} - X^T X\mathbf{b} = 0. \tag{1.2}$$

We can combine equations (1.1) and (1.2) to obtain the *augmented system*

$$\begin{bmatrix} I & X \\ X^T & 0 \end{bmatrix} \begin{pmatrix} \mathbf{r} \\ \mathbf{b} \end{pmatrix} = 0.$$

[1]Matrices whose coefficients are constant along diagonals also arise frequently. Such matrices are called *Toeplitz*.

1.4 Issues in Data Fitting

In the previous section, we followed a three-step process to fit the original data to a function:

1. We chose to model the data using a polynomial approximation
2. We formed the normal equations, whose solution would yield the best approximation possible given the chosen model
3. We solved the linear system to obtain the approximation.

We will now discuss some issues that commonly arise from these three stages of data fitting.

1.4.1 Modeling with Polynomial Approximations

Implicitly we assumed that we know x_i exactly. We can drop this assumption and use the statistical model

$$y_j = p_\ell(x_j) + \varepsilon_j$$

where ε_j is noise, and

$$x_j = \xi_j + \eta_j$$

where η_j is noise, uncorrelated to the noise ε_j. We might want to consider minimizing both

$$(\mathbf{y} - X\mathbf{b})^T(\mathbf{y} - X\mathbf{b})$$

and

$$(\mathbf{x} - \xi)^T(\mathbf{x} - \xi).$$

1.4.2 Forming the Normal Equations

We have assumed that X^TX is formed exactly, but we need to consider error in computing the entries of X^TX. For example, if we need to compute x_i^5 where $x_i \approx 10^3$, $x_i^5 \approx 10^{15}$, and therefore we need 15 places to compute the matrix X^TX exactly. Because precision is limited, roundoff error must be taken into account when computing the entries.

Can we reduce the effect of roundoff error by choosing another basis? Suppose that we know our values of x_i are approximately some value a. Then we can shift by a to obtain the representation

$$p_\ell(x) = a_0 + a_1(x - a) + \cdots + a_\ell(x - a)^\ell$$

resulting in entries of smaller magnitude in X^TX. Another strategy is to use the approximation

$$p_\ell(x) = c_0 q_0(x) + \cdots + c_\ell q_\ell(x)$$

where $q_\ell(x)$ is a polynomial of degree ℓ and

$$\sum_{i=1}^{N} q_r(x_i) q_s(x_i) = 0$$

whenever $r \neq s$. Such polynomials are called *orthogonal polynomials*. Using

$$Q = \begin{bmatrix} q_0(x_1) & q_1(x_1) & \cdots & q_\ell(x_1) \\ \vdots & & & \vdots \\ q_0(x_N) & \cdots & \cdots & q_\ell(x_N) \end{bmatrix}$$

we obtain

$$X^TX = Q^TQ = \begin{bmatrix} d_0 & & \\ & \ddots & \\ & & d_\ell \end{bmatrix}$$

where

$$d_r = \sum_{i=1}^{N} (q_r(x_i))^2.$$

1.4.3 Updating and Downdating

In forming the normal equations, we need to consider the situation where the value of the degree ℓ is increased. In other words, we want to increase the order of the approximating polynomial so

$$X_{\ell+1} = [X_\ell \quad x^{\ell+1}]$$

$$\begin{aligned} X_{\ell+1}^T X_{\ell+1} &= \begin{bmatrix} X_\ell^T \\ (\mathbf{x}^{\ell+1})^T \end{bmatrix} \begin{bmatrix} X_\ell & \mathbf{x}^{\ell+1} \end{bmatrix} \\ &= \begin{bmatrix} X_\ell^T X_\ell & X_\ell^T \mathbf{x}^{\ell+1} \\ (\mathbf{x}^{\ell+1})^T X_\ell & (\mathbf{x}^{\ell+1})^T \mathbf{x}^{\ell+1} \end{bmatrix} \end{aligned}$$

resulting in the updated normal equations

$$X_{\ell+1}^T X_{\ell+1} \mathbf{b}_{\ell+1} = X_{\ell+1}^T \mathbf{y}_{\ell+1}.$$

Note that in the case of orthogonal polynomials, $Q_{\ell+1}^T Q_{\ell+1}$ is a diagonal matrix so that the coefficients $c_0, \ldots, c_\ell$ do not change.

Another common scenario is that we might add one more observation

$$x_N^r \rightarrow x_{N+1}^r = \begin{bmatrix} x_1^r \\ \vdots \\ x_N^r \\ x_{N+1}^r \end{bmatrix}.$$

These modifications are examples of *updating problems*, where we seek to obtain the solution of a modified problem more efficiently by updating the solution to the original problem, rather than recomputing the solution to the modified problem from the beginning. It is also not uncommon to have to solve *downdating problems*, where the solution is affected by the removal of data from the original problem. The above discussion shows that orthogonal polynomials are particularly well-suited to updating and downdating, which is just one reason why they are used so frequently in data-fitting applications.

1.4.4 Solving Linear Equations

In solving systems of linear equations, the following questions arise:

- How can we perform computations accurately?
- Are there alternative methods of solution?
- Can specialized methods for structured matrices such as Hankel matrices or Toeplitz matrices be used?

In many applications, systems involving *sparse* matrices must be solved, and as a result many methods, some of which we will discuss, have been developed for these cases. Furthermore, the advent of vector and parallel architectures have led to the development of new methods for solving linear systems.

A common difficulty is the occurrence of an *ill-posed problem*. The question that must be answered when solving any linear system is, can small changes in data make a large change in solution? A priori bounds and a posteriori bounds can be used to help answer this question.

Another consideration is that many linear systems must be solved subject to constraints such as

$$C^T \mathbf{b} = 0$$

for some given matrix C, or

$$\mathbf{b}^T\mathbf{b} = \alpha^2$$

for some known constant α. For example, one may wish to fit data using different functions on different intervals, in which case such constraints can be used to make the composite function smooth. This is the idea behind cubic splines.

Finally, it makes sense to ask, why minimize $\phi(\mathbf{b}) = \sum_{j=1}^{N}(y_j - p_\ell(x_j))^2$? Instead, why not minimize

$$\phi(\mathbf{b}) = \sum_{j=1}^{N} |y_j - p_\ell(x_j)|?$$

Minimizing alternative measures of error such as this is possible, but the sum of squares lends itself to minimization more naturally than these other measures, as will be discussed later.

Chapter 2

Matrix Analysis

This chapter establishes the fundamental concepts used in numerical linear algebra:

- Vector and matrix norms,
- Gerschgorin theorem,
- Singular value decomposition,
- Projections and pseudo-inverses,
- Jordan canonical form.

2.1 Vector Norms

A real-valued function $f(\mathbf{x})$ defined on a vector space is said to be a *norm* if

1. For any vector $\mathbf{x}$, $f(\mathbf{x}) \geq 0$, and $f(\mathbf{x}) = 0$ if and only if $\mathbf{x} = \mathbf{0}$.
2. $f(\alpha\mathbf{x}) = |\alpha| f(\mathbf{x})$ for any scalar α.
3. $f(\mathbf{x} + \mathbf{y}) \leq f(\mathbf{x}) + f(\mathbf{y})$ for any two vectors $\mathbf{x}$ and $\mathbf{y}$.

We indicate $f(x) = \|x\|$.

Consider

$$\nu(\mathbf{x}) = \sum_{i=1}^{n} |x_i|.$$

Is this a norm?

1. Clearly $\nu(\mathbf{x}) \geq 0$, and the only way that $\nu(\mathbf{x}) = 0$ is if $\mathbf{x} = \mathbf{0}$.

2. We have

$$\begin{aligned}\nu(\alpha\mathbf{x}) &= \sum_{i=1}^{n} |\alpha x_i| \\ &= |\alpha| \sum_{i=1}^{n} |x_i| \\ &= |\alpha|\nu(\mathbf{x}).\end{aligned}$$

3. Using the triangle inequality for scalars, we obtain

$$\begin{aligned}\nu(\mathbf{x}+\mathbf{y}) &= \sum_{i=1}^{n} |x_i + y_i| \\ &\leq \sum_{i=1}^{n} |x_i| + |y_i| \\ &\leq \nu(\mathbf{x}) + \nu(\mathbf{y}).\end{aligned}$$

We define the p-norm $\|\mathbf{x}\|_p$ by

$$\|\mathbf{x}\|_p = \left(|x_1|^p + \cdots + |x_n|^p\right)^{1/p}.$$

The 1-norm

$$\|\mathbf{x}\|_1 = \sum_{i=1}^{n} |x_i|$$

is the same as the function $\nu(x)$ discussed above. Another common norm is the 2-norm

$$\|\mathbf{x}\|_2 = \left(\sum_{i=1}^{n} |x_i|^2\right)^{1/2}.$$

It can be shown that for any p, we have

$$(\max_i |x_i|^p)^{1/p} \leq \|\mathbf{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p\right)^{1/p} \leq (n \max_i |x_i|^p)^{1/p}$$

from which it follows that

$$\max_i |x_i| \leq \|\mathbf{x}\|_p \leq n^{1/p} \max_i |x_i|.$$

Therefore, as $p \to \infty$, we obtain the *infinity norm*

$$\|\mathbf{x}|_\infty = \lim_{p\to\infty} \|\mathbf{x}\|_p = \max_i |x_i|.$$

This norm is also known as the *Chebyshev norm*. It is easy to verify that it is in fact a norm.

A variation of the p-norm is the *weighted p-norm*, defined by

$$\|\mathbf{x}\|_{p,\mathbf{w}} = \left(\sum_{i=1}^{n} w_i |x_i|^p \right)^{1/p}.$$

It can be shown that this is a norm as long as the *weights* w_i, $i = 1, \ldots, n$, are strictly positive. Another common norm is the A-norm, defined in terms of a matrix A by

$$\|\mathbf{x}\|_A = (\mathbf{x}^T A \mathbf{x})^{1/2},$$

which is a norm provided that the matrix A is positive definite.

We now highlight some additional, and useful, relationships involving norms. First of all, the triangle inequality generalizes directly to sums of more than two vectors:

$$\|\mathbf{x} + \mathbf{y} + \mathbf{z}\| \leq \|\mathbf{x} + \mathbf{y}\| + \|\mathbf{z}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\| + \|\mathbf{z}\|,$$

and in general,

$$\left\| \sum_{i=1}^{m} \mathbf{x}_i \right\| \leq \sum_{i=1}^{m} \|\mathbf{x}_i\|.$$

What can we say about the norm of the difference of two vectors? While we know that $\|\mathbf{x} - \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|$, we can obtain a more useful relationship as follows: From

$$\|\mathbf{x}\| = \|(\mathbf{x} - \mathbf{y}) + \mathbf{y}\| \leq \|\mathbf{x} - \mathbf{y}\| + \|\mathbf{y}\|$$

we obtain

$$\|\mathbf{x} - \mathbf{y}\| \geq \|\mathbf{x}\| - \|\mathbf{y}\|.$$

Similarly, from

$$\|\mathbf{y}\| = \|\mathbf{y} - \mathbf{x} + \mathbf{x}\| = \|\mathbf{x} - \mathbf{y}\| + \|\mathbf{x}\|$$

it follows that

$$\|\mathbf{x} - \mathbf{y}\| \geq \|\mathbf{y}\| - \|\mathbf{x}\|$$

and therefore

$$|\|\mathbf{x}\| - \|\mathbf{y}\|| \leq \|\mathbf{x} - \mathbf{y}\|.$$

There are also interesting relationships among different norms. First and foremost, all norms are, in some sense, equivalent. In particular, given two

norms $\|\mathbf{x}\|_\alpha$ and $\|\mathbf{x}\|_\beta$, there exist constants c_1 and c_2, independent of $\mathbf{x}$, such that

$$c_1\|\mathbf{x}\|_\alpha \leq \|\mathbf{x}\|_\beta \leq c_2\|\mathbf{x}\|_\alpha.$$

For example, from the definition of the ∞-norm, we have

$$\|\mathbf{x}\|_\infty \leq \|\mathbf{x}\|_2 \leq \sqrt{n}\|\mathbf{x}\|_\infty.$$

It is also not hard to show that

$$\frac{1}{n}\|\mathbf{x}\|_1 \leq \|\mathbf{x}\|_\infty \leq \|\mathbf{x}\|_1.$$

We also have a relationship that applies to products of norms, the *Hölder inequality*

$$|\mathbf{x}^T\mathbf{y}| \leq \|\mathbf{x}\|_p\|\mathbf{y}\|_q, \quad \frac{1}{p}+\frac{1}{q}=1.$$

A well-known corollary arises when $p = q = 2$, the *Cauchy-Schwarz inequality*

$$|\mathbf{x}^T\mathbf{y}| \leq \|\mathbf{x}\|_2\|\mathbf{y}\|_2.$$

It is interesting to note that by setting $\mathbf{x} = (1, 1, \ldots, 1)$, the Hölder inequality yields the relationships

$$|\sum_{i=1}^{n} y_i| \leq \sum_{i=1}^{n} |y_i|,$$

$$|\sum_{i=1}^{n} y_i| \leq n \max_i |y_i|,$$

and

$$|\sum_{i=1}^{n} y_i| \leq \sqrt{n}\left(\sum_{i=1}^{n} |y_i|^2\right)^{1/2}.$$

A very important property of norms is that they are all continuous functions of the entries of their arguments. It follows that a sequence of vectors $\mathbf{x}_0, \mathbf{x}_1, \ldots$ converges to a vector $\mathbf{x}$ if and only if

$$\lim_{k\to\infty} \|\mathbf{x}_k - \mathbf{x}\| = 0$$

for any norm. For this reason, norms are very useful to measure the error in an approximation. Three commonly used measures of the error in an approximation $\hat{\mathbf{x}}$ to a vector $\mathbf{x}$ are the *absolute error*

$$\varepsilon_{abs} = \|\mathbf{x} - \hat{\mathbf{x}}\|,$$

the *relative error*

$$\varepsilon_{rel} = \frac{\|\mathbf{x} - \hat{\mathbf{x}}\|}{\|\mathbf{x}\|},$$

and the *point-wise error*

$$\varepsilon_{elem} = \|\mathbf{y}\|, \quad y_i = \frac{\hat{x}_i - x_i}{x_i}.$$

2.2 Matrix Norms

A real-valued function $f(A)$ defined on the space of $m \times n$ matrices is called a *norm* if

1. $f(A) \geq 0$, and $f(A) = 0$ if and only if $A = 0$,
2. $f(A + B) \leq f(A) + f(B)$,
3. $f(\alpha A) = |\alpha| f(A)$.

Often, we add the condition that $f(A)$ satisfy the *submultiplicative property*

$$f(AB) \leq f(A)f(B).$$

We write $\|A\| = f(A)$. An example of a matrix norm is the *Frobenius norm*

$$\|A\|_F = \left(\sum_{i=1}^{m} \sum_{i=1}^{n} |a_{ij}|^2 \right)^{1/2}.$$

On the other hand, the function

$$f(A) = \max_{i,j} |a_{ij}|$$

is not a norm because it does not satisfy the submultiplicative property, but for an appropriate choice of a constant c, the function $f(A) = c \max_{i,j} |a_{ij}|$ is a norm.

Note that the submultiplicative property implies that

$$\|A^n\| \leq \|A\|^n,$$

from which it follows that if $\|A\| < 1$, then, as $n \to \infty$, $\|A^n\| \to 0$, and therefore $A^n \to 0$ as well, due to the continuity of the matrix norm.

How can we define a matrix norm? It is natural to want to define matrix norms in terms of vector norms, but we must be careful in doing so. For

example, if we choose to view an $m \times n$ matrix A as a vector α with mn elements, then we have $\|A\|_F = \|\alpha\|_2$. However, if we define a norm $\|A\|_\nu = \|\alpha\|_\infty$, then the resulting matrix norm does not satisfy the submultiplicative property. To see this, take

$$A = \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}.$$

Then

$$AB = \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix},$$

but $\|A\|_\alpha = \|B\|_\alpha = 1$, while $\|AB\|_\alpha = 2$.

Instead, we take the approach of defining the *natural norm*

$$\|A\| = \sup_{\mathbf{x} \neq \mathbf{0}} \frac{\|A\mathbf{x}\|}{\|\mathbf{x}\|} = \max_{\|\mathbf{y}\|=1} \|A\mathbf{y}\|.$$

where $\|\mathbf{x}\|$ is any given vector norm. We say that the vector norm $\|\mathbf{x}\|$ *induces* the matrix norm $\|A\|$. Since the set of all vectors $\{\mathbf{y} | \|\mathbf{y}\| = 1\}$ is compact and the norm $\|\mathbf{y}\|$ is continuous, there is some vector $\mathbf{y}$ such that $\|\mathbf{y}\| = 1$ and $\|A\| = \|A\mathbf{y}\|$.

We will now verify that the natural norm does in fact define a valid matrix norm.

1. If $A = 0$, then clearly $A\mathbf{y} = \mathbf{0}$ for any $\mathbf{y}$, so we must have $\|A\| = 0$. Otherwise, $a_{ij} \leq 0$ for some i, j. If we let $\mathbf{y}$ be the vector containing all zero elements except for a 1 in the jth position, then $\|\mathbf{y}\| > 0$ and $\|A\mathbf{y}\| > 0$, since $A\mathbf{y}$ is the jth column of A, which is a nonzero vector. Therefore
$$\|A\| \geq \frac{\|A\mathbf{y}\|}{\|\mathbf{y}\|} > 0.$$

2. Let $\mathbf{y}_0$ be a vector satisfying $\|\mathbf{y}_0\| = 1$ and $\|A\| = \|A\mathbf{y}_0\|$. For any constant α, we have
$$\max_{\|\mathbf{y}\|=1} \|\alpha A\mathbf{y}\| = \|\alpha A\mathbf{y}_0\| = |\alpha| \|A\mathbf{y}_0\| = |\alpha| \|A\|.$$

3. Before we prove that the triangle inequality holds, it is useful to note that for any matrix A and vector $\mathbf{z}$,
$$\begin{aligned} \|A\mathbf{z}\| &= \left\| A\|\mathbf{z}\| \frac{\mathbf{z}}{\|\mathbf{z}\|} \right\| \\ &= \|\mathbf{z}\| \left\| A\frac{\mathbf{z}}{\|\mathbf{z}\|} \right\| \\ &\leq \|\mathbf{z}\| \|A\|. \end{aligned}$$

Then, for some vector $\mathbf{y}_0$ satisfying $\|\mathbf{y}_0\| = 1$, we have

$$\begin{aligned} \|A+B\| &= \|(A+B)\mathbf{y}_0\| \\ &\leq \|A\mathbf{y}_0\| + \|B\mathbf{y}_0\| \\ &\leq \|A\| + \|B\|. \end{aligned}$$

4. Using the property $\|A\mathbf{z}\| \leq \|A\|\|\mathbf{z}\|$, it follows that for some vector $\mathbf{y}_0$ satisfying $\|\mathbf{y}_0\| = 1$, we have

$$\begin{aligned} \|AB\| &= \|A(B\mathbf{y}_0)\| \\ &\leq \|A\|\|B\mathbf{y}_0\| \\ &\leq \|A\|\|B\|. \end{aligned}$$

We denote by $\|\cdot\|_p$ the natural norm induced by the vector p-norm. The most commonly used matrix norms are $\|\cdot\|_1$, $\|\cdot\|_2$ and $\|\cdot\|_\infty$.

2.3 The Matrix ∞-norm

We will now try to arrive at an explicit expression for $\|A\|_\infty$. We have

$$\begin{aligned} \|A\|_\infty &= \max_{\|\mathbf{y}\|_\infty=1} \|A\mathbf{y}\| \\ &= \max_{\|\mathbf{y}\|_\infty=1} \max_i \left| \sum_{j=1}^n a_{ij} y_j \right| \\ &\leq \max_{\|\mathbf{y}\|_\infty=1} \max_i \sum_{j=1}^n |a_{ij}||y_j| \\ &\leq \max_{\|\mathbf{y}\|_\infty=1} \max_i \max_j |y_j| \sum_{j=1}^n |a_{ij}| \\ &\leq \max_i \sum_{j=1}^n |a_{ij}|. \end{aligned}$$

Now, suppose that

$$\max_i \sum_{j=1}^n |a_{ij}| = \sum_{j=1}^n |a_{Ij}|.$$

Let $\mathbf{y}$ be the vector with elements

$$y_j = \begin{cases} +1 & a_{Ij} \geq 0 \\ -1 & a_{Ij} < 0 \end{cases}$$

Then $\|\mathbf{y}\|_\infty = 1$ and

$$\|A\mathbf{y}\|_\infty = \sum_{j=1}^{n} |a_{Ij}| = \max_i \sum_{j=1}^{n} |a_{ij}|.$$

Therefore the upper bound we obtained for $\|A\|_\infty$ is actually assumed for some unit vector $\mathbf{y}$, from which it follows that

$$\|A\|_\infty = \max_i \sum_{j=1}^{n} |a_{ij}|.$$

Similarly, the vector 1-norm

$$\|\mathbf{x}\|_1 = \sum_{i=1}^{n} |x_i|$$

induces the matrix 1-norm

$$\|A\|_1 = \max_j \sum_{i=1}^{n} |a_{ij}|.$$

2.4 The Matrix 2-Norm

It is natural to ask whether there is a similar explicit expression for the matrix 2-norm induced by the vector 2-norm

$$\|\mathbf{x}\|_2 = \left(\sum_{i=1}^{n} |x_i|^2 \right)^{1/2}.$$

One might suggest the Frobenius norm

$$\|A\|_F = \left(\sum_{i=1}^{n} \sum_{j=1}^{n} |a_{ij}|^2 \right)^{1/2},$$

but that is incorrect. We will now derive an explicit expression for the induced matrix 2-norm.

Recalling the definition of the matrix 2-norm,

$$\|A\|_2 = \max_{\mathbf{x} \neq 0} \frac{\|A\mathbf{x}\|_2}{\|\mathbf{x}\|_2},$$

we examine the expression

$$\|A\mathbf{x}\|_2^2 = x^T A^T A x.$$

The matrix $A^T A$ is symmetric and positive semi-definite, i.e. $\mathbf{x}^T(A^T A)\mathbf{x} \geq 0$ for all nonzero $\mathbf{x}$. As such, it has the decomposition

$$A^T A = U \Sigma U^T$$

where U is an orthogonal matrix whose columns are the eigenvectors of $A^T A$, and Σ is a diagonal matrix of the form

$$\begin{bmatrix} \sigma_1^2 & & \\ & \ddots & \\ & & \sigma_n^2 \end{bmatrix}$$

where each σ_i is nonnegative and an eigenvalue of $A^T A$. These eigenvalues can be ordered such that

$$\sigma_1^2 \geq \sigma_2^2 \geq \cdots \geq \sigma_r^2 > 0, \quad \sigma_{r+1}^2 = \cdots = \sigma_n^2 = 0,$$

where r is the rank of A. If we define $\mathbf{w} = U^T \mathbf{x}$, then we obtain

$$\begin{aligned} \frac{\|A\mathbf{x}\|_2^2}{\|\mathbf{x}\|_2^2} &= \frac{\mathbf{x}^T A^T A \mathbf{x}}{\mathbf{x}^T \mathbf{x}} \\ &= \frac{\mathbf{x}^T U \Sigma U^T \mathbf{x}}{\mathbf{x}^T U^T U \mathbf{x}} \\ &= \frac{\mathbf{w}^T \Sigma \mathbf{w}}{\mathbf{w}^T \mathbf{w}} \\ &= \frac{\sum_{i=1}^n \sigma_i^2 |w_i|^2}{\sum_{i=1}^n |w_i|^2} \\ &\leq \sigma_1^2. \end{aligned}$$

It follows that

$$\frac{\|A\mathbf{x}\|_2}{\|\mathbf{x}\|_2} \leq \sigma_1$$

for all nonzero $\mathbf{x}$, but we must now determine whether this inequality is actually an equality for any $\mathbf{x}$. Since U is an orthogonal matrix, it follows that there exists an $\mathbf{x}$ such that

$$\mathbf{w} = U^T \mathbf{x} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \mathbf{e}_1,$$

in which case

$$\mathbf{x}^T A^T A \mathbf{x} = \mathbf{e}_1^T \Sigma \mathbf{e}_1 = \sigma_1^2.$$

In fact, this vector $\mathbf{x}$ is the eigenvector of $A^T A$ corresponding to the eigenvalue σ_1^2.

We conclude that

$$\|A\|_2 = \sigma_1.$$

2.5 The Spectral Radius

The matrix 2-norm is also known as the *spectral norm*. This name is connected to the fact that the norm is given by the square root of the largest eigenvalue of $A^T A$, and, in general, the *spectral radius* $\rho(A)$ of a matrix A is defined in terms of its largest eigenvalue:

$$\rho(A) = \max_i |\lambda_i|, \quad A\mathbf{x}_i = \lambda_i \mathbf{x}_i, \quad \mathbf{x}_i \neq \mathbf{0}.$$

We now discuss some relationships between the norm of a matrix and its spectral radius. First, suppose that

$$A\mathbf{x}_i = \lambda_i \mathbf{x}_i.$$

Then, for any matrix norm,

$$\|A\mathbf{x}_i\| = \|\lambda_i \mathbf{x}_i\| = |\lambda_i| \|\mathbf{x}_i\|.$$

Therefore

$$|\lambda_i| = \frac{\|A\mathbf{x}_i\|}{\|\mathbf{x}_i\|} \leq \|A\|.$$

Since this holds for *any* eigenvalue of A, it follows that

$$\max_i |\lambda_i| = \rho(A) \leq \|A\|.$$

We also have the following result:

Theorem For every $\epsilon > 0$, there exists a matrix norm $\|A\|_\alpha$ such that

$$\|A\|_\alpha \leq \rho(A) + \epsilon.$$

The norm is dependent on the matrix A.

This result suggests that the largest eigenvalue of a matrix can be easily approximated. A simple example is the case of the identity matrix I, whose

only eigenvalue is 1, and whose norm is equal to 1 for any natural matrix norm.

As another example, let

$$A = \begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & 1 \\ & & & -1 & 2 \end{bmatrix}.$$

The eigenvalues of this matrix, which arises frequently in numerical methods for solving differential equations, are known to be

$$\lambda_j = 2 + 2\cos\frac{j\pi}{n+1}, \quad j = 1, 2, \ldots, n.$$

The largest eigenvalue is

$$|\lambda_1| = 2 + 2\cos\frac{\pi}{n+1} \leq 4,$$

and $\|A\|_\infty = 4$, so in this case, the ∞-norm provides an excellent approximation.

On the other hand, suppose

$$A = \begin{bmatrix} 1 & 10^6 \\ 0 & 1 \end{bmatrix}.$$

We have $\|A\|_\infty = 10^6 + 1$, but $\rho(A) = 1$, so in this case the norm yields a poor approximation. However, suppose

$$D = \begin{bmatrix} \epsilon & 0 \\ 0 & 1 \end{bmatrix}.$$

Then

$$DAD^{-1} = \begin{bmatrix} 1 & 10^6\epsilon \\ 0 & 1 \end{bmatrix},$$

and $\|DAD^{-1}\|_\infty = 1 + 10^{-6}\epsilon$,, which for sufficiently small ϵ, yields a much better approximation to $\rho(DAD^{-1}) = \rho(A)$.

We can also show that all matrix norms are equivalent: for any two matrix norms $\|A\|_\alpha$ and $\|A\|_\beta$, there exist constants c_1 and c_2 such that

$$c_2\|A\|_\alpha \leq \|A\|_\beta \leq c_1\|A\|_\alpha,$$

and the constants c_1 and c_2 are independent of the matrix A. For example, for any $m \times n$ matrix A,

$$\frac{1}{\sqrt{n}}\|A\|_\infty \leq \|A\|_2 \leq \sqrt{m}\|A\|_\infty.$$

2.6 Gerschgorin's Theorem

Suppose that λ is an eigenvalue of A with corresponding eigenvector $\mathbf{x}$; i.e.

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

Then, for $i = 1, 2, \ldots, n$, we have

$$\sum_{j=1}^{n} a_{ij}x_j = \lambda x_j.$$

Rearranging, we obtain

$$(a_{ii} - \lambda)x_i = -\sum_{j\neq i} a_{ij}x_j,$$

and, taking absolute values yields

$$|a_{ii} - \lambda||x_i| \leq \sum_{j\neq i} |a_{ij}||x_j|.$$

Now, suppose that $|x_I| \geq |x_i|$, for $i = 1, 2, \ldots, n$. Then

$$|a_{II} - \lambda| \leq \sum_{j\neq I} |a_{Ij}|\frac{|x_j|}{|x_I|} \leq \sum_{j\neq I} |a_{Ij}|.$$

Since this bound applies to any eigenvalue of A, we can conclude that each eigenvalue of A lies within the union of the *Gerschgorin disks* defined by

$$|\lambda - a_{ii}| \leq r_i, \quad r_i = \sum_{j\neq i} |a_{ij}|, \quad i = 1, 2, \ldots, n.$$

This result is known as *Gerschgorin's Theorem*.

Suppose that $A = D + K$, where D is a diagonal matrix with diagonal entries $d_{ii} = a_{ii}$, and K represents the off-diagonal portion of A, with entries

$$K_{ij} = \begin{cases} a_{ij} & i \neq j \\ 0 & i = j \end{cases}$$

Then, define $A(\epsilon) = D + \epsilon K$. Then $A(0) = D$ and $A(1) = A$. The eigenvalues of $A(\epsilon)$ are continuous functions of ϵ, so we can approximate the eigenvalues of A by examining how the Gerschgorin disks change as ϵ changes. In particular, we can determine how many eigenvalues lie within individual disks or unions of disks.

2.7 The Singular Value Decomposition

Suppose A is an $m \times n$ real matrix with $m \geq n$. Then we can write

$$A = U\Sigma V^T,$$

where

$$U^T U = I_m, \quad V^T V = I_n, \quad \Sigma = \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_n \\ & 0 & \end{bmatrix}.$$

The diagonal elements σ_i, $i = 1, \ldots, n$, are all nonnegative, and can be ordered such that

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0, \quad \sigma_{r+1} = \cdots = \sigma_n = 0$$

where r is the rank of A. This decomposition of A is called the *singular value decomposition*, or SVD. The values σ_i, for $i = 1, 2, \ldots, n$, are the *singular values* of A. The columns of U are the *left singular vectors*, and the columns of V are the *right singular vectors*.

An alternative decomposition of A omits the singular values that are equal to zero:

$$A = \tilde{U}\tilde{\Sigma}\tilde{V}^T,$$

where $\tilde{U}$ is an $m \times r$ matrix satisfying $\tilde{U}^T\tilde{U} = I_r$, $\tilde{V}$ is an $n \times r$ matrix satisfying $\tilde{V}^T\tilde{V} = I_r$, and $\tilde{\Sigma}$ is an $r \times r$ diagonal matrix with diagonal elements $\sigma_1, \ldots, \sigma_r > 0$. The columns of $\tilde{U}$ are the left singular vectors corresponding to the nonzero singular values of A, and form an orthogonal basis for the range of A. The columns of $\tilde{V}$ are the right singular vectors corresponding to the nonzero singular values of A, and are each orthogonal to the null space of A.

If A is an $m \times m$ matrix and $\sigma_m > 0$, then

$$A^{-1} = (V^T)^{-1}\Sigma^{-1}U^{-1} = V\Sigma^{-1}U^T.$$

We will see that this representation of the inverse can be used to obtain a *generalized inverse* of a matrix A in the case where A does not actually have an inverse.

While the previous discussion assumed that A was a real matrix, the SVD exists for complex matrices as well. In this case, the decomposition takes the form

$$A = U\Sigma V^*$$

where, for general A, $A^* = \bar{A}^T$, the complex conjugate of the transpose. A^* is often written as A^H, and is equivalent to the transpose for real matrices.

We now mention some additional properties of the singular values and singular vectors. We have

$$A^T A = V\Sigma^T U^T U\Sigma V^T = V(\Sigma^T\Sigma)V^T.$$

The matrix $\Sigma^T\Sigma$ is a diagonal matrix with diagonal elements σ_i^2, $i = 1, \ldots, n$, which are also the eigenvalues of $A^T A$, with corresponding eigenvectors v_i, where v_i is the ith column of V. Similarly, $AA^T = U\Sigma\Sigma^T U^T$, from which we can easily see that the columns of U are eigenvectors of AA^T, corresponding to the eigenvalues σ_i^2, $i = 1, \ldots, n$.

In section 2.4, we have shown that

$$\|A\|_2 = [\rho(A^T A)]^{1/2}.$$

Since the eigenvalues of $A^T A$ are simply the squares of the singular values of A, we can also say that

$$\|A\|_2 = \sigma_1.$$

Another way to arrive at this same conclusion is to take advantage of the fact that the 2-norm of a vector is invariant under multiplication by an orthogonal matrix, i.e. if $Q^T Q = I$, then $\|\mathbf{x}\|_2 = \|Q\mathbf{x}\|_2$. Therefore

$$\|A\|_2 = \|U\Sigma V^T\|_2 = \|\Sigma\|_2 = \sigma_1.$$

2.8 Existence of the SVD

We will now prove the existence of the SVD. We define

$$\tilde{A} = \begin{bmatrix} 0 & A \\ A^* & 0 \end{bmatrix}.$$

It is easy to verify that $\tilde{A} = \tilde{A}^*$, and therefore $\tilde{A}$ has the decomposition $\tilde{A} = Z\Lambda Z^*$ where Z is an orthogonal matrix and Λ is a diagonal matrix with real diagonal elements. If $\mathbf{z}$ is a column of Z, then we can write

$$\tilde{A}\mathbf{z} = \sigma\mathbf{z}, \quad \mathbf{z} = \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix}$$

and therefore

$$\begin{bmatrix} 0 & A \\ A^* & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} = \sigma \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix}$$

or, equivalently,

$$A\mathbf{y} = \sigma\mathbf{x}, \quad A^*\mathbf{x} = \sigma\mathbf{y}.$$

Now, suppose that we apply $\tilde{A}$ to the vector obtained from $\mathbf{z}$ by negating $\mathbf{y}$. Then we have

$$\begin{bmatrix} 0 & A \\ A^* & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ -\mathbf{y} \end{bmatrix} = \begin{bmatrix} -A\mathbf{y} \\ A^*\mathbf{x} \end{bmatrix} = \begin{bmatrix} -\sigma\mathbf{x} \\ \sigma\mathbf{y} \end{bmatrix} = -\sigma \begin{bmatrix} \mathbf{x} \\ -\mathbf{y} \end{bmatrix}.$$

In other words, if $\sigma \neq 0$ is an eigenvalue, then $-\sigma$ is also an eigenvalue.

Suppose that we normalize the eigenvector $\mathbf{z}$ of $\tilde{A}$ so that $\mathbf{z}^*\mathbf{z} = 2$. Since $\tilde{A}$ is symmetric, eigenvectors corresponding to different eigenvalues are orthogonal, so it follows that

$$\begin{bmatrix} \mathbf{x}^* & \mathbf{y}^* \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} = 0.$$

This yields the system of equations

$$\begin{aligned} \mathbf{x}^*\mathbf{x} + \mathbf{y}^*\mathbf{y} &= 2 \\ \mathbf{x}^*\mathbf{x} - \mathbf{y}^*\mathbf{y} &= 0 \end{aligned}$$

which has the unique solution

$$\mathbf{x}^*\mathbf{x} = 1, \quad \mathbf{y}^*\mathbf{y} = 1.$$

From the relationships $A\mathbf{y} = \sigma\mathbf{x}$, $A^*\mathbf{x} = \mathbf{y}$, we obtain

$$A^*A\mathbf{y} = \sigma^2\mathbf{y}, \quad AA^*\mathbf{x} = \sigma^2\mathbf{x}.$$

Therefore, if $\pm\sigma$ are eigenvalues of $\tilde{A}$, then σ^2 is an eigenvalue of both AA^* and A^*A.

To complete the proof, we note that we can represent the matrix of normalized eigenvectors of $\tilde{A}$ corresponding to nonzero eigenvalues as

$$\tilde{Z} = \frac{1}{\sqrt{2}} \begin{bmatrix} X & X \\ Y & -Y \end{bmatrix}.$$

It follows that

$$\begin{aligned}
\tilde{A} &= \tilde{Z}\Lambda\tilde{Z}^* \\
&= \frac{1}{2}\begin{bmatrix} X & X \\ Y & -Y \end{bmatrix}\begin{bmatrix} \Sigma_r & 0 \\ 0 & -\Sigma_r \end{bmatrix}\begin{bmatrix} X^* & Y^* \\ X^* & -Y^* \end{bmatrix} \\
&= \frac{1}{2}\begin{bmatrix} X\Sigma_r & -X\Sigma_r \\ Y\Sigma_r & Y\Sigma_r \end{bmatrix}\begin{bmatrix} X^* & Y^* \\ X^* & -Y^* \end{bmatrix} \\
&= \frac{1}{2}\begin{bmatrix} 0 & 2X\Sigma_r Y^* \\ 2Y\Sigma_r X^* & 0 \end{bmatrix} \\
&= \begin{bmatrix} 0 & X\Sigma_r Y^* \\ Y\Sigma_r X^* & 0 \end{bmatrix}
\end{aligned}$$

and therefore

$$A = X\tilde{\Sigma}Y^*, \quad A^* = Y\Sigma_r$$

where X is an $m \times r$ matrix, Σ is $r \times r$, and Y is $n \times r$, and r is the rank of A. This represents the "condensed" SVD.

2.9 Projections and pseudo-inverses

The singular value decomposition is very useful in studying the linear least squares problem. Suppose that we are given an m-vector $\mathbf{b}$ and an $m \times n$ matrix A, and we wish to find $\mathbf{x}$ such that

$$\|\mathbf{b} - A\mathbf{x}\|_2 = \text{minimum}.$$

From the SVD of A, we can simplify this minimization problem as follows:

$$\begin{aligned}
\|\mathbf{b} - A\mathbf{x}\|_2^2 &= \|\mathbf{b} - U\Sigma V^T\mathbf{x}\|_2^2 \\
&= \|U^T\mathbf{b} - \Sigma V^T\mathbf{x}\|_2^2 \\
&= \|\mathbf{c} - \Sigma\mathbf{y}\|_2^2 \\
&= (c_1 - \sigma_1 y_1)^2 + \cdots + (c_r - \sigma_r y_r)^2 + \\
&\quad c_{r+1}^2 + \cdots + c_m^2
\end{aligned}$$

where $\mathbf{c} = U^T\mathbf{b}$ and $\mathbf{y} = V^T\mathbf{x}$. We see that in order to minimize $\|A\mathbf{x} - \mathbf{b}\|_2$, we must set $y_i = c_i/\sigma_i$ for $i = 1, \ldots, r$, but the unknowns y_i, for $i = r+1, \ldots, m$, can have any value, since they do not influence $\|\mathbf{c} - \Sigma\mathbf{y}\|_2$. Therefore, if A does not have full rank, there are infinitely many solutions to the least squares problem. However, we can easily obtain the unique solution of minimum 2-norm by setting $y_{r+1} = \cdots = y_m = 0$.

In summary, the solution of minimum length to the linear least squares problem is

$$\begin{aligned} \mathbf{x} &= V\mathbf{y} \\ &= V\Sigma^{+}\mathbf{c} \\ &= V\Sigma^{+}U^{T}\mathbf{b} \\ &= A^{+}\mathbf{b} \end{aligned}$$

where Σ^{+} is a diagonal matrix with entries

$$\Sigma^{+} = \begin{bmatrix} \sigma_1^{-1} & & & & & \\ & \ddots & & & & \\ & & \sigma_r^{-1} & & & \\ & & & 0 & & \\ & & & & \ddots & \\ & & & & & 0 \end{bmatrix}$$

and $A^{+} = V\Sigma^{+}U^{T}$. The matrix A^{+} is called the *pseudo-inverse* of A. In the case where A has full rank, the pseudo-inverse is equal to A^{-1}. Note that A^{+} is independent of $\mathbf{b}$.

The solution $\mathbf{x}$ of the least-squares problem minimizes $\|A\mathbf{x} - \mathbf{b}\|$, and therefore is the vector that solves the system $A\mathbf{x} = \mathbf{b}$ as closely as possible. However, we can use the SVD to show that $\mathbf{x}$ is the exact solution to a related system of equations.

We write $\mathbf{b} = \mathbf{b}_1 + \mathbf{b}_2$, where

$$\mathbf{b}_1 = AA^{+}\mathbf{b}, \quad \mathbf{b}_2 = (I - AA^{+})\mathbf{b}.$$

The matrix AA^{+} has the form

$$AA^{+} = U\Sigma V^{T}V\Sigma^{+}U^{T} = U\Sigma\Sigma^{+}U^{T} = U \begin{bmatrix} I_r & 0 \\ 0 & 0 \end{bmatrix}.$$

It follows that $\mathbf{b}_1$ is a linear combination of $\mathbf{u}_1, \ldots, \mathbf{u}_r$, the columns of U that form an orthogonal basis for the range of A. From $\mathbf{x} = A^{+}\mathbf{b}$ we obtain

$$A\mathbf{x} = AA^{+}\mathbf{b} = P\mathbf{b} = \mathbf{b}_1,$$

where $P = AA^{+}$. Therefore, the solution to the least squares problem, is also the exact solution to the system $A\mathbf{x} = P\mathbf{b}$.

It can be shown that the matrix P has the properties

1. $P = P^T$
2. $P^2 = P$

In other words, the matrix P is a *projection*. In particular, it is a projection onto the space spanned by the columns of A, i.e. the range of A.

The following theorem is due to Moore and Penrose:

Theorem. *Let A be a real m-by-n matrix. If thre exists a matrix X such that*

1. $AXA = A$,
2. $XAX = X$,
3. $(AX)^T = AX$,
4. $(XA)^T = XA$,

then X is unique, and $X = A^+$.

One can easily verify that A^+ satisfies properties (1) through (4). Also, if A has full column rank, then $(A^TA)^{-1}A^T$ also satisfies these conditions, so we have

$$A^+ = (A^TA)^{-1}A^T$$

provided $\text{rank}(A) = n$.

2.10 Further applications of the SVD

Using the SVD, we can easily establish a lower bound for the largest single value σ_1 of A, which also happens to be equal to $\|A\|_2$, as previously discussed. First, let us consider the case where A is symmetric and positive definite. Then, we can write $A = U\Lambda U^*$ where U is an orthogonal matrix and Λ is a diagonal matrix with real and positive diagonal elements $\lambda_1 \geq \cdots \geq \lambda_n$ which are the eigenvalues of A. We can then write

$$\max_{x \neq 0} \frac{x^*Ax}{x^*x} = \max_{x \neq 0} \frac{\mathbf{x}^*U\Lambda U^*\mathbf{x}}{\mathbf{x}^*UU^*\mathbf{x}} = \max_{y \neq 0} \frac{\mathbf{y}^*\Lambda\mathbf{y}}{\mathbf{y}^*\mathbf{y}} \leq \lambda_1$$

where $\mathbf{y} = U^*\mathbf{x}$. Now, we consider the case of general A and try to find an upper bound for the expression

$$\max_{\mathbf{u},\mathbf{v} \neq 0} \frac{|\mathbf{u}^*A\mathbf{v}|}{\|\mathbf{u}\|_2\|\mathbf{v}\|_2}.$$

We have, by the Cauchy-Schwarz inequality,

$$\begin{aligned}
\max_{\mathbf{u},\mathbf{v}\neq 0} \frac{|\mathbf{u}^* A \mathbf{v}|}{\|\mathbf{u}\|_2 \|\mathbf{v}\|_2} &= \max_{\mathbf{u},\mathbf{v}\neq 0} \frac{|\mathbf{u}^* U \Sigma V^* \mathbf{v}|}{\|U^*\mathbf{u}\|_2 \|V^*\mathbf{v}\|_2} \\
&= \max_{\mathbf{x},\mathbf{y}\neq 0} \frac{|\mathbf{x}^* \Sigma \mathbf{y}|}{\|\mathbf{x}\|_2 \|\mathbf{y}\|_2} \\
&\leq \sigma_1 \max_{\mathbf{x},\mathbf{y}\neq 0} \frac{|\mathbf{x}^* \mathbf{y}|}{\|\mathbf{x}\|_2 \|\mathbf{y}\|_2} \\
&\leq \sigma_1.
\end{aligned}$$

2.11 Jordan Canonical Form

An $n \times n$ matrix A can be decomposed as

$$A = QJQ^{-1}$$

where the matrix J is a block diagonal matrix

$$J = \begin{bmatrix} J_1 & & \\ & \ddots & \\ & & J_k \end{bmatrix}$$

and each block J_r, for $r = 1, \ldots, k$, has the form

$$J_r = \begin{bmatrix} \lambda_r & 1 & & \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ & & & \lambda_r \end{bmatrix}$$

where J_r is $n_r \times n_r$. This decomposition of A is known as the *Jordan canonical form*.

The Jordan canonical form provides valuable information about the eigenvalues of A. The values λ_j, for $j = 1, \ldots, k$, are the eigenvalues of A. For each distinct eigenvalue λ, the *number* of Jordan blocks having λ as a diagonal element is equal to the number of linearly independent eigenvectors associated with λ. This number is called the *geometric multiplicity* of the eigenvalue λ. The *sum* of the sizes of all of these blocks is called the *algebraic multiplicity* of λ.

We now consider J_r's eigenvalues. We have

$$\lambda(J_r) = \lambda_r, \ldots, \lambda_r$$

where λ_r is repeated n_r times. But, because

$$J_r - \lambda_r I = \begin{bmatrix} 0 & 1 & & \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ & & & 0 \end{bmatrix}$$

is a matrix of rank $n_r - 1$, it follows that the homogeneous system $(J_r - \lambda_r I)\mathbf{x} = \mathbf{0}$ has only one vector (up to a scalar multiple) for a solution, and therefore there is only one eigenvector associated with this Jordan block.

The unique unit vector that solves $(J_r - \lambda_r I)\mathbf{x} = \mathbf{0}$ is the vector $\mathbf{e}_1 = \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}$. Now, consider the matrix

$$\begin{aligned} (J_r - \lambda_r I)^2 &= \begin{bmatrix} 0 & 1 & & \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ & & & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & & \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ & & & 0 \end{bmatrix} \\ &= \begin{bmatrix} 0 & 0 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & 1 \\ & & & \ddots & 0 \\ & & & & 0 \end{bmatrix}. \end{aligned}$$

It is easy to see that $(J_r - \lambda_r I)^2 \mathbf{e}_2 = 0$. Continuing in this fashion, we can conclude that

$$(J_r - \lambda_r I)^k \mathbf{e}_k = \mathbf{0}, \quad k = 1, \ldots, n_r - 1.$$

The Jordan form can be used to easily compute powers of a matrix. For example, one can easily show that

$$A^2 = QJQ^{-1}QJQ^{-1} = QJ^2Q$$

and, in general,

$$A^k = QJ^kQ.$$

Due to its structure, it is easy to compute powers of a Jordan block J_r. We

have

$$\begin{aligned} J_r^k &= \begin{bmatrix} \lambda_r & 1 & & \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ & & & \lambda_r \end{bmatrix}^k \\ &= (\lambda_r I + K)^k, \quad K = \begin{bmatrix} 0 & 1 & & \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ & & & 0 \end{bmatrix} \\ &= \sum_{j=0}^{k} \begin{pmatrix} k \\ j \end{pmatrix} \lambda_r^{k-j} K^j \end{aligned}$$

which yields, for $k > n_r$,

$$J_r^k = \begin{bmatrix} \lambda_r^k & \begin{pmatrix} k \\ 1 \end{pmatrix} \lambda_r^{k-1} & \begin{pmatrix} k \\ 2 \end{pmatrix} \lambda_r^{k-2} & \cdots & \begin{pmatrix} k \\ n_r - 1 \end{pmatrix} \lambda_r^{k-(n_r-1)} \\ & \ddots & \ddots & & \vdots \\ & & \ddots & \ddots & \vdots \\ & & & \ddots & \vdots \\ & & & & \lambda_r^k \end{bmatrix}.$$

For example,

$$\begin{bmatrix} \lambda & 1 & 0 \\ 0 & \lambda & 1 \\ 0 & 0 & \lambda \end{bmatrix}^3 = \begin{bmatrix} \lambda^3 & 3\lambda^2 & 3\lambda \\ 0 & \lambda^3 & 3\lambda^2 \\ 0 & 0 & \lambda^3 \end{bmatrix}.$$

We now consider an application of the Jordan canonical form. Consider the system of differential equations

$$\mathbf{y}'(t) = A\mathbf{y}(t), \quad \mathbf{y}(t_0) = \mathbf{y}_0.$$

Using the Jordan form, we can rewrite this system as

$$\mathbf{y}'(t) = QJQ^{-1}\mathbf{y}(t).$$

Multiplying through by Q^{-1} yields

$$Q^{-1}\mathbf{y}'(t) = JQ^{-1}\mathbf{y}(t),$$

which can be rewritten as

$$\mathbf{z}'(t) = J\mathbf{z}(t),$$

where $\mathbf{z} = Q^{-1}\mathbf{y}(t)$. This new system has the initial condition

$$\mathbf{z}(t_0) = \mathbf{z}_0 = Q^{-1}\mathbf{y}_0.$$

If we assume that J is a diagonal matrix (which is true in the case where A has a full set of linearly independent eigenvectors), then the system decouples into scalar equations of the form

$$z_i'(t) = \lambda_i z_i(t),$$

where λ_i is an eigenvalue of A. This equation has the solution

$$z_i(t) = e^{\lambda_i(t-t_0)} z_i(0),$$

and therefore the solution to the original system is

$$\mathbf{y}(t) = Q \begin{bmatrix} e^{\lambda_1(t-t_0)} & & \\ & \ddots & \\ & & e^{\lambda_n(t-t_0)} \end{bmatrix} Q^{-1}\mathbf{y}_0.$$

Although the Jordan canonical form is a valuable tool in theoretical linear algebra, it is difficult to compute in practice, because the Jordan block structure is very sensitive to perturbations. To see why, consider the matrix

$$A = \begin{bmatrix} 1 & 10^{-6} \\ 0 & 1 \end{bmatrix}.$$

This matrix is not diagonalizable; it is similar to the Jordan block

$$J = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}.$$

However, the nearby matrix

$$\tilde{A} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

is diagonalizable with two 1×1 Jordan blocks. Clearly, the Jordan form is very difficult to compute in the presence of round-off errors.[1]

[1] A related problem is the ill-conditioning of eigenvalue problems involving matrices with multiple eigenvalues. In contrast, the problem of computing singular values is numerically stable, since it comes from the eigenvalues of symmetric matrices.

2.12 Some Results Involving Norms

If $\|A\| < 1$, then $\|A^m\| \to 0$ as $m \to \infty$. Since $\|A\|$ is a continuous function of the elements of A, it follows that $A^m \to 0$. However, if $\|A\| > 1$, it does not follow that $\|A^m\| \to \infty$. For example, suppose

$$A = \begin{bmatrix} 0.99 & 10^6 \\ 0 & 0.99 \end{bmatrix}.$$

In this case, $\|A\|_\infty > 1$, but because $\rho(A) < 1$, there must exist some norm $\|A\|_\alpha$ such that $\|A\|_\alpha < 1$.

For matrices A such that $\|A\| < 1$ for some natural norm, we also have the following result.

Theorem If, for some natural norm, $\|A\| < 1$, then

1. $I - A$ is nonsingular
2. $$\frac{1}{1 + \|A\|} \leq \|(I - A)^{-1}\| \leq \frac{1}{1 - \|A\|}.$$

Proof

1. Assume $I - A$ is singular. Then, there exists a vector $\mathbf{z} \neq \mathbf{0}$ such that $(I - A)\mathbf{z} = \mathbf{0}$. Therefore $\mathbf{z} = A\mathbf{z}$ and

   $$\|\mathbf{z}\| = \|A\mathbf{z}\| \leq \|A\|\|\mathbf{z}\|.$$

   Therefore $\|A\| \geq 1$, which is a contradiction.

2. Since $I = (I - A)(I - A)^{-1}$, we have

   $$\|I\| \leq \|(I - A)\|\|(I - A)^{-1}\|$$

   but since we are using a natural norm, $\|I\| = 1$, so, by the triangle inequality, we have

   $$1 \leq (1 + \|A\|)\|(I - A)^{-1}\|,$$

   thus proving the left inequality. For the right inequality, $(I-A)^{-1}(I-A) = I$ yields

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

or

$$(I - A)^{-1} = I + (I - A)^{-1}A.$$

Taking norms, we obtain

$$\|(I - A)^{-1}\| \leq 1 + \|A\|\|(I - A)^{-1}\|$$

which, by the fact that $\|A\| < 1$, proves the right inequality true.

Chapter 3

Inexact Computation

Computer arithmetic is necessarily inexact, since only a finite amount of memory is available. This chapter presents a model for inexact arithmetic that is appropriate for most architectures, and discusses issues that arise from inexact computation.

3.1 Number representation

Real numbers can be represented using *floating-point* notation: a *floating-point representation* such as

$$y = \pm d_1 \cdots d_s d_{s+1} \cdots 10^e$$

A real number may not necessarily have a unique floating-point representation, as the following examples indicate:

$$\begin{aligned} y &= \pm 0.899 \cdots 9 \cdots \times 10^0 \\ &= \pm 0.90 \cdots 0 \cdots \times 10^0 \end{aligned}$$

or even worse,

$$\begin{aligned} y &= +0.99 \cdots 9 \times 10^0 \\ &= +0.10 \cdots 0 \times 10^1 \end{aligned}$$

How can we represent y? We can use a *chopped representation*

$$\tilde{y} = \pm .d_1 \cdots d_s \times 10^3$$

or possibly

$$\tilde{y} = \pm .d_1 \cdots d_{s-1} \bar{d}_s \times 10^{\bar{e}}$$

where

$$\bar{d}_s = \begin{cases} d_s & d_{s+1} < 5 \\ (d_s + 1) \bmod 9 & d_{s+1} > 5 \\ ? & d_{s+1} = 5 \end{cases}$$

where the value of $\bar{d}_s$ when $d_s = 5$ depends on what convention is used for rounding. Note that e can change if $d_{s+1} > 5$. We often write $\tilde{y} = fl(y)$ where $fl(\cdot)$ means "floating-point representation of y".

Floating point numbers can be represented in base β as

$$y = \pm .d_1 \cdots d_s \times \beta^e$$

where $m \le e \le M$ and the digits d_j satisfy

$$1 \le d_1 \le \beta - 1, \quad 0 \le d_j \le \beta - 1, \quad j = 2, \ldots, s.$$

This is a *normalized floating-point number*. The sequence of *significant* digits $d_1 \cdots d_s$ is called the *mantissa*, and the number e is called the *exponent*.

Suppose $s = 1$, $m = -1$, and $M = 1$. Then the representable numbers are

$$\begin{array}{cccc} +0.1 \times 10^{-1} & +0.2 \times 10^{-1} & \cdots & +0.9 \times 10^{-1} \\ +0.1 \times 10^{0} & +0.2 \times 10^{0} & \cdots & +0.9 \times 10^{0} \\ +0.1 \times 10^{1} & +0.2 \times 10^{1} & \cdots & +0.9 \times 10^{1} \end{array}$$

and 27 negative numbers. Note that the distribution is not uniform. In the previous example, the representable numbers are

$$-9, -8, \ldots, -1, -0.9, -0.8, \ldots, -0.1, -0.09, \ldots, -0.01, 0, 0.1, \ldots$$

Note that there are large gaps between integers.

Now, suppose that $s = 3$, and that we multiply two numbers $y_1 = 0.999$ and $y_2 = 0.999$. The exact product is $y_1 \times y_2 = 0.998001$, but the computed product is $fl(y_1 \times y_2) = 0.998$. We will write $fl(y_1 \, op \, y_2)$ to indicate some numerical calculation.

On the IBM 360, we have base 16 (hexadecimal). In general,

$$\begin{aligned} y &= (\pm .d_1 \cdots d_s)_\beta \times \beta^e, \quad \beta = 16 \\ &= \pm \left(\frac{d_1}{\beta} + \frac{d_2}{\beta^2} + \cdots + \frac{d_s}{\beta_s} \right) \times \beta^3 \end{aligned}$$

with $1 \le d_1 \le \beta - 1$ and $0 \le d_j \le \beta_1$ for $2 \le j \le s$. For the IBM 360, floating-point numbers are represented using $s = 6$, $m = -64$ and $M = 63$, while double-precision floating-point numbers are represented using $s = 14$, $m = -64$, and $M = 63$. If, for the result of any operation, $e < -64$, then *underflow* has occurred. On the other hand, the scenario $e > 63$ is called *overflow*.

3.2 IEEE floating point numbers

Notes in this section are due to Lieven Vandenberghe of UCLA.

3.2.1 Floating point numbers with base 10

Notation:

$$x = \pm(.d_1 d_2 \cdots d_n)_{10} \cdot 10^e$$

- $.d_1 d_2 \cdots d_n$ is the *mantissa* (d_i integer, $0 \leq d_i \leq 9$, $d_1 \neq 0$ if $x \neq 0$)
- n is the *mantissa length* (or *precision*)
- e is the *exponent* ($e_{\min} \leq e \leq e_{\max}$)

Interpretation: $x = \pm(d_1 10^{-1} + d_2 10^{-2} + \cdots + d_n 10^{-n}) \cdot 10^e$

Example (with $n = 7$):

$$\begin{aligned} 12.625 &= +(.1262500)_{10} \cdot 10^2 \\ &= +(1 \cdot 10^{-1} + 2 \cdot 10^{-2} + 6 \cdot 10^{-3} + 2 \cdot 10^{-4} + 5 \cdot 10^{-5} \\ &\quad +0 \cdot 10^{-6} + 0 \cdot 10^{-7}) \cdot 10^2 \end{aligned}$$

used in pocket calculators

Properties

- a finite set of numbers
- unequally spaced distance between floating point numbers varies
  - the smallest number greater than 1 is $1 + 10^{-n+1}$
  - the smallest number greater than 10 is $10 + 10^{-n+2}, \ldots$
- largest positive number:

$$+(.999 \cdots 9)_{10} \cdot 10^{e_{\max}} = (1 - 10^{-n}) 10^{e_{\max}}$$

- smallest positive number:

$$x_{\min} = +(.100 \cdots 0)_{10} \cdot 10^{e_{\min}} = 10^{e_{\min}-1}$$

3.2.2 Floating point numbers with base 2

Notation:

$$x = \pm(.d_1 d_2 \cdots d_n)_2 \cdot 2^e$$

- $.d_1 d_2 \cdots d_n$ is the *mantissa* ($d_i \in \{0, 1\}$, $d_1 = 1$ if $x \neq 0$)
- n is the *mantissa length* (or *precision*)
- e is the *exponent* ($e_{\min} \leq e \leq e_{\max}$)

Interpretation: $x = \pm(d_1 2^{-1} + d_2 2^{-2} + \cdots + d_n 2^{-n}) \cdot 2^e$

Example (with $n = 8$):

$$\begin{aligned} 12.625 &= +(.11001010)_2 \cdot 2^4 \\ &= +(1 \cdot 2^{-1} + 1 \cdot 2^{-2} + 0 \cdot 2^{-3} + 0 \cdot 2^{-4} + 1 \cdot 2^{-5} \\ &\quad +0 \cdot 2^{-6} + 1 \cdot 2^{-7} + 0 \cdot 2^{-8}) \cdot 2^4 \end{aligned}$$

used in almost all computers

a finite set of unequally spaced numbers

- largest positive number:

$$x_{\max} = +(.1 \cdots 1)_2 \cdot 2^{e_{\max}} = (1 - 2^{-n}) 2^{e_{\max}}$$

- smallest positive number:

$$x_{\min} = +(.100 \cdots 0)_2 \cdot 2^{e_{\min}} = 2^{e_{\min} - 1}$$

- in practice, the number system includes *subnormal numbers*: unnormalized small numbers ($d_1 = 0, e = e_{\min}$), and the number 0

3.2.3 IEEE floating point standard

specifies two binary floating point number formats

IEEE standard single precision:

$$n = 24, \quad e_{\min} = -125, \quad e_{\max} = 128$$

requires 32 bits: 1 sign bit, 23 bits for mantissa, 8 bits for exponent

IEEE standard double precision:

$$n = 53, \quad e_{\min} = -1021, \quad e_{\max} = 1024$$

requires 64 bits: 1 sign bit, 52 bits for mantissa, 11 bits for exponent

used in almost all modern computers

3.2.4 Machine precision

Definition: the *machine precision* of a binary floating point number system with mantissa length n is defined as

$$\epsilon_M = 2^{-n}$$

Example: IEEE standard double precision ($n = 53$):

$$\epsilon_M = 2^{-53} \approx 1.1102 \cdot 10^{-16}$$

Interpretation: $1 + 2\epsilon_M$ is the smallest floating point number greater than 1:

$$(.10\cdots01)_2 \cdot 2^1 = 1 + 2^{1-n} = 1 + 2\epsilon_M$$

3.2.5 Rounding error

a floating-point number system is a finite set of numbers; all other numbers must be rounded

Notation: $fl(x)$ is the floating-point representation of x

Rounding rules used in practice:

- numbers are rounded to the nearest floating-point number
- in case of a tie: round to the number with least significant bit 0 ("round to nearest even")

Example: numbers $x \in (1, 1 + 2\epsilon_M)$ are rounded to 1 or $1 + 2\epsilon_M$:

- $fl(x) = 1$ if $1 < x \leq 1 + \epsilon_M$
- $fl(x) = 1 + 2\epsilon_M$ if $1 + \epsilon_M < x < 1 + 2\epsilon_M$

gives another interpretation of ϵ_M: numbers between 1 and $1 + \epsilon_M$ are indistinguishable from 1

3.2.6 Rounding error and machine precision

General result:

$$\frac{|fl(x) - x|}{|x|} \leq \epsilon_M$$

(no proof)

- machine precision gives a bound on the relative error due to rounding
- number of correct (decimal) digits in $fl(x)$ is roughly

  $$-\log_{10} \epsilon_M$$

  i.e., about 15 or 16 in IEEE precision
- fundamental limit on accuracy of numerical computation

3.3 Roundoff Error in Arithmetic Operations

We need to consider the error arising from computing the results of arithmetic expressions. In general,

$$z = fl(x \, op \, y) = (x \, op \, y)(1 + \delta)$$

where $|\delta| \leq \beta^{-(s-1)} \equiv \mathbf{u}$ when results are truncated, or $|\delta| \leq \frac{1}{2}\beta^{-(s-1)}$ when results are rounded. Therefore the relative error in $fl(x \, op \, y)$ is

$$\left| \frac{fl(x \, op \, y) - (x \, op \, y)}{x \, op \, y} \right| = |\delta| \leq \mathbf{u}$$

where $\mathbf{u}$ is known as the *unit roundoff*.

We can use this idea sequentially to bound the error in more complex computations. Suppose we want to compute

$$s_n = x_1 + x_2 + \cdots + x_n.$$

If we use the numerical algorithm

$$\begin{aligned} s_0 &= 0 \\ s_1 &= s_0 + x_1 \\ s_2 &= s_1 + x_2 \\ &\vdots \\ s_n &= s_{n-1} + x_n \end{aligned}$$

the corresponding computer algorithm is

$$\begin{aligned}
\sigma_0 &= 0 \\
\sigma_1 &= fl(\sigma_0 + x_1) \\
\sigma_2 &= fl(\sigma_1 + x_2) \\
&\vdots \\
\sigma_n &= fl(\sigma_{n-1} + x_n)
\end{aligned}$$

Expanding the expressions for the σ_i, we obtain

$$\begin{aligned}
\sigma_1 &= fl(\sigma_0 + x_1) = (\sigma_0 + x_1)(1 + \delta_1) \\
\sigma_2 &= fl(\sigma_1 + x_2) = (\sigma_1 + x_2)(1 + \delta_2) \\
&= \sigma_1(1 + \delta_2) + x_2(1 + \delta_2) \\
&= (\sigma_1 + x_1)(1 + \delta_1)(1 + \delta_2) + x_2(1 + \delta_2) \\
&\vdots \\
\sigma_n &= x_1(1 + \delta_1)\cdots(1 + \delta_n) + x_2(1 + \delta_2)\cdots(1 + \delta_n) + \\
&\quad \cdots + x_n(1 + \delta_n) \\
&= \sum_{j=1}^{n} x_j \prod_{k=j}^{n}(1 + \delta_k) \\
&= \sum_{j=1}^{n} x_j(1 + \eta_j), \quad (1 + \eta_j) = \prod_{k=j}^{n}(1 + \delta_k) \\
&= x_1 + \cdots + x_n + \sum_{j=1}^{n} \eta_j x_j
\end{aligned}$$

In summary,

$$\left| \sigma_n - \sum_{i=1}^{n} x_i \right| \leq \sum_{j=1}^{n} |\eta_j||x_j|, \quad |\eta_j| \leq (n - j + 1)\mathbf{u} + O(\mathbf{u})^2$$

if $0 < x_{i_1} \leq x_{i_2} \leq \cdots \leq x_{i_n}$. It would seem to imply that we should add the numbers in that order since the numbers are weighted. A better procedure is to add the numbers in pairs, resulting in the bound

$$|1 + \eta_i| \leq 1 + p\mathbf{u}$$

where $n = 2^p$. In this case, the error is uniformly distributed.

For the product of n numbers $p = x_1 \cdots x_n$, we obtain the computed results

$$\begin{aligned}
\Pi_1 &= fl(x_1 x_2) = (x_1 x_2)(1+\epsilon_1) \\
\Pi_2 &= fl(\Pi_2 x_3) = (x_1 x_2 x_3)(1+\epsilon_1)(1+\epsilon_2) \\
&\vdots \\
\Pi_n &= fl(\Pi_{n-1} x_n) \\
&= (x_1 \cdots x_n)(1+\epsilon_1)\cdots(1+\epsilon_{n-1}) \\
&= p(1+\eta_{n-1})
\end{aligned}$$

where

$$|\eta_{n-1}| \leq (n-1)\mathbf{u} + O(\mathbf{u}^2).$$

3.4 Common Computations

It is frequently necessary to compute the *inner product*

$$s = \sum_{i=1}^{n} u_i v_i$$

Now

$$fl(u_i \times v_i) = (u_i \times v_i)(1+\gamma_i), \quad |\gamma_i| \leq \mathbf{u}.$$

Therefore, if we wish to compute

$$fl(s) = \sum_{j=1}^{n} x_j y_j (1+\eta_j)$$

and $(1+\eta_j)$ will depend on how the computation is performed. If we do serial addition, then

$$(1+\eta_j) = \prod_{k=j}^{n} (1+\delta_k)(1+\gamma_j).$$

If we do pairwise addition and $n = 2^p$, then

$$(1+\eta_j) = \left(\sum_{k=1}^{p} (1+\delta_{i_{k_j}}) \right) (1+\gamma_j).$$

In computing inner products one must be quite careful; otherwise the underflow or overflow problem can be quite serious.

Suppose one wishes to compute

$$s = \sum_{i=1}^{n} x_i^2.$$

It is quite possible that overflow will occur. One solution is to scale. Suppose

$$|x_{\pm}| \geq |x_j|, \quad j = 1, 2, \ldots, n$$

and $x_{\pm} = \alpha\beta^p$. Then one should compute

$$s = \left(\sum_{i=1}^{n} \left(\frac{x_i}{\beta_p} \right)^2 \right) \beta^{2p}.$$

This requires two passes; there are better ways of performing the calculation.

We can build up analysis of many problems. This has been done extensively for linear algebra by J. H. Wilkinson. The analysis can be automated; R. Moore has used interval arithmetic for determining bounds on the solution.

It is a general principle that one should add terms of a similar sign. Suppose we wish to compute

$$s^2 = \sum_{i=1}^{n} (x_i - \bar{x})^2, \quad \bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i. \tag{3.1}$$

It is well known that

$$s^2 = \sum_{i=1}^{n} x_i^2 - n\bar{x}^2 \tag{3.2}$$

and this formula is frequently used. But it is very bad, especially if the x_i's are large but s^2 is small. Then, you can be subtracting two large numbers. Formula (3.1) is more stable but requires two passes. First, you must compute the mean and then the quantity s^2. We could try writing

$$s^2 = \sum_{i=1}^{n} (x_i - \nu)^2 + \eta(\nu^2 - \bar{x}^2)$$

where ν is some approximation to $\bar{x}$.

3.5 Issues with Floating-point Arithmetic

We conclude our discussion of floating-point arithmetic by highlighting two issues that frequently arise in practice.

First of all, relationships among numbers that are known to be true in exact arithmetic do not necessarily hold when using floating-point arithmetic. For example, suppose that $x > y > 0$, and that $a > 0$. Then, in exact arithmetic, $ax > ay > 0$, but in floating-point arithmetic, we can only assume that $ax \geq ay \geq 0$.

Second, the order in which floating-point arithmetic operations are performed can drastically affect the result. For example, suppose that we want to compute e^{-x}, where $x > 0$. Using the Taylor series for e^x, we know that

$$e^{-x} = 1 - x + \frac{x^2}{2} - \frac{x^3}{3!} + \cdots$$

but for sufficiently large x, this means obtaining small numbers by subtracting larger ones, and therefore the computation is susceptible to a phenomenon known as *catastrophic cancellation*, in which subtracting numbers that are nearly equal causes the loss of significant digits (since such digits in the result of the subtraction are equal to zero). An alternative approach is to compute

$$e^{-x} = \frac{1}{1 + x + \frac{x^2}{2} + \frac{x^3}{3!} + \cdots}$$

which avoids this problem.

As a rule, it is best to try to avoid subtractions altogether, instead trying to add numbers that are guaranteed to be the same sign. For example, given the quadratic equation $x^2 + bx + c = 0$, we can compute the roots by applying the quadratic formula as follows:

$$x_+ = \frac{-b + \text{sign}(-b)\sqrt{b^2 - 4c}}{2}, \quad x_- = \frac{c}{x_+}.$$

Chapter 4

Direct methods for the solutions of linear systems

4.1 Direct vs. Iterative methods

Modern algorithms for solving linear systems $Ax = b$ are largely divided into two categories:

- *Direct methods* are based on explicit manipulation of the entries in A. Most involve factoring A into a product of "simpler" matrices (e.g. triangular or orthogonal matrices). In exact arithmetic, direct methods yield the exact solution in a finite number of steps (which is usually $O(n^3)$ unless sparsity structure is exploited).

- *Iterative methods* attempt to generate a sequence of approximations $x^{(i)}$ which converges to the true solution x. Such algorithms are not expected to yield the exact solution at any finite step k (although some do; such methods are said to have "finite termination property"). In general, iterative methods do not attempt to manipulate the entries of A directly, but instead rely on performing matrix-vector products Av, which makes them especially useful when A is not stored explicitly (*e.g.* matrices that arise from the discretization of a differential operator). Convergence behavior (and hence the algorithmic complexity) tends to be highly dependent on the properties of the matrix itself.

In this chapter, we focus our study on direct methods. We will consider iterative methods in Chapter 6.

4.2 Perturbation Theory for linear systems

Suppose we want to solve

$$A\mathbf{x} = \mathbf{b}.$$

Because of inexact arithmetic, we actually have an approximation ξ such that

$$\mathbf{x} = \xi + \mathbf{e}.$$

The question is, how can we use norms to bound the relative error in ξ? We define the *residual* $\mathbf{r}$ by

$$\begin{aligned} \mathbf{r} &= \mathbf{b} - A\xi \\ &= A(\mathbf{x} - \xi) \\ &= A\mathbf{e} \end{aligned}$$

Note that $\mathbf{r} = \mathbf{0}$ if $A\mathbf{x} = \mathbf{b}$. From the relations $\|\mathbf{r}\| \leq \|A\|\|\mathbf{e}\|$ and $\|\mathbf{e}\| \leq \|A^{-1}\|\|\mathbf{r}\|$, we obtain

$$\frac{\|\mathbf{r}\|}{\|A\|} \leq \|\mathbf{e}\| \leq \|A^{-1}\|\|\mathbf{r}\|.$$

It follows that the relative error is bounded as follows:

$$\frac{\|\mathbf{r}\|}{\|A\|\|\mathbf{x}\|} \leq \frac{\|\mathbf{e}\|}{\|\mathbf{x}\|} \leq \frac{\|A^{-1}\|\|\mathbf{r}\|}{\|\mathbf{x}\|} \leq \|A^{-1}\|\|A\|\frac{\|\mathbf{r}\|}{\|\mathbf{b}\|}$$

since $\|A\|\|\mathbf{x}\| \geq \|\mathbf{b}\|$. The quantity

$$\kappa(A) = \|A^{-1}\|\|A\|$$

is called the *condition number* of A. The condition number serves as a measure of how perturbation in the data of the problem $A\mathbf{x} = \mathbf{b}$ affects the solution.

How does a perturbation in A affect the solution $\mathbf{x}$? To answer this question, we define $A(\epsilon)$ to be a function of the size of the perturbation ϵ, with $A(0) = A$. Starting with

$$A(\epsilon)A^{-1}(\epsilon) = I$$

and differentiating with respect to epsilon yields

$$A(\epsilon)\frac{dA^{-1}(\epsilon)}{d\epsilon} + \frac{dA(\epsilon)}{d\epsilon}A^{-1}(\epsilon) = 0$$

or

$$\frac{dA^{-1}(\epsilon)}{d\epsilon} = -A^{-1}(\epsilon)\frac{dA(\epsilon)}{d\epsilon}A^{-1}(\epsilon).$$

Now, suppose that $\mathbf{x}(\epsilon)$ satisfies

$$(A + \epsilon E)\mathbf{x}(\epsilon) = \mathbf{b}.$$

Using Taylor series, we obtain

$$\begin{aligned}
\mathbf{x}(\epsilon) &= (A + \epsilon E)^{-1}\mathbf{b} \\
&= \mathbf{x}(0) + \epsilon \left.\frac{d\mathbf{x}(\epsilon)}{d\epsilon}\right|_{\epsilon=0} + O(\epsilon^2) \\
&= \mathbf{x}(0) + \epsilon \left(-A^{-1}(\epsilon)\frac{dA(\epsilon)}{d\epsilon}A^{-1}(\epsilon)\right)\mathbf{b} + O(\epsilon^2) \\
&= \mathbf{x}(0) + \epsilon(-A^{-1}EA^{-1})\mathbf{b} + O(\epsilon^2) \\
&= \mathbf{x}(0) + \epsilon(-A^{-1}E)\mathbf{x} + O(\epsilon^2)
\end{aligned}$$

Taking norms, we obtain

$$\|\mathbf{x}(\epsilon) - \mathbf{x}(0)\| \leq |\epsilon|\|A^{-1}\|^2\|E\|\|\mathbf{b}\| + O(\epsilon^2)$$

from which it follows that the relative perturbation in $\mathbf{x}$ is

$$\begin{aligned}
\frac{\|\mathbf{x}(\epsilon) - \mathbf{x}\|}{\|\mathbf{x}\|} &\leq |\epsilon|\|A^{-1}\|\|A\|\frac{\|E\|}{\|A\|} + O(\epsilon^2) \\
&\leq \kappa(A)\rho + O(\epsilon^2)
\end{aligned}$$

where $\rho = \|\epsilon E\|/\|A\|$ is the relative perturbation in A.

Since the exact solution to $A\mathbf{x} = \mathbf{b}$ is given by $\mathbf{x} = A^{-1}\mathbf{b}$, we are also interested in examining $(A + E)^{-1}$ where E is some perturbation. Can we say something about $\|(A+E)^{-1} - A^{-1}\|$? We assume that $\|A^{-1}E\| = r < 1$. We have

$$\begin{aligned}
A + E &= A(I + A^{-1}E) \\
&= A(I - F), \quad F = -A^{-1}E
\end{aligned}$$

which yields

$$\|(I - F)^{-1}\| \leq \frac{1}{1 - r}.$$

From

$$\begin{aligned}
(A + E)^{-1} - A^{-1} &= (I + A^{-1}E)^{-1}A^{-1} - A^{-1} \\
&= (I + A^{-1}E)(A^{-1} - (I + A^{-1}E)A^{-1}) \\
&= (I + A^{-1}E)^{-1}(-A^{-1}EA^{-1})
\end{aligned}$$

we obtain

$$\|(A+E)^{-1} - A^{-1}\| \leq \frac{1}{1-r}\|A^{-1}\|^2\|E\|$$

or

$$\frac{\|(A+E)^{-1} - A^{-1}\|}{\|A^{-1}\|} \leq \frac{1}{1-r}\kappa(A)\frac{\|E\|}{\|A\|}.$$

4.3 A Special Case: Rank-1 Updates and the Inverse

Suppose that we know how to solve the problem $A\mathbf{x} = \mathbf{b}$, but we now wish to solve the perturbed problem

$$(A + \mathbf{u}\mathbf{v}^T)\mathbf{y} = \mathbf{b}.$$

Such a perturbation is called a *rank-one update* of A, since the matrix $\mathbf{u}\mathbf{v}^T$ has rank 1. As an example, we might find that there was an error in the element a_{11} and we update it with the value $\bar{a}_{11}$. We can accomplish this update by setting

$$\bar{A} = A + (\bar{a}_{11} - a_{11})\mathbf{e}_1\mathbf{e}_1^T, \quad \mathbf{e}_1 = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

For a general rank-one update, we can use the *Sherman-Morrison formula*, which we will derive here. Multiplying through the equation $(A+\mathbf{u}\mathbf{v}^T)\mathbf{y} = \mathbf{b}$ by A^{-1} yields

$$(I + A^{-1}\mathbf{u}\mathbf{v}^T)\mathbf{y} = A^{-1}\mathbf{b} = \mathbf{x}.$$

We therefore need to find $(I + \mathbf{w}\mathbf{v}^T)^{-1}$ where $\mathbf{w} = A^{-1}\mathbf{u}$. We assume that $(I + \mathbf{w}\mathbf{v}^T)^{-1}$ is a matrix of the form $(I + \sigma\mathbf{w}\mathbf{v}^T)$ where σ is some constant. From the relationship

$$(I + \mathbf{w}\mathbf{v}^T)(I + \sigma\mathbf{w}\mathbf{v}^T) = I$$

we obtain

$$\sigma\mathbf{w}\mathbf{v}^T + \mathbf{w}\mathbf{v}^T + \sigma\mathbf{w}\mathbf{v}^T\mathbf{w}\mathbf{v}^T = 0.$$

However, the quantity $\mathbf{v}^T\mathbf{w}$ is a scalar, so this simplifies to

$$(\sigma + 1 + \sigma\mathbf{v}^T\mathbf{w})\mathbf{w}\mathbf{v}^T = 0$$

which yields

$$\sigma = -\frac{1}{1+\mathbf{v}^T\mathbf{w}}.$$

It follows that the solution $\mathbf{y}$ to the perturbed problem is given by

$$\mathbf{y} = (I + \sigma\mathbf{w}\mathbf{v}^T)\mathbf{x} = \mathbf{x} + \sigma\mathbf{v}^{\mathbf{x}}\mathbf{w}$$

and the perturbed inverse is given by

$$\begin{aligned}(A+\mathbf{u}\mathbf{v}^T)^{-1} &= (I + A^{-1}\mathbf{u}\mathbf{v}^T)^{-1}A \\ &= \left(I - \frac{1}{1+\mathbf{v}^T\mathbf{w}}\mathbf{w}\mathbf{v}^T\right)A^{-1} \\ &= A^{-1} - \frac{1}{1+\mathbf{v}^TA^{-1}\mathbf{u}}A^{-1}\mathbf{u}\mathbf{v}^TA^{-1}.\end{aligned}$$

An efficient algorithm for solving the perturbed problem $(A+\mathbf{u}\mathbf{v}^T)\mathbf{y} = \mathbf{b}$ can therefore proceed as follows:

1. Solve $A\mathbf{x} = \mathbf{b}$
2. Solve $A\mathbf{w} = \mathbf{u}$
3. Compute $\sigma = -\frac{1}{1+\mathbf{v}^T\mathbf{w}}$
4. Compute $\mathbf{y} = \mathbf{x} + \sigma(\mathbf{v}^T\mathbf{x})\mathbf{w}$

An alternative approach is to note that

$$\begin{aligned}(A+\mathbf{u}\mathbf{v}^T)^{-1} &= [A(I + A^{-1}\mathbf{u}\mathbf{v}^T)]^{-1} \\ &= (I + \sigma A^{-1}\mathbf{u}\mathbf{v}^T)A^{-1} \\ &= A^{-1} + \sigma A^{-1}\mathbf{u}\mathbf{v}^TA^{-1}\end{aligned}$$

which yields

$$\begin{aligned}(A+\mathbf{u}\mathbf{v}^T)^{-1}\mathbf{b} &= A^{-1}(I + \sigma\mathbf{u}\mathbf{v}^TA^{-1})\mathbf{b} \\ &= A^{-1}(\mathbf{b} + \sigma(\mathbf{v}^TA^{-1}\mathbf{b})\mathbf{u})\end{aligned}$$

and therefore we can solve $(A+\mathbf{u}\mathbf{v}^T)\mathbf{y} = \mathbf{b}$ by solving a problem of the form $A\mathbf{x} = \mathbf{b}$ where the right-hand side $\mathbf{b}$ is perturbed.

4.4 Gaussian Elimination and the LU Factorization

4.4.1 Basic Algorithm

We often wish to solve

$$A\mathbf{x} = \mathbf{b}$$

where A is an $m \times n$ matrix and $\mathbf{b}$ is an m-vector. For now, we assume that $m = n$ and that A has rank n. If we can write

$$A = PQ$$

then we can solve the system $A\mathbf{x} = PQ\mathbf{x} = \mathbf{b}$ by solving

$$P\mathbf{y} = \mathbf{b}$$
$$Q\mathbf{x} = \mathbf{y}$$

Therefore we would like to find such a decomposition where the above systems are simple to solve. We now discuss a few scenarios where this is the case.

1. If the matrix A is diagonal, then the system $A\mathbf{x} = \mathbf{b}$ has the solution $x_i = b_i/a_{ii}$ for $i = 1, \ldots, n$. The solution can be computed in only n divisions. Furthermore, each component of $\mathbf{x}$ can be computed independently, and therefore the algorithm can be parallelized.

2. If $AA^T = I$, then $A\mathbf{x} = \mathbf{b}$ can be solved simply by computing the matrix-vector product $\mathbf{x} = A^T\mathbf{b}$. This requires only $O(n^2)$ multiplications and additions, and can also be parallelized.

3. If A is a lower triangular matrix, i.e. if $a_{ij} = 0$ for $i < j$, then the system of equations $A\mathbf{x} = \mathbf{b}$ takes the form

$$\begin{array}{ccccccc} a_{11}x_1 & & & & & = & b_1 \\ a_{12}x_1 & + & a_{22}x_2 & & & = & b_2 \\ & & & & & \vdots & \\ a_{n1}x_1 & + & \cdots & + & a_{nn}x_n & = & b_n \end{array}$$

which can be solved by the process of *forward substitution*

$$\begin{aligned} x_1 &= b_1/a_{11} \\ x_2 &= (b_2 - a_{21}x_1)/a_{22} \\ &\vdots \\ x_n &= \left(b_n - \sum_{j=1}^{n-1} a_{nj}x_j \right) /a_{nn} \end{aligned}$$

This algorithm cannot be parallelized, since each component x_i depends on x_j for $j < i$, but it is still efficient, as it requires only $O(n^2)$ multiplications and additions. In the case where A is an *upper triangular matrix*, i.e. $a_{ij} = 0$ whenever $i > j$, a similar process known as *back substitution* can be used.

Note that the solution method for the problem $A\mathbf{x} = \mathbf{b}$ depends on the structure of A. A may be a sparse or dense matrix, or it may have one of many well-known structures, such as being a banded matrix, or a Hankel matrix. We will consider the general case of a dense, unstructured matrix A, and obtain a decomposition $A = LU$, where L is lower triangular and U is upper triangular.

This decomposition is achieved using Gaussian elimination. We write out the system $Ax = b$ as

$$\begin{array}{ccccccccc} a_{11}x_1 & + & \cdots & + & a_{1n}x_n & = & b_1 \\ a_{21}x_1 & + & \cdots & + & a_{2n}x_n & = & b_2 \\ \vdots & + & \cdots & + & \vdots & \vdots & \vdots \\ a_{n1}x_1 & + & \cdots & + & a_{nn}x_n & = & b_n \end{array}$$

We proceed by multiplying the first equation by $-a_{21}/a_{11}$ and adding it to the second equation, and in general multiplying the first equation by $-a_{i1}/a_{11}$ and adding it to equation i. We obtain the following equivalent system

$$\begin{array}{ccccccccccc} a_{11}x_1 & + & a_{12}x_2 & + & \cdots & + & a_{1n}x_n & = & b_1 \\ 0x_1 & + & a'_{22}x_2 & + & \cdots & + & a'_{2n}x_n & = & b'_2 \\ \vdots & + & \vdots & + & \cdots & + & \vdots & \vdots & \vdots \\ 0x_1 & + & a'_{n2}x_2 & + & \cdots & + & a'_{nn}x_n & = & b'_n \end{array}$$

Continuing in this fashion, adding multiples of the second equation to each subsequent equation to make all elements below the diagonal equal to zero, we obtain an upper triangular system.

This process of transforming A to an upper triangular matrix U is equivalent to multiplying A by a sequence of matrices to obtain U. Specifically, we have $M_1 A = A_2$ where

$$A_2 = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ 0 & a_{22}^{(2)} & \cdots & a_{2n}^{(2)} \\ \vdots & \vdots & & \vdots \\ 0 & a_{n2}^{(2)} & \cdots & a_{nn}^{(2)} \end{bmatrix}$$

and

$$M_1 = \begin{bmatrix} 1 & & & 0 \\ -\ell_{21} & 1 & & \\ \vdots & 0 & \ddots & \\ -\ell_{n1} & & & 1 \end{bmatrix}, \quad \ell_{i1} = \frac{a_{i1}}{a_{11}}.$$

Similarly, if we define M_2 by

$$M_2 = \begin{bmatrix} 1 & & & & \\ 0 & 1 & & & \\ 0 & -\ell_{32} & 1 & & \\ \vdots & \vdots & & \ddots & \\ 0 & -\ell_{n2} & & & 1 \end{bmatrix}, \quad \ell_{i2} = \frac{a_{i2}^{(2)}}{a_{22}^{(2)}}$$

then

$$M_2 A_2 = A_3 = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ 0 & a_{22}^{(2)} & a_{23}^{(2)} & \cdots & a_{2n}^{(2)} \\ 0 & 0 & a_{33}^{(3)} & \cdots & a_{3n}^{(3)} \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & a_{n3}^{(3)} & \cdots & a_{nn}^{(3)} \end{bmatrix}$$

In general, we have

$$M_k = \begin{bmatrix} 1 & & & & & \\ 0 & \ddots & & & & \\ \vdots & \ddots & 1 & & & \\ \vdots & & -\ell_{k+1,k} & 1 & & \\ \vdots & & \vdots & & \ddots & \\ 0 & & -\ell_{nk} & & & 1 \end{bmatrix}$$

and

$$M_{n-1}M_{n-2}\cdots M_1 A = A_n \equiv \begin{bmatrix} u_{11} & \cdots & & u_{1n} \\ 0 & u_{22} & \cdots & u_{2n} \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & 0 & u_{nn} \end{bmatrix}$$

or, equivalently,

$$A = M_1^{-1}M_2^{-1}\cdots M_{n-1}^{-1}U.$$

It turns out that M_j^{-1} is very easy to compute. We claim that

$$M_1^{-1} = \begin{bmatrix} 1 & & & 0 \\ \ell_{21} & 1 & & \\ \vdots & 0 & \ddots & \\ \ell_{n1} & & & 1 \end{bmatrix}$$

To see this, consider the product

$$M_1 M_1^{-1} = \begin{bmatrix} 1 & & & 0 \\ -\ell_{21} & 1 & & \\ \vdots & 0 & \ddots & \\ -\ell_{n1} & & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & 0 \\ \ell_{21} & 1 & & \\ \vdots & 0 & \ddots & \\ \ell_{n1} & & & 1 \end{bmatrix}$$

which can easily be verified to be equal to the identity matrix. In general, we have

$$M_k^{-1} = \begin{bmatrix} 1 & & & & & \\ 0 & \ddots & & & & \\ \vdots & \ddots & 1 & & & \\ \vdots & & \ell_{k+1,k} & 1 & & \\ \vdots & & \vdots & & \ddots & \\ 0 & & \ell_{nk} & & & 1 \end{bmatrix}$$

Now, consider the product

$$M_1^{-1}M_2^{-1} = \begin{bmatrix} 1 & & & 0 \\ \ell_{21} & 1 & & \\ \vdots & 0 & \ddots & \\ \ell_{n1} & & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & & \\ 0 & 1 & & & \\ 0 & \ell_{32} & 1 & & \\ \vdots & \vdots & & \ddots & \\ 0 & \ell_{n2} & & & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & & & & \\ \ell_{21} & 1 & & & \\ \vdots & \ell_{32} & 1 & & \\ \vdots & \vdots & & \ddots & \\ \ell_{n1} & \ell_{n2} & & & 1 \end{bmatrix}$$

It can be shown that

$$M_1^{-1}M_2^{-1}\cdots M_{n-1}^{-1} = \begin{bmatrix} 1 & & & & \\ \ell_{21} & \ddots & & & \\ \vdots & \ell_{32} & \ddots & & \\ \vdots & \vdots & \ddots & \ddots & \\ \ell_{n1} & \ell_{n2} & \cdots & \ell_{n,n-1} & 1 \end{bmatrix}$$

It follows that under proper circumstances, we can write $A = LU$ where

$$L = \begin{bmatrix} 1 & & & & \\ \ell_{21} & \ddots & & & \\ \vdots & \ell_{32} & \ddots & & \\ \vdots & \vdots & \ddots & \ddots & \\ \ell_{n1} & \ell_{n2} & \cdots & \ell_{n,n-1} & 1 \end{bmatrix}, \quad U = \begin{bmatrix} u_{11} & \cdots & & u_{1n} \\ 0 & u_{22} & \cdots & u_{2n} \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & 0 & u_{nn} \end{bmatrix}$$

Given this decomposition, we can easily compute the determinant of A:

$$\det A = \det LU = \det L \det U = 1 \cdot \prod_{i=1}^{n} u_{ii}$$

What exactly are proper circumstances? We must have $a_{kk}^{(k)} \neq 0$, or we cannot proceed with the decomposition. For example, if

$$A = \begin{bmatrix} 0 & 1 & 11 \\ 3 & 7 & 2 \\ 2 & 9 & 3 \end{bmatrix} \quad \text{or} \quad A = \begin{bmatrix} 1 & 3 & 4 \\ 2 & 6 & 4 \\ 7 & 1 & 2 \end{bmatrix}$$

Gaussian elimination will fail. In the first case, it fails immediately; in the second case, it fails after the subdiagonal entries in the first column are zeroed, and we find that $a_{22}^{(k)} = 0$. In general, we must have $\det A_{ii} \neq 0$ for $i = 1, \ldots, n$ where

$$A_{ii} = \begin{bmatrix} a_{11} & \cdots & a_{1i} \\ \vdots & & \vdots \\ a_{i1} & \cdots & a_{ii} \end{bmatrix}$$

for the LU factorization to exist.

4.4.2 Pivoting

How can we obtain the LU factorization for a general non-singular matrix? If A is nonsingular, then *some* element of the first column must be nonzero. If $a_{i1} \neq 0$, then we can interchange row i with row 1 and proceed. This is equivalent to multiplying A by a *permutation matrix* Π_1 that interchanges row 1 and row i:

$$\Pi_1 = \begin{bmatrix} 0 & \cdots & \cdots & \cdots & 1 & 0 & \cdots & 0 \\ & 1 & & & & & & \\ & & \ddots & & & & & \\ & & & 1 & & & & \\ 1 & 0 & \cdots & \cdots & 0 & 0 & \cdots & 0 \\ & & & & & 1 & & \\ & & & & & & \ddots & \\ 0 & & & & & & & 1 \end{bmatrix}$$

Thus $M_1 \Pi_1 A = A_2$. Then, since A_2 is nonsingular, some element of column 2 of A_2 below the diagonal must be nonzero. Proceeding as before, we compute $M_2 \Pi_2 A_2 = A_3$, where Π_2 is another permutation matrix. Continuing, we obtain

$$A = (M_{n-1} \Pi_{n-1} \cdots M_1 \Pi_1)^{-1} U.$$

It can easily be shown that $\Pi A = LU$ where Π is a permutation matrix.

Most often, Π_i is chosen so that row i is interchanged with row j, where $a_{ij}^{(i)} = \max_{i \leq j \leq n} |a_{ij}^{(i)}|$. This guarantees that $|\ell_{ij}| \leq 1$. This strategy is known as *partial pivoting*. Another common strategy, *complete pivoting*, uses both row and column interchanges to ensure that at step i of the algorithm, the element a_{ii} is the largest element in absolute value from the entire submatrix obtained by deleting the first $i-1$ rows and columns. Often, however, other criteria is used to guide pivoting, due to considerations such as preserving sparsity.

4.4.3 Uniqueness of the LU Decomposition

It is natural ask whether the LU decomposition is unique. To determine this, we assume that A has two LU decompositions, $A = L_1U_1$ and $A = L_2U_2$. From $L_1U_1 = L_2U_2$ we obtain $L_2^{-1}L_1 = U_2U_1^{-1}$. The inverse of a unit lower triangular matrix is a unit lower triangular matrix, and the product of two unit lower triangular matrices is a unit lower triangular matrix, so $L_2^{-1}L_1$ must be a unit lower triangular matrix. Similarly, $U_2U_1^{-1}$ is an upper triangular matrix. The only matrix that is both upper triangular and unit lower triangular is the identity matrix I, so we must have $L_1 = L_2$ and $U_1 = U_2$.

4.4.4 Computing the Inverse

Using the LU decomposition, one can compute the inverse of a matrix. A natural method to compute the inverse of an $n \times n$ matrix A is to solve the matrix equation

$$AX = I$$

by solving the systems of equations

$$A\mathbf{x}_j = \mathbf{e}_j, \quad j = 1, \ldots, n.$$

Since only the right-hand side is different in each of these systems, we need only compute the LU decomposition of A once, which requires $2n^3/3$ operations. For simplicity, we assume that pivoting is not required, and note that the case where pivoting is required can be handled in a similar fashion.

Given the LU decomposition, we compute A^{-1} by solving the systems $L\mathbf{y}_j = \mathbf{e}_j$ and $U\mathbf{x}_j = \mathbf{y}_j$ for $j = 1, \ldots, n$. Computing each column $\mathbf{x}_j$ of A^{-1} requires n^2 operations, resulting in a total of $2n^3/3 + n(n^2) = 5n^3/3$ operations.

We can compute A^{-1} more efficiently by noting that $A^{-1} = U^{-1}L^{-1}$ and computing U^{-1}, L^{-1}, and the product $U^{-1}L^{-1}$ directly. Since the inverse of an upper triangular matrix is also an upper triangular matrix, computing column j of U^{-1} requires only approximately $j^2/2$ operations, since, in solving the system $U\mathbf{x}_j = \mathbf{e}_j$, we can ignore the last $n - j$ components of $\mathbf{x}_j$ since we know that they are equal to zero. As a result, computing U^{-1} requires only $n^3/6$ operations. A similar result holds for computing L^{-1}, which is a lower triangular matrix.

To compute the product $A^{-1} = U^{-1}L^{-1}$, we note that if we number the northeast-to-southwest diagonals of A^{-1} starting with 1 for the upper left diagonal (the (1,1) element) and n for the lower right diagonal (the (n, n)

element), then elements along diagonal j require only $n-j+1$ multiplications to compute. It follows that the total operation count to compute the product of U^{-1} and L^{-1} is

$$(2n-1) + 2(2n-3) + 3(2n-5) + \cdots + (n-1)2 + n \approx \frac{1}{3}n^3.$$

Therefore, the overall operation count to compute A^{-1} using this method is $4n^3/3$.

4.4.5 Gauss-Jordan Elimination

A variant of Gaussian elimination is called *Gauss-Jordan elimination*. It entails zeroing elements above the diagonal as well as below, using elementary column operations that are similar to the elementary row operations used in Gaussian elimination. The result is a decomposition $A = LDM^T$, where L is a unit lower triangular matrix, D is a diagonal matrix, and M is also a unit lower triangular matrix. We can then solve the system $A\mathbf{x} = \mathbf{b}$ by solving the systems

$$L\mathbf{y} = \mathbf{b}, \quad D\mathbf{z} = \mathbf{y}, \quad M^T\mathbf{x} = \mathbf{z}.$$

The benefit of Gauss-Jordan elimination is that the diagonal system $D\mathbf{z} = \mathbf{y}$ can be solved in parallel, since the elements of $\mathbf{z}$ can be computed independently. The drawback is that the elimination process can be numerically unstable, since the multipliers can be large.

4.4.6 Parallelism of Gaussian Elimination

Suppose that we wish to perform Gaussian elimination on the matrix $A = \begin{bmatrix} \mathbf{a}_1 & \cdots & \mathbf{a_n} \end{bmatrix}$. During the first step of the elimination, we compute

$$P^{(1)}\Pi_1 A = \begin{bmatrix} P^{(1)}\Pi_1\mathbf{a}_1 & \cdots & P^{(1)}\Pi_1\mathbf{a}_n \end{bmatrix}.$$

Clearly we can work on each column independently, leading to a parallel algorithm. As the elimination proceeds, we obtain less benefit from parallelism since fewer columns are being modified at each step.

4.5 The Cholesky Decomposition

4.5.1 Positive Definite Matrices

A matrix A is *positive definite* if $\mathbf{x}^T A\mathbf{x} > 0$ for all nonzero $\mathbf{x}$. A positive definite matrix has real and positive eigenvalues, and its leading principal

submatrices all have positive determinants. From the definition, it is easy to see that all diagonal elements are positive.

To solve the system $A\mathbf{x} = \mathbf{b}$ where A is positive definite, we can compute the *Cholesky decomposition* $A = F^T F$ where F is upper triangular. This decomposition exists if and only if A is symmetric and positive definite. In fact, attempting to compute the Cholesky decomposition of A is an efficient method for checking whether A is symmetric positive definite. There are several ways to write $A = GG^T$ for some matrix G since

$$A = FF^T = FQQ^T F = (FQ)(FQ)^T = GG^T$$

for any orthogonal matrix Q, but for the Cholesky decomposition, we require that F is lower triangular, with positive diagonal elements.

The Cholesky decomposition is also called the *square root factorization*, although it is important to note that the matrix F in $A = F^T F$ is not the square root of A, since it does not hold that $F^2 = A$ unless A is a diagonal matrix. The square root of A can be computed by using the fact that A has the decomposition $A = U\Lambda U^T$ where Λ is a diagonal matrix whose diagonal elements are the eigenvalues of A and U is an orthogonal matrix whose columns are the eigenvectors of A. It follows that

$$A = U\Lambda U^T = (U\Lambda^{1/2}U^T)(U\Lambda^{1/2}U^T) = SS$$

where $S = U\Lambda^{1/2}U^T$ is the square root of A.

We can compute F by examining the matrix equation $A = FF^T$ on an element-by-element basis, writing

$$\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ a_{21} & \cdots & a_{2n} \\ \vdots & & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix} = \begin{bmatrix} f_{11} & & & \\ f_{21} & f_{22} & & \\ \vdots & \vdots & \ddots & \\ f_{n1} & f_{n2} & \cdots & f_{nn} \end{bmatrix} \begin{bmatrix} f_{11} & f_{21} & \cdots & f_{n1} \\ & f_{22} & & \vdots \\ & & \ddots & \vdots \\ & & & f_{nn} \end{bmatrix}.$$

From the above matrix multiplication we see that $f_{11}^2 = a_{11}$, from which it follows that

$$f_{11} = \sqrt{a_{11}}.$$

From the relationship $f_{11}f_{i1} = a_{i1}$ and the fact that we already know f_{11}, we obtain

$$f_{i1} = \frac{a_{i1}}{f_{11}}, \quad i = 2, \ldots, n.$$

Proceeding to the second column of F, we see that $f_{21}^2 + f_{22}^2 = a_{22}$. Since we already know f_{21}, we have

$$f_{22} = \sqrt{a_{22} - f_{21}^2}.$$

Next, we use the relation $f_{21}f_{i1} + f_{22}f_{i2} = a_{2i}$ to compute

$$f_{i1} = \frac{a_{2i} - f_{21}f_{i1}}{f_{22}}.$$

In general, we can use the relationship $a_{ij} = \mathbf{f}_i^T\mathbf{f}_j$ to compute f_{ij}, where $\mathbf{f}_i$ is the ith column of F:

For $k = 1, \ldots, n$:

$$\begin{aligned} f_{kk} &= \left(a_{kk} - \sum_{j=1}^{k-1} f_{jk}^2 \right)^{1/2} \\ f_{kj} &= \left(a_{kj} - \sum_{\ell=1}^{k-1} f_{\ell k} f_{\ell j} \right) \Big/ f_{kk}, \quad j = k+1, \ldots, n \end{aligned}$$

This algorithm requires roughly half as many operations as Gaussian elimination.

Another method for computing the Cholesky decomposition is to compute

$$\mathbf{f}_1 = \frac{1}{\sqrt{a_{11}}}\mathbf{a}_1$$

where $\mathbf{a}_i$ is the ith column of A. Then we set $A^{(1)} = A$ and compute

$$A^{(2)} = A^{(1)} - \mathbf{f}_1\mathbf{f}_1^T = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 0 & & & \\ \vdots & & A_2 & \\ 0 & & & \end{bmatrix}.$$

We partition the matrix A_2 into columns, writing $A_2 = \begin{bmatrix} \mathbf{a}_2^{(2)} & \mathbf{a}_3^{(2)} & \cdots & \mathbf{a}_n^{(2)} \end{bmatrix}$ and then compute

$$\mathbf{f}_2 = \frac{1}{\sqrt{a_{22}^{(2)}}} \begin{bmatrix} 0 \\ \mathbf{a}_2^{(2)} \end{bmatrix}.$$

We then compute

$$A_3 = A^{(2)} - \mathbf{f}_2\mathbf{f}_2^T$$

and so on.

Note that

$$a_{kk} = f_{k1}^2 + f_{k2}^2 + \cdots + f_{kk}^2,$$

which implies that

$$|f_{ki}| \leq |a_{kk}|.$$

In other words, the elements of F are bounded. We also have the relationship

$$\det A = \det F \det F^T = (\det F)^2 = f_{11}^2 f_{22}^2 \cdots f_{nn}^2.$$

4.5.2 Uniqueness of the Cholesky Factorization

Is the Cholesky decompositon unique? Employing a similar approach to the one used to prove the uniquess of the LU decomposition, we assume that A has two Cholesky decompositions

$$A = F_1 F_1^T = F_2 F_2^T.$$

Then

$$F_2^{-1} F_1 = F_2^T F_1^{-T},$$

but since F_1 and F_2 are lower triangular, both matrices must be diagonal. Furthermore, they must have diagonal elements equal to ± 1. Since we require that the diagonal elements be positive, it follows that the decomposition is unique.

In computing the Cholesky decomposition, no row interchanges are necessary because A is positive definite, so the number of operations required to compute F is approximately $n^3/3$.

A variant of the Cholesky decomposition is known as the *square-root-free Cholesky decomposition*, and has the form

$$A = LDL^T$$

where L is a unit lower triangular matrix, and D is a diagonal matrix with positive diagonal elements. This is a special case of the $A = LDM^T$ factorization previously discussed. The LDL^T and Cholesky decompositions are related by

$$F = LD^{1/2}.$$

4.6 Banded Matrices

A *banded* matrix has all of its nonzero elements contained within a "band" consisting of select diagonals. Specifically, a matrix A that has *upper bandwidth* p and *lower bandwidth* q has the form

$$A = \begin{bmatrix} a_{11} & \cdots & a_{1,p+1} & & \\ a_{21} & \ddots & a_{2,p+1} & a_{2,p+2} & \\ \vdots & & & & \\ a_{q+1,1} & \cdots & a_{q+1,q+1} & \cdots & a_{q+1,n} \\ & & \ddots & \ddots & \vdots \end{bmatrix}.$$

Matrices of this form arise frequently from discretization of partial differential equations.

The simplest banded matrix is a *tridiagonal* matrix, which has upper bandwidth 1 and lower bandwidth 1. Such a matrix can be stored using only three vectors instead of a two-dimensional array. Computing the LU decomposition of a tridiagonal matrix without pivoting requires only $O(n)$ operations, and produces bidiagonal L and U. When pivoting is used, this desirable structure is lost, and the process as a whole is more expensive in terms of computation time and storage space.

Various applications, such as the solution of partial differential equations in two or more space dimensions, yield symmetric *block tridiagonal* matrices, which have a block Cholesky decomposition:

$$\begin{bmatrix} A_1 & B_2^T & & \\ B_2 & \ddots & \ddots & \\ & \ddots & \ddots & B_n^T \\ & & B_n & A_n \end{bmatrix} = \begin{bmatrix} F_1 & & & \\ G_2 & \ddots & & \\ & \ddots & \ddots & \\ & & G_n & F_n \end{bmatrix} \begin{bmatrix} F_1^T & G_2^T & & \\ & \ddots & \ddots & \\ & & \ddots & G_n^T \\ & & & F_n^T \end{bmatrix}.$$

From the above matrix equation, we determine that

$$A_1 = F_1 F_1^T, \quad B_2 = G_2 F_1^T$$

from which it follows that we can compute the Cholesky decomposition of A_1 to obtain F_1, and then compute $G_2 = B_2(F_1^T)^{-1}$. Next, we use the relationship $A_2 = G_2 G_2^T + F_2 F_2^T$ to obtain

$$F_2 F_2^T = A_2 - G_2 G_2^T = A_2 - B_2(F_1^T)^{-1} F_1^{-1} B_2^T = A_2 - B_2 A_1^{-1} B_2.$$

It is interesting to note that in the case of $n = 2$, the matrix $A_2 - B_2 A_1^{-1} B_2$. is known as the *Schur complement* of A_1.

Continuing with the block tridiagonal case with $n = 2$, suppose that we wish to compute the factorization

$$\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} = \begin{bmatrix} F \\ G \end{bmatrix} \begin{bmatrix} F^T & G^T \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & X \end{bmatrix}.$$

It is easy to see that $X = -B^T A^{-1} B$, but this matrix is *negative definite*. Therefore, we cannot compute a block Cholesky decomposition, but we can achieve the factorization

$$\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} = \begin{bmatrix} F & 0 \\ G & K \end{bmatrix} \begin{bmatrix} F^T & G^T \\ 0 & -K^T \end{bmatrix}$$

where K is the Cholesky factor of the positive definite matrix $B^T A^{-1} B$.

4.7 Error Analysis of Gaussian Elimination

4.7.1 Condition Numbers and Error Bounds

Suppose that we wish to solve the system $A\mathbf{x} = \mathbf{b}$. Our computed solution $\tilde{\mathbf{x}}$ satisfies a perturbed system $(A + \Delta)\tilde{\mathbf{x}} = \mathbf{b}$. It can be shown that

$$\begin{aligned} \frac{\|\mathbf{x} - \tilde{\mathbf{x}}\|}{\|\mathbf{x}\|} &\leq \frac{\|A^{-1}\|\|\Delta\|}{1 - \|A^{-1}\|\|\Delta\|} \\ &\leq \frac{\|A\|\|A^{-1}\|\frac{\|\Delta\|}{\|A\|}}{1 - \|A\|\|A^{-1}\|\frac{\|\Delta\|}{\|A\|}} \\ &\leq \frac{\kappa(A) r}{1 - \kappa(A) r} \end{aligned}$$

where $\kappa(A) = \|A\|\|A^{-1}\|$ is the *condition number* of A and $r = \|\Delta\|/\|A\|$. The condition number has the following properties:

- $\kappa(\alpha A) = \kappa(A)$ where α is a nonzero scalar.
- $\kappa(I) = 1$
- $\kappa(Q) = 1$ when $Q^T Q = I$.

The perturbation matrix Δ is typically a function of the algorithm used to solve $A\mathbf{x} = \mathbf{b}$.

In this section, we will consider the case of Gaussian elimination and perform a detailed error analysis, illustrating the analysis originally carried out by J.H. Wilkinson. The process of solving $A\mathbf{x} = \mathbf{b}$ consists of three stages:

1. Factoring $A = LU$, resulting in an approximate LU decomposition $A + E = \bar{L}\bar{U}$. We assume that partial pivoting is used.

2. Solving $L\mathbf{y} = \mathbf{b}$, or, numerically, computing $\mathbf{y}$ such that

$$(\bar{L} + \delta\bar{L})(\mathbf{y} + \delta\mathbf{y}) = \mathbf{b}$$

3. Solving $U\mathbf{x} = \mathbf{y}$, or, numerically, computing $\mathbf{x}$ such that

$$(\bar{U} + \delta\bar{U})(\mathbf{x} + \delta\mathbf{x}) = \mathbf{y} + \delta\mathbf{y}.$$

Combining these stages, we see that

$$\begin{aligned}
\mathbf{b} &= (\bar{L} + \delta\bar{L})(\bar{U} + \delta\bar{U})(\mathbf{x} + \delta\mathbf{x}) \\
&= (\bar{L}\bar{U} + \delta\bar{L}\bar{U} + \bar{L}\delta\bar{U} + \delta\bar{L}\delta\bar{U})(\mathbf{x} + \delta\mathbf{x}) \\
&= (A + E + \delta\bar{L}\bar{U} + \bar{L}\delta\bar{U} + \delta\bar{L}\delta\bar{U})(\mathbf{x} + \delta\mathbf{x}) \\
&= (A + \Delta)(\mathbf{x} + \delta\mathbf{x})
\end{aligned}$$

where $\Delta = E + \delta\bar{L}\bar{U} + \bar{L}\delta\bar{U} + \delta\bar{L}\delta\bar{U}$.

In this analysis, we will view the computed solution $\bar{\mathbf{x}} = \mathbf{x} + \delta\mathbf{x}$ as the exact solution to the perturbed problem $(A + \Delta)\mathbf{x} = \mathbf{b}$. This perspective is the idea behind *backward error analysis*, which we will use to determine the size of the perturbation Δ, and, eventually, arrive at a bound for the error in the computed solution $\bar{\mathbf{x}}$.

4.7.2 Error in the LU Factorization

Let $A^{(k)}$ denote the matrix A after $k-1$ steps of Gaussian elimination have been performed, where a step denotes the process of making all elements below the diagonal within a particular column equal to zero. Then, in exact arithmetic, the elements of $A^{(k+1)}$ are given by

$$a_{ij}^{(k+1)} = a_{ij}^{(k)} - m_{ik}a_{kj}^{(k)}, \quad m_{ik} = \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}}. \tag{4.1}$$

Let $B^{(k)}$ denote the matrix A after $k-1$ steps of Gaussian elimination have been performed using floating-point arithmetic. Then the elements of $B^{(k+1)}$ are

$$b_{ij}^{(k+1)} = a_{ij}^{(k)} - s_{ik} b_{kj}^{(k)} + \epsilon_{ij}^{(k+1)}, \quad s_{ik} = \frac{b_{ik}^{(k)}}{b_{kk}^{(k)}} + \cdots \tag{4.2}$$

For $j \geq i$, we have

$$\begin{aligned} b_{ij}^{(2)} &= b_{ij}^{(1)} - s_{i1} b_{1j}^{(1)} + \epsilon_{ij}^{(2)} \\ b_{ij}^{(3)} &= b_{ij}^{(2)} - s_{i1} b_{1j}^{(2)} + \epsilon_{ij}^{(3)} \\ &\vdots \\ b_{ij}^{(i)} &= b_{ij}^{(i-1)} - s_{i1} b_{1j}^{(i-1)} + \epsilon_{ij}^{(i)} \end{aligned}$$

Combining these equations yields

$$\sum_{k=2}^{i} b_{ij}^{(k)} = \sum_{k=1}^{i-1} b_{ij}^{(k)} - \sum_{k=1}^{i-1} s_{ik} b_{kj}^{(k)} + \sum_{k=2}^{i} \epsilon_{ij}^{(k)}$$

Cancelling terms, we obtain

$$b_{ij}^{(1)} = b_{ij}^{(i)} + \sum_{k=1}^{i-1} s_{ik} b_{kj}^{(k)} + e_{ij}, \quad j \geq i \tag{4.3}$$

For $i > j$,

$$\begin{aligned} b_{ij}^{(2)} &= b_{ij}^{(1)} - s_{i1} b_{1j}^{(1)} + \epsilon_{ij}^{(2)} \\ &\vdots \\ b_{ij}^{(j)} &= b_{ij}^{(j-1)} - s_{i1} b_{j-1,j}^{(j-1)} + \epsilon_{ij}^{(j)} \end{aligned}$$

where $s_{ij} = b_{ij}^{(j)} / b_{jj}^{(j)} + \eta_{ij}$, and therefore

$$\begin{aligned} 0 &= b_{ij}^{(j)} - s_{ij} b_{jj}^{(j)} + b_{jj}^{(j)} \eta_{ij} \\ &= b_{ij}^{(j)} - s_{ij} b_{jj}^{(j)} + \epsilon_{ij}^{(j+1)} \\ &= b_{ij}^{(1)} - \sum_{k=1}^{j} s_{ik} b_{kj}^{(k)} + e_{ij} \end{aligned} \tag{4.4}$$

From (4.3) and (4.4), we obtain

$$\bar{L}\bar{U} = \begin{bmatrix} 1 & & & \\ s_{21} & 1 & & \\ \vdots & & \ddots & \\ s_{n1} & \cdots & \cdots & 1 \end{bmatrix} \begin{bmatrix} b_{11}^{(1)} & b_{12}^{(1)} & \cdots & b_{1n}^{(1)} \\ & \ddots & & \vdots \\ & & \ddots & \vdots \\ & & & b_{nn}^{(n)} \end{bmatrix} = A + E.$$

where

$$s_{ik} = fl(b_{ik}^{(k)}/b_{kk}^{(k)}) = \frac{b_{ik}^{(k)}}{b_{kk}^{(k)}}(1+\eta_{ik}), \quad |\eta_{ik}| \leq \mathbf{u}$$

Then,

$$fl(s_{ik}b_{kj}^{(k)}) = s_{ik}b_{kj}^{(k)}(1+\theta_{ij}^{(k)}), \quad |\theta_{ij}^{(k)}| \leq \mathbf{u}$$

and so,

$$\begin{aligned} b_{ij}^{(k+1)} &= fl(b_{ij}^{(k)} - s_{ik}b_{kj}^{(k)}(1+\theta_{ij}^{(k)})) \\ &= (b_{ij}^{(k)} - s_{ik}b_{kj}^{(k)}(1+\theta_{ij}^{(k)}))(1+\varphi_{ij}^{(k)}), \quad |\varphi_{ij}^{(k)}| \leq \mathbf{u}. \end{aligned}$$

After some manipulations, we obtain

$$\epsilon_{ij}^{(k+1)} = b_{ij}^{(k+1)}\left(\frac{\varphi_{ij}^{(k)}}{1+\varphi_{ij}^{(k)}}\right) - s_{ik}b_{kj}^{(k)}\theta_{ij}^{(k)}.$$

With partial pivoting, $|s_{ik}| \leq 1$, provided that $|fl(a/b)| \leq 1$ whenever $|a| \leq |b|$. In most modern implementations of floating-point arithmetic, this is in fact the case. It follows that

$$|\epsilon_{ij}^{(k+1)}| \leq |b_{ij}^{(k+1)}|\frac{\mathbf{u}}{1-\mathbf{u}} + 1 \cdot |b_{ij}^{(k)}|\mathbf{u}.$$

How large can the elements of $B^{(k)}$ be? Returning to exact arithmetic, we assume that $|a_{ij}| \leq a$ and from (4.1), we obtain

$$\begin{aligned} |a_{ij}^{(2)}| &\leq |a_{ij}^{(1)}| + |a_{kj}^{(1)}| \leq 2a \\ |a_{ij}^{(3)}| &\leq 4a \\ &\vdots \\ |a_{ij}^{(n)}| &= |a_{nn}^{(n)}| \leq 2^{n-1}a. \end{aligned}$$

We can show that a similar result holds in floating-point arithmetic:

$$|b_{ij}^{(k)}| \leq 2^{k-1}a + O(\mathbf{u}).$$

This upper bound is achievable, but in practice it rarely occurs.

For complete pivoting, Wilkinson gave a bound, denoted G, or *growth factor*. Until 1990, it was conjectured that $G \leq k$. It was shown to be true for $n \leq 5$, but there have been examples constructed for $n > 5$ where $G \geq n$. In any event, we have the following bound for the entries of E:

$$|E| \leq 2\mathbf{u}Ga \begin{bmatrix} 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ 1 & \cdots & \cdots & \cdots & \cdots & 1 \\ 1 & 2 & \cdots & \cdots & \cdots & 2 \\ \vdots & \vdots & 3 & \cdots & \cdots & 3 \\ & & & \ddots & \cdots & \vdots \\ 1 & 2 & 3 & \cdots & n-1 & n-1 \end{bmatrix} + O(\mathbf{u}^2).$$

4.7.3 Error Analysis of Forward Substitution

We now study the process of forward substitution, to solve

$$\begin{bmatrix} t_{11} & & 0 \\ \vdots & \ddots & \\ t_{n1} & & t_{nn} \end{bmatrix} \begin{bmatrix} u_1 \\ \vdots \\ u_n \end{bmatrix} = \begin{bmatrix} h_1 \\ \vdots \\ h_n \end{bmatrix}.$$

Using forward substitution, we obtain

$$\begin{aligned} u_1 &= h_1/t_{11} \\ &\vdots \\ u_k &= \frac{h_k - t_{k1}u_1 - \cdots - t_{k,k-1}u_{k-1}}{t_{kk}} \end{aligned}$$

which yields

$$fl(u_k) = \frac{h_k(1+\epsilon_k)(1+\eta_k) - \sum_{i=1}^{k-1} t_{ki}u_i(1+\xi_{ki})(1+\epsilon_k)(1+\eta_k)}{t_{kk}} = \frac{h_k - \sum_{i=1}^{k-1} t_{ki}u_i(1+\xi_{ki})}{\frac{t_{kk}}{(1+\epsilon_k)(1+\eta_k)}}$$

or

$$\sum_{i=1}^{k} u_i t_{ki}(1+\lambda_{ki}) = h_k$$

which can be rewritten in matrix notation as

$$T\mathbf{u} + \begin{bmatrix} \lambda_{11}t_{11} & & \\ \lambda_{12}t_{12} & \lambda_{22}t_{22} & \\ \vdots & \vdots & \ddots \end{bmatrix} \mathbf{u} = \mathbf{h}.$$

In other words, the computed solution $\mathbf{u}$ is the exact solution to the perturbed problem $(T + \delta T)\mathbf{u} = \mathbf{h}$, where

$$|\delta T| \leq \mathbf{u} \begin{bmatrix} |t_{11}| & & & \\ |t_{21}| & 2|t_{22}| & & \\ \vdots & & \ddots & \\ (n-1)|t_{n1}| & \cdots & \cdots & 2|t_{nn}| \end{bmatrix} + O(\mathbf{u}^2).$$

Note that the perturbation δT actually depends on $\mathbf{h}$.

4.7.4 Bounding the perturbation in A

Recall that our computed solution $\mathbf{x} + \delta\mathbf{x}$ solves

$$(A + \delta A)\bar{\mathbf{x}} = \mathbf{b}$$

where δA is a perturbation that has the form

$$\delta A = E + \bar{L}\delta\bar{U} + \delta\bar{L}\bar{U} + \delta\bar{L}\delta\bar{U}.$$

For partial pivoting, $|\bar{l}_{ij}| \leq 1$, and we have the bounds

$$\begin{aligned} \max_{i,j} |\delta\bar{L}_{ij}| &\leq n\mathbf{u} + O(\mathbf{u}^2), \\ \max_{i,j} |\delta\bar{U}_{ij}| &\leq n\mathbf{u}Ga + O(\mathbf{u}^2) \end{aligned}$$

were $a = \max_{i,j} |a_{ij}|$ and G is the growth factor for partial pivoting. Putting our bounds together, we have

$$\begin{aligned} \max_{i,j} |\delta A_{ij}| &\leq \max_{i,j} |e_{ij}| + \max_{i,j} |\bar{L}\delta\bar{U}_{ij}| + \max_{i,j} |\bar{U}\delta\bar{L}_{ij}| + \max_{i,j} |\delta\bar{L}\delta\bar{U}_{ij}| \\ &\leq 2\mathbf{u}Gan + n^2Ga\mathbf{u} + n^2Ga\mathbf{u} + O(\mathbf{u}^2) \end{aligned}$$

from which it follows that

$$\|\delta A\|_\infty \leq 2n^2(n+1)\mathbf{u}Ga + O(\mathbf{u}^2).$$

We conclude that Gaussian elimination is *backward stable*.

4.7.5 Bounding the error in the solution

Let $\bar{\mathbf{x}} = \mathbf{x} + \delta\mathbf{x}$ be the computed solution. Then, from $(A + \delta A)\bar{\mathbf{x}} = \mathbf{b}$ we obtain

$$\delta A\bar{\mathbf{x}} = \mathbf{b} - A\bar{\mathbf{x}} = \mathbf{r}$$

where $\mathbf{r}$ is called the *residual vector*. From our previous analysis,

$$\frac{\|\mathbf{r}\|_\infty}{\|\bar{\mathbf{x}}\|_\infty} \leq \|\delta A\|_\infty \leq 2n^2(n+1)Ga\mathbf{u}.$$

Also, recall

$$\frac{\|\delta\mathbf{x}\|}{\|\mathbf{x}\|} \leq \frac{\kappa(A)}{1 - \kappa(A)\frac{\|\delta A\|}{\|A\|}} \frac{\|\delta A\|}{\|A\|}.$$

We know that $\|A\|_\infty \leq na$, so

$$\frac{\|\delta A\|_\infty}{\|A\|_\infty} \leq 2n(n+1)G\mathbf{u}.$$

Note that if $\kappa(A)$ is large and G is large, our solution can be very inaccurate. The important factors in the accuracy of the computed solution are:

- The growth factor G
- The condition number κ
- The accuracy $\mathbf{u}$

In particular, κ must be large with respect to the accuracy in order to be troublesome. For example, consider the scenario where $\kappa = 10^2$ and $\mathbf{u} = 10^{-3}$, as opposed to the case where $\kappa = 10^2$ and $\mathbf{u} = 10^{-50}$.

4.8 Improving the accuracy of solutions

4.8.1 Iterative Refinement

The process of *iterative refinement* proceeds as follows to find a solution to $A\mathbf{x} = \mathbf{b}$:

$$\begin{aligned} \mathbf{x}^{(0)} &= \mathbf{0} \\ \mathbf{r}^{(i)} &= \mathbf{b} - A\mathbf{x}^{(i)} \\ A\delta^{(i)} &= \mathbf{r}^{(i)} \\ \mathbf{x}^{(i+1)} &= \mathbf{x}^{(i)} + \delta^{(i)} \end{aligned}$$

Numerically, this translates to

$$\begin{aligned}(A+\delta A^{(i)})\delta^{(i)} &= (I+E^{(i)})\mathbf{r}^{(i)}\\ \mathbf{x}^{(i+1)} &= (I+F^{(i)})(\mathbf{x}^{(i)}+\delta^{(i)})\end{aligned}$$

where the matrices $E^{(i)}$ and $F^{(i)}$ denote roundoff error. Let $\mathbf{z}^{(i)} = \mathbf{x} - \mathbf{x}^{(i)}$. Then

$$\begin{aligned}\mathbf{x}^{(i+1)} - \mathbf{x} &= (I+F^{(i)})(\mathbf{x}^{(i)}+\delta^{(i)}) - \mathbf{x}\\ &= (I+F^{(i)})(\mathbf{x}^{(i)}-\mathbf{x}) + F^{(i)}\mathbf{x} + (I+F^{(i)})\delta^{(i)}\\ &= (I+F^{(i)})(-\mathbf{z}^{(i)} + (I+A^{-1}\delta A^{(i)})^{-1}\mathbf{z}^{(i)} +\\ &\quad (I+A^{-1}\delta A^{(i)})^{-1}(A^{-1}E^{(i)}A)\mathbf{z}^{(i)}) + F^{(i)}\mathbf{x}\\ &= (I+F^{(i)})(I+A^{-1}\delta A^{(i)})^{-1}(A^{-1}\delta A^{(i)}\mathbf{z}^{(i)} + A^{-1}E^{(i)}A\mathbf{z}^{(i)}) + F^{(i)}\mathbf{x}\end{aligned}$$

which we rewrite as

$$\mathbf{z}^{(i+1)} = K^{(i)}\mathbf{z}^{(i)} + \mathbf{c}^{(i)}$$

Taking norms yields

$$\|\mathbf{z}^{(i+1)}\| \le \|K^{(i)}\|\|\mathbf{z}^{(i)}\| + \|\mathbf{c}^{(i)}\|.$$

Under the assumptions

$$\|K^{(i)}\| \le \tau, \quad \|\mathbf{c}^{(i)}\| \le \sigma\|\mathbf{x}\|$$

we obtain

$$\begin{aligned}\|\mathbf{z}^{(i+1)}\| &\le \tau\|\mathbf{z}^{(i)}\| + \sigma\|\mathbf{x}\|\\ &\le \tau^{i+1}\|\mathbf{z}^{(0)}\| + \sigma(1+\tau+\cdots+\tau^{i})\|\mathbf{x}\|\\ &\le \tau^{i+1}\|\mathbf{z}^{(0)}\| + \sigma\frac{1-\tau^{(i+1)}}{1-\tau}\|\mathbf{x}\|\end{aligned}$$

Assuming $\|A^{-1}\|\|\delta A^{(i)}\| \le \alpha$ and $\|E^{(i)}\| \le \omega$,

$$\tau = \frac{(1+\epsilon)(\alpha+\kappa(A)\omega)}{1-\alpha}$$

where $\|F^{(i)}\| \le \epsilon$. For sufficiently large i, we have

$$\frac{\|\mathbf{z}^{(i)}\|}{\|\mathbf{x}\|} \le \frac{\epsilon}{1-\tau} + O(\epsilon^2)$$

From

$$1 - \tau = \frac{(1-\alpha) - (1+\epsilon)(\alpha + \kappa(A)\omega)}{1-\alpha}$$

we obtain

$$\frac{1}{1-\tau} = \frac{1-\alpha}{(1-\alpha) - (1+\epsilon)(\alpha+\kappa(A)\omega)} \approx \frac{1-\alpha}{1-2\alpha-\kappa(A)\omega}.$$

Therefore, $1/(1-\tau) \leq 2$ whenever $\alpha \leq \frac{1}{3} - \frac{2}{3}\kappa(A)\omega$, approximately.

It can be shown that if the vector $\mathbf{r}^{(k)}$ is computed using double or extended precision that $\mathbf{x}^{(k)}$ converges to a solution where almost all digits are correct when $\kappa(A)\mathbf{u} \leq 1$.

4.8.2 Scaling and Equilibration

As we have seen, the bounds for the error depend on $\kappa(A) = \|A\|\|A^{-1}\|$. Perhaps we can re-scale the equations so that the condition number is changed. We replace the system

$$A\mathbf{x} = \mathbf{b}$$

by the equivalent system

$$DA\mathbf{x} = D\mathbf{b}$$

or possibly

$$DAE\mathbf{y} = D\mathbf{b}$$

where D and E are diagonal matrices and $y = E^{-1}\mathbf{x}$.

The answer will depend upon the norm used to compute the condition number.

Suppose A is symmetric positive definite. We want to replace A by DAD; i.e. $a_{ij} \leftarrow d_i d_j a_{ij}$. Can we choose D so that $\kappa(DAD)$ is minimized?

It turns out that for a class of symmetric matrices, this is the case. A symmetric positive definite matrix A is said to have *Property A* if there exists a permutation matrix Π such that

$$\Pi A \Pi^T = \begin{bmatrix} D & F \\ F^T & D \end{bmatrix}$$

where D is a diagonal matrix. All tridiagonal matrices that are symmetric positive definite have Property A.

For example, suppose

$$A = \begin{bmatrix} 50 & 7 \\ 7 & 1 \end{bmatrix}.$$

Then $\lambda_{\max} \approx 51$ and $\lambda_{\min} \approx 1/51$, which means that $\kappa(A) \approx 2500$. However,

$$DAD = \begin{bmatrix} \frac{1}{\sqrt{50}} & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 50 & 7 \\ 7 & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{50}} & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & \frac{7}{\sqrt{50}} \\ \frac{7}{\sqrt{50}} & 1 \end{bmatrix}$$

and

$$\kappa = \frac{1 + \frac{7}{\sqrt{50}}}{1 - \frac{7}{\sqrt{50}}} \approx 200.$$

One scaling strategy is called *equilibration*. The idea is to set $A^{(0)} = A$ and compute $A^{(1/2)} = D^{(1)}A^{(0)} = \{d_i^{(1)} a_{ij}\}$, choosing the diagonal matrix D_1 so that $d_i^{(1)} \sum_{j=1}^n |a_{ij}^{(0)}| = 1$. Then, we compute $A^{(1)} = A^{(1/2)}E^{(1)} = \{a_{ij}^{(1/2)} e_j^{(1)}\}$, choosing each element of the diagonal matrix $E^{(1)}$ so that $e_j^{(1)} \sum_{i=1}^n |a_{ij}^{(1/2)}| = 1$. We then repeat this process, which yields

$$\begin{aligned} A^{(k+1/2)} &= D^{(k+1)}A^{(k)} \\ A^{(k+1)} &= A^{(k+1/2)}E^{(k+1)} \end{aligned}$$

Under very general conditions, the $A(k)$ converge to a matrix whose row and column sums are all equal.

4.9 Estimating the Condition Number

Consider the condition number

$$\kappa_\infty(A) = \|A\|_\infty \|A^{-1}\|_\infty.$$

Of course,

$$\|A\|_\infty = \max_i \sum_{j=1}^n |a_{ij}|,$$

but how do we compute $\|A^{-1}\|_\infty$? If $A^{-1} = B$, then $\|A^{-1}\|_\infty = \max_i \sum_{j=1}^n |b_{ij}|$. Suppose $A\mathbf{y} = \mathbf{d}$ or $\mathbf{y} = A^{-1}\mathbf{d}$. Then $\|\mathbf{y}\|_\infty \leq \|A^{-1}\|_\infty \|\mathbf{d}\|_\infty$, and therefore

$$\|A^{-1}\|_\infty \geq \frac{\|\mathbf{y}\|_\infty}{\|\mathbf{d}\|_\infty}.$$

This suggests an algorithm for estimating the condition number: we can choose $\mathbf{d}$ to maximize $\|\mathbf{y}\|_\infty$. To illustrate the process, we let

$$A = T = \begin{bmatrix} t_{11} & t_{12} & \cdots & t_{1n} \\ & \ddots & & \vdots \\ & & & t_{nn} \end{bmatrix}$$

and examine the process of solving $T\mathbf{y} = \mathbf{d}$. Writing out this system of equations yields

$$\begin{aligned} t_{11}y_1 + t_{12}y_2 + \cdots + t_{1n}y_n &= d_1 \\ &\vdots \\ t_{nn}y_n &= d_n \end{aligned}$$

Considering the last equation $y_n = d_n/t_{nn}$, we choose $d_n = +1$ if $t_{nn} > 0$, and -1 otherwise. Next, we have

$$t_{n-1,n-1}y_{n-1} + t_{n-1,n}y_n = d_{n-1},$$

which yields

$$y_{n-1} = \frac{d_{n-1} - t_{n-1,n}y_n}{t_{n-1,n-1}}.$$

If $t_{n-1,n}y_n > 0$, we choose $d_{n-1} = -1$, otherwise, we set $d_{n-1} = +1$. We continue this process, consistently choosing $d_i = \pm 1$ depending on which choice increases $\|\mathbf{y}\|_\infty$. There are other more sophisticated strategies than this.

Appendix: the Simplex Method

In order to implement the Simplex Algorithm, it is necessary to solve three systems of linear equations at each iteration; namely

$$Bx = b \tag{4.5}$$
$$B^T w = \tilde{c} \tag{4.6}$$
$$Bt^{(r)} = -a^{(r)} \tag{4.7}$$

If the LU decomposition of B is known, then it is easy to solve the three systems of equations. We have already shown how to solve systems (4.5) and (4.7), using the LU decomposition. Since $B^T = U^T L^T$, solving (4.6) merely requires the solving of $U^T y = \tilde{c}$ and then $L^T w = y$.

In the Simplex Algorithm we change only one column of B at a time. If the LU decomposition of B is known, we can determine the LU decomposition of the new matrix B by simply updating the previous decomposition. This process can be done efficiently and in a manner that insures numerical stability.

Suppose Gaussian elimination with partial pivoting has been used on B so that

$$P^{(m-1)}\Pi^{(m-1)} \cdots P^{(1)}\Pi^{(1)}B = U.$$

Let

$$B = [b^{(1)}, b^{(2)}, \ldots, b^{(m)}] \quad \text{and} \quad U = [u^{(1)}, u^{(2)}, \ldots, u^{(m)}].$$

Because the last $(m-k)$ components of $u^{(k)}$ are zero,

$$P^{(k+1)}\Pi^{(k+1)}u^{(k)} = u^{(k)}$$

since $P^{(k+1)}\Pi^{(k+1)}$ linearly combines the bottom $(m-k)$ elements of $u^{(k)}$. Thus,

$$u^{(k)} = P^{(k)}\Pi^{(k)}P^{(k-1)}\Pi^{(k-1)}\cdots P^{(1)}\Pi^{(1)}b^{(k)}.$$

If we let

$$\bar{B} = [b^{(1)}, b^{(2)}, \ldots, b^{(s-1)}, g, b^{(s+1)}, \ldots, b^{(m)}],$$

and

$$T^{(k)} = P^{(k)}\Pi^{(k)}\cdots P^{(1)}\Pi^{(1)}.$$

Then

$$\begin{aligned} T^{(s-1)} &= [T^{(1)}b^{(1)}, \ldots, T^{(s-1)}b^{(s-1)}, T^{(s-1)}g, T^{(s-1)}b^{(s+1)}, \ldots, T^{(s-1)}b^{(m)}] \\ &= [u^{(1)}, \ldots, u^{(s-1)}, T^{(s-1)}g, T^{(s-1)}b^{(s+1)}, \ldots, T^{(s-1)}b^{(m)}]. \end{aligned}$$

Therefore, to find the new LU decomposition of $\bar{B}$ we need only compute $\bar{\Pi}^{(s)}$, $\bar{P}^{(s)}$, ..., $\bar{\Pi}^{(m-1)}$, $\bar{P}^{(m-1)}$ so that

$$\bar{P}^{(m-1)}\Pi^{(m-1)}\cdots\bar{P}^{(s)}\bar{\Pi}^{(s)}T^{(s-1)}[g, b^{(s+1)}, \ldots, b^{(m)}] = [\bar{u}^{(s)}, \bar{u}^{(s+1)}, \ldots, \bar{u}^{(m)}],$$

where $\bar{u}^{(k)}$ is a new vector whose last $(m-k)$ components are zero. If g replaces $b^{(m)}$, then about $m^2/2$ multiplications are required to compute the new $\bar{U}$. However, if g replaces $b^{(1)}$, the decomposition must be completely recomputed.

We can update the LU decomposition in a more efficient manner which unfortunately requires more storage. Let us write

$$B_0 = L_0U_0.$$

Let the column s_0 of B_0 be replaced by the column vector g_0. As long as we revise the ordering of the unknowns accordingly we may insert g_0 into the last column position, shifting columns s_0+1 through m of B_0 one position to the left to make room. We will call the result B_1, and we can easily check that it has the decomposition

$$B_1 = L_0H_1,$$

where H_1 is a matrix that is *upper Hessenberg* in its last $m-s_0+1$ columns, and upper-triangular in its first s_0-1 columns.

The first s_0-1 columns of H_1 are identical with those of U_0. The next $m-s_0$ are identical with the last $m-s_0$ columns of U_0, and the last column of H_1 is the vector $L_0^{-1}g_0$.

H_1 can be reduced to upper-triangular form by Gaussian elimination with row interchanges. Here, however, we need only concern ourselves with the interchanges of pairs of adjacent rows. Thus, U_1 is gotten from H_1 by applying a sequence of simple transformations:

$$U_1 = P_1^{(m-1)}\Pi_1^{(m-1)}\cdots P_1^{(s_0)}\Pi_1^{(s_0)}H_1 \tag{4.8}$$

where each $P_1^{(k)}$ is the identity matrix with a single nonzero subdiagonal element $g_k^{(1)}$ in the $(k+1,k)$ position, and each $\Pi^{(k)}$ is either the identity matrix or the identity matrix with the kth and $(k+1)st$ rows exchanged, the choice being made so that $|g_k^{(1)}| \leq 1$.

The essential information in all of the transformations can be stored in $m-s_0$ locations plus an additional $m-s_0$ bits (to indicate the interchanges). If we let

$$L_1^{-1} = P_1^{(m-1)}\Pi_1^{(m-1)}\cdots P_1^{(s_0)}\Pi_1^{(s_0)}L_0^{-1},$$

then we have achieved the decomposition

$$B_1 = L_1U_1.$$

The transition from B_1 to B_{i+1}, where i represents the ith time through steps (2)-(7) of the Simplex Algorithm, is to be made exactly as the transition from B_0 to B_1. Any system of linear equations involving the matrix B_i for any i is to be solved by applying the sequences of transformations defined by (4.8) and then solving the upper triangular system of equations.

As we have already pointed out, it requires

$$m^3/3 + O(m^2)$$

multiplication-type operations to produce an initial LU decomposition,

$$B_0x = v.$$

The solution for any system $B_ix = v$ must be found according to the LU decomposition method by computing

$$y = L_i^{-1}v, \tag{4.9}$$

followed by solving

$$U_i x = y. \tag{4.10}$$

The application of L_0^{-1} to v in (4.9) will require $m(m-1)/2$ operations. The application of the remaining transformations in L_i^{-1} will require at most $i(m-1)$ operations. Solving (4.10) costs $m(m+1)/2$ operations. Hence, the cost of (4.9) and (4.10) together is not greater than

$$m^2 + i(m-1)$$

operations, and a reasonable expected figure would be $m^2 + \frac{i}{2}(m-1)$.

Chapter 5

Least-Squares Problems

5.1 The Full-rank Linear Least Squares Problem

Given an $m \times n$ matrix A, with $m \geq n$, and an m-vector $\mathbf{b}$, we consider the *overdetermined* system of equations $A\mathbf{x} = \mathbf{b}$, in the case where A has full column rank. If $\mathbf{b}$ is in the range of A, then there exists a unique solution $\mathbf{x}^*$. For example, there exists a unique solution in the case of

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 0 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix},$$

but not if $\mathbf{b} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}^T$. In such cases, when $\mathbf{b}$ is not in the range of A, then we seek to minimize $\|A\mathbf{x} - \mathbf{b}\|_p$ for some p.

Different norms give different solutions. If $p = 1$ or $p = \infty$, then the function we seek to minimize, $f(x) = \|A\mathbf{x} - \mathbf{b}\|_p$ is not differentiable, so we cannot use standard minimization techniques. However, if $p = 2$, $f(x)$ is differentiable, and thus the problem is more tractable. We now consider two methods.

The first approach is to take advantage of the fact that the 2-norm is invariant under orthogonal transformations, and seek an orthogonal matrix Q such that the transformed problem

$$\min \|A\mathbf{x} - \mathbf{b}\|_2 = \min \|Q^T(A\mathbf{x} - \mathbf{b})\|_2$$

is "easy" to solve. Let

$$A = Q \begin{bmatrix} R \\ 0 \end{bmatrix} = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} R \\ 0 \end{bmatrix} = Q_1 R.$$

Then $Q_1^T A = R$ and

$$\begin{aligned} \min \|A\mathbf{x} - \mathbf{b}\|_2 &= \min \|Q^T(A\mathbf{x} - \mathbf{b})\|_2 \\ &= \min \|(Q^T A)\mathbf{x} - Q^T\mathbf{b}\|_2 \\ &= \min \left\| \begin{bmatrix} R \\ 0 \end{bmatrix} \mathbf{x} - Q^T\mathbf{b} \right\|_2 \end{aligned}$$

If we partition

$$Q^T\mathbf{b} = \begin{bmatrix} \mathbf{c} \\ \mathbf{d} \end{bmatrix}$$

then

$$\min \|A\mathbf{x} - \mathbf{b}\|_2^2 = \min \left\| \begin{bmatrix} R \\ 0 \end{bmatrix} \mathbf{x} - \begin{bmatrix} \mathbf{c} \\ \mathbf{d} \end{bmatrix} \right\|_2^2 = \min \|R\mathbf{x} - \mathbf{c}\|_2^2 + \|\mathbf{d}\|_2^2.$$

Therefore, the minimum is achieved by the vector $\mathbf{x}$ such that $R\mathbf{x} = \mathbf{c}$ and therefore

$$\min_{\mathbf{x}} \|A\mathbf{x} - \mathbf{b}\|_2 = \|\mathbf{d}\|_2 \equiv \rho_{LS}.$$

The second method is to define $\phi(\mathbf{x}) = \frac{1}{2}\|A\mathbf{x} - \mathbf{b}\|_2^2$, which is a differentiable function of $\mathbf{x}$. We can minimize $\phi(\mathbf{x})$ by noting that $\nabla\phi(\mathbf{x}) = A^T(A\mathbf{x} - \mathbf{b})$, which means that $\nabla\phi(\mathbf{x}) = \mathbf{0}$ if and only if $A^T A\mathbf{x} = A^T\mathbf{b}$. This system of equations is called the *normal equations*, and they were used by Gauss to solve the least squares problem. If $m >> n$ then $A^T A$ is $n \times n$, which is a much smaller system to solve than $A\mathbf{x} = \mathbf{b}$, and if $\kappa(A^T A)$ is not too large, we can use the LU factorization to solve for $\mathbf{x}$.

Which is the better method? This is not a simple question to answer. The normal equations produce an $\mathbf{x}^*$ whose relative error depends on $\kappa(A)^2$, whereas the QR factorization produces an $\mathbf{x}^*$ whose relative error depends on $\mathbf{u}(\kappa_2(A) + \rho_{LS}\kappa_2(A)^2)$. The normal equations involve much less arithmetic when $m >> n$ and they require less storage, but the QR factorization is often applicable if the normal equations break down.

5.2 The QR Factorization

Let A be an $m \times n$ matrix with full column rank. The QR *factorization* of A is a decomposition $A = QR$, where Q is an $m \times m$ orthogonal matrix and R is an $m \times n$ upper triangular matrix. There are two common ways to compute this decomposition:

1. Using Householder matrices, developed by Alston S. Householder;

2. Using Givens rotations, also known as Jacobi rotations, used by W. Givens and originally invented by Jacobi for use with in solving the symmetric eigenvalue problem in 1846.

A third, less frequently used approach, called *Gram-Schmidt orthogonalization*, will also be discussed.

5.2.1 Givens (Jacobi) rotations

We illustrate the process in the case where A is a 2×2 matrix. In Gaussian elimination, we compute $L^{-1}A = U$ where L^{-1} is unit lower triangular and U is upper triangular. Specifically,

$$\begin{bmatrix} 1 & 0 \\ m_{21} & 1 \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = \begin{bmatrix} a_{11}^{(2)} & a_{12}^{(2)} \\ 0 & a_{22}^{(2)} \end{bmatrix}, \quad m_{21} = -\frac{a_{21}}{a_{11}}.$$

By contrast, the QR decomposition takes the form

$$\begin{bmatrix} \gamma & \sigma \\ -\sigma & \gamma \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} \\ 0 & r_{22} \end{bmatrix}$$

where $\gamma^2 + \sigma^2 = 1$. From the relationship $-\sigma a_{11} + \gamma a_{21} = 0$ we obtain

$$\begin{aligned} \gamma a_{21} &= \sigma a_{11} \\ \gamma^2 a_{21}^2 &= \sigma^2 a_{11}^2 = (1 - \gamma^2) a_{11}^2 \end{aligned}$$

which yields

$$\gamma = \pm \frac{a_{11}}{\sqrt{a_{21}^2 + a_{11}^2}}.$$

It is conventional to choose the + sign. Then, we obtain

$$\sigma^2 = 1 - \gamma^2 = 1 - \frac{a_{11}^2}{a_{21}^2 + a_{11}^2} = \frac{a_{21}^2}{a_{21}^2 + a_{11}^2},$$

or

$$\sigma = \pm \frac{a_{21}}{\sqrt{a_{21}^2 + a_{11}^2}}.$$

Again, we choose the + sign. As a result, we have

$$r_{11} = a_{11} \frac{a_{11}}{\sqrt{a_{21}^2 + a_{11}^2}} + a_{21} \frac{a_{21}}{\sqrt{a_{21}^2 + a_{11}^2}} = \sqrt{a_{21}^2 + a_{11}^2}.$$

The matrix

$$Q^T = \begin{bmatrix} \gamma & \sigma \\ -\sigma & \gamma \end{bmatrix}$$

is called a *Givens rotation*. It is called a rotation because it is orthogonal, and therefore length-preserving, and also because there is an angle θ such that $\sin\theta = \sigma$ and $\cos\theta = \gamma$, and its effect is to rotate a vector through the angle θ. In particular,

$$\begin{bmatrix} \gamma & \sigma \\ -\sigma & \gamma \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \rho \\ 0 \end{bmatrix}$$

where $\rho = \sqrt{\alpha^2 + \beta^2}$, $\alpha = \rho\cos\theta$ and $\beta = \rho\sin\theta$. It is easy to verify that the product of two rotations is itself a rotation. Now, in the case where A is an $n \times n$ matrix, suppose that we have the vector

$$\begin{bmatrix} \times \\ \vdots \\ \times \\ \alpha \\ \times \\ \vdots \\ \times \\ \beta \\ \times \\ \vdots \\ \times \end{bmatrix}.$$

Then

$$\begin{bmatrix} 1 & & & & & & & & & & \\ & \ddots & & & & & & & & & \\ & & 1 & & & & & & & & \\ & & & \gamma & & & & \sigma & & & \\ & & & & 1 & & & & & & \\ & & & & & \ddots & & & & & \\ & & & & & & 1 & & & & \\ & & & -\sigma & & & & \gamma & & & \\ & & & & & & & & 1 & & \\ & & & & & & & & & \ddots & \\ & & & & & & & & & & 1 \end{bmatrix} \begin{bmatrix} \times \\ \vdots \\ \times \\ \alpha \\ \times \\ \vdots \\ \times \\ \beta \\ \times \\ \vdots \\ \times \end{bmatrix} = \begin{bmatrix} \times \\ \vdots \\ \times \\ \rho \\ \times \\ \vdots \\ \times \\ 0 \\ \times \\ \vdots \\ \times \end{bmatrix}.$$

So, in order to transform A into an upper triangular matrix R, we can find a product of rotations Q such that $Q^T A = R$. It is easy to see that $O(n^2)$ rotations are required.

5.2.2 Householder reflections

It is natural to ask whether we can introduce more zeros with each orthogonal rotation. To that end, we examine *Householder reflections*. Consider a matrix of the form $P = I - \tau \mathbf{u}\mathbf{u}^T$, where $\mathbf{u} \neq \mathbf{0}$ and τ is a nonzero constant. It is clear that P is a symmetric rank-1 change of I. Can we choose τ so that P is also orthogonal? From the desired relation $P^T P = I$ we obtain

$$\begin{aligned} P^T P &= (I - \tau \mathbf{u}\mathbf{u}^T)^T (I - \tau \mathbf{u}\mathbf{u}^T) \\ &= I - 2\tau \mathbf{u}\mathbf{u}^T + \tau^2 \mathbf{u}\mathbf{u}^T \mathbf{u}\mathbf{u}^T \\ &= I - 2\tau \mathbf{u}\mathbf{u}^T + \tau^2 (\mathbf{u}^T \mathbf{u}) \mathbf{u}\mathbf{u}^T \\ &= I - (\tau^2 \mathbf{u}^T \mathbf{u} - 2\tau) \mathbf{u}\mathbf{u}^T \\ &= I + \tau(\tau \mathbf{u}^T \mathbf{u} - 2) \mathbf{u}\mathbf{u}^T. \end{aligned}$$

It follows that if $\tau = 2/\mathbf{u}^T\mathbf{u}$, then $P^T P = I$ for any nonzero $\mathbf{u}$. Without loss of generality, we can stipulate that $\mathbf{u}^T\mathbf{u} = 1$, and therefore P takes the form $P = I - 2\mathbf{v}\mathbf{v}^T$, where $\mathbf{v}^T\mathbf{v} = 1$.

Why is the matrix P called a reflection? This is because for any nonzero vector $\mathbf{x}$, $P\mathbf{x}$ is the reflection of $\mathbf{x}$ across the hyperplane that is normal to $\mathbf{v}$. To see this, we consider the 2×2 case and set $\mathbf{v} = \begin{bmatrix} 1 & 0 \end{bmatrix}^T$ and $\mathbf{x} = \begin{bmatrix} 1 & 2 \end{bmatrix}^T$. Then

$$\begin{aligned} P &= I - 2\mathbf{v}\mathbf{v}^T \\ &= I - 2 \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - 2 \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \\ &= \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \end{aligned}$$

Therefore

$$P\mathbf{x} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} -1 \\ 2 \end{bmatrix}.$$

Now, let $\mathbf{x}$ be any vector. We wish to construct P so that $P\mathbf{x} = \alpha \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}^T =$

$\alpha\mathbf{e}_1$ for some α. From the relations

$$\begin{aligned} \|P\mathbf{x}\|_2 &= \|\mathbf{x}\|_2 \\ \|\alpha\mathbf{e}_1\|_2 &= |\alpha|\|\mathbf{e}_1\|_2 = |\alpha| \end{aligned}$$

we obtain $\alpha = \pm\|\mathbf{x}\|_2$. To determine P, we observe that

$$\begin{aligned} \mathbf{x} &= P^{-1}(\alpha\mathbf{e}_1) \\ &= \alpha P\mathbf{e}_1 \\ &= \alpha(I - 2\mathbf{v}\mathbf{v}^T)\mathbf{e}_1 \\ &= \alpha[\mathbf{e}_1 - 2\mathbf{v}\mathbf{v}^T\mathbf{e}_1] \\ &= \alpha[\mathbf{e}_1 - 2\mathbf{v}v_1] \end{aligned}$$

which yields the system of equations

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \alpha \begin{bmatrix} 1 - 2v_1^2 \\ -2v_1v_2 \\ \vdots \\ -2v_1v_n \end{bmatrix}.$$

From the first equation $x_1 = \alpha(1 - 2v_1^2)$ we obtain

$$v_1 = \pm\sqrt{\frac{1}{2}\left(1 - \frac{x_1}{\alpha}\right)}.$$

It is best to choose α to have the opposite sign of x_1 to avoid cancellation in the computation of v_1. Then for $i = 2, \ldots, n$, we have

$$v_i = -\frac{x_i}{2\alpha v_1}.$$

Note that the matrix P is never formed explicitly. For any vector $\mathbf{b}$, the product $P\mathbf{b}$ can be computed as follows:

$$P\mathbf{b} = (I - 2\mathbf{v}\mathbf{v}^T)\mathbf{b} = \mathbf{b} - 2(\mathbf{v}^T\mathbf{b})\mathbf{v}.$$

This process requires only $O(2n)$ operations. It is easy to see that we can represent P simply by storing only $\mathbf{v}$.

To complete the QR factorization, suppose that that $\mathbf{x} = \mathbf{a}_1$ is the first column of a matrix A. First we construct a Householder reflection $H_1 = I - 2\mathbf{u}_1\mathbf{u}_1^T$ such that $H\mathbf{x} = \alpha\mathbf{e}_1$, so that we have

$$A^{(2)} = H_1A = \begin{bmatrix} r_{11} & & & \\ 0 & & & \\ \vdots & \mathbf{a}_2^{(2)} & \cdots & \mathbf{a}_n^{(2)} \\ 0 & & & \end{bmatrix},$$

where we denote the constant α by r_{11}, as it is the $(1,1)$ element of the updated matrix $A^{(2)}$. Now, we can construct H_2 such that

$$H_2\mathbf{a}^{(2)} = \begin{bmatrix} a_{12}^{(2)} \\ r_{22} \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad u_{12} = 0, \quad H_2 = \begin{bmatrix} 1 & 0 \\ 0 & \\ \vdots & h_{ij} \\ 0 & \end{bmatrix}.$$

Note that the first column of $A^{(2)}$ is unchanged by H_2. Continuing this process, we obtain

$$H_{n-1} \cdots H_1 A = A^{(n)} = R$$

where R is an upper triangular matrix. We have thus factored $A = QR$, where $Q = H_1 H_2 \cdots H_{n-1}$ is an orthogonal matrix. Note that

$$A^T A = R^T Q^T Q R = R^T R,$$

and thus R is the Cholesky factor of $A^T A$.

Remark. Because each Jacobi rotation only modifies two rows of A, it is possible to interchange the order of rotations that affect different rows, and thus apply sets of rotations in parallel. This is the main reason why Jacobi rotations can be preferable to Householder reflections. Other reasons are that they are easy to use when the QR factorization needs to be updated as a result of adding a row to A or deleting a column of A. They are also more efficient when A is sparse.

5.2.3 Gram-Schmidt and Modified Gram-Schmidt orthogonalization

Consider the QR factorization

$$A = \begin{bmatrix} \mathbf{a}_1 & \cdots & \mathbf{a}_n \end{bmatrix} = \begin{bmatrix} \mathbf{q}_1 & \cdots & \mathbf{q}_n \end{bmatrix} \begin{bmatrix} r_{11} & \cdots & r_{1n} \\ & \ddots & \vdots \\ & & r_{nn} \end{bmatrix}.$$

From the above matrix product we can see that $\mathbf{a}_1 = r_{11}\mathbf{q}_1$, from which it follows that

$$r_{11} = \pm\|\mathbf{a}_1\|_2, \quad \mathbf{q}_1 = \frac{1}{\|\mathbf{a}_1\|_2}\mathbf{a}_1.$$

Next, from $\mathbf{a}_2 = r_{12}\mathbf{q}_1 + r_{22}\mathbf{q}_2$ we obtain

$$r_{12} = \mathbf{q}_1^T\mathbf{a}_2, \quad r_{22} = \pm\|\mathbf{a}_2 - r_{12}\mathbf{q}_1\|_2, \quad \mathbf{q}_2 = \frac{1}{r_{22}}(\mathbf{a}_2 - r_{12}\mathbf{q}_1).$$

In general, we use the relation

$$\mathbf{a}_k = \sum_{j=1}^{k} r_{jk}\mathbf{q}_j$$

to obtain

$$\mathbf{q}_k = \frac{1}{r_{kk}}\left(\mathbf{a}_k - \sum_{j=1}^{k-1} r_{jk}\mathbf{q}_j\right), \quad r_{jk} = \mathbf{q}_j^T\mathbf{a}_k.$$

Note that $\mathbf{q}_k$ can be rewritten as

$$\mathbf{q}_k = \frac{1}{r_{kk}}\left(\mathbf{a}_k - \sum_{j=1}^{k-1}(\mathbf{q}_j^T\mathbf{a}_k)\mathbf{q}_j\right) = \frac{1}{r_{kk}}\left(\mathbf{a}_k - \sum_{j=1}^{k-1}\mathbf{q}_j\mathbf{q}_j^T\mathbf{a}_k\right) = \frac{1}{r_{kk}}\left(I - \sum_{j=1}^{k-1}\mathbf{q}_j\mathbf{q}_j^T\right)\mathbf{a}_k.$$

If we define $P_i = \mathbf{q}_i\mathbf{q}_i^T$, then P_i is a *symmetric projector* that satisfies $P_i^2 = P_i$, and $P_iP_j = \delta_{ij}$. Thus we can write

$$\mathbf{q}_k = \frac{1}{r_{kk}}\left(I - \sum_{j=0}^{k-1} P_j\right)\mathbf{a}_k = \frac{1}{r_{kk}}\prod_{j=1}^{k-1}(I - P_j)\mathbf{a}_k.$$

The main deficiency of the classical Gram-Schmidt process is that it is numerically unstable. If $\mathbf{a}_1$ and $\mathbf{a}_2$ are almost parallel, then $\mathbf{a}_2 - r_{12}\mathbf{q}_1$ is almost zero and roundoff error becomes significant. The *Modified Gram-Schmidt* method alleviates this difficulty. Recall

$$A = QR = \begin{bmatrix} r_{11}\mathbf{q}_1 & r_{12}\mathbf{q}_1 + r_{22}\mathbf{q}_2 & \cdots \end{bmatrix}$$

We define

$$A^{(k)} = \sum_{i=1}^{k-1}\mathbf{q}_i\mathbf{r}_i^T, \quad \mathbf{r}_i^T = \begin{bmatrix} r_{i1} & r_{i2} & \cdots & r_{ii} \end{bmatrix}$$

which means

$$A - \sum_{i=1}^{k-1}\mathbf{q}_i\mathbf{r}_i^T = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & A^{(k)} \end{bmatrix}.$$

If we write

$$A^{(k)} = \begin{bmatrix} \mathbf{z} & B \end{bmatrix}$$

then

$$r_{kk} = \|\mathbf{z}\|_2, \quad \mathbf{q}_k = \frac{1}{r_{kk}}\mathbf{z}.$$

We then compute

$$\begin{bmatrix} r_{k,k+1} & \cdots & r_{k,n} \end{bmatrix} = \mathbf{q}_k^T B$$

which yields

$$A^{(k+1)} = B - \mathbf{q}_k \begin{bmatrix} r_{1k} & \cdots & r_{kk} \end{bmatrix}$$

This process is numerically stable. One can show that

$$\hat{Q}_1^T \hat{Q}_1 = I + E_{MGS}, \quad \|E_{MGS}\| \approx \mathbf{u}\kappa_2(A),$$

and $\hat{Q}_1$ can be computed in approximately $2mn^2$ flops, whereas with Householder QR,

$$\hat{Q}_1^T \hat{Q}_1 = I + E_n, \quad \|E_n\| \approx \mathbf{u},$$

with $\hat{Q}_1$ being computed in approximately $2mn^2 - 2n^2/3$ flops to factor A and an additional $2mn^2 - 2n^2/3$ flops to obtain the n columns of Q.

Note that the error bound for modified Gram-Schmidt depends on the condition number of A, whereas the bound for Householder doesn't. This can be an important consideration when dealing with ill-conditioned systems. An example of such systems is the *Hilbert matrices*, which are defined by

$$H = \begin{bmatrix} 1 & 1/2 & 1/3 & \cdots & 1/n \\ 1/2 & 1/3 & \cdots & & \\ \vdots & & & & \end{bmatrix}, \quad h_{ij} = \frac{1}{i+j-1}.$$

It is very ill-conditioned, but H^{-1} is known, and its entries are all integers.

5.3 Solution using Normal Equations

We can solve the linear least squares problem using the normal equations

$$A^T A\mathbf{x} = A^T \mathbf{b}$$

as follows: first, we solve the above system to obtain an approximate solution $\hat{\mathbf{x}}$, and compute the residual vector $\mathbf{r} = \mathbf{b} - A\hat{\mathbf{x}}$. Now, because

$$A^T \mathbf{r} = A^T \mathbf{b} - A^T A\hat{\mathbf{x}} = \mathbf{0},$$

we obtain the system

$$\begin{aligned} \mathbf{r} + A\hat{\mathbf{x}} &= \mathbf{b} \\ A^T \mathbf{r} &= \mathbf{0} \end{aligned}$$

or, in matrix form,

$$\begin{bmatrix} I & A \\ A^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{r} \\ \mathbf{x} \end{bmatrix} = \begin{bmatrix} \mathbf{b} \\ \mathbf{0} \end{bmatrix}.$$

This is a large system, but it preserves the sparsity of A. It can be used in connection with iterative refinement, but unfortunately this procedure does not work well because it is very sensitive to the residual.

5.4 Perturbation Theory for Least-Squares Problems

Suppose that we are solving the perturbed least squares problem

$$A(\epsilon)\mathbf{x}(\epsilon) = \mathbf{b}, \quad A(\epsilon) = A + \epsilon E.$$

How does the residual vector $\mathbf{r}(\epsilon) = \mathbf{b} - A\mathbf{x}(\epsilon)$ and the solution $\mathbf{x}(\epsilon)$ change as a function of ϵ?

Before computing any bounds, let us first note that the computed solution $\hat{\mathbf{x}} = A^+\mathbf{b}$ is very sensitive to the residual. To see this, suppose that $\mathbf{b}$ is replaced by $\mathbf{b} + \alpha\mathbf{r}$, where α is a constant. Then

$$\begin{aligned} A^+(\mathbf{b} + \alpha\mathbf{r}) &= \hat{\mathbf{x}} + \alpha A^+\mathbf{r} \\ &= \hat{\mathbf{x}} + \alpha A^+(I - AA^+)\mathbf{b} \\ &= \hat{\mathbf{x}} + \alpha[A^+\mathbf{b} - A^+AA^+\mathbf{b}] \\ &= \hat{\mathbf{x}} \end{aligned}$$

so the computed solution is unchanged, even if α is large.

To compute an actual bound, we use the fact that $PA = AA^+A = A$, and we differentiate with respect to ϵ and obtain

$$P\frac{dA}{d\epsilon} + \frac{dP}{d\epsilon}A = \frac{dA}{d\epsilon}.$$

It follows that

$$\frac{dP}{d\epsilon}A = (I - P)\frac{dA}{d\epsilon} = P^\perp\frac{dA}{d\epsilon}.$$

Multiplying through by A^+, we obtain

$$\frac{dP}{d\epsilon}P = P^\perp\frac{dA}{d\epsilon}A^+.$$

Because P is a projection,

$$\frac{d(P^2)}{d\epsilon} = P\frac{dP}{d\epsilon} + \frac{dP}{d\epsilon}P = \frac{dP}{d\epsilon},$$

so, using the relationship $A^T P = A^T$,

$$\frac{dP}{d\epsilon} = P^\perp \frac{dA}{d\epsilon}A^+ + (A^+)^T \frac{dA^T}{d\epsilon}P^\perp.$$

Now, using a Taylor expansion around $\epsilon = 0$, we obtain

$$\begin{aligned}
\mathbf{r}(\epsilon) &= \mathbf{r}(0) + \epsilon\frac{dP^\perp}{d\epsilon}\mathbf{b} + O(\epsilon^2) \\
&= \mathbf{r}(0) - \epsilon\frac{dP}{d\epsilon}\mathbf{b} + O(\epsilon^2) \\
&= \mathbf{r}(0) - \epsilon[P^\perp E\hat{\mathbf{x}}(0) + (A^+)^T E^T \mathbf{r}(0)] + O(\epsilon^2)
\end{aligned}$$

from the relations $\hat{\mathbf{x}} = A^+\mathbf{b}$ and $\mathbf{r} = P^\perp\mathbf{b}$. Taking norms, we obtain

$$\frac{\|\mathbf{r}(\epsilon) - \mathbf{r}(0)\|_2}{\|\hat{\mathbf{x}}\|_2} = |\epsilon|\|E\|_2\left(1 + \|A^+\|_2\frac{\|\mathbf{r}(0)\|_2}{\|\hat{\mathbf{x}}(0)\|_2}\right) + O(\epsilon^2).$$

Note that if A is scaled so that $\|A\|_2 = 1$, then the second term above involves the condition number $\kappa_2(A)$. We also have

$$\frac{\|\mathbf{x}(\epsilon) - \mathbf{x}(0)\|_2}{\|\hat{\mathbf{x}}\|_2} = |\epsilon|\|E\|_2\left(2\kappa(A) + \kappa_2(A)^2\frac{\|\mathbf{r}(0)\|_2}{\|\hat{\mathbf{x}}(0)\|_2}\right) + O(\epsilon^2).$$

Note that a small perturbation residual does not imply a small perturbation in the solution.

5.5 Rank-deficient Least Squares

We seek a decomposition of the form $A = Q^T R\Pi$ where Π is chosen so that the diagonal elements of R are maximized at each stage. Specifically, suppose

$$H_1 A = \begin{bmatrix} r_{11} & \\ 0 & \\ \vdots & * \\ 0 & \end{bmatrix}, \quad r_{11} = \|\mathbf{a}_1\|_2.$$

So, we choose Π_1 so that $\|\mathbf{a}_1\|_2 \geq \|\mathbf{a}_j\|_2$ for $j \geq 2$. For Π_2, look at the lengths of the columns of the submatrix. We don't need to recompute the

lengths each time, because we can update by subtracting the square of the first component from the square of the total length. Eventually, we get

$$Q^T \begin{bmatrix} R & S \\ 0 & 0 \end{bmatrix} \Pi_1 \cdots \Pi_r = A$$

where R is upper triangular. Using this decomposition, we can solve the linear least squares problem $A\mathbf{x} = \mathbf{b}$ by observing that

$$\begin{aligned} \|\mathbf{b} - A\mathbf{x}\|_2^2 &= \left\| \mathbf{b} - Q^T \begin{bmatrix} R & S \\ 0 & 0 \end{bmatrix} \Pi\mathbf{x} \right\|_2^2 \\ &= \left\| Q\mathbf{b} - \begin{bmatrix} R & S \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix} \right\|_2^2 \\ &= \left\| \begin{bmatrix} \mathbf{c} \\ \mathbf{d} \end{bmatrix} - \begin{bmatrix} R\mathbf{u} + S\mathbf{v} \\ \mathbf{0} \end{bmatrix} \right\|_2^2 \\ &= \|\mathbf{c} - R\mathbf{u} - S\mathbf{v}\|_2^2 + \|\mathbf{d}\|_2^2. \end{aligned}$$

Thus $\min \|\mathbf{b} - A\mathbf{x}\|_2^2 = \|\mathbf{d}\|_2^2$ provided that $R\mathbf{u} + S\mathbf{v} = \mathbf{c}$. A basic solution is obtained by choosing $\mathbf{v} = \mathbf{0}$. A second solution is to choose $\mathbf{u}$ and $\mathbf{v}$ so that $\|\mathbf{u}\|_2^2 + \|\mathbf{v}\|_2^2$ is minimized. This criterion is related to the pseudoinverse of A.

Suppose

$$A = Q^T \begin{bmatrix} R & S \\ 0 & 0 \end{bmatrix} \Pi$$

where R is upper triangular. Then

$$A^T = \Pi^T \begin{bmatrix} R^T & 0 \\ S^T & 0 \end{bmatrix} Q$$

where R^T is lower triangular. We apply Householder reflections so that

$$H_i \cdots H_2 H_1 \begin{bmatrix} R^T & 0 \\ S^T & 0 \end{bmatrix} = \begin{bmatrix} U & 0 \\ 0 & 0 \end{bmatrix}.$$

Then

$$A^T = Z^T \begin{bmatrix} U & 0 \\ 0 & 0 \end{bmatrix} Q$$

where $Z = H_i \cdots H_1 \Pi$. In other words,

$$A = Q^T \begin{bmatrix} L & 0 \\ 0 & 0 \end{bmatrix} Z$$

where L is a lower triangular matrix of size $r \times r$, where r is the rank of A. This is the *complete orthogonal decomposition* of A.

Recall that X is the *pseudoinverse* of A if

1. $AXA = A$
2. $XAX = X$
3. $(XA)^T = XA$
4. $(AX)^T = AX$

Given the above complete orthogonal decomposition of A, the pseudoinverse of A, denoted A^+, is given by

$$A^+ = Z^T \begin{bmatrix} L^{-1} & 0 \\ 0 & 0 \end{bmatrix} Q.$$

Let $\mathcal{X} = \{\mathbf{x} | \|\mathbf{b} - A\mathbf{x}\|_2 = \min\}$. If $\mathbf{x} \in \mathcal{X}$ and we desire $\|\mathbf{x}\|_2 = \min$, then $\mathbf{x} = A^+\mathbf{b}$. Note that in this case,

$$\mathbf{r} = \mathbf{b} - A\mathbf{x} = \mathbf{b} - AA^+\mathbf{b} = (I - AA^+)\mathbf{b}$$

where the matrix $(I - AA^+)$ is a projection matrix $P^\perp$. To see that $P^\perp$ is a projection, note that

$$\begin{aligned} P &= AA^+ \\ &= Q^T \begin{bmatrix} L & 0 \\ 0 & 0 \end{bmatrix} ZZ^T \begin{bmatrix} L^{-1} & 0 \\ 0 & 0 \end{bmatrix} Q \\ &= Q^T \begin{bmatrix} I_r & 0 \\ 0 & 0 \end{bmatrix} Q. \end{aligned}$$

5.6 Least Squares with Linear Constraints

Suppose that we wish to fit data as in the least squares problem, except that we are using different functions to fit the data on different subintervals. A common example is the process of fitting data using cubic splines, with a different cubic polynomial approximating data on each subinterval.

Typically, it is desired that the functions assigned to each piece form a function that is continuous on the entire interval within which the data lies. This requires that *constraints* be imposed on the functions themselves. It is also not uncommon to require that the function assembled from these pieces

also has a continuous first or even second derivative, resulting in additional constraints. The result is a *least squares problem with linear constraints*, as the constraints are applied to coefficients of predetermined functions chosen as a basis for some function space, such as the space of polynomials of a given degree.

The general form of a least squares problem with linear constraints is as follows: we wish to find an n-vector $\mathbf{x}$ that minimizes $\|A\mathbf{x} - \mathbf{b}\|_2$, subject to the constraint $C^T\mathbf{x} = \mathbf{d}$, where C is a known $n \times p$ matrix and $\mathbf{d}$ is a known p-vector.

This problem is usually solved using *Lagrange multipliers*. We define

$$f(x; \lambda) = \|\mathbf{b} - A\mathbf{x}\|_2^2 + 2\lambda^T C^T \mathbf{x}.$$

Then

$$\nabla f = 2(A^T A\mathbf{x} - A^T\mathbf{b} + C\lambda).$$

To minimize f, we can solve the system

$$\begin{bmatrix} A^T A & C \\ C^T & 0 \end{bmatrix} \begin{bmatrix} \hat{\mathbf{x}} \\ \lambda \end{bmatrix} = \begin{bmatrix} A^T\mathbf{b} \\ \mathbf{d} \end{bmatrix}.$$

From $A^T A\mathbf{x} = A^T\mathbf{b} - C\lambda$, we see that we can first compute $\mathbf{x} = \hat{\mathbf{x}} - (A^T A)^{-1}C\lambda$ where $\hat{\mathbf{x}}$ is the solution to the unconstrained least squares problem. Then, from the equation $C^T\mathbf{x} = \mathbf{d}$ we obtain the equation $C^T(A^T A)^{-1}C\lambda = C^T\hat{\mathbf{x}} - \mathbf{d}$ which we can now solve for λ. The algorithm proceeds as follows:

1. Solve the unconstrained least squares problem $A\mathbf{x} = \mathbf{b}$ for $\hat{\mathbf{x}}$.
2. Compute $A = QR$.
3. Form $W = (R^T)^{-1}C$.
4. Compute $W = PU$, the QR factorization of W.
5. Solve $U^T U\lambda = \eta = C^T\hat{\mathbf{x}} - \mathbf{d}$ for λ. Note that

$$\begin{aligned} U^T U &= (P^T W)^T (P^T W) \\ &= W^T P P^T W \\ &= C^T R^{-1} (R^T)^{-1} C \\ &= C^T (R^T R)^{-1} C \\ &= C^T (R^T Q^T Q R)^{-1} C \\ &= C^T (A^T A)^{-1} C \end{aligned}$$

6. Set $\mathbf{x} = \hat{\mathbf{x}} - (A^T A)^{-1} C\lambda$.

This method is not the most practical since it has more unknowns than the unconstrained least squares problem, which is odd because the constraints should have the effect of eliminating unknowns, not adding them. We now describe an alternate approach.

Suppose that we compute the QR factorization of C to obtain

$$Q^T C = \begin{bmatrix} R \\ 0 \end{bmatrix}$$

where R is a $p \times p$ upper triangular matrix. Then the constraint $C^T\mathbf{x} = \mathbf{d}$ takes the form

$$R^T\mathbf{u} = \mathbf{d}, \quad Q^T\mathbf{x} = \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix}.$$

Then

$$\begin{aligned} \|\mathbf{b} - A\mathbf{x}\|_2 &= \|\mathbf{b} - AQQ^T\mathbf{x}\| \\ &= \left\|\mathbf{b} - \tilde{A}\begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix}\right\|_2, \quad \tilde{A} = AQ \\ &= \left\|\mathbf{b} - \begin{bmatrix} \tilde{A}_1 & \tilde{A}_2 \end{bmatrix}\begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix}\right\|_2 \\ &= \|\mathbf{b} - \tilde{A}_1\mathbf{u} - \tilde{A}_2\mathbf{v}\|_2 \end{aligned}$$

Thus we can obtain $\mathbf{x}$ by the following procedure:

1. Compute the QR factorization of C
2. Compute $\tilde{A} = AQ$
3. Solve $R^T\mathbf{u} = \mathbf{d}$
4. Solve the new least squares problem of minimizing $\|(\mathbf{b} - \tilde{A}_1\mathbf{u}) - \tilde{A}_2\mathbf{v}\|_2$
5. Compute
$$\mathbf{x} = Q\begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix}.$$

This approach has the advantage that there are fewer unknowns in each system that needs to be solved, and also that $\kappa(\tilde{A}_2) \leq \kappa(A)$. The drawback is that sparsity can be destroyed.

5.7 Least Squares with Quadratic Constraints

We wish to solve the problem

$$\|\mathbf{b} - A\mathbf{x}\|_2 = \min, \quad \|\mathbf{x}\|_2 = \alpha, \quad \alpha \leq \|A^+\mathbf{b}\|_2.$$

This problem is known as *least squares with quadratic constraints*. To solve this problem, we define

$$\varphi(\mathbf{x};\mu) = \|\mathbf{b} - A\mathbf{x}\|_2^2 + \mu\|\mathbf{x}^2 - \alpha^2\|$$

and seek to minimize φ. From

$$\nabla\varphi = 2A^T\mathbf{b} - 2A^TA\mathbf{x} + 2\mu\mathbf{x}$$

we obtain the system

$$(A^TA + \mu I)\mathbf{x} = A^T\mathbf{b}.$$

If we denote the eigenvalues of A^TA by

$$\lambda_i(A^TA) = \lambda_1, \ldots, \lambda_n, \quad \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \geq 0$$

then

$$\lambda_i(A^TA + \mu I) = \lambda_1 + \mu, \cdots, \lambda_n + \mu.$$

If $\mu \geq 0$, then $\kappa(A^TA + \mu I) \leq \kappa(A^TA)$, because

$$\frac{\lambda_1 + \mu}{\lambda_n + \mu} \leq \frac{\lambda_1}{\lambda_n},$$

so $A^TA + \mu I$ is better conditioned.

Solving the least squares problem with quadratic constraints arises in many literatures, including

1. Statistics: Ridge Regression
2. Regularization: Tichonov
3. Generalized cross-validation (GCV)

To solve this problem, we see that we need to compute

$$\mathbf{x} = (A^TA + \mu I)^{-1}A^T\mathbf{b}$$

where

$$\mathbf{x}^T\mathbf{x} = \mathbf{b}^TA(A^TA + \mu I)^{-2}A^T\mathbf{b} = \alpha^2.$$

If $A = U\Sigma V^T$ is the SVD of A, then we have

$$\begin{aligned}
\alpha^2 &= \mathbf{b}^T U\Sigma V^T (V\Sigma^T\Sigma V^T + \mu I)^{-2} V\Sigma^T U^T \mathbf{b} \\
&= \mathbf{c}^T \Sigma(\Sigma^T\Sigma + \mu I)^{-2}\Sigma^T \mathbf{c}, \quad U^T\mathbf{b} = \mathbf{c} \\
&= \sum_{i=1}^{r} \frac{c_i^2\sigma_i^2}{(\sigma_i^2+\mu)^2} \\
&= \chi(\mu)
\end{aligned}$$

The function $\chi(\mu)$ has poles at $-\sigma_i^2$ for $i = 1, \ldots, n$. Furthermore, $\lim_{\mu\to\infty}\chi(\mu) = 0$.

We now have the following procedure for solving this problem, given A, $\mathbf{b}$, and α^2:

1. Compute the SVD of A to obtain $A = U\Sigma V^T$.
2. Compute $\mathbf{c} = U^T\mathbf{b}$.
3. Solve $\chi(\mu^*) = \alpha^2$ where $\mu^* \geq 0$. Don't use Newton's method on this equation directly; solving $1/\chi(\mu) = 1/\alpha^2$ is much better.
4. Use the SVD to compute

$$\mathbf{x} = (A^TA + \mu I)^{-1}A^T\mathbf{b} = V(\Sigma^T\Sigma + \mu I)^{-1}\Sigma^T U^T\mathbf{b}.$$

5.8 Applications of the SVD

5.8.1 Minimum-norm least squares solution

One of the most well-known applications of the SVD is that it can be used to obtain the solution to the problem

$$\|\mathbf{b} - A\mathbf{x}\|_2 = \min, \quad \|\mathbf{x}\|_2 = \min.$$

The solution is

$$\hat{\mathbf{x}} = A^+\mathbf{b} = V\Sigma^+U^T\mathbf{b}$$

where A^+ is the *pseudo-inverse* of A.

5.8.2 Closest Orthogonal Matrix

Let $\mathcal{Q}_n$ be the set of all $n \times n$ orthogonal matrices. Given an $n \times n$ matrix A, we wish to find the matrix Q that satisfies

$$\|A - Q\|_F = \min, \quad Q \in \mathcal{Q}_n, \quad \sigma_i(Q) = 1.$$

Given $A = U\Sigma V^T$, if we compute $\hat{Q} = UIV^T$, then

$$\begin{aligned} \|A - \hat{Q}\|_F^2 &= \|U(\Sigma - I)V^T\|_F^2 \\ &= \|\Sigma - I\|_F^2 \\ &= (\sigma_1 - 1)^2 + \cdots + (\sigma_n - 1)^2 \end{aligned}$$

It can be shown that this is in fact the minimum.

A more general problem is to find $Q \in \mathcal{Q}_n$ such that

$$\|A - BQ\|_F = \min$$

for given matrices A and B. The solution is

$$\hat{Q} = UV^T, \quad B^T A = U\Sigma V^T.$$

5.8.3 Low-Rank Approximations

Let $\mathcal{M}_{m,n}^{(r)}$ be the set of all $m \times n$ matrices of rank r, and let $A \in \mathcal{M}_{m,n}^{(r)}$. We wish to find $B \in \mathcal{M}_{m,n}^{(k)}$, where $k < r$, such that $\|A - B\|_F = \min$.

To solve this problem, let $A = U\Sigma V^T$ be the SVD of A, and let $\hat{B} = U\Omega_k V^T$ where

$$\Omega_k = \begin{bmatrix} \sigma_1 & & & & & \\ & \ddots & & & & \\ & & \sigma_k & & & \\ & & & 0 & & \\ & & & & \ddots & \\ & & & & & 0 \end{bmatrix}.$$

Then

$$\begin{aligned} \|A - \hat{B}\|_F^2 &= \|U(\Sigma - \Omega_k)V^T\|_F^2 \\ &= \|\Sigma - \Omega_k\|_F^2 \\ &= \sigma_{k+1}^2 + \cdots + \sigma_r^2. \end{aligned}$$

We now consider a variation of this problem. Suppose that B is a perturbation of A such that $A = B + E$, where $\|E\|_F^2 \leq \epsilon^2$. We wish to find $\hat{B}$ such that $\|A - \hat{B}\|_F^2 \leq \epsilon^2$, where the rank of $\hat{B}$ is minimized. We know that if $B_k = U\Omega_k V^T$ then

$$\|A - B_K\|_F^2 = \sigma_{k+1}^2 + \cdots + \sigma_r^2.$$

It follows that $\hat{B} = B_k$ is the solution if

$$\sigma_{k+1} + \cdots + \sigma_r^2 \leq \epsilon^2, \quad \sigma_k^2 + \cdots + \sigma_r^2 > \epsilon^2.$$

Note that

$$\|A^+ - \hat{B}^+\|_F^2 = \left(\frac{1}{\sigma_{k+1}^2} + \cdots + \frac{1}{\sigma_r^2} \right).$$

5.9 Total Least Squares

In the ordinary least squares problem, we are solving

$$A\mathbf{x} = \mathbf{b} + \mathbf{r}, \quad \|\mathbf{r}\|_2 = \min.$$

In the *total least squares* problem, we wish to solve

$$(A + E)\mathbf{x} = \mathbf{b} + \mathbf{r}, \quad \|E\|_F^2 + \lambda^2 \|\mathbf{r}\|_2^2 = \min.$$

From $A\mathbf{x} - \mathbf{b} + E\mathbf{x} - \mathbf{r}$ we obtain the system

$$\begin{bmatrix} A & \mathbf{b} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ -1 \end{bmatrix} + \begin{bmatrix} E & \mathbf{r} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ -1 \end{bmatrix} = \mathbf{0},$$

or

$$(C + F)\mathbf{z} = \mathbf{0}.$$

We need the matrix $C + F$ to have rank $\leq n + 1$, and we want to minimize $\|F\|$.

To solve this problem, we compute the SVD of $C = \begin{bmatrix} A & \mathbf{b} \end{bmatrix} = U\Sigma V^T$. Let $\hat{C} = U\Omega_n V^T$. Then, if $\mathbf{v}_i$ is the ith column of V, we have

$$\hat{C}\mathbf{v}_{n+1} = U\Omega_n V^T \mathbf{v}_{n+1} = \mathbf{0}.$$

Our solution is

$$\begin{bmatrix} \hat{\mathbf{x}} \\ -1 \end{bmatrix} = -\frac{1}{v_{n+1,n+1}} \mathbf{v}_{n+1}$$

provided that $v_{n+1,n+1} \neq 0$.

Now, suppose that only some of the data is contaminated, i.e. $E = \begin{bmatrix} 0 & E_1 \end{bmatrix}$ where the first p columns of E are zero. Then, in solving $(C + F)\mathbf{z} = \mathbf{0}$, we use Householder transformations to compute $Q^T(C+F)$ where the first p columns are zero below the diagonal. Since $\|F\|_F = \|Q^T F\|_F$, we then have a block upper triangular system

$$\begin{bmatrix} R_{11} & R_{12} + F_{12} \\ 0 & R_{22} + F_{22} \end{bmatrix} \mathbf{z} = \mathbf{0}, \quad \mathbf{z} = \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix}.$$

We can find the total least squares solution of

$$(R_{22} + F_{22})\mathbf{v} = \mathbf{0},$$

and then set $F_{12} = 0$ and solve

$$R_{11}\mathbf{u} + R_{12}\mathbf{v} = \mathbf{0}.$$

Chapter 6

Iterative Methods

The topic of interest of this chapter is iterative methods, which are methods that attempt to generate a sequence of approximations $x^{(i)}$ that converge to the true solution x. Such methods are most useful in solving very large, naturally sparse problems that arise from applications, such as the discretization of PDEs. They are also used in some dense problems in which the coefficient matrix is structured.

6.1 Stationary methods

Given $A\mathbf{x} = \mathbf{b}$, write $M\mathbf{x} = N\mathbf{x} + \mathbf{b}$ (where M is invertible) and construct the iteration

$$M\mathbf{x}^{(k+1)} = N\mathbf{x}^{(k)} + \mathbf{b}.$$

Subtracting these equations, we obtain

$$M(\mathbf{x} - \mathbf{x}^{(k+1)}) = N(\mathbf{x} - \mathbf{x}^{(k)}).$$

Therefore if we denote the error in $\mathbf{x}^{(k)}$ by $\mathbf{e}^{(k)} = \mathbf{x} - \mathbf{x}^{(k)}$, then

$$e^{(k+1)} = M^{-1}N\mathbf{e}^{(k)} \equiv B\mathbf{e}^{(k)}.$$

Thus $\mathbf{e}^{(k)} = B^k\mathbf{e}^{(0)}$, which suggests the following theorem:

Theorem $\mathbf{e}^{(k)} \to 0$ as $k \to \infty$ for all $\mathbf{e}^{(0)}$ if and only if $\rho(B) < 1$.

Convergence can still occur if $\rho(B) = 1$, but in that case we must be careful in how we choose $\mathbf{x}^{(0)}$.

Note that from $\mathbf{e}^{(k)} = B^k\mathbf{e}^{(0)}$, it follows that

$$\frac{\|\mathbf{e}^{(k)}\|}{\|\mathbf{e}^{(0)}\|} \leq \|B\|^k.$$

6.1.1 The Jacobi Method

We now develop a simple iterative method. If we rewrite $A\mathbf{x} = \mathbf{b}$ as

$$\sum_{j=1}^{n} a_{ij}x_j = b_i, \quad i = 1, \ldots, n,$$

then

$$a_{ii}x_i = b_i - \sum_{i \neq j} a_{ij}x_j,$$

or

$$x_i = \frac{1}{a_{ii}} \left(b_i - \sum_{j \neq i} a_{ij}x_j \right).$$

In other words,

$$M = \begin{bmatrix} a_{11} & & \\ & \ddots & \\ & & a_{nn} \end{bmatrix}, \quad N = - \begin{bmatrix} 0 & a_{12} & \cdots & a_{1n} \\ a_{21} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ a_{n1} & \cdots & a_{n,n-1} & 0 \end{bmatrix}.$$

Our iteration is therefore

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j \neq i} a_{ij}x^{(k)} \right),$$

known as the *Jacobi method*, with

$$M^{-1}N = \begin{bmatrix} 0 & \frac{a_{12}}{a_{11}} & \cdots & \frac{a_{1n}}{a_{11}} \\ \frac{a_{21}}{a_{22}} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ \frac{a_{n1}}{a_{nn}} & \cdots & \frac{a_{n,n-1}}{a_{nn}} & 0 \end{bmatrix} \equiv B_J.$$

So, if

$$\|M^{-1}N\|_\infty = \max_{1 \leq i \leq n} \sum_{j \neq i} \left| \frac{a_{ij}}{a_{ii}} \right| < 1,$$

i.e. if B_J is *strictly diagonally dominant*, then the iteration converges.

For example, suppose

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

Then $\|B_J\|_\infty = \frac{1}{2}$, so the Jacobi method converges rapidly. On the other hand, if

$$A = \begin{bmatrix} 2 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2 \end{bmatrix},$$

which arises from discretizing the Laplacian, then $\|B_J\|_\infty = 1$. A more subtle analysis can be used to show convergence, but it is slow.

Note that for these two examples, $\mathbf{x}^{(0)} \leftarrow \mathbf{x}^{(1)}$ when *all* elements of $\mathbf{x}^{(1)}$ have been computed. This is a waste of storage; we need only $(n+2)$ elements of storage of A above. This shows that the ordering of equations is *very* important. If we reorder the equations in such a way that odd-numbered equations and even-numbered equations are grouped separately, then we obtain, for the latter example,

$$A = \begin{bmatrix} 2 & & & & & -1 & & & & \\ & 2 & & & & -1 & -1 & & & \\ & & \ddots & & & & -1 & -1 & & \\ & & & \ddots & & & & \ddots & \ddots & \\ & & & & & & & & & \\ -1 & -1 & & & & 2 & & & & \\ & -1 & -1 & & & & 2 & & & \\ & & \ddots & \ddots & & & & \ddots & & \\ & & & \ddots & \ddots & & & & \ddots & \\ & & & & -1 & & & & & 2 \end{bmatrix}.$$

Then, we can solve for all odd indices, then all even indicies, independently of each other. Not only does this approach save storage space but it also lends itself to parallelism.

6.1.2 The Gauss-Seidel Method

In the Jacobi method, we compute $x_i^{(k+1)}$ using the elements of $\mathbf{x}^{(k)}$, even though $x_1^{(k+1)}, \ldots, x_{i-1}^{(k+1)}$ are already known. The *Gauss-Seidel* method is designed to take advantage of the latest information available about $\mathbf{x}$:

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)} \right).$$

To derive this method, we write $A = L + D + U$ where

$$L = \begin{bmatrix} 0 & & & \\ a_{21} & \ddots & & \\ \vdots & & \ddots & \\ a_{n1} & \cdots & a_{n,n-1} & 0 \end{bmatrix}, \quad D = \begin{bmatrix} a_{11} & & & \\ & \ddots & & \\ & & a_{nn} & \end{bmatrix}, \quad U = \begin{bmatrix} 0 & a_{12} & \cdots & a_{1n} \\ & \ddots & & \vdots \\ & & \ddots & a_{n-1,n} \\ & & & 0 \end{bmatrix}.$$

Thus the Gauss-Seidel iteration can be written as

$$D\mathbf{x}^{(k+1)} = \mathbf{b} - L\mathbf{x}^{(k+1)} - U\mathbf{x}^{(k)},$$

or

$$(D + L)\mathbf{x}^{(k+1)} = \mathbf{b} - U\mathbf{x}^{(k)}$$

which yields

$$\mathbf{x}^{(k+1)} = -(D + L)^{-1} U\mathbf{x}^{(k)} + (D + L)^{-1}\mathbf{b}.$$

Thus the iteration matrix for the Gauss-Seidel method is $B_{GS} = -(D + L)^{-1}U$, as opposed to the iteration matrix for the Jacobi method, $B_J = -D^{-1}(L + U)$. In some cases, $\rho(B_{GS}) = (\rho(B_J))^2$, so the Gauss-Seidel method converges twice as fast. (We defer the analysis of convergence to section 6.3.1.) On the other hand, note that Gauss-Seidel is very sequential; i.e. it does not lend itself to parallelism.

6.1.3 The SOR Method

The method of *successive overrelaxation* (SOR) is the iteration

$$x_i^{(k+1)} = \frac{\omega}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{N} a_{ij} x_j^{(k)} \right) + (1 - \omega) x_i^{(k)}.$$

The quantity ω is called the *relaxation parameter*. If $\omega = 1$, then the SOR method reduces to the Gauss-Seidel method.

In matrix form, the iteration can be written as

$$D\mathbf{x}^{(k+1)} = \omega(\mathbf{b} - L\mathbf{x}^{(k+1)} - U\mathbf{x}^{(k)}) + (1-\omega)D\mathbf{x}^{(k)}$$

which can be rearranged to obtain

$$(D + \omega L)\mathbf{x}^{(k+1)} = \omega\mathbf{b} + [(1-\omega)D - \omega U]\mathbf{x}^{(k)}$$

or

$$\mathbf{x}^{(k+1)} = \left(\frac{1}{\omega}D + L\right)^{-1}\left[\left(\frac{1}{\omega} - 1\right)D - U\right]\mathbf{x}^{(k)} + \left(\frac{1}{\omega}D + L\right)^{-1}\mathbf{b}.$$

Define

$$\mathcal{L}_\omega = \left(\frac{1}{\omega}D + L\right)^{-1}\left[\left(\frac{1}{\omega} - 1\right)D - U\right].$$

Then

$$\begin{aligned}
\det \mathcal{L}_\omega &= \det\left(\frac{1}{\omega}D + L\right)^{-1} \det\left[\left(\frac{1}{\omega} - 1\right)D - U\right] \\
&= \frac{1}{\det\left(\frac{1}{\omega}D + L\right)} \det\left[\left(\frac{1}{\omega} - 1\right)D - U\right] \\
&= \frac{\omega^n}{\prod_{i=1}^n a_{ii}} \frac{(1-\omega)^n \prod_{i=1}^n a_{ii}}{\omega^n} \\
&= (1-\omega)^n.
\end{aligned}$$

Therefore, $\prod_{i=1}^n \lambda_i = (1-\omega)^n$ where $\lambda_1, \ldots, \lambda_n$ are the eigenvalues of $\mathcal{L}_\omega$, with $|\lambda_1| \geq \cdots \geq |\lambda_n|$. Therefore $|\lambda_1|^n \geq (1-\omega)^n$. Since we must have $|\lambda_1| < 1$ for convergence, it follows that a necessary condition for convergence of SOR is

$$0 < \omega < 2.$$

6.2 Poisson's Equation

Consider the standard problem of solving Poisson's equation on a domain R in two dimensions,

$$-\Delta u = f, \quad (x,y) \in R, \quad \Delta u = u_{xx} + u_{yy},$$

$$u = g, \quad (x,y) \in \partial R.$$

We take R to be the unit rectangle $[0,1] \times [0,1]$ and discretize the problem using a uniform grid with spacing $h = 1/(N+1)$ in the x and y directions, and gridpoints $x_i = ih$, $i = 0, \ldots, N+1$, and $y_j = jh$, $j = 0, \ldots, N+1$. Then, for $i, j = 1, \ldots, N$, we replace the differential equation by a difference approximation

$$\frac{-u_{i-1,j} + 2u_{ij} - u_{i+1,j}}{h^2} + \frac{-u_{i,j+1} + 2u_{ij} - u_{i,j+1}}{h^2} = f_{ij},$$

where $u_{ij} = u(x_i, y_j)$ and $f_{ij} = f(x_i, y_j)$. From the boundary conditions, we have

$$u_{0j} = g(x_0, y_j), \quad j = 1, 2, \ldots, N,$$

and similar conditions for the other gridpoints along the boundary.

Let $\mathbf{u}_j = \begin{bmatrix} u_{1j} & \cdots & u_{Nj} \end{bmatrix}^T$. Then

$$-\mathbf{u}_{j-1} + T\mathbf{u}_j - \mathbf{u}_{j+1} = \tilde{\mathbf{f}}_j$$

where

$$T = \begin{bmatrix} 4 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 4 \end{bmatrix}, \quad [\tilde{f}_j]_i = \begin{cases} h^2 f_{1j} + g(x_0, j) & i = 1 \\ h^2 f_{ij} & i = 2, \ldots, N-1 \\ h^2 f_{Nj} + g(x_N, j) & i = N \end{cases}$$

Thus we can solve the problem on the entire domain by solving $A\mathbf{u} = \tilde{\mathbf{f}}$ where

$$A = \begin{bmatrix} T & -I & & & \\ -I & T & -I & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & -I \\ & & & -I & T \end{bmatrix}.$$

We say that A is a *block tridiagonal matrix*. A is also a *band* matrix, but the band is sparse and Gaussian elimination may fill-in the whole band. However, the equations can be re-ordered to avoid fill-in.

6.2.1 Eigenvalues of Tridiagonal Toeplitz Matrices

We will now show how we can find eigenvalues and eigenvectors of certain tridiagonal toeplitz matrices that frequently arise in difference approxima-

tions. Let

$$\hat{T} = \begin{bmatrix} 0 & 1 & & \\ 1 & \ddots & \ddots & \\ & \ddots & \ddots & 1 \\ & & 1 & 0 \end{bmatrix}, \quad T(a,b) = \begin{bmatrix} a & b & & \\ b & \ddots & \ddots & \\ & \ddots & \ddots & b \\ & & b & a \end{bmatrix} = aI + b\hat{T}.$$

Note that $\lambda_j(T(a,b)) = a + b\lambda_j(\hat{T})$. We first study the case where $a = 0$ and $b = 1$; then we will consider the case $a = 4$, $b = -1$ arising from Poisson's equation.

Consider $\hat{T}\mathbf{v} = \lambda\mathbf{v}$. We can write this as a system of equations

$$\begin{aligned} v_{j-1} + v_{j+1} &= \lambda v_j \\ v_2 &= \lambda v_1 \\ v_{N-1} &= \lambda v_N \end{aligned}$$

Since $\hat{T}$ is symmetric, it has the decomposition $\hat{T} = V\Lambda V^T$, and therefore we can write $T(a,b) = V\Lambda(a,b)V^T$ where $\Lambda(a,b) = aI + b\Lambda$.

We guess that

$$v_j = A\sin j\theta + B\cos j\theta.$$

Substituting this representation into $T\mathbf{v} = \lambda\mathbf{v}$ yields

$$\begin{aligned} \lambda v_j &= \lambda(A\sin j\theta + B\cos j\theta) \\ &= A\sin(j-1)\theta + B\cos(j-1)\theta + A\sin(j+1)\theta + B\cos(j+1)\theta \\ &= A[\sin(j-1)\theta + \sin(j+1)\theta] + B[\cos(j-1)\theta + \cos(j+1)\theta] \\ &= A(2\sin j\theta\cos\theta) + B(2\cos\theta\cos j\theta) \\ &= 2\cos\theta v_j \end{aligned}$$

which yields $\lambda = 2\cos\theta$.

We use the boundary conditions to find θ. Our representation of v_j yields

$$\begin{aligned} A\sin 2\theta + B\cos 2\theta &= 2\cos\theta(A\sin\theta + B\cos\theta) \\ A\sin(N-1)\theta + B\cos(N-1)\theta &= 2\cos\theta(A\sin N\theta + B\cos N\theta) \end{aligned}$$

which can be written as a system of two equations for the two unknowns A and B,

$$\begin{aligned} (\sin 2\theta - 2\cos\theta\sin\theta)A + (\cos 2\theta - 2\cos\theta\sin\theta)B &= 0 \\ (\sin(N-1)\theta - 2\cos\theta\sin N\theta)A + (\cos(N-1)\theta - 2\cos\theta\cos\theta) &= 0 \end{aligned}$$

or, in matrix form,

$$\begin{bmatrix} 0 & -1 \\ \times & \times \end{bmatrix} \begin{bmatrix} A \\ B \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

which yields $B = 0$. In order for A to be nonzero, we must have

$$\begin{aligned} 0 &= \sin(N_1)\theta - 2\cos\theta \sin N\theta \\ &= \sin N\theta \cos\theta - \sin\theta \cos N\theta - 2\cos\theta \sin N\theta \\ &= -\sin N\theta \cos\theta - \sin\theta \cos N\theta \\ &= -\sin(N+1)\theta \end{aligned}$$

which yields

$$\theta_k = \frac{j\pi}{N+1}, \quad \lambda_k = 2\cos\left(\frac{k\pi}{N+1}\right).$$

Thus the largest eigenvalue is $\lambda_1 = 2\cos\pi h \approx 2 = \|\hat{T}\|_\infty$. Note that the eigenvalues are not uniformly distributed on the interval $[0, 2]$.

The eigenvectors are given by

$$v_{kj} = A\sin\left(\frac{kj\pi}{N+1}\right).$$

We want normalized eigenvectors, so we take A so that $\|\mathbf{v}_k\|_2^2 = 1$, which yields

$$A = \sqrt{\frac{2}{N+1}}.$$

Recall that $T(a, b) = aI + b\hat{T}$, where $\hat{T} = V\Lambda V^T$ and $V = \begin{bmatrix} \mathbf{v}_1 & \cdots & \mathbf{v}_N \end{bmatrix}$. Thus $\lambda_k(a, b) = a + 2b\cos\frac{k\pi}{N+1}$.

Suppose $T(a, b)\mathbf{u} = \mathbf{e}$. Then the solution $\mathbf{u}$ is given by

$$\mathbf{u} = V\Lambda^{-1}V^T\mathbf{e} = V\Lambda^{-1}\hat{\mathbf{e}}$$

where

$$\hat{e}_k = \sum_{i=1}^{N} \sqrt{\frac{2}{N+1}} \sin\left(\frac{ik\pi}{N+1}\right) e_i = \mathbf{v}_k^T\mathbf{e}.$$

This can be computed quickly using the FFT. Similarly, we can use the inverse FFT to compute $V(\Lambda^{-1}\hat{\mathbf{e}})$.

We now wish to find the eigenvalues of

$$A = \begin{bmatrix} T & -I & & & \\ -I & T & -I & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & -I \\ & & & -I & T \end{bmatrix}.$$

If we define

$$Q = \begin{bmatrix} V & & \\ & \ddots & \\ & & V \end{bmatrix},$$

then

$$Q^T A Q = \hat{A} = \begin{bmatrix} \Lambda & -I & & & \\ -I & \Lambda & -I & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & -I \\ & & & -I & \Lambda \end{bmatrix}.$$

The system $\hat{A}\mathbf{w} = \mu\mathbf{w}$ has equations of the form

$$-w_{i,j-1} + \lambda w_{ij} - w_{i,j+1} = \mu w_{ij}, \quad i = 1, \ldots, N.$$

If we reorder the unknowns by columns instead of rows, then we obtain a block diagonal matrix where each diagonal block is a tridiagonal block of the form $T_k(\lambda_k, -1)$, where λ_k is an eigenvalue of T. The matrix $T_k(\lambda_k, -1)$ has eigenvalues

$$\lambda_j(T_k(\lambda_k, -1)) = \lambda_k - 2\cos\frac{j\pi}{N+1}, \quad j = 1, \ldots, N+1.$$

Therefore the eigenvalues of A are given by

$$\mu_{rs} = 4 - 2\cos\frac{r\pi}{N+1} - 2\cos\frac{s\pi}{N+1}, \quad r, s = 1, \ldots, N+1.$$

It follows that

$$\mu_{\min} = 4 - 4\cos\frac{\pi}{N+1}, \quad \mu_{\max} = 4 - 4\cos\frac{N\pi}{N+1} = 4 + 4\cos\frac{\pi}{N+1}.$$

Observe that $\mu_{\max} \leq \|A\|_\infty = 8$. However, as $N \to \infty$, $\mu_{\min} \to 0$, so the matrix becomes ill-conditioned quite rapidly as $N \to \infty$.

6.2.2 The Helmholtz Equation

Poisson's equation $-\Delta u = f$ occurs very often in applications, as does the *Helmholtz equation*

$$-\Delta u + \sigma(x,y)u = f.$$

For this equation, we have the discretization

$$A\mathbf{u} + h^2\Sigma\mathbf{u} = \mathbf{f}$$

where A is an $N^2 \times N^2$ matrix of the form

$$A = \begin{bmatrix} T & -I & & \\ -I & \ddots & \ddots & \\ & \ddots & \ddots & -I \\ & & -IT & \end{bmatrix},$$

and

$$\Sigma = \begin{bmatrix} \sigma_{11} & & \\ & \ddots & \\ & & \sigma_{NN} \end{bmatrix}, \quad \sigma_{ij} = \sigma(x_i, y_j).$$

Recall from last time that A has eigenvalues

$$\mu_{rs} = 4 - 2\cos\frac{\pi r}{N+1} - 2\cos\frac{\pi s}{N+1}, \quad r,s = 1,\dots N.$$

To solve this system, we can use the iteration

$$A\mathbf{u}^{(k+1)} = h^2\Sigma\mathbf{u}^{(k)} + \mathbf{f}.$$

To determine whether this iteration converges, we will try to bound

$$\|h^2A^{-1}\Sigma\| \le h^2\|A^{-1}\|\|\Sigma\|.$$

We assume that $|\sigma(x,y)| \le \bar{\sigma}$ and note that

$$\|A^{-1}\|_2 = \frac{1}{4(1 - cos\frac{pi}{N+1})} = \frac{1}{4(1 - cos\pi h)}.$$

Using the facts

$$|\frac{\sin x}{x}| \le 1, \quad \lim_{x\to 0}\frac{\sin x}{x} = 1,$$

it follows that

$$\|h^2 A^{-1}\Sigma\| \leq \frac{h^2\bar{\sigma}}{8\sin^2\frac{\pi h}{2}} = \frac{\bar{\sigma}}{8\left(\frac{\sin\frac{\pi h}{2}}{h}\right)^2} \approx \bar{\sigma}/2\pi^2.$$

Note that this bound is independent of h, and the method converges if $\bar{\sigma} \leq 20$. Thus, the rate of convergence is essentially independent of the mesh size h, which is very desirable.

6.3 Convergence Analysis

6.3.1 Convergence of Gauss-Seidel

Recall the basic iterative methods based on the splitting $A = D + L + U$, the *Jacobi method*

$$D\mathbf{x}^{(k+1)} = -(L+U)\mathbf{x}^{(k)} + \mathbf{b}$$

and the *Gauss-Seidel method*

$$(D+L)\mathbf{x}^{(k+1)} = -U\mathbf{x}^{(k)} + \mathbf{b}.$$

These are examples of *one-step stationary method*, which is an iteration of the form

$$M\mathbf{x}^{(k+1)} = N\mathbf{x}^{(k)} + \mathbf{b},$$

where $A = M - N$.

Let $B = M^{-1}N$, and define $\mathbf{e}^{(k)} = \mathbf{x} - \mathbf{x}^{(k)}$. Then $\mathbf{e}^{(k+1)} = B\mathbf{e}^{(k)} = B^{k+1}\mathbf{e}^{(0)}$. Recall that if $\rho(B^k) < 1$ then $\mathbf{e}^{(k)} \to 0$ for all choices of $\mathbf{x}^{(0)}$. Also, recall that for all consistent norms, $\rho(B) \leq \|B\|$.

Therefore, a sufficient condition for convergence of the Jacobi method is $\|B\|_\infty < 1$ where

$$b_{ij} = \begin{cases} -\frac{a_{ij}}{a_{ii}} & i \neq j \\ 0 & i = j \end{cases}$$

Note that

$$\|B\|_\infty = \max_i \sum_{j\neq i} \left|\frac{a_{ij}}{a_{ii}}\right| < 1$$

if B is *diagonally dominant*.

Now, define

$$r_i = \sum_{i\neq j} \left|\frac{a_{ij}}{a_{ii}}\right|, \quad r = \max_i r_i.$$

Then we have the following result:

Theorem If $r < 1$, then $\rho(B_{GS}) < 1$. In other words, the Gauss-Seidel iteration converges if A is diagonally dominant.

Proof The proof proceeds using induction on the elements of $\mathbf{e}^{(k)}$. We have

$$(D + L)\mathbf{e}^{(k+1)} = U\mathbf{e}^{(k)},$$

which can be written as

$$\sum_{j=1}^{i} a_{ij} e_j^{(k+1)} = -\sum_{j=i+1}^{N} a_{ij} e_j^{(k)}, \quad i = 1, \ldots, N.$$

Thus

$$e_i^{(k+1)} = -\sum_{j=i+1}^{N} \frac{a_{ij}}{a_{ii}} e_j^{(k)} - \sum_{j=1}^{i-1} \frac{a_{ij}}{a_{ii}} e_j^{(k+1)}, \quad i = 1, \ldots, N.$$

For $i = 1$, we have

$$|e_1^{(k+1)}| \leq \sum_{j=2}^{N} \left| \frac{a_{ij}}{a_{ii}} \right| |e_j^{(k)}| \leq \|\mathbf{e}^{(k)}\|_\infty r_1.$$

Assume that for $p = 1, \ldots, i-1$,

$$|e_p^{(k+1)}| \leq \|\mathbf{e}^{(k)}\|_\infty r_p \leq r\|\mathbf{e}^{(k)}\|_\infty.$$

Then,

$$\begin{aligned}
|e_i^{(k+1)}| &\leq \sum_{j=1}^{i-1} \left| \frac{a_{ij}}{a_{ii}} \right| |e_j^{(k+1)}| + \sum_{j=i+1}^{N} \left| \frac{a_{ij}}{a_{ii}} \right| |e_j^{(k)}| \\
&\leq r\|\mathbf{e}^{(k)}\|_\infty \sum_{j=1}^{i-1} \left| \frac{a_{ij}}{a_{ii}} \right| + \|\mathbf{e}\|_\infty \sum_{j=i+1}^{N} \left| \frac{a_{ij}}{a_{ii}} \right| \\
&\leq \|\mathbf{e}^{(k)}\|_\infty \sum_{j \neq i} \left| \frac{a_{ij}}{a_{ii}} \right| \\
&= r_i \|\mathbf{e}^{(k)}\|_\infty \\
&\leq r\|\mathbf{e}^{(k)}\|_\infty.
\end{aligned}$$

Therefore

$$\|\mathbf{e}^{(k+1)}\|_\infty \leq r\|\mathbf{e}^{(k)}\|_\infty \leq r^{k+1}\|\mathbf{e}^{(0)}\|_\infty,$$

from which it follows that

$$\lim_{k\to\infty} \|\mathbf{e}^{(k)}\| = 0$$

since $r < 1$. □

We see that the Jacobi method and the Gauss-Seidel method both converge if A is diagonally dominant, but convergence can be slow in some cases. For example, if

$$A = \begin{bmatrix} 2 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2 \end{bmatrix}$$

is of size $N \times N$ then

$$-D^{-1}(L+U) = \begin{bmatrix} 0 & 1/2 & & \\ 1/2 & \ddots & \ddots & \\ & \ddots & \ddots & 1/2 \\ & & 1/2 & 0 \end{bmatrix}$$

and therefore

$$\rho(B_J) = cos\frac{\pi}{N+1} = \cos \pi h \approx 1 - \frac{\pi^2 h^2}{2} + \cdots$$

which is approximately 1 for small $h = \frac{1}{N+1}$. We would like to develop a method where $\rho(B) \approx 1 - ch$.

NOw, suppose $B = B^T$. Then

$$\frac{\|\mathbf{e}^{(k)}\|_2}{\|\mathbf{e}^{(0)}\|_2} \leq \|B\|_2^k = \rho(B)^k.$$

We want $\|\mathbf{e}^{(k)}\|_2 / \|\mathbf{e}^{(0)}\|_2 \leq \epsilon$, so if we let $\rho^k = \epsilon$, then

$$k = \frac{-\log \epsilon}{-\log \rho}$$

is the number of iterations necessary for convergence. The quantity $-\log \rho$ is called the rate of convergence.

6.3.2 Convergence of SOR on Positive-Definite Systems

To analyze the convergence of SOR, we need the following result.

Lemma Let $A = M - N$, where $A = A^*$ and M is invertible, and define $Q = M + M^* - A$. If Q and A are both positive definite, then $\rho(M^{-1}N) < 1$.

Proof Define $B = M^{-1}N = I - M^{-1}A$. It follows that if $B\mathbf{u} = \lambda\mathbf{u}$, then

$$A\mathbf{u} = (1 - \lambda)M\mathbf{u},$$

where $\lambda \neq 1$ since A is invertible. Taking the inner product of both sides with $\mathbf{u}$ yields

$$\mathbf{u}^*A\mathbf{u} = (1 - \lambda)\mathbf{u}^*M\mathbf{u},$$

but since A is symmetric positive definite, we also have

$$\mathbf{u}^*A\mathbf{u} = (1 - \bar{\lambda})\mathbf{u}^*M^*\mathbf{u}.$$

Adding these relaions yields

$$\begin{aligned} \mathbf{u}^*(M + M^*)\mathbf{u} &= \left(\frac{1}{1-\lambda} + \frac{1}{1-\bar{\lambda}}\right)\mathbf{u}^*A\mathbf{u} \\ &= 2\Re\left(\frac{1}{1-\lambda}\right)\mathbf{u}^*A\mathbf{u} \end{aligned}$$

which can be rewritten as

$$\frac{\mathbf{u}^*(Q + A)\mathbf{u}}{\mathbf{u}^*A\mathbf{u}} = 1 + \frac{\mathbf{u}^*Q\mathbf{u}}{\mathbf{u}^*A\mathbf{u}} = 2\Re\left(\frac{1}{1-\lambda}\right).$$

Since both Q and A are positive definite, we must have $2\Re\left(\frac{1}{1-\lambda}\right) > 1$. If we write $\lambda = \alpha + i\beta$, then it follows that

$$\frac{2(1-\alpha)}{(1-\alpha)^2 + \beta^2} > 1$$

which yields $\alpha^2 + \beta^2 = |\lambda|^2 < 1$. □

Let $A = D + L + U$ be positive definite with $D = I$. Then the iteration matrix for SOR is

$$\mathcal{L}_\omega = \left(\frac{1}{\omega}I + L\right)^{-1}\left(\left(\frac{1}{\omega} - 1\right)I - U\right).$$

Then $Q = M + M^* - A$ is

$$Q = \left(\frac{1}{\omega}I + L\right) + \left(\frac{1}{\omega}I + U\right) - (I + L + U) = \left(\frac{2}{\omega} - 1\right) I.$$

For convergence, we want Q to be positive definite, so we must have $2/\omega - 1 > 0$ or $0 < \omega < 2$. It follows that SOR will converge for all $0 < \omega < 2$ when A is positive definite.

6.4 Block Methods

Recall that in solving Poisson's equation on a rectangle, we needed to solve systems of the form

$$-\mathbf{v}_j + T\mathbf{v}_j - \mathbf{v}_{j+1} = \mathbf{g}_j.$$

This can be accomplished using an iteration

$$T\mathbf{v}^{(k+1)} = \mathbf{g}_j + \mathbf{v}_{j-1}^{(k)} + \mathbf{v}_{j+1}^{(k)},$$

which is an example of a *block Jacobi* iteration, since it involves solving the system $A\mathbf{u} = \mathbf{g}$ by applying the Jacobi method to A, except each block of size $N \times N$ is treated as a single element. Similarly, we can use the *block Gauss-Seidel* iteration

$$T\mathbf{v}_j^{(k+1)} = \mathbf{g}_j + \mathbf{v}_{j-1}^{(k+1)} + \mathbf{v}_j^{(k)}.$$

6.5 Richardson's Method

Consider the iteration

$$\begin{aligned} \mathbf{x}^{(k+1)} &= (I - \alpha A)\mathbf{x}^{(k)} + \alpha\mathbf{b} \\ &= \mathbf{x}^{(k)} + \alpha(\mathbf{b} - A\mathbf{x}^{(k)}) \\ &= \mathbf{x}^{(k)} + \alpha\mathbf{r}^{(k)} \end{aligned}$$

This is known as the *Richardson method*. If we define the error $\mathbf{e}^{(k)} = \mathbf{x} - \mathbf{x}^{(k)}$, then $\mathbf{e}^{(k+1)} = B_\alpha \mathbf{e}^{(k)}$ where $B_\alpha = I - \alpha A$; we want to choose the parameter α a priori so as to minimize $\|B_\alpha\|$.

Suppose A is symmetric positive definite, with eigenvalues

$$\mu_1 \geq \mu_2 \geq \cdots \mu_n > 0.$$

Since $B = I - \alpha A$, $\lambda_i = 1 - \alpha\mu_i$. We want to choose α so that $\|B_\alpha\|_2$ is minimized; i.e.

$$\min_\alpha \max_{1\le i\le n} |\lambda_i(\alpha)| = \min_\alpha \max_{1\le i\le n} |1 - \alpha\mu_i|.$$

The optimal parameter $\hat{\alpha}$ is found by solving

$$1 - \hat{\alpha}\mu_n = -(1 - \hat{\alpha}\mu_1)$$

which yields

$$\hat{\alpha} = \frac{2}{\mu_1 + \mu_n}.$$

Note that When $1 - \alpha\mu_n = -1$ that the iteration diverges, from which it follows that the method converges for $0 < \alpha < 2/\mu_n$. However, this iteration is sensitive to perturbation, and therefore bad numerically. For example, if $\mu_1 = 10$ and $\mu_n = 10^{-4}$, then the optimal α is $2/(10 + 10^{-4}$, but this is close to a value of α for which the iteration diverges, $\alpha = 2/10$.

Also, note that

$$\lambda_1(\hat{\alpha}) = 1 - \frac{2}{\mu_1 + \mu_n}\mu_1 = \frac{\mu_n - \mu_1}{\mu_1 + \mu_n},$$

and similarly,

$$\lambda_n(\hat{\alpha}) = \frac{\mu_1 - \mu_n}{\mu_1 + \mu_n} = \frac{\frac{\mu_1}{\mu_n} - 1}{\frac{\mu_1}{\mu_n} + 1} = \frac{\kappa(A) - 1}{\kappa(A) + 1}.$$

Therefore the convergence rate depends on $\kappa(A)$. (Compare this with the method for solving the Helmholtz equation in section 6.2.2, whose convergence is independent of $\kappa(A)$.)

6.6 Steepest Descent

An alternative approach is to consider the iteration

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \mathbf{r}^{(k)}$$

where α_k varies from iteration to iteration. It follows that

$$\mathbf{r}^{(k+1)} = \mathbf{b} - A\mathbf{x}^{(k+1)} = \mathbf{b} - A\mathbf{x}^{(k)} - \alpha_k A\mathbf{r}^{(k)} = \mathbf{r}^{(k)} - \alpha_k A\mathbf{r}^{(k)}.$$

We wish to choose α_k so that $[\mathbf{r}^{(k+1)}]^T A^{-1}\mathbf{r}^{(k+1)}$ is minimized. Now

$$\begin{aligned}[\mathbf{r}^{(k+1)}]^T A^{-1}\mathbf{r}^{(k+1)} &= ([\mathbf{r}^{(k)}]^T - \alpha_k[\mathbf{r}^{(k)}]^T A)A^{-1}(\mathbf{r}^{(k)} - \alpha_k A\mathbf{r}^{(k)} \\ &= [\mathbf{r}^{(k)}]^T A^{-1}\mathbf{r}^{(k)} - 2\alpha_k[\mathbf{r}^{(k)}]^T\mathbf{r}^{(k)} + \alpha_k^2[\mathbf{r}^{(k)}]^T A\mathbf{r}^{(k)} \quad (6.1)\end{aligned}$$

To find the minimium, we differentiate with respect to α_k and obtain

$$\frac{d}{d\alpha_k}([\mathbf{r}^{(k)}]^T A^{-1}\mathbf{r}^{(k+1)}) = -2[\mathbf{r}^{(k)}]^T\mathbf{r}^{(k)} + 2\alpha_k[\mathbf{r}^{(k)}]^T A\mathbf{r}^{(k)}$$

which yields

$$\hat{\alpha}_k = \frac{[\mathbf{r}^{(k)}]^T\mathbf{r}^{(k)}}{[\mathbf{r}^{(k)}]^T A\mathbf{r}^{(k)}}$$

which is well-defined since A is symmetric positive definite. This method is known as the *method of steepest descent.*

Note that

$$0 < \lambda_{\min}(A) \le \frac{\mathbf{x}^T A\mathbf{x}}{\mathbf{x}^T\mathbf{x}} \le \lambda_{\max}(A)$$

and therefore

$$\frac{1}{\lambda_{\max}(A)} \le \hat{\alpha}_k \le \frac{1}{\lambda_{\min}(A)}.$$

Substituting $\hat{\alpha}_k$ into (6.1) yields

$$\begin{aligned}[\mathbf{r}^{(k+1)}]^T A^{-1}\mathbf{r}^{(k+1)} &= [\mathbf{r}^{(k)}]^T A^{-1}\mathbf{r}^{(k)} - 2[\mathbf{r}^{(k)}]^T\mathbf{r}^{(k)}\frac{[\mathbf{r}^{(k)}]^T\mathbf{r}^{(k)}}{[\mathbf{r}^{(k)}]^T A\mathbf{r}^{(k)}} + \left(\frac{[\mathbf{r}^{(k)}]^T\mathbf{r}^{(k)}}{[\mathbf{r}^{(k)}]^T A\mathbf{r}^{(k)}}\right)^2 [\mathbf{r}^{(k)}]^T A\mathbf{r}^{(k)} \\ &= [\mathbf{r}^{(k)}]^T A^{-1}\mathbf{r}^{(k)} - \frac{([\mathbf{r}^{(k)}]^T\mathbf{r}^{(k)})^2}{[\mathbf{r}^{(k)}]^T A\mathbf{r}^{(k)}}\end{aligned}$$

and therefore

$$\frac{\|\mathbf{r}^{(k+1)}\|_{A^{-1}}^2}{\|\mathbf{r}^{(k)}\|_{A^{-1}}^2} = 1 - \frac{([\mathbf{r}^{(k)}]^T\mathbf{r}^{(k)})^2}{([\mathbf{r}^{(k)}]^T A^{-1}\mathbf{r}^{(k)})([\mathbf{r}^{(k)}]^T A\mathbf{r}^{(k)})}.$$

The *Kantorovich inequality*, which comes up very often in applications such as optimization and statistics, states that

$$\frac{\mathbf{x}^T A\mathbf{x}\cdot\mathbf{x}^T A^{-1}\mathbf{x}}{(\mathbf{x}^T\mathbf{x})^2} \le \left(\frac{\sqrt{\kappa} + (\sqrt{\kappa})^{-1}}{2}\right)^2, \quad \kappa = \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)}.$$

It follows that

$$\frac{\|\mathbf{r}^{(k+1)}\|_{A^{-1}}^2}{\|\mathbf{r}^{(k)}\|_{A^{-1}}^2} \le \left(\frac{\kappa - 1}{\kappa + 1}\right)^2.$$

Thus,

$$\frac{\|\mathbf{r}^{(1)}\|_{A^{-1}}}{\|\mathbf{r}^{(0)}\|_{A^{-1}}} \cdot \frac{\|\mathbf{r}^{(2)}\|_{A^{-1}}}{\|\mathbf{r}^{(1)}\|_{A^{-1}}} \cdot \cdots \cdot \frac{\|\mathbf{r}^{(k)}\|_{A^{-1}}}{\|\mathbf{r}^{(k-1)}\|_{A^{-1}}} \leq \left(\frac{\kappa-1}{\kappa+1}\right)^k$$

which yields

$$\frac{\|\mathbf{r}^{(k)}\|_{A^{-1}}}{\|\mathbf{r}^{(0)}\|_{A^{-1}}} \leq \left(\frac{\kappa-1}{\kappa+1}\right)^k.$$

In other words, the rate of convergence is the same as when the parameter α_k is chosen a priori to be $\hat{\alpha} = \frac{2}{\mu_1+\mu_n}$. Which method is preferable? For the first approach, the problem is that we must know μ_1 and μ_n. For the second approach, we must compute α_k at each step, which is worse for computation, but in practice works better for certain problems.

Now consider the iteration

$$\mathbf{x}^{(k+1)} = (I - \alpha_k A)\mathbf{x}^{(k)} + \alpha_k \mathbf{b}.$$

Since the exact solution $\mathbf{x}$ satisfies

$$\mathbf{x} = (I - \alpha_k A)\mathbf{x} + \alpha_k \mathbf{b},$$

it follows that

$$\mathbf{e}^{(k+1)} = (I - \alpha_k A)\mathbf{e}^{(k)}.$$

So, we have

$$\begin{aligned} \mathbf{e}^{(1)} &= (I - \alpha_0 A)\mathbf{e}^{(0)} \\ &\vdots \\ \mathbf{e}^{(k)} &= (I - \alpha_{k-1} A)(I - \alpha_{k-2} A) \cdots (I - \alpha_0 A)\mathbf{e}^{(0)}. \end{aligned}$$

In other words,

$$\mathbf{e}^{(k)} = P_k(A)\mathbf{e}^{(0)}$$

where P_k is a polynomial of degree k.

By the Cayley-Hamilton theorem,

$$\psi(A) = \prod_{i=0}^{d-1} (A - \mu_i I) = 0$$

where d is the number of distinct eigenvalues μ_i of A, when $A = A^T$. In other words

$$\psi(A) = \prod_{i=0}^{d-1} (I - \frac{1}{\mu_i} A) = 0$$

so we could choose $\alpha_i = 1/\mu_i$, but this choice is nonsense because one almost never knows the eigenvalues of A and even so, this choice is unstable because μ_i can vary immensely in magnitude. However, we have

$$\frac{\|\mathbf{e}^{(k)}\|}{\|\mathbf{e}^{(0)}\|} \leq \|P_k(A)\|_2,$$

so we will now use approximation theory to find a suitable P_k.

If $A = Q\Lambda Q^T$, then $P_k(A) = QP_k(\Lambda)Q^T$, and therefore

$$\frac{\|\mathbf{e}^{(k)}\|}{\|\mathbf{e}^{(0)}\|} \leq \|P_k(\Lambda)\|_2.$$

And since

$$P_k(\Lambda) = \begin{bmatrix} P_k(\lambda_1) & & \\ & \ddots & \\ & & P_k(\lambda_n) \end{bmatrix},$$

it follows that

$$\|P_k(\Lambda)\|_2 \leq \max_{1\leq i\leq n} |P_k(\lambda_i)|.$$

So, because $P_k(0) = I$, we want to find a polynomial $\hat{p}_k(\lambda)$ such that $\hat{p}_k(0) = 1$ and

$$\max_{1\leq i\leq n} |\hat{p}_k(\lambda_i)| = \min_{p_k(0)=1} \max_{1\leq i\leq n} |p_k(\lambda_i)|.$$

But clearly,

$$\min_{p_k(0)=1} \max_{1\leq i\leq n} |p_k(\lambda_i)| \leq \min_{p_k(0)=1} \max_{\lambda_n\leq\lambda\leq\lambda_1} |p_k(\lambda)|.$$

Therefore, we will try to find the polynomial $\hat{p}_k$ that satisfies $\hat{p}(0) = 1$ and is of minimum absolute value on the interval $[\lambda_n, \lambda_1]$. The solution to this problem is given by the *Chebyshev polynomials*.

The Chebyshev polynomial of degree k is defined to be

$$C_k(x) = \begin{cases} \cos[k\cos^{-1}(x)] & |x| \leq 1 \\ \cosh[k\cosh^{-1}(x)] & |x| > 1 \end{cases}$$

For example,

$$C_0(x) = 1, \quad C_1(x) = x, \quad C_2(x) = 2x^2 - 1.$$

These polynomials are designed to be bounded by 1 in absolute value on the interval $|x| \leq 1$.

If $\theta = \cos^{-1} x$ then, using the trigonometric identities

$$\begin{aligned} \cos(k+1)\theta &= \cos k\theta \cos\theta - \sin k\theta \sin\theta \\ \cos(k-1)\theta &= \cos k\theta \cos\theta + \sin k\theta \sin\theta \end{aligned}$$

we obtain

$$\cos(k+1)\theta = 2\cos k\theta \cos\theta - \cos(k-1)\theta$$

which yields the three-term recurrence relation of the Chebyshev polynomials

$$C_{k+1}(x) = 2xC_k(x) - C_{k-1}(x).$$

Since this relation leads to a leading coefficient of 2^{k-1} for $C_k(x)$ when $k \geq 1$, it is customary to normalize, defining

$$T_k(x) = \frac{C_k(x)}{2^{k-1}}, \quad k \geq 1.$$

We now claim that for $k = 2$, $\hat{p}_2(x)$ is $T_2(x) = x^2 - \frac{1}{2}$, scaled and translated appropriately so as to be small on the interval $[\lambda_n, \lambda_1]$ and satisfy $\hat{p}_2(0) = 1$.

Note that on $[-1, 1]$, $T_2(x)$ has a maximum at $x = -1$ and $x = 1$, and a local minimum at $x = 0$. Now, suppose that there is another polynomial $p_2(x) = x^2 + bx + c$ such that $p_2(-1) < T_2(-1)$, $p_2(1) < T_2(1)$, and $p_2(0) > T_2(0)$. Then the polynomial $q_1(x) = T_2(x) - p_2(x)$ has three sign changes in the interval $[-1, 1]$, but since $T_2(x)$ and $p_2(x)$ have the same leading coefficient, $q_1(x)$ can have degree at most 1, so it must be identically zero.

6.7 Chebyshev Iteration

Consider the iterative method

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_{k+1}\mathbf{r}^{(k)}$$

for solving $A\mathbf{x} = \mathbf{b}$. If we define $\mathbf{e}^{(k)} = \mathbf{x} - \mathbf{x}^{(k)}$, then

$$\mathbf{e}^{(k)} = P_k(A)\mathbf{e}^{(0)}$$

where

$$P_k(A) = (I - \alpha_k A)(I - \alpha_{k-1}A)\cdots(I - \alpha_1 A).$$

Therefore

$$\begin{aligned} \frac{\|\mathbf{e}^{(k)}\|_2}{\|\mathbf{e}^{(0)}\|_2} &\leq \|P_k(A)\|_2 \\ &\leq \max_{1 \leq i \leq N} |P_k(\lambda_i)| \\ &\leq \max_{a \leq \lambda_i \leq b} |P_k(\lambda)| \end{aligned}$$

where $P_k(0) = I$ and $b = \lambda_1 \geq \cdots \geq \lambda_N = a$ are the eigenvalues of A.

Recall that a good choice for the polynomial P_k arises from the Chebyshev polynomials

$$C_k(\cos\theta) = \cos k\theta, \quad \theta = \cos^{-1} x.$$

If we fix k, then we have

$$\alpha_j^{(k)} = \left(\frac{b+a}{2} - \left(\frac{b-a}{2} \right) \cos \frac{(2j+1)\pi}{2k} \right)^{-1}, \quad j = 0, \ldots, k-1.$$

Note that

$$\alpha_0^{(1)} = \frac{2}{b+a},$$

which is the same optimal parameter obtained using a different analysis.

Therefore, we can select k and then use the parameters $\alpha_0^{(k)}, \ldots, \alpha_{k-1}^{(k)}$. If $\|\mathbf{r}^{(k)}\|/\|\mathbf{r}^{(0)}\| \leq \epsilon$, we can stop; otherwise, we simply recycle these parameters. The process should not be stopped before the full cycle, because a partial polynomial may not be small on the interval $[a, b]$. Also, using the parameters in an arbitrary order may lead to numerical instabilities even though mathematically the order does not matter. For a long time, the determination of a suitable ordering was an open problem, but it has now been solved. It has been shown that when solving Laplace's equation using 128 parameters, a simple left-to-right ordering results in $\|\mathbf{e}^{(128)}\| \approx 10^{35}$, while the optimal ordering yields $\|\mathbf{e}^{(128)}\| \approx 10^{-7}$.

In the absence of roundoff error, using Chebyshev polynomials yields

$$\frac{\|\mathbf{e}^{(k)}\|_2}{\|\mathbf{e}^{(0)}\|_2} \leq \frac{2}{\left(\frac{\sqrt{\kappa}+1}{\sqrt{\kappa}-1}\right)^k + \left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^k} \approx \left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1} \right)^k$$

whereas, with steepest descent,

$$\frac{\|\mathbf{e}^{(k)}\|_2}{\|\mathbf{e}^{(0)}\|_2} \approx \left(\frac{\kappa-1}{\kappa+1} \right)^k.$$

6.8 Convergence Acceleration

Consider the iteration

$$M\mathbf{x}^{(k+1)} = N\mathbf{x}^{(k)} + \mathbf{b},$$

where $A = M - N$ is symmetric positive definite. This iteration can be rewritten as

$$\mathbf{x}^{(k+1)} = B\mathbf{x}^{(k)} + \mathbf{c}$$

where $B = M^{-1}N$ and $\mathbf{c} = M^{-1}\mathbf{b}$. Therefore $\mathbf{e}^{(k+1)} = B\mathbf{e}^{(k)}$ where $\mathbf{e}^{(k)} = \mathbf{x} - \mathbf{x}^{(k)}$. In an attempt to accelerate convergence, we define

$$\mathbf{y}^{(k)} = \sum_{\ell=0}^{k} a_{k\ell} x^{(\ell)}, \quad \sum_{\ell=0}^{k} a_{k\ell} = 1.$$

Then

$$\mathbf{x} - \mathbf{y}^{(k)} = \sum_{\ell=0}^{k} a_{k\ell}(\mathbf{x} - \mathbf{x}^{(\ell)}) = \sum_{\ell=0}^{k} a_{k\ell} B^{\ell} \mathbf{e}^{(0)}$$

which yields

$$\hat{\mathbf{e}}^{(k)} = P_k(B)\mathbf{e}^{(0)}$$

where $\hat{\mathbf{e}}^{(k)} = \mathbf{x} - \mathbf{y}^{(k)}$ and

$$P_k(\lambda) = \sum_{\ell=0}^{k} a_{k\ell} \lambda^{\ell}, \quad P_k(1) = 1.$$

It follows that

$$\frac{\|\hat{\mathbf{e}}^{(k)}\|_2}{\|\hat{\mathbf{e}}^{(0)}\|_2} \leq \|P_k(B)\|_2.$$

If B is symmetric, then we can write $B = Q\Lambda Q^T$ and obtain

$$\|P_k(B)\|_2 = \|P_k(\Lambda)\|_2 = \max_{\lambda=\lambda_i} |P_k(\lambda)| \leq \max_{a \leq \lambda \leq b} |P_k(\lambda)|.$$

Recall that the Chebyshev polynomials $C_k(x)$ satisfy the three-term recurrence relation

$$C_{k+1}(x) = 2xC_k(x) - C_{k-1}(x).$$

If we let $B = I - \alpha A$ where $\alpha = \frac{2}{a+b}$, then B is symmetric and we can use the iteration

$$\mathbf{y}^{(\ell+1)} = \omega_{\ell+1}(B\mathbf{y}^{(\ell)} + \mathbf{c} - \mathbf{y}^{(\ell-1)}) + \mathbf{y}^{(\ell-1)}$$

with initial vectors

$$\mathbf{y}^{(0)} = \mathbf{x}^{(0)}, \quad \mathbf{y}^{(1)} = B\mathbf{y}^{(0)} + \mathbf{c}.$$

The parameters $\omega_{\ell+1}$ are defined by

$$\omega_{\ell+1} = \left(1 - \frac{\rho^2 \omega_\ell}{4}\right)^{-1}, \quad \ell \geq 1, \quad \rho = \frac{b-a}{b+a}.$$

It follows that

$$\omega_2 \geq \omega_3 \geq \cdots \geq \omega^* > 1$$

where

$$\omega_* = \lim_{\ell \to \infty} \omega_\ell.$$

What is the limit ω^*? This limit satisfies

$$\omega^* = \left(1 - \frac{\rho^2 \omega^*}{4}\right)^{-1}$$

which is a quadratic equation with solutions

$$\omega^* = \frac{1 \pm \sqrt{1-\rho^2}}{\rho^2/2}.$$

Choosing the plus sign, we have

$$1 < \omega_* = \frac{2}{1+\sqrt{1-\rho^2}} < 2.$$

Recall that for solving Poisson's equation, $\rho = 1 - ch^2 + O(h^4)$ for the Jacobi method, while $\rho = 1 - c'h + O(h^2)$ for the Chebyshev method.

6.9 Two-Cyclic Systems

Let A be symmetric positive definite. Then we can use diagonal scaling to obtain a matrix $D^{-1/2}AD^{-1/2}$ with all diagonal elements equal to 1 by setting

$$D = \begin{bmatrix} a_{11} & & \\ & \ddots & \\ & & a_{nn} \end{bmatrix}.$$

Then we can check whether the new matrix is two-cyclic. A matrix A is said to be *two-cyclic* (or to have *Property A*) if there is a permutation matrix Π such that

$$\Pi^T A \Pi = \begin{bmatrix} I_p & F \\ F^T & I_q \end{bmatrix}. \tag{6.2}$$

For example, suppose

$$A = \begin{bmatrix} 1 & a_1 & & \\ a_1 & \ddots & \ddots & \\ & \ddots & \ddots & a_{n-1} \\ & & a_{n-1} & 1 \end{bmatrix}.$$

Then by choosing Π so that odd-numbered rows and columns are grouped together, followed by even-numbered rows and columns, we obtain

$$\Pi^T A \Pi = \begin{bmatrix} 1 & & & & a_1 & & & \\ & \ddots & & & a_2 & \ddots & & \\ & & \ddots & & & \ddots & \ddots & \\ & & & 1 & & & \ddots & \ddots \\ a_1 & a_2 & & & 1 & & & \\ & \ddots & \ddots & & & \ddots & & \\ & & \ddots & \ddots & & & \ddots & \\ & & & \ddots & & & & \ddots \end{bmatrix}.$$

This matrix has all kinds of nice properties. In particular, it allows decoupling of equations.

It should be noted that a matrix arising from the discretization of a PDE in two dimensions using a 5-point stencil has Property A, but a matrix based on a 9-point stencil does not. However, the latter matrix does have *block* Property A. For example, if

$$A = \begin{bmatrix} A_1 & B_1 & & \\ B_1^T & \ddots & \ddots & \\ & \ddots & \ddots & B_{n-1} \\ & & B_{n-1}^T & A_n \end{bmatrix}$$

then we can choose Π so that

$$\Pi^T A \Pi = \begin{bmatrix} A_1 & & & & B_1^T & & & \\ & A_3 & & & B_2^T & B_3^T & & \\ & & \ddots & & & \ddots & \ddots & \\ & & & 1 & & & \ddots & \ddots \\ B_1 & B_2 & & & A_2 & & & \\ & B_3 & \ddots & & & A_4 & & \\ & & \ddots & \ddots & & & \ddots & \\ & & & \ddots & & & & \ddots \end{bmatrix}.$$

We now show that for a matrix of the form (6.2), we can choose an optimal parameter ω for the SOR method. Let F be a $p \times q$ matrix with $p \geq q$, and let $F = U\Sigma V^T$ be the SVD of F. Then

$$\begin{aligned} A &= \begin{bmatrix} UU^T & U\Sigma V^T \\ V\Sigma^T U^T & VV^T \end{bmatrix} \\ &= \begin{bmatrix} U & 0 \\ 0 & V \end{bmatrix} \begin{bmatrix} I & \Sigma \\ \Sigma^T & I \end{bmatrix} \begin{bmatrix} U^T & 0 \\ 0 & V^T \end{bmatrix}. \end{aligned}$$

Since the left and right matrices above denote a similarity transformation, it follows that

$$\lambda(A) = \lambda(\tilde{A}), \quad \tilde{A} = \begin{bmatrix} I & \Sigma \\ \Sigma^T & I \end{bmatrix}.$$

Reordering the rows and columns of $\tilde(A)$, we obtain a block diagonal matrix, where each diagonal block is a 2×2 matrix of the form

$$\begin{bmatrix} 1 & \sigma_i \\ \sigma_i & 1 \end{bmatrix}, \quad i = 1, \ldots, q.$$

The eigenvalues of $\tilde{A}$ are the eigenvalues of all of these diagonal blocks, which are $\lambda = 1 \pm \sigma_i$. These eigenvalues must be positive since A is positive definite, so it follows that

$$0 < \sigma_i < 1, \quad i = 1, \ldots, q.$$

Now, consider the SOR operator

$$\begin{aligned} \mathcal{L}_\omega &= \left(\frac{1}{\omega} I + L\right)^{-1} \left(\left(\frac{1}{\omega} - 1\right) I - U\right) \\ &= \begin{bmatrix} \frac{1}{\omega} I & 0 \\ F^T & \frac{1}{\omega} I \end{bmatrix}^{-1} \begin{bmatrix} \left(\frac{1}{\omega} - 1\right) I & -F \\ 0 & \left(\frac{1}{\omega} - 1\right) I \end{bmatrix} \end{aligned}$$

where

$$L = \begin{bmatrix} 0 & 0 \\ F^T & 0 \end{bmatrix}, \quad U = \begin{bmatrix} 0 & F \\ 0 & 0 \end{bmatrix}.$$

We can explicitly invert the first matrix to obtain

$$\begin{aligned} \mathcal{L}_\omega &= \begin{bmatrix} \omega I & 0 \\ -\omega^2 F^T & \omega I \end{bmatrix} \begin{bmatrix} \left(\frac{1}{\omega} - 1\right) I & -F \\ 0 & \left(\frac{1}{\omega} - 1\right) I \end{bmatrix} \\ &= \begin{bmatrix} (1-\omega) I & -\omega F \\ (\omega^2 - \omega) F^T & (1-\omega) I + \omega^2 F^T F \end{bmatrix}. \end{aligned}$$

Using the SVD of F again, we obtain

$$\begin{aligned} \mathcal{L}_\omega &= \begin{bmatrix} (1-\omega) U U^T & -\omega U \Sigma V^T \\ (\omega^2 - \omega) V \Sigma^T U^T & (1-\omega) V V^T + \omega^2 V \Sigma^T \Sigma V^T \end{bmatrix} \\ &= \begin{bmatrix} U & 0 \\ 0 & V \end{bmatrix} \begin{bmatrix} (1-\omega) I & -\omega \Sigma \\ (\omega^2 - \omega) \Sigma^T & (1-\omega) I + \omega^2 \Sigma^T \Sigma \end{bmatrix} \begin{bmatrix} U^T & 0 \\ 0 & V^T \end{bmatrix}. \end{aligned}$$

Define

$$\Gamma(\omega) = \begin{bmatrix} (1-\omega) I & -\omega \Sigma \\ (\omega^2 - \omega) \Sigma^T & (1-\omega) I + \omega^2 \Sigma^T \Sigma \end{bmatrix}.$$

Then $\lambda(\mathcal{L}_\omega) = \lambda(\Gamma(\omega))$ and $\|\mathcal{L}_\omega\|_2 = \|\Gamma(\omega)\|_2$. Recall that

$$\mathbf{e}^k = \mathcal{L}_\omega^k \mathbf{e}^{(0)}.$$

Ideally, we want to choose ω so that $\|\mathcal{L}_\omega^k\|$ is minimized, but this is an open problem. However, Young showed how to compute ω so that $\rho(\mathcal{L}_\omega)$ is minimized. Since each block of $\Gamma(\omega)$ is a diagonal matrix, we can use the same reordering trick as before to obtain a block diagonal matrix, where each diagonal block is a 2×2 matrix of the form

$$\Gamma_i = \begin{bmatrix} (1-\omega) & -\omega \sigma_i \\ (\omega^2 - \omega) \sigma_i & (1-\omega) + \omega^2 \sigma_i^2 \end{bmatrix}, \quad i = 1, \ldots, q.$$

The eigenvalues μ of Γ_i satisfy the characteristic equation

$$(1 - \omega - \mu)^2 - \mu \sigma_i^2 \omega^2 = 0.$$

Note that when $\omega = 0$, then $|\mu| = 1$, indicating divergence. If $\omega = 1$, corresponding to the Gauss-Seidel method, then $\mu = 0$ or $\mu = \sigma_i^2$. If $\omega = 2$, then the eigenvalues are complex conjugates with $|\mu| = 1$. Therefore there exists an ω where μ becomes complex:

$$\hat{\omega} = \frac{2}{1 + \sqrt{1 - \sigma_i^2}}.$$

Thus, $|\mu(\omega_1)| > |\mu(\omega_2)|$ for $\omega_1 > \omega_2 > \hat{\omega}$.

Note that the eigenvalues of the Gauss-Seidel matrix are 0 or σ_i^2, while the eigenvalues of the Jacobi matrix are $\pm\sigma_i$. Therefore we can expect Gauss-Seidel to converge twice as fast as Jacobi for matrices with Property A.

6.10 Conjugate Gradient Method

Many iterative methods for solving $A\mathbf{x} = \mathbf{b}$ have the form

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k-1)} + \omega_{k+1}(\alpha_k \mathbf{z}^{(k)} - \mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}) \tag{6.3}$$

where

$$M\mathbf{z}^{(k)} = \mathbf{r}^{(k)} = \mathbf{b} - A\mathbf{x}^{(k)} \tag{6.4}$$

for some M. In particular, if $\omega_k \equiv 1$ and $\alpha_k \equiv 1$ then this reduces to

$$\mathbf{x}^{(k+1)} = M^{-1}(\mathbf{b} - A\mathbf{x}^{(k)}) - \mathbf{x}^{(k)}$$

or

$$M\mathbf{x}^{(k+1)} = \mathbf{b} - (A - M)\mathbf{x}^{(k)} = N\mathbf{x}^{(k)} + \mathbf{b}$$

where $A = M - N$. Our goal is to choose the parameters α_k and ω_k so that $\|P_k(M^{-1}A)\mathbf{e}^{(0)}\|$ is minimized, where $\mathbf{e}^{(k)} = \mathbf{x} - \mathbf{x}^{(k)}$ and $\mathbf{e}^{(k)} = P_k(M^{-1}A)\mathbf{e}^{(0)}$.

Suppose that we can impose the condition that

$$(\mathbf{z}^{(k)}, M\mathbf{z}^{(k)}) = \delta_{jk}$$

where both M and A are $n \times n$ and required to be symmetric positive definite. If this is possible, then it follows that $\mathbf{z}^{(n+1)} = \mathbf{0}$, and therefore $\mathbf{r}^{(n+1)} = \mathbf{0}$, implying convergence in n iterations.

It follows from (6.3) that

$$\mathbf{b} - A\mathbf{x}^{(k+1)} = \mathbf{b} - A\mathbf{x}^{(k-1)} - \omega_{k+1}(\alpha_k A\mathbf{z}^{(k)} + A\mathbf{x}^{(k)} - \mathbf{b} + \mathbf{b} - A\mathbf{y}^{(k-1)})$$

which simplifies to

$$\mathbf{r}^{(k+1)} = \mathbf{r}^{(k-1)} - \omega_{k+1}(\alpha_k A\mathbf{z}^{(k)} - \mathbf{r}^{(k)} + \mathbf{r}^{(k-1)}).$$

From (6.4), we obtain

$$M\mathbf{z}^{(k+1)} = M\mathbf{z}^{(k-1)} - \omega_{k+1}(\alpha_k A\mathbf{z}^{(k)} - M\mathbf{z}^{(k)} + M\mathbf{z}^{(k-1)}).$$

We use the induction hypothesis

$$(\mathbf{z}^{(p)}, M\mathbf{z}^{(q)}) = 0, \quad p \neq q, \quad p = 1, 2, \ldots, k.$$

Then

$$(\mathbf{z}^{(k)}, M\mathbf{z}^{(k+1)}) = (\mathbf{z}^{(k)}, M\mathbf{z}^{(k-1)}) - \omega_{k+1}[(\alpha_k \mathbf{z}^{(k)}, A\mathbf{z}^{(k)}) - (\mathbf{z}^{(k)}, M\mathbf{z}^{(k)}) + (\mathbf{z}^{(k)}, M\mathbf{z}^{(k-1)})]$$

which yields

$$\alpha_k = \frac{(\mathbf{z}^{(k)}, M\mathbf{z}^{(k)})}{(\mathbf{z}^{(k)}, A\mathbf{z}^{(k)})}.$$

Similarly,

$$(\mathbf{z}^{(k-1)}, M\mathbf{z}^{(k+1)}) = (\mathbf{z}^{(k-1)}, M\mathbf{z}^{(k-1)}) - \omega_{k+1}[(\alpha_k \mathbf{z}^{(k-1)}, A\mathbf{z}^{(k)}) - (\mathbf{z}^{(k-1)}, M\mathbf{z}^{(k)}) + (\mathbf{z}^{(k-1)}, M\mathbf{z}^{(k-1)})]$$

which yields

$$\omega_{k+1} = \frac{(\mathbf{z}^{(k-1)}, M\mathbf{z}^{(k-1)})}{\alpha_k(\mathbf{z}^{(k-1)}, A\mathbf{z}^{(k)}) + (\mathbf{z}^{(k-1)}, M\mathbf{z}^{(k-1)})}.$$

We can simplify this expression for ω_{k+1} by noting that by symmetry,

$$(\mathbf{z}^{(k-1)}, A\mathbf{z}^{(k)}) = (\mathbf{z}^{(k)}, A\mathbf{z}^{(k-1)})$$

and therefore

$$\begin{aligned}(\mathbf{z}^{(k)}, M\mathbf{z}^{(k)}) &= (\mathbf{z}^{(k)}, M\mathbf{z}^{(k-2)}) + \omega_k(\alpha_{k-1}(\mathbf{z}^{(k)}, A\mathbf{z}^{(k-1)}) - (\mathbf{z}^{(k)}, M\mathbf{z}^{(k-1)}) + (\mathbf{z}^{(k)}, M\mathbf{z}^{(k-2)})) \\ &= \omega_k(\alpha_{k-1}(\mathbf{z}^{(k)}, A\mathbf{z}^{(k-1)}))\end{aligned}$$

which yields

$$\omega_{k+1} = \frac{(\mathbf{z}^{(k-1)}, M\mathbf{z}^{(k-1)})}{-\frac{\alpha_k}{\alpha_{k+1}}\frac{1}{\omega_k}(\mathbf{z}^{(k)}, M\mathbf{z}^{(k)}) + (\mathbf{z}^{(k-1)}, M\mathbf{z}^{(k-1)})}$$

or

$$\omega_{k+1} = \left(1 - \frac{\alpha_k}{\alpha_{k-1}}\frac{1}{\omega_k}\frac{(\mathbf{z}^{(k)}, M\mathbf{z}^{(k)})}{(\mathbf{z}^{(k-1)}, M\mathbf{z}^{(k-1)})}\right)^{-1}.$$

We have shown that

$$(\mathbf{z}^{(k)}, M\mathbf{z}^{(k+1)}) = (\mathbf{z}^{(k-1)}, M\mathbf{z}^{(k+1)}) = 0.$$

It can easily be shown that

$$(\mathbf{z}^{(\ell)}, M\mathbf{z}^{(k+1)}) = 0, \quad \ell < k - 1.$$

We now state the *classical conjugate gradient* algorithm:

$\mathbf{x}^{(0)}$ given
Solve $M\mathbf{z}^{(0)} = \mathbf{r}^{(0)}$
$\mathbf{p}^{(0)} = \mathbf{z}^{(0)}$
for $k = 0, \ldots$
    $\alpha_k = \frac{(\mathbf{z}^{(k)}, M\mathbf{z}^{(k)})}{(\mathbf{p}^{(k)}, A\mathbf{p}^{(k)})}$
    $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \mathbf{p}^{(k)}$
    $\mathbf{r}^{(k+1)} = \mathbf{r}^{(k)} - \alpha_k A\mathbf{p}^{(k)}$
    Test for convergence
    Solve $M\mathbf{z}^{(k+1)} = \mathbf{r}^{(k+1)}$
    $\beta_{k+1} = \frac{(\mathbf{z}^{(k+1)}, M\mathbf{z}^{(k+1)})}{(\mathbf{z}^{(k)}, M\mathbf{z}^{(k)})}$
    $\mathbf{p}^{(k+1)} = \mathbf{z}^{(k+1)} + \beta_{k+1}\mathbf{p}^{(k)}$
end

It can be shown that

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(0)} + P_k(K)\mathbf{z}^{(0)}$$

where $K = M^{-1}A$. Furthermore, amongst all methods which generate a polynomial for a given $\mathbf{x}^{(0)}$, the conjugate gradient method minimizes the quantity

$$\epsilon^{k+1} = [\mathbf{e}^{(k+1)}]^T A\mathbf{e}^{(k+1)}.$$

Most notable of all is that if A has p distinct eigenvalues, then the conjugate gradient method converges in p steps. This is particularly useful in domain decomposition, where the interface between two subdomains consists of only a small number of points.