1 Elimination

Gaussian elimination is the standard way to solve a system of linear equations, but other schemes have some advantages. A brief discussion of elimination with Givens transformations will show this and illuminate our study of Gaussian elimination. We begin with a review of Gaussian elimination for solving a system of $n$ equations in $n$ unknowns,

$$Ax = b$$

(1)

A description of the process for the first column of $A$ (the elimination of $x_1$) shows how the computation goes. The scheme requires a non-zero pivot $a_{1,1}$, so if necessary, we swap the first row with a row that has $a_{k,1} \neq 0$. Thereafter Gaussian elimination keeps the entries of the first row fixed and subtracts a multiple $m_{1,p}$ of row 1 from row $p$ for rows $p = 2, \ldots, n$. This changes the entries in row $p$ to

$$a'_{p,j} = a_{p,j} - m_{1,p}a_{1,j}$$

The unknown $x_1$ is eliminated from equation $p$ by making $a'_{p,1} = 0$. This is accomplished by taking $m_{1,p} = a_{p,1}/a_{1,1}$, which is possible because the pivot $a_{1,1} \neq 0$. Because this operation on the rows can be reversed, it results in an equivalent linear system. If the pivot is small, the multiplier $m_{1,p}$ can be large and correspondingly, some of the new entries $a'_{p,j}$ can be large. This can lead to large errors in the numerical solution. Instead of pivoting only when $a_{1,1} = 0$ and swapping the first row with any row that has $a_{k,1} \neq 0$, we always pivot and we swap the first row with a row that has the biggest $|a_{k,1}|$. This guarantees that $|m_{1,p}| \leq 1$ for all $p$, which helps control the growth of entries in the matrix as elimination proceeds. It is often useful to view Gaussian elimination as a matrix factorization: $PA = LU$ where $P$ is a permutation matrix, $L$ is a lower triangular matrix, and $U$ is an upper triangular matrix.

The key difference with Givens transformations is that when modifying row $p$ to eliminate $a_{p,1}$, we also modify row 1. Pivoting is not necessary. Indeed, we can eliminate $a_{p,1}$ even when $a_{1,1} = 0$. If $a_{p,1} \neq 0$, the coefficients

$$c = \frac{a_{1,1}}{d} \quad \text{and} \quad s = \frac{a_{p,1}}{d} \quad \text{where} \quad d = \sqrt{a_{1,1}^2 + a_{p,1}^2}$$

are well-defined. With these coefficients, the two rows become

$$a'_{1,j} = c a_{1,j} + s a_{p,j}$$
$$a'_{p,j} = -s a_{1,j} + c a_{p,j}$$

A little calculation shows that $a'_{p,1} = 0$, which is to say that $x_1$ has been eliminated from equation $p$. Because this operation on the rows can be reversed,
it results in an equivalent linear system. Another calculation shows that for \( j = 1, \ldots, n \),
\[
(a'_{1,j})^2 + (a'_{p,j})^2 = (a_{1,j})^2 + (a_{p,j})^2
\]
This states that the sum of the squares of all the entries in the matrix is constant as elimination proceeds. This provides a much better control of the growth of entries than when Gaussian elimination is done. Summing up, elimination with Givens transformations is simpler than Gaussian elimination because no pivoting is necessary and it is more effective at controlling the growth of entries. On the other hand, each step processes two rows instead of one with Gaussian elimination, so Givens transformations are roughly twice as expensive. What is more important in some contexts is that they do not preserve as many zeros in the matrix as Gaussian elimination. Givens transformations are used in practice in certain situations, but a closely related transformation due to Householder that eliminates all entries in a column at a time is preferred for most purposes. Elimination with Givens transformations can be viewed as a matrix factorization: 
\[
A = QR
\]
where \( Q \) is an orthogonal matrix (meaning that \( Q^{-1} = Q^T \)) and \( R \) is an upper triangular matrix.

On the class site is a program \texttt{givens.m} that solves (1) by elimination with Givens transformations and a program \texttt{givensQR} that calculates the QR decomposition of a square matrix \( A \).

## 2 Condition

We investigate the condition of a solution \( x \) of (1) with respect to perturbations of both the matrix and the right hand side. We begin with the simpler case of the perturbation induced in the solution by a perturbation of the right hand side:
\[
A(x + \Delta x) = b + \Delta b
\]
Using (1), this says that \( \Delta x = A^{-1} \Delta b \) and if we write \( A^{-1} = (\alpha_{i,j}) \), then
\[
\Delta x_i = \sum_{p=1}^{n} \alpha_{i,p} \Delta b_p \quad i = 1, \ldots, n \tag{2}
\]
This is an exact formula for the absolute change in \( x_i \) as a function of absolute changes in the \( b_p \). If the appropriate quantities are non-zero, a little manipulation gives a similar exact formula for relative changes:
\[
\frac{\Delta x_i}{x_i} = \sum_{p=1}^{n} \alpha_{i,p} \frac{b_p}{x_i} \left( \frac{\Delta b_p}{b_p} \right) \quad i = 1, \ldots, n \tag{3}
\]
Perturbations of the matrix are more complicated because if the perturbations \( \Delta A \) are too big, the matrix \( A + \Delta A \) might be singular. However, if \( A \) has an inverse and the perturbations are sufficiently small, it can be shown that
A + ΔA has an inverse, hence this is not an issue in an analysis based upon “small” perturbations. The perturbed solution is defined by

\[(A + ΔA)(x + Δx) = b\]

Multiplying out the left hand side, using (1), and then multiplying on the left by \(A^{-1}\) leads to

\[Δx + A^{-1} ΔA x + A^{-1} ΔA Δx = 0\]

Component \(i\) of this expression is

\[Δx_i + \sum_{k=1}^{n} α_{i,k} (ΔA x)_k + \sum_{k=1}^{n} α_{i,k} (ΔA Δx)_k = 0\]

The expression is simplified greatly if we consider the effect of perturbing a single entry of \(A\), so let us suppose that all entries of \(ΔA\) are zero except for \(Δa_{p,q}\). Each sum reduces to a single term and we are left with

\[Δx_i + α_{i,p} Δa_{p,q} x_q + α_{i,p} Δa_{p,q} Δx_q = 0\]

We now linearize this expression by dropping the product of small quantities and solving for

\[Δx_i ≈ -α_{i,p} Δa_{p,q} x_q\]  \hspace{1cm} (4)

If the appropriate quantities are non-zero, a little manipulation gives the relative change in the solution component due to a small relative change in a component of the matrix:

\[\frac{Δx_i}{x_i} ≈ -α_{i,p} \frac{a_{p,q}}{a_{p,q}} \left( \frac{x_q}{x_i} \right)\]

\hspace{1cm} (5)

The expressions developed are the result of linearization, so the effects of multiple perturbations are additive. For example, if we perturb \(a_{p,q}\) and \(a_{r,s}\), we have

\[\frac{Δx_i}{x_i} ≈ -α_{i,p} a_{p,q} \left( \frac{Δa_{p,q}}{a_{p,q}} \right) \left( \frac{x_q}{x_i} \right) - α_{i,r} a_{r,s} \left( \frac{Δa_{r,s}}{a_{r,s}} \right) \left( \frac{x_s}{x_i} \right)\]

Of course, we can handle perturbations of the right hand side in the same way.

## 3 Backward Error Analysis

Suppose that we have computed somehow an approximation \(z\) to the solution \(x\) of (1). The residual of this approximation is

\[r = b - Az\]

If we define \(Δb = -r\), we find that \(z\) is the exact solution of the perturbed problem

\[Az = b + Δb\]
If \( b \) is the result of measurement or computation and the perturbation is comparable to the error present in the right hand side, we would consider \( z \) to be a good solution of the linear system. This is a particularly simple way to study the quality of a numerical solution, but there are other possibilities. One is to regard \( z \) as the solution of a problem with the entries of \( A \) perturbed. Indeed, if \( r^Tz \neq 0 \), it is easy to verify that \( z \) is the exact solution of the perturbed problem

\[
(A + \Delta A)z = b
\]

with \( \Delta A = rr^T / r^Tz \).

Here is a strong result about relative perturbations in both \( b \) and \( A \). Let \( |A| \) be the matrix \( (|a_{ij}|) \), which is what you get from \texttt{abs(A)} in MATLAB. We consider perturbations \( \Delta A \) and \( \Delta b \) for which

\[
(A + \Delta A)z = b + \Delta b
\]

and define \( \rho \) so that

\[
|\Delta A| \leq \rho |A| \quad \text{and} \quad |\Delta b| \leq \rho |b|
\]

Here the inequalities are to be interpreted as applying to corresponding entries in arrays so that \( \rho \) is a uniform bound on the relative size of perturbations to the entries of both the matrix and the right hand side. A sharp bound on the smallest possible value of \( \rho \) for which there are perturbations such that (6) holds is

\[
\rho = \max_i \frac{|r_i|}{(|A||z| + |b|)_i}
\]

This quantity is easily computed in MATLAB as

\[
\text{>> rho = max( abs(r) ./ (abs(A)*abs(z) + abs(b)) ) ;}
\]

To recapitulate, for this value of \( \rho \) there are perturbations \( \Delta A \) and \( \Delta b \) for which (6) and (7) hold and it is the smallest value for which this is true. We see then that \( z \) is the exact solution of a problem which differs from the given problem by relative changes of no more than \( \rho \) in both the matrix and right hand side.

4 Backwards Stability of Back Substitution

To illustrate an important result, we do a backward error analysis of back substitution for a system of two equations:

\[
\begin{align*}
    u_{11}x_1 + u_{12}x_2 &= b_1 \\
    u_{22}x_2 &= b_2
\end{align*}
\]

We shall see that the computed solution \( x_1^*, x_2^* \) is the exact solution of a system

\[
\begin{align*}
    u_{11}^*x_1^* + u_{12}^*x_2^* &= b_1 \\
    u_{22}^*x_2^* &= b_2
\end{align*}
\]
with coefficients that are close to those of (8,9). Notice that we do not need to perturb \( b_1 \) and \( b_2 \).

The first step of solving (8,9) with back substitution is

\[
x_2^* = b_2 \odot u_{22} = \frac{b_2}{u_{22}} (1 + \delta_1)
\]

With the definition

\[
u_{22}^* = \frac{u_{22}}{1 + \delta_1} \approx u_{22} (1 - \delta_1)
\]

the computed \( x_2^* \) is the exact solution of (11). The second step is

\[
x_1^* = (b_1 \odot x_2^* \odot u_{12}) \odot u_{11}
\]

We have first that

\[
x_2^* \odot u_{12} = x_2^* u_{12} (1 + \delta_2)
\]

and with the definition

\[
u_{12}^* = u_{12} (1 + \delta_2)
\]

this is

\[
x_2^* \odot u_{12} = x_2^* u_{12}^*
\]

Then

\[
b_1 \odot x_2^* u_{12}^* = (b_1 - x_2^* u_{12}^*) (1 + \delta_3)
\]

and finally

\[
x_1^* = \frac{(b_1 - x_2^* u_{12}^*) (1 + \delta_3)}{u_{11}} (1 + \delta_4)
\]

With the definition

\[
u_{11}^* = \frac{u_{11}}{(1 + \delta_3)(1 + \delta_4)} \approx u_{11} (1 - \delta_3 - \delta_4)
\]

the computed \( x_1^* \) is the exact solution of (10).

In this analysis each \( \delta_m \) is bounded in magnitude by a unit roundoff, so each \( u_{ij}^* \) differs from \( u_{ij} \) by no more than about two units of roundoff. Because the computed solution is the exact solution of a system with coefficients that are (very) close to the ones given, we say that back substitution is backwards stable for systems of the form (8,9). The result can be established for \( n \times n \) systems in a similar way.