ME 360 Lecture 15

• Recap

Forward Error: $\|x - x_a\|_\infty$

Backward Error: $\|b - Ax_a\|_\infty$

• Relative Forward & Backward Error

Denote the residual by $r = b - Ax_a$. The relative backward error of system $Ax = b$ is defined to be $\frac{\|r\|_\infty}{\|b\|_\infty}$,

and the relative forward error is $\frac{\|x - x_a\|_\infty}{\|x\|_\infty}$.

• Error Magnification factor

The error magnification factor for $Ax = b$ is the ratio of the two, or

$$\text{error magnification factor} = \frac{\text{relative forward error}}{\text{relative backward error}} = \frac{\frac{\|x - x_a\|_\infty}{\|x\|_\infty}}{\frac{\|r\|_\infty}{\|b\|_\infty}}$$

For the example mentioned above:

The relative backward error is

$$\frac{0.0001}{2.0001} \approx 0.00005 = 0.005\%$$

and the relative forward error is

$$\frac{2.0001}{1} = 2.0001 \approx 200\%$$

The error magnification factor is $2.0001/(0.0001/2.0001) = 40004.0001$

• Condition Number

The condition number of a square matrix A, $\text{cond}(A)$, is the maximum possible error magnification factor for solving $Ax = b$, over all right-hand sides b.

- **Matrix Norm**

The matrix norm of an $n \times n$ matrix A is

$$\|A\|_\infty = \text{maximum absolute row sum}$$

Theorem: (for Proof, see Sauer P.94-95)
The condition number of the $n \times n$ matrix A is

$$\text{Cond}(A) = \|A\| \cdot \|A^{-1}\|$$

Using this theorem for the previous coefficient matrix

$$A = \begin{bmatrix} 1 & 1 \\ 1.0001 & 1 \end{bmatrix} \Rightarrow \|A\| = 2.0001$$

Finding the inverse of A

$$A^{-1} = \begin{bmatrix} -10000 & 10000 \\ 10001 & -10000 \end{bmatrix} \Rightarrow \|A^{-1}\| = 20001$$

The condition number of A is

$$\text{cond}(A) = (2.0001)(20001) = 40004.001$$

This is exactly the error magnification we found in the previous example, which achieves the worst case. The error magnification factor for any other b in this system will be less than or equal to 40004.0001.

*** Why do we care about condition number?**

- In floating point arithmetic, the relative backward error cannot be expected to be less than ϵ_{mach}, since storing the entries of b already causes error of that size.
- According to the definition of error magnification factor, relative forward errors of size $\epsilon_{mach} \cdot \text{cond}(A)$ are possible in solving $Ax = b$.
- In other words, if $\text{cond}(A) \approx 10^k$, we should prepare to lose k digits of accuracy in computing x.

- **Definition of vector/matrix/1/operator norms**

(a). Vector norm $\|x\|$ satisfies three properties.

(i). $\|x\| \geq 0$ with equality if and only if $x = [0, \dots, 0]$

(ii). for each scalar α and vector x, $\|\alpha x\| = |\alpha| \cdot \|x\|$

(iii). for vectors x, y, $\|x+y\| \leq \|x\| + \|y\|$

(b). Matrix norm $\|A\|_\infty$ satisfies three similar properties

(i). $\|A\| \geq 0$ with equality if and only if $A = 0$

(ii) for each scalar α and matrix A, $\|\alpha A\| = |\alpha| \cdot \|A\|$

(iii). for matrices A, B, $\|A+B\| \leq \|A\| + \|B\|$

(c). 1-norm

(i). the vector 1-norm of the vector $x = [x_1, \dots, x_n]$ is $\|x\|_1 = |x_1| + \dots + |x_n|$

(ii). the matrix 1-norm of the $n \times n$ matrix A is $\|A\|_1$ = maximum absolute column sum.

(d). operator norm

$$\|A\| = \max \frac{\|Ax\|}{\|x\|}$$

2.3.2 Swamping

This source of error is fixable. We demonstrate swamping with the next example.

Example: Consider the system of equations.

$$10^{-20} x_1 + x_2 = 1$$
$$x_1 + 2x_2 = 4$$

We will use three ways to solve these equations.

1. Exact solution

$$\left[\begin{array}{cc|c} 10^{-20} & 1 & 1 \\ 1 & 2 & 4 \end{array}\right] \Rightarrow \begin{array}{c}\text{subtract } 10^{20} \times \text{row 1} \\ \text{from row 2}\end{array} \Rightarrow \left[\begin{array}{cc|c} 10^{-20} & 1 & 1 \\ 0 & 2-10^{20} & 4-10^{20} \end{array}\right]$$

Bottom equation becomes

$$(2-10^{20})x_2 = 4 - 10^{20} \Rightarrow x_2 = \frac{4-10^{20}}{2-10^{20}}$$

Top equation becomes

$$10^{-20}x_1 + \frac{4-10^{20}}{2-10^{20}} = 1 \Rightarrow x_1 = 10^{20}\left(1 - \frac{4-10^{20}}{2-10^{20}}\right) = \frac{-2\times10^{20}}{2-10^{20}}$$

The exact solution is

$$[x_1, x_2] = \left[\frac{2\times10^{20}}{10^{20}-2}, \frac{4-10^{20}}{2-10^{20}}\right] \approx [2, 1]$$

2. IEEE double precision.

We start with the same Gaussian Elimination.

$$\left[\begin{array}{cc|c} 10^{-20} & 1 & 1 \\ 1 & 2 & 4 \end{array}\right] \Rightarrow \text{subtract } 10^{20} \times \text{row } 1 \text{ from row } 2 \Rightarrow \left[\begin{array}{cc|c} 10^{-20} & 1 & 1 \\ 0 & 2-10^{20} & 4-10^{20} \end{array}\right]$$

- In IEEE double precision, $2-10^{20}$ is the same as -10^{20} due to rounding. Similarly, $4-10^{20}$ is stored as -10^{20}.
  Now the bottom equation is

$$-10^{20}x_2 = -10^{20} \Rightarrow x_2 = 1$$

The machine arithmetic version of the top equation becomes

$$10^{-20}x_1 + 1 = 1 \Rightarrow x_1 = 0$$

The computed solution is exactly $[x_1, x_2] = [0, 1]$, which has large relative error compared with the exact solution.

3. IEEE double precision, after row exchange.

We use Gaussian Elimination after row exchange.

$$\left[\begin{array}{cc|c} 1 & 2 & 4 \\ 10^{-20} & 1 & 1 \end{array}\right] \Rightarrow \text{subtract } 10^{-20} \times \text{row } 1 \text{ from row } 2 \Rightarrow \left[\begin{array}{cc|c} 1 & 2 & 4 \\ 0 & 1-2\times10^{-20} & 1-4\times10^{-20} \end{array}\right]$$

- In IEEE double precision, $1-2\times10^{-20}$ is stored as 1 and so is $1-4\times10^{-20}$.

The equations are now

$$\begin{cases} x_1 + 2x_2 = 4 \\ \quad\quad\ \ x_2 = 1 \end{cases} \Rightarrow \begin{cases} x_1 = 2 \\ x_2 = 1 \end{cases}$$

This is not the exact answer, but it is correct up to 16 digits, which is the best one can do with double precision arithmetics.

- **Effect of swamping.**

In method 2, the effect of subtracting 10^{20} times the top equation from the bottom equation was to overpower, or "swamp", the bottom equation.
By doing so, after elimination, we lose the information from the second equation, and now we have essentially two copies of the first equation.
Since the bottom equation has disappeared, we cannot expect the computed solution to satisfy the bottom equation, and it does not.

In method 3, swamping does not happen as the multiplier is 10^{-20}. After the elimination, the original two equations are still largely existent, hence we get a more accurate solution,

- **Take away**

We should keep the multiplier of Gaussian Elimination as small as possible to avoid swamping.
This row exchange protocol, is called partial pivoting, which we will talk about in the next section.

2.4. The PA=LU Factorization

So far the Gaussian Elimination is considered "naive" because of two series difficulties

(a). zero pivot

(b). swamping.

For a nonsingular matrix (matrix that has an inverse), we can improve both difficulties using an efficient protocol for exchanging rows of the coefficient matrix, called partial pivoting.

2.4.1 Partial Pivoting

- **Partial Pivoting Protocol**
  (i) compare numbers before carrying out each elimination
  (ii) locate largest entry of the first column
  (iii). Swap row from (ii) with the pivot row.

- **A more detailed Explanation**
  (i). At the start of Gaussian elimination, partial pivoting ask us to select pth row. where

$$|a_{p1}| \geq |a_{i1}| \quad \text{for all} \quad 1 \leq i \leq n$$

(ii) Exchange rows 1 and p.
(iii). Perform Gaussian elimination using the "new" version of a_{11} as pivot. The multiplier used to eliminate a_{i1} will be

$$m_{i1} = \frac{a_{i1}}{a_{11}}, \quad \& \quad |m_{i1}| \leq 1$$

(iv). Apply the same check for every pivot. When deciding on the second pivot, we start with the current a_{22} and check all entries directly below.
We select row p such that

$$|a_{p2}| \geq |a_{i2}| \quad \text{for all} \quad 2 \leq i \leq n$$

if $p \neq 2$, exchange row 2 & row p. Row 1 is not involved in this process.
(v). We apply this protocol for every column elimination. Before eliminating column k, the p with $k \leq p \leq n$ and largest $|a_{pk}|$ is located, and rows k & p are exchanged if necessary.

With this protocol, we ensure that all multipliers, or entries in matrix L, will be no greater than 1 in absolute value.
This protocol effectively avoid 0 pivot and swamping.

Example: apply Gaussian Elimination with partial pivoting to solve

$$\left[\begin{array}{cc|c} 1 & 1 & 3 \\ 3 & -4 & 2 \end{array}\right]$$

According to partial pivoting, we compare $|a_{11}|$ and $|a_{21}|$. Since $|a_{21}| > |a_{11}|$, we apply row exchange before elimination.

$$\left[\begin{array}{cc|c} 3 & -4 & 2 \\ 1 & 1 & 3 \end{array}\right] \Rightarrow \begin{array}{c}\text{subtract } 1/3 \times \text{row 1} \\ \text{from row 2}\end{array} \Rightarrow \left[\begin{array}{cc|c} 3 & -4 & 2 \\ 0 & 7/3 & 7/3 \end{array}\right]$$

Apply back substitution, the solution is $x_2 = 1$ & $x_1 = 2$, the same as we found earlier.

When we solved it the first time, the multiplier was 3, this will not occur under partial pivoting.

Example: apply Gaussian elimination with partial pivoting to solve the system.

$$x_1 - x_2 + 3x_3 = -3$$
$$-x_1 - 2x_3 = 1$$
$$2x_1 + 2x_2 + 4x_3 = 0$$

Step 1: we write in tableau form

$$\left[\begin{array}{ccc|c} 1 & -1 & 3 & -3 \\ -1 & 0 & -2 & 1 \\ 2 & 2 & 4 & 0 \end{array}\right]$$

Step 2: partial pivoting for column 1

We compare $|a_{11}|$, $|a_{21}|$, and $|a_{31}|$, and choose $|a_{31}|$ for the new pivot. We exchange row 1 & row 3.

$$\left[\begin{array}{ccc|c} 1 & -1 & 3 & -3 \\ -1 & 0 & -2 & 1 \\ 2 & 2 & 4 & 0 \end{array}\right] \Rightarrow \begin{array}{c}\text{exchange row1} \\ \text{and row3}\end{array} \Rightarrow \left[\begin{array}{ccc|c} 2 & 2 & 4 & 0 \\ -1 & 0 & -2 & 1 \\ 1 & -1 & 3 & -3 \end{array}\right]$$

$$\left[\begin{array}{ccc|c} 2 & 2 & 4 & 0 \\ -1 & 0 & -2 & 1 \\ 1 & -1 & 3 & -3 \end{array}\right] \Rightarrow \begin{array}{c}\text{subtract } -\frac{1}{2} \times \text{row1} \\ \text{from row2}\end{array} \Rightarrow \left[\begin{array}{ccc|c} 2 & 2 & 4 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & -1 & 3 & -3 \end{array}\right]$$

$$\left[\begin{array}{ccc|c} 2 & 2 & 4 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & -1 & 3 & -3 \end{array}\right] \Rightarrow \text{subtract } \tfrac{1}{2} \times \text{row1 from row 3} \Rightarrow \left[\begin{array}{ccc|c} 2 & 2 & 4 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & -2 & 1 & -3 \end{array}\right]$$

Step 3: Partial pivoting for column 2

We compare the current $|a_{22}|$ & $|a_{32}|$, and choose $|a_{32}|$ to be the new pivot, we switch row 2 & row 3.

$$\left[\begin{array}{ccc|c} 2 & 2 & 4 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & -2 & 1 & -3 \end{array}\right] \Rightarrow \text{exchange row 2 and row 3} \Rightarrow \left[\begin{array}{ccc|c} 2 & 2 & 4 & 0 \\ 0 & -2 & 1 & -3 \\ 0 & 1 & 0 & 1 \end{array}\right]$$

$$\left[\begin{array}{ccc|c} 2 & 2 & 4 & 0 \\ 0 & -2 & 1 & -3 \\ 0 & 1 & 0 & 1 \end{array}\right] \Rightarrow \text{subtract } -\tfrac{1}{2} \times \text{row2 from row3} \Rightarrow \left[\begin{array}{ccc|c} 2 & 2 & 4 & 0 \\ 0 & -2 & 1 & -3 \\ 0 & 0 & \tfrac{1}{2} & -\tfrac{1}{2} \end{array}\right]$$

Note that all three multipliers are less than 1 in absolute value.

Step 4: back substitution.

$$\begin{cases} \tfrac{1}{2} x_3 = -\tfrac{1}{2} \\ -2x_2 + x_3 = -3 \\ 2x_1 + 2x_2 + 4x_3 = 0 \end{cases} \Rightarrow \begin{cases} x_3 = -1 \\ x_2 = 1 \\ x_1 = 1 \end{cases} \Rightarrow \boxed{x = [1, 1, -1]}$$

Partial pivot avoids zero pivot when there is at least one non-zero value in the column. If there is no such nonzero entry at or below diagonal entry, then the matrix is singular and Gaussian Elimination will fail anyway.