UNIVERSITY OF WATERLOO
FACULTY OF ENGINEERING
Department of Electrical &
Computer Engineering

ECE 204 *Numerical methods*

Linear algebra

Douglas Wilhelm Harder, LEL, M.Math.
dwharder@uwaterloo.ca
dwharder@gmail.com

Introduction

- In this topic, we will
  - Look at how floating-point numbers can seriously affect the results of Gaussian elimination and backward substitution
  - Describe the Gaussian elimination algorithm with partial pivoting
  - Look at the Jacobi method for approximating the solution to a system of linear equations using iteration
  - Describe the condition number of a matrix

Linear algebra

- Systems of linear equations are often the only ones that can be approximated numerically with reasonable certainty
  - Many non-linear systems are often approximated using linear systems
    - For example, small-signal analysis of non-linear devices
  - The modelling of objects with momentum can generally be done locally in time by linear approximations
    - This of course fails if, for example, an object strikes another object

Linear algebra

- Consider a system of n linear equations in n unknowns

$$A\mathbf{u} = \mathbf{v}$$

- To solve such a system, we:
  - Create the $n \times (n + 1)$ augmented matrix $\left(A \;\vdots\; \mathbf{v}\right)$
  - We apply row operations on the augmented matrix until the matrix is in row-echelon form
    - The three elementary row operations are:
      - Swapping two rows
      - Adding a multiple of one row onto another
      - Multiplying a row by a non-zero scalar
    - The first two are used for this process of Gaussian elimination
  - If rank(A) = n, we may use backward substitution to find the unique solution

Issues with linear algebra

- Issues:
  - If the matrix A is dense, the number of floating-point operations (FLOPS) can be as high as

$$\frac{2}{3}n^3 - \frac{1}{2}n^2 - \frac{1}{6}n$$

  - Many times, in adding a multiple of one row onto another, there is the possibility of subtractive cancellation
  - Additionally, adding a large multiple of one row onto another may result in $x + y = y$ when x is the critical value

Issues with linear algebra

- Consider the following system:

$$\left(\begin{array}{cc|c} \texttt{+441000} & \texttt{+491000} & \texttt{+491000} \\ \texttt{+491000} & \texttt{+491000} & \texttt{+492000} \end{array}\right) = \left(\begin{array}{cc|c} 0.00001 & 1 & 1 \\ 1 & 1 & 2 \end{array}\right)$$

- By observation, the solution should be close to

$$\mathbf{u} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

  - The exact solution is

$$\mathbf{u} = \begin{pmatrix} \frac{100000}{99999} \\ \frac{99998}{99999} \end{pmatrix} = \begin{pmatrix} 1.\overline{00001} \\ 0.\overline{99998} \end{pmatrix}$$

  - To four significant digits, this is $\mathbf{u} \approx \begin{pmatrix} \texttt{+491000} \\ \texttt{+491000} \end{pmatrix}$

Issues with linear algebra

- Now, applying Gaussian elimination:

$$\left(\begin{array}{cc|c} 0.00001 & 1 & 1 \\ 1 & 1 & 2 \end{array}\right)$$

  - Add –100 000 times Row 1 onto Row 2, resulting the calculations:

$$-100\,000 + 1 = -99\,999 = -541000$$
$$-100\,000 + 2 = -99\,998 = -541000$$

  - Thus, we are left with

$$\left(\begin{array}{cc|c} 0.0001 & 1 & 1 \\ 0 & -100000 & -100000 \end{array}\right)$$

  - Thus $u_2 = 1$
  - Substituting this into Row 1, we get that $u_1 = 0$

$$\mathbf{u} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Issues with linear algebra

- Thus, while the best answer is

$$\mathbf{u} \approx \begin{pmatrix} +491000 \\ +491000 \end{pmatrix}$$

Gaussian elimination and backward substitution gave us

$$\mathbf{u} \approx \begin{pmatrix} +000000 \\ +491000 \end{pmatrix}$$

- What happened?
  - When we added a huge multiple of Row 1 onto Row 2, this had the effect of swamping out Row 2
    - Recall that $a_{2,j} + ca_{1,j} = ca_{1,j}$ if $ca_{1,j} >> a_{2,j}$
    - Consequently, the matrix ended up looking like:

$$\left(\begin{array}{cc:c} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \end{array}\right) \sim \left(\begin{array}{cc:c} a_{1,1} & a_{1,2} & a_{1,3} \\ 0 & -ca_{1,2} & -ca_{1,3} \end{array}\right)$$

Partial pivoting

- Thus, by adding a large multiple of one row onto another,
  we lost all information about that second row

- Recall in linear algebra,
  if you had a zero in the pivot position, you'd swap two rows:

$$\left(\begin{array}{cc:c} 0 & 1 & 1 \\ 1 & 1 & 2 \end{array}\right) \sim \left(\begin{array}{cc:c} 1 & 1 & 2 \\ 0 & 1 & 1 \end{array}\right)$$

- Gaussian elimination with *partial pivoting* says to swap the row with the largest-in-magnitude entry on or below the pivot to the pivot row

Partial pivoting

- Applying this here, we would now have:

$$\left(\begin{array}{cc:c} 0.00001 & 1 & 1 \\ 1 & 1 & 2 \end{array}\right) \sim \left(\begin{array}{cc:c} 1 & 1 & 2 \\ 0.00001 & 1 & 1 \end{array}\right)$$

  - Adding –0.00001 times Row 1 onto Row 2 now leaves Row 2 unchanged:

$$\sim \left(\begin{array}{cc:c} 1 & 1 & 2 \\ 0 & 1 & 1 \end{array}\right)$$

  - Applying backward substitution, gives us that

$$\mathbf{u} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

Partial pivoting

- You may think this is unlikely to occur with double-precision floating-point numbers, but consider:

$$\left(\begin{array}{ccc|c} 2.1 & 0.7 & 7 & 9.8 \\ 0.3 & 0.1 & 0 & 0.4 \\ 0 & 1 & 1 & 2 \end{array}\right)$$

  - The unique solution is clearly $\mathbf{u} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$

  - Let us use Gaussian elimination as taught in first year

Partial pivoting

- Neither 0.3 nor 2.1 can be stored exactly in binary, and there is also round-off error in calculating $-\frac{0.3}{2.1}$
  - Adding this times Row 1 onto Row 2 yields:

$$\left(\begin{array}{ccc|c} 2.1 & 0.7 & 7 & 9.8 \\ 0.3 & 0.1 & 0 & 0.4 \\ 0 & 1 & 1 & 2 \end{array}\right) \sim \left(\begin{array}{ccc|c} 2.1 & 0.7 & 7 & 9.8 \\ 0 & 2^{-56} & 1 & 1 \\ 0 & 1 & 1 & 2 \end{array}\right)$$

  - Technically, the entry at (2, 2) is non-zero, so next we add -2^{56} times Row 2 onto Row 3:

$$\left(\begin{array}{ccc|c} 2.1 & 0.7 & 7 & 9.8 \\ 0.3 & 0.1 & 0 & 0.4 \\ 0 & 1 & 1 & 2 \end{array}\right) \sim \left(\begin{array}{ccc|c} 2.1 & 0.7 & 7 & 9.8 \\ 0 & 2^{-56} & 1 & 1 \\ 0 & 0 & 2^{56} & 2^{56} \end{array}\right)$$

Partial pivoting

- Applying backward substitution:

$$\left(\begin{array}{ccc|c} 2.1 & 0.7 & 7 & 9.8 \\ 0.3 & 0.1 & 0 & 0.4 \\ 0 & 1 & 1 & 2 \end{array}\right) \sim \left(\begin{array}{ccc|c} 2.1 & 0.7 & 7 & 9.8 \\ 0 & 2^{-56} & 1 & 1 \\ 0 & 0 & 2^{56} & 2^{56} \end{array}\right)$$

  - First, $u_3 = \dfrac{2^{56}}{2^{56}} = 1$
  - Next, $2^{-56}u_2 + u_3 = 1$, so $2^{-56}u_2 + 1 = 1$, and hence $u_2 = 0$
  - Finally, $2.1u_1 + 0.7u_2 + 7u_3 = 9.8$, so $2.1u_1 + 7 = 9.8$,

$$\text{hence } u_1 = \frac{2.8}{2.1} = 1.\overline{3} \text{ so } \mathbf{u} = \begin{pmatrix} 1.\overline{3} \\ 0 \\ 1 \end{pmatrix} \text{ and not } \mathbf{u} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

Partial pivoting

- Note that $\begin{pmatrix} 2.1 & 0.7 & 7 \\ 0.3 & 0.1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 9.8 \\ 0.4 \\ 2 \end{pmatrix}$

  while $\begin{pmatrix} 2.1 & 0.7 & 7 \\ 0.3 & 0.1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1.\overline{3} \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 9.8 \\ 0.4 \\ 1 \end{pmatrix}$

  – This is not a “pathological” matrix,
    one that might significantly amplify any error

Partial pivoting

- Notice that that swapping Rows 2 and 3 first yields

$$\left(\begin{array}{cccc} 2.1 & 0.7 & 7 & 9.8 \\ 0 & 2^{-56} & 1 & 1 \\ 0 & 1 & 1 & 2 \end{array}\right) \sim \left(\begin{array}{cccc} 2.1 & 0.7 & 7 & 9.8 \\ 0 & 1 & 1 & 2 \\ 0 & 2^{-56} & 1 & 1 \end{array}\right)$$

- Next adding -2^{-56} times Row 2 onto Row 3 makes no change

$$\sim \left(\begin{array}{cccc} 2.1 & 0.7 & 7 & 9.8 \\ 0 & 1 & 1 & 2 \\ 0 & 0 & 1 & 1 \end{array}\right)$$

- Thus, backward substitution yields the solution $\mathbf{u} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$

Gaussian elimination with partial pivoting

- Thus, you could describe the Gaussian elimination algorithm where A:$\mathbf{R}^n \rightarrow \mathbf{R}^m$ so that A is an $m \times n$ matrix as follows:
  1. Create the $m \times (n+1)$ augmented matrix $\left(A \;\vdots\; \mathbf{v} \right)$
  2. Assign $i \leftarrow 1$ and $j \leftarrow 1$
  3. While $j \leq n+1$,
     - i. If Column j contains no leading non-zero entries, update $j \leftarrow j+1$ and return to Step 3.
     - ii. Find the row with the largest-in-magnitude non-zero leading entry in Column j (if there are multiple such entries, pick one), and swap that row with Row i
     - iii. For each other Row k that has a leading non-zero entry in Column j, add $-\dfrac{a_{k,j}}{a_{i,j}}$ times Row i onto Row k
     - iv. Update $i \leftarrow i+1$ and $j \leftarrow j+1$; and return to Step 3.

Gaussian elimination with partial pivoting

- Why does this help?
  - If the largest entry in absolute value is moved to the pivot, then when we add a multiple of one row onto another, we are guaranteed that

$$\left| -\frac{a_{k,j}}{a_{i,j}} \right| \leq 1$$

  - Thus, in general, we will avoid the issue of adding a significant multiple of one row onto another

Gaussian elimination with partial pivoting

- For example, suppose we are to solve

$$\begin{pmatrix} -1.5 & -6 & 5.8 & 7 \\ 4 & 4 & -8.8 & -1.6 \\ 5 & 2 & 1 & -1 \\ 3 & 7.2 & 1.6 & 2.4 \end{pmatrix} \mathbf{u} = \begin{pmatrix} 12.7 \\ -34.4 \\ 8 \\ -12.2 \end{pmatrix}$$

Gaussian elimination with partial pivoting

- First, we create the augmented matrix:

$$\left(\begin{array}{cccc|c} -1.5 & -6 & 5.8 & 7 & 12.7 \\ 4 & 4 & -8.8 & -1.6 & -34.4 \\ 5 & 2 & 1 & -1 & 8 \\ 3 & 7.2 & 1.6 & 2.4 & -12.2 \end{array}\right)$$

Gaussian elimination with partial pivoting

- Starting in the first column, the largest entry in absolute value on or below entry (1, 1) is 5 in Row 3
  - Swap Rows 1 and 3

$$\left(\begin{array}{cccc|c} -1.5 & -6 & 5.8 & 7 & 12.7 \\ 4 & 4 & -8.8 & -1.6 & -34.4 \\ 5 & 2 & 1 & -1 & 8 \\ 3 & 7.2 & 1.6 & 2.4 & -12.2 \end{array}\right)$$

$$\sim \left(\begin{array}{cccc|c} 5 & 2 & 1 & -1 & 8 \\ 4 & 4 & -8.8 & -1.6 & -34.4 \\ -1.5 & -6 & 5.8 & 7 & 12.7 \\ 3 & 7.2 & 1.6 & 2.4 & -12.2 \end{array}\right)$$

Gaussian elimination with partial pivoting

- Now, add appropriate multiples of Row 1 onto Rows 2, 3 and 4:
  - Add –4/5 = –0.8 times Row 1 onto Row 2
  - Add –(–1.5)/5 = 0.3 times Row 1 onto Row 3
  - Add –3/5 = –0.6 times Row 1 onto Row 4

$$\sim \left(\begin{array}{cccc|c} 5 & 2 & 1 & -1 & 8 \\ 4 & 4 & -8.8 & -1.6 & -34.4 \\ -1.5 & -6 & 5.8 & 7 & 12.7 \\ 3 & 7.2 & 1.6 & 2.4 & -12.2 \end{array}\right)$$

$$\sim \left(\begin{array}{cccc|c} 5 & 2 & 1 & -1 & 8 \\ 0 & 2.4 & -9.6 & -0.8 & -40.8 \\ 0 & -5.4 & 6.1 & 6.7 & 15.1 \\ 0 & 6 & 1 & 3 & -17 \end{array}\right)$$

Gaussian elimination with partial pivoting

- Continuing in the second column, the largest entry in absolute value on or below entry (2, 2) is 6 in Row 4
  - Swap Rows 2 and 4

$$\sim \left(\begin{array}{cccc|c} 5 & 2 & 1 & -1 & 8 \\ 0 & 2.4 & -9.6 & -0.8 & -40.8 \\ 0 & -5.4 & 6.1 & 6.7 & 15.1 \\ 0 & 6 & 1 & 3 & -17 \end{array}\right)$$

$$\sim \left(\begin{array}{cccc|c} 5 & 2 & 1 & -1 & 8 \\ 0 & 6 & 1 & 3 & -17 \\ 0 & -5.4 & 6.1 & 6.7 & 15.1 \\ 0 & 2.4 & -9.6 & -0.8 & -40.8 \end{array}\right)$$

Gaussian elimination with partial pivoting

- Now, add appropriate multiples of Row 2 onto Rows 3 and 4:
  - Add –(–5.4)/6 = 0.9 times Row 2 onto Row 3
  - Add –2.4/6 = –0.4 times Row 2 onto Row 4

$$\sim \left(\begin{array}{cccc|c} 5 & 2 & 1 & -1 & 8 \\ 0 & 6 & 1 & 3 & -17 \\ 0 & -5.4 & 6.1 & 6.7 & 15.1 \\ 0 & 2.4 & -9.6 & -0.8 & -40.8 \end{array}\right)$$

$$\sim \left(\begin{array}{cccc|c} 5 & 2 & 1 & -1 & 8 \\ 0 & 6 & 1 & 3 & -17 \\ 0 & 0 & 7 & 9.4 & -0.2 \\ 0 & 0 & -10 & -2 & -34 \end{array}\right)$$

Gaussian elimination with partial pivoting

- Continuing in the third column, the largest entry in absolute value on or below entry (3, 3) is –10 in Row 4
  - Swap Rows 3 and 4

$$\sim \left(\begin{array}{cccc|c} 5 & 2 & 1 & -1 & 8 \\ 0 & 6 & 1 & 3 & -17 \\ 0 & 0 & 7 & 9.4 & -0.2 \\ 0 & 0 & -10 & -2 & -34 \end{array}\right)$$

$$\sim \left(\begin{array}{cccc|c} 5 & 2 & 1 & -1 & 8 \\ 0 & 6 & 1 & 3 & -17 \\ 0 & 0 & -10 & -2 & -34 \\ 0 & 0 & 7 & 9.4 & -0.2 \end{array}\right)$$

Gaussian elimination with partial pivoting

- Now, add an appropriate multiple of Row 3 onto Row 4:
  - Add –7/(–10) = 0.7 times Row 3 onto Row 4

$$\sim \left(\begin{array}{cccc|c} 5 & 2 & 1 & -1 & 8 \\ 0 & 6 & 1 & 3 & -17 \\ 0 & 0 & -10 & -2 & -34 \\ 0 & 0 & 7 & 9.4 & -0.2 \end{array}\right)$$

$$\sim \left(\begin{array}{cccc|c} 5 & 2 & 1 & -1 & 8 \\ 0 & 6 & 1 & 3 & -17 \\ 0 & 0 & -10 & -2 & -34 \\ 0 & 0 & 0 & 8 & -24 \end{array}\right)$$

Gaussian elimination with partial pivoting

- We can now use backward substitution to find the solution:

$$\left(\begin{array}{cccc:c} 5 & 2 & 1 & -1 & 8 \\ 0 & 6 & 1 & 3 & -17 \\ 0 & 0 & -10 & -2 & -34 \\ 0 & 0 & 0 & 8 & -24 \end{array}\right)$$

$$u_4 = \frac{-24}{8} = -3$$

$$u_3 = \frac{-34 + 2u_4}{-10} = \frac{-34 + 2(-3)}{-10} = 4$$

$$u_2 = \frac{-17 - u_3 - 3u_4}{6} = \frac{-17 - 4 + 9}{6} = -2$$

$$u_1 = \frac{8 - 2u_2 - u_3 + u_4}{5} = \frac{8 + 4 - 4 - 3}{5} = 1$$

$$\mathbf{u} = \begin{pmatrix} 1 \\ -2 \\ 4 \\ -3 \end{pmatrix}$$

Iteration

- In MATLAB, you can do this as follows:

```
>> A = [-1.5 -6.0  5.8  7.0
         4.0  4.0 -8.8 -1.6
         5.0  2.0  1.0 -1.0
         3.0  7.2  1.6  2.4];
>> v = [12.7 -34.4 8.0 -12.2]';
>> u = A\v
    ans =
          1
         -2
          4
         -3
```

$$\mathbf{u} = \begin{pmatrix} 1 \\ -2 \\ 4 \\ -3 \end{pmatrix}$$

Iteration

- Why use \ as an operator?

$$\frac{1}{A}Au = \frac{1}{A}v$$

Gaussian elimination with partial pivoting

- Question:
  - Suppose we followed the rules of Gaussian elimination, and one of the "largest entries in magnitude" we found was very small relative to other diagonal entries

$$\left(\begin{array}{cccc|c} 5 & 2 & 1 & -1 & 8 \\ 0 & 6 & 1 & 3 & -17 \\ 0 & 0 & -10^{-10} & -2 & -34 \\ 0 & 0 & 0 & 8 & -24 \end{array}\right)$$

  - Is the original matrix A still invertible?
    - We'll discuss this later…

Iteration

- Recall the fixed-point theorem:
  - If we are trying to solve $x = f(x)$, one technique is to start with an initial approximation x_0 and then iterate $x_{k+1} = f(x_k)$ until either
    - This sequence appears to converge
      - Successive values are *close enough*
    - A maximum number of iterations has occurred

- Sometimes, it is possible to rewrite an equation so that it is in this form:

$$x^2 + x - 3 = 0 \qquad \frac{-1 \pm \sqrt{13}}{2}$$

You don't have to know how to find these

  - This can be rewritten as either

$$\cancel{x = 3 - x^2} \qquad x = \frac{3 - x}{x} = \frac{3}{x} - 1 \qquad x = \frac{3}{x + 1}$$

Iteration

- Now, take a look at this equation:

$$A\mathbf{u} = \mathbf{v}$$

- It is not of the form

$$\mathbf{u} = \mathbf{f}(\mathbf{u})$$

  - Can we rewrite it?

Iteration

- There are some properties of matrices that are seen in engineering:
  - Matrices may be strictly diagonally dominant
    - That is, each diagonal entry is greater than the sum of the absolute values of all other entries in their rows or their column columns
  - This ensures that the diagonal entries are all non-zero
  - It also guarantees the matrix is invertible
  - Of these two matrices, the right is diagonally dominant

$$\begin{pmatrix} -1.5 & -6 & 5.8 & 7 \\ 4 & 4 & -8.8 & -1.6 \\ 5 & 2 & 1 & -1 \\ 3 & 7.2 & 1.6 & 2.4 \end{pmatrix} \qquad \begin{pmatrix} 20 & 0.3 & -0.4 & 0.5 \\ 0.1 & 5 & 1.2 & -0.3 \\ 0.7 & 0.2 & 4 & -1.1 \\ 0.4 & 1.3 & 0.6 & 10 \end{pmatrix}$$

Iteration

- Consider this equation:

$$A\mathbf{u} = \mathbf{v}$$

- We can rewrite A as the sum of a diagonal matrix and an off-diagonal matrix

$$A = A_{\text{diag}} + A_{\text{off}}$$

  - For example,

$$\begin{pmatrix} 20 & 0.3 & -0.4 & 0.5 \\ 0.1 & 5 & 1.2 & -0.3 \\ 0.7 & 0.2 & 4 & -1.1 \\ 0.4 & 1.3 & 0.6 & 10 \end{pmatrix} = \begin{pmatrix} 20 & 0 & 0 & 0 \\ 0 & 5 & 0 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 10 \end{pmatrix} + \begin{pmatrix} 0 & 0.3 & -0.4 & 0.5 \\ 0.1 & 0 & 1.2 & -0.3 \\ 0.7 & 0.2 & 0 & -1.1 \\ 0.4 & 1.3 & 0.6 & 0 \end{pmatrix}$$

Iteration

- Thus, we may rewrite this equation

$$A\mathbf{u} = \mathbf{v}$$

$$\left(A_{\text{diag}} + A_{\text{off}}\right)\mathbf{u} = \mathbf{v}$$

- From linear algebra, you know that $(A + B)\mathbf{u} = A\mathbf{u} + B\mathbf{u}$
  - Thus

$$A_{\text{diag}}\mathbf{u} + A_{\text{off}}\mathbf{u} = \mathbf{v}$$

- The problem is, we still need to isolate a $\mathbf{u}$…

Iteration

- Of these two matrices, which is invertible?

$$\begin{pmatrix} 20 & 0 & 0 & 0 \\ 0 & 5 & 0 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 10 \end{pmatrix} \qquad \begin{pmatrix} 0 & 0.3 & -0.4 & 0.5 \\ 0.1 & 0 & 1.2 & -0.3 \\ 0.7 & 0.2 & 0 & -1.1 \\ 0.4 & 1.3 & 0.6 & 0 \end{pmatrix}$$

- If you said "both", you're right, but how?
  - After all, this matrix is singular (not invertible):

$$\begin{pmatrix} 0 & 0.3 & -0.4 & 0.5 \\ 0.1 & 0 & 1.6 & -0.3 \\ 0.7 & 0.2 & 0 & -1.1 \\ 1.0 & 1.3 & 0.6 & 0 \end{pmatrix}$$

Iteration

- From linear algebra,
  a diagonal matrix is invertible if and only if
  all the diagonal entries are non-zero
- The inverse of an invertible diagonal matrix is that matrix with the reciprocals of the diagonal entries

$$A_{\text{diag}} = \begin{pmatrix} 20 & 0 & 0 & 0 \\ 0 & 5 & 0 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 10 \end{pmatrix} \qquad A_{\text{diag}}^{-1} = \begin{pmatrix} 0.05 & 0 & 0 & 0 \\ 0 & 0.2 & 0 & 0 \\ 0 & 0 & 0.25 & 0 \\ 0 & 0 & 0 & 0.1 \end{pmatrix}$$

- Recall that if A is invertible, then $\mathbf{u} = A^{-1}\mathbf{v}$ solves $A\mathbf{u} = \mathbf{v}$
  - Normally, we don't want to find the inverse,
    but for diagonal matrices, it is a numerically safe operation

Iteration

- Question:
  - Is this matrix still invertible?

$$\begin{pmatrix} 20 & 0 & 0 & 0 \\ 0 & 5 & 0 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 10^{-10} \end{pmatrix}$$

  - We'll discuss this later…

Iteration

- Okay, so given this derivation:

$$A\mathbf{u} = \mathbf{v}$$

$$\left(A_{\text{diag}} + A_{\text{off}}\right)\mathbf{u} = \mathbf{v}$$

$$A_{\text{diag}}\mathbf{u} + A_{\text{off}}\mathbf{u} = \mathbf{v}$$

  - The diagonal matrix is invertible, so:
    - Bring the vector $A_{\text{off}}\,\mathbf{u}$ to the other side:

$$A_{\text{diag}}\mathbf{u} = \mathbf{v} - A_{\text{off}}\mathbf{u}$$

    - Multiply both sides by the inverse of A_{diag}:

$$A_{\text{diag}}^{-1}\left(A_{\text{diag}}\mathbf{u}\right) = A_{\text{diag}}^{-1}\left(\mathbf{v} - A_{\text{off}}\mathbf{u}\right)$$

$$\left(A_{\text{diag}}^{-1}A_{\text{diag}}\right)\mathbf{u} = A_{\text{diag}}^{-1}\left(\mathbf{v} - A_{\text{off}}\mathbf{u}\right)$$

$$\mathbf{u} = A_{\text{diag}}^{-1}\left(\mathbf{v} - A_{\text{off}}\mathbf{u}\right)$$

Iteration

- Thus, we have now transformed

$$A\mathbf{u} = \mathbf{v}$$

into the equivalent equation

$$\mathbf{u} = A_{\text{diag}}^{-1}\left(\mathbf{v} - A_{\text{off}}\mathbf{u}\right)$$

- This is of the form $\mathbf{u} = \mathbf{f}(\mathbf{u})$
  - Thus, if we find a solution to the second, that solution is also a solution to the first

Iteration

- This is not something to do by hand, so let us go to MATLAB:

```
>> A = [20.0  0.3 -0.4  0.5
         0.1  5.0  1.2 -0.3
         0.7  0.2  4.0 -1.1
         0.4  1.3  0.6 10.0];
>> v = [0.3 0.5 -0.2 0.4]';
>> u = A\v
    u =
        0.011602268277939 14
        0.113449902472233 0
       -0.050060355176282 02
        0.027791043258069 07

>> Adiag = diag( diag( A ) );
>> Aoff = A - Adiag;
>> InvAdiag = inv( Adiag );
```

Iteration

```
>> f = @(u)( InvAdiag*(v - Aoff*u) );
>> u0 = InvAdiag*v;          % Solution to Adiag u0 = v
>> for k = 1:100
       previous_u0 = u0;
       u0 = f( u0 );

       if norm( u0 - previous_u0 ) < 1e-6
           u0
           break;
       end
   end
     u0 =
         0.011602263173223 44
         0.1134498919522636
        -0.05006031299952349
         0.027791017535565 67
>> k
    k = 8
>> norm( u0 - u );
    5.076666249750968e-08
```

$$A_{\text{diag}}^{-1}\left(\mathbf{v} - A_{\text{off}}\mathbf{u}\right)$$

$$A\mathbf{u} = \mathbf{v}$$

$$A_{\text{diag}}\mathbf{u}_0 = \mathbf{v}$$

```
u =
    0.0116022682779391 4
    0.1134499024722330
   -0.0500603551762820 2
    0.0277910432580690 7
```

Iteration

- This is called the Jacobi method
  - It is guaranteed to converge to a solution if the matrix is strictly diagonally dominant
  - In other cases, it may diverge, even if the matrix is only diagonally dominant
  - Later, we will see a variation called the Gauss-Seidel method

Iteration

- Question:
  - Which should we choose?

$$\mathbf{u} = A_{\text{diag}}^{-1}\left(\mathbf{v} - A_{\text{off}}\mathbf{u}\right)$$

$$\mathbf{u} = A_{\text{diag}}^{-1}\mathbf{v} - \left(A_{\text{diag}}^{-1}A_{\text{off}}\right)\mathbf{u}$$

  - The first requires an $O(n)$ calculation each iteration
  - The second requires an $O(n + n^2)$ calculation up front
  - If the number of iterations is significantly less than the number of equations, we should use the first

Ill-conditioned systems

- We have covered how to minimize numeric error when solving a system of linear equations
  - There are systems, however, that are inherently numerically unstable
  - One problem in engineering is that nothing is exact:

$$A\mathbf{u} = \mathbf{v}$$

    - There are errors in the matrix A
    - There are errors in the target vector $\mathbf{v}$
    - There are round-off errors and errors due to subtractive cancellation
  - Can these errors be magnified in $\mathbf{u}$?
  - Can we accidentally design a system that is *ill-conditioned*

Ill-conditioned systems

- Consider this matrix:

$$A = \begin{pmatrix} 4 & -3 & 4 \\ 2 & 4 & 3 \\ 3 & 3 & 4 \end{pmatrix}$$

  - The determinant is one: $\det(A) = 1$

- If $\mathbf{v} = \begin{pmatrix} 5 \\ 9 \\ 10 \end{pmatrix}$, the solution to $A\mathbf{u} = \mathbf{v}$ is $\mathbf{u} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$

Ill-conditioned systems

- What happens if there is an error in the matrix?

$$\tilde{A} = \begin{pmatrix} 3.99 & -2.99 & 3.99 \\ 1.99 & 3.99 & 3.01 \\ 3.01 & 3.01 & 3.99 \end{pmatrix} \qquad A = \begin{pmatrix} 4 & -3 & 4 \\ 2 & 4 & 3 \\ 3 & 3 & 4 \end{pmatrix}$$

- The solution to $\tilde{A}\tilde{\mathbf{u}} = \mathbf{v}$ is $\tilde{\mathbf{u}} = \begin{pmatrix} 200 \\ 33.5 \\ -173.643 \end{pmatrix}$
  - Recall that the solution to $A\mathbf{u} = \mathbf{v}$ was $\mathbf{u} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$
  - The relative error is $\dfrac{\|\mathbf{u} - \tilde{\mathbf{u}}\|_2}{\|\mathbf{u}\|_2} \approx 153.0$ or 15300%

Ill-conditioned systems

- Consider this matrix: $A = \begin{pmatrix} 4 & -3 & 4 \\ 2 & 4 & 3 \\ 3 & 3 & 4 \end{pmatrix}$
- If $\tilde{\mathbf{v}} = \begin{pmatrix} 4.99 \\ 8.99 \\ 10.01 \end{pmatrix}$, the solution to $A\tilde{\mathbf{u}} = \tilde{\mathbf{v}}$ is $\tilde{\mathbf{u}} = \begin{pmatrix} 0.44 \\ 0.91 \\ 1.49 \end{pmatrix}$
  - Recall that the solution to $A\mathbf{u} = \mathbf{v}$ was $\mathbf{u} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$
  - A relative error of $\frac{\|\mathbf{v} - \tilde{\mathbf{v}}\|_2}{\|\mathbf{v}\|_2} \approx 0.0012$ in the target vector

    results in the relative error of $\frac{\|\mathbf{u} - \tilde{\mathbf{u}}\|_2}{\|\mathbf{u}\|_2} \approx 0.43$ in the solution

Ill-conditioned systems

- In first year, you were repeatedly told to check if the determinant was non-zero when checking for invertibility
  - Unfortunately, due to numeric error, the determinant of even clearly non-invertible matrices is non-zero
  - Consider

```
>> B = [1 2 3; 4 5 6; 7 8 9];
>> det( B )
    ans =
        -9.5162e-16
>> 2*B(:,2) - B(:,1)
    ans =
        3
        6
        9
```

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

Ill-conditioned systems

- What is better to check are the singular values
  - Specifically, the ratio between the largest singular value and the smallest singular value
    - The largest singular value is the most a matrix stretches the norm of a vector
    - The smallest singular value is the least a matrix stretches the norm of a vector

```
>> A = [4 -3 4; 2 4 3; 3 3 4];
>> svd( A )
    ans =
        8.5702
        5.5274
        0.0211
```

$$A = \begin{pmatrix} 4 & -3 & 4 \\ 2 & 4 & 3 \\ 3 & 3 & 4 \end{pmatrix}$$

Ill-conditioned systems

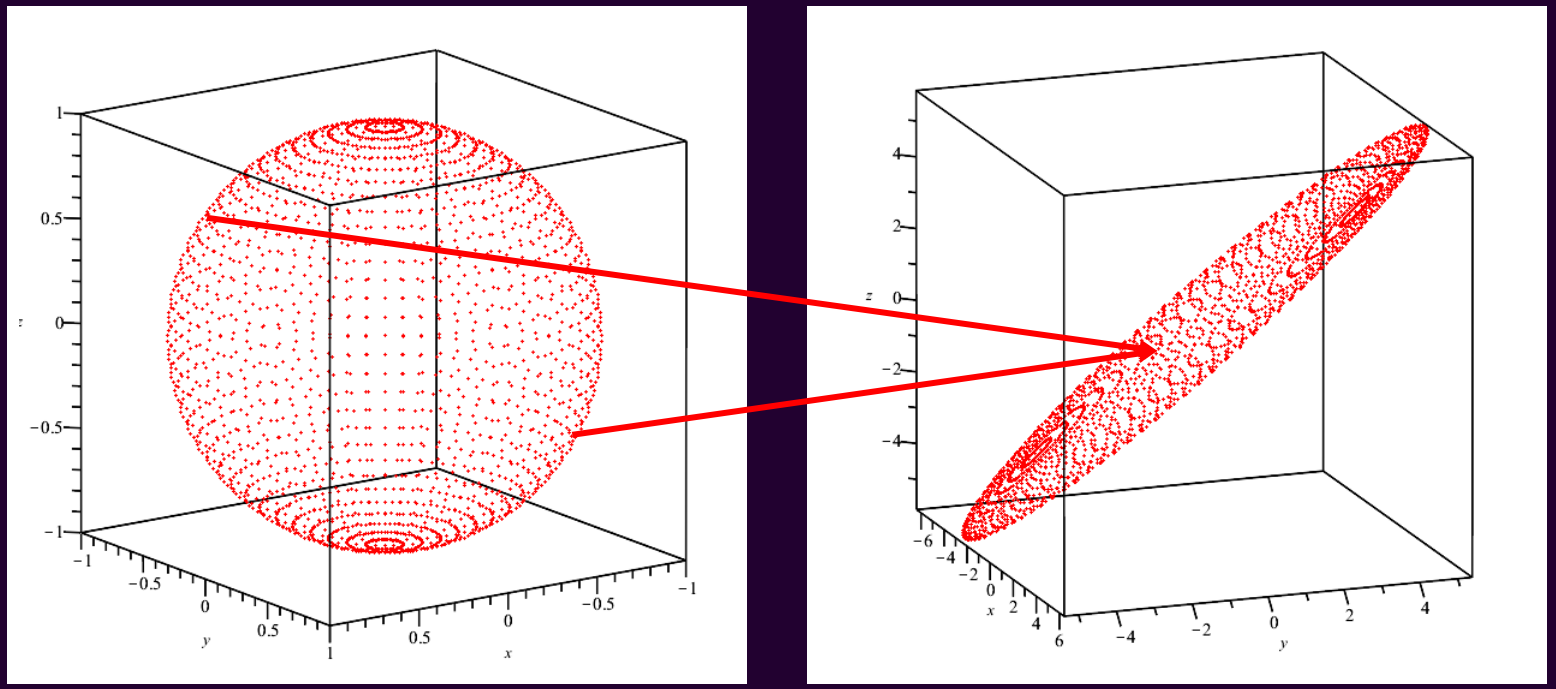
- Suppose a 3×3 matrix A has $\text{rank}(A) = 2$
  - The matrix is not invertible
  - The image of the unit sphere will be an ellipse centered at the origin on a plane passing through the origin

- Suppose now 3×3 matrix A has $\text{rank}(A) = 3$, but where the image of the unit sphere is now pancake shaped
  - In our example,
    - The sphere is stretched by a factor of almost 10
    - In another perpendicular direction by a factor of 5
    - In another perpendicular direction it is shrunk by a factor of 50
  - The matrix is invertible, but it is close to a matrix that is not

Ill-conditioned Systems

- This matrix is not invertible, and the image of the unit sphere is an ellipse in $\mathbf{R}^3$

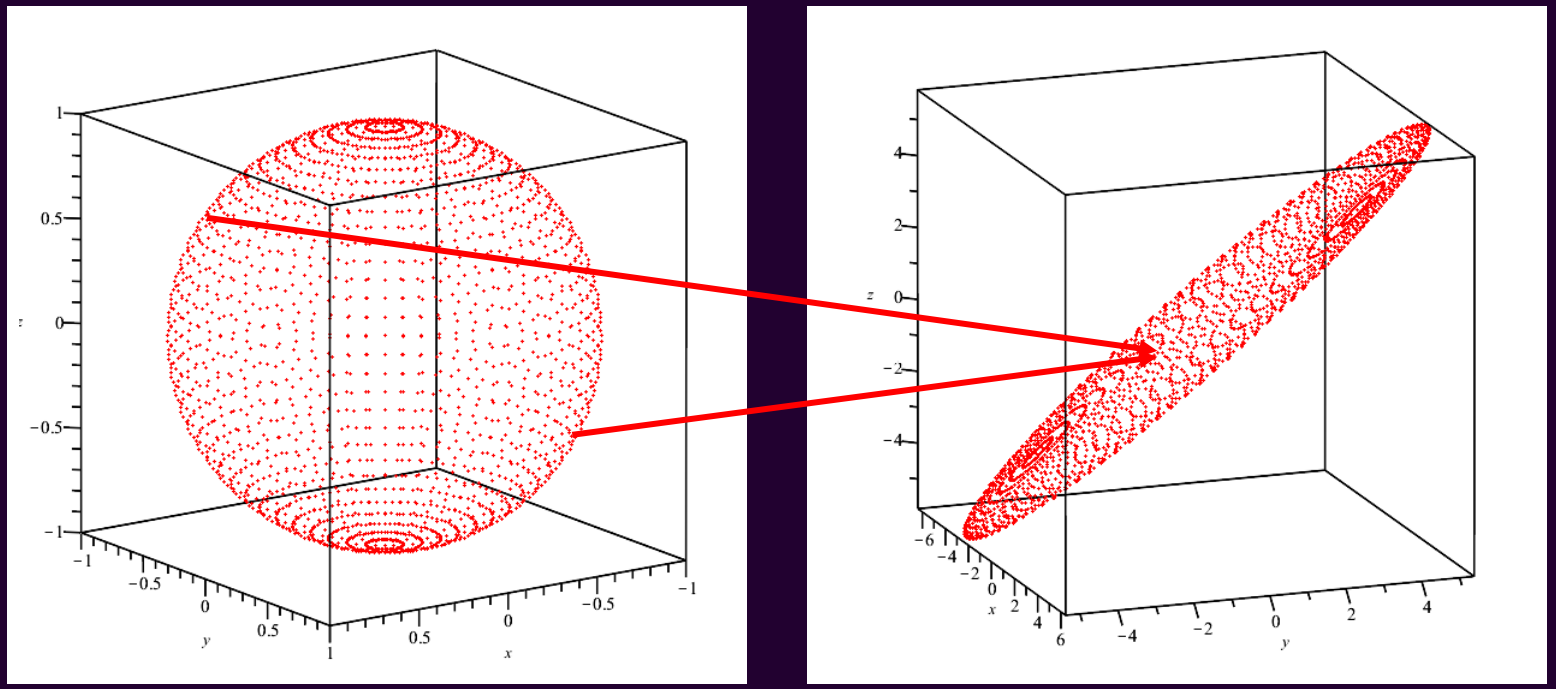
$$\tilde{A} = \begin{pmatrix} 4 & -3 & 4 \\ 2 & 4 & 3 \\ 3.04 & 3 & 4 \end{pmatrix}$$



Ill-conditioned Systems

- This matrix is invertible, but very close to the previous matrix

$$A = \begin{pmatrix} 4 & -3 & 4 \\ 2 & 4 & 3 \\ 3 & 3 & 4 \end{pmatrix}$$



Ill-conditioned systems

- This ratio between the largest and smallest stretch is called the *condition number* and it describes the maximum increase in the relative error when solving a system of linear equations

```
>> A = [4 -3 4; 2 4 3; 3 3 4];
>> cond( A )
    ans =
        405.9726
>> format hex
>> A \ [5 9 10]'
    ans =
       3ff0000000000019
       3ff0000000000004
       3feffffffffffd4
```

```
…000000011001
…000000000100
…111111010100
```

You don't have to know how to calculate the condition number

Ill-conditioned systems

- The larger the condition number:
  - The larger small errors in the matrix will be magnified
  - The larger small errors in the target vector will be magnified
  - The larger round-off error and effects of subtractive cancellation will be magnified

- For the purposes of this course,
  you must simply be aware that a large condition number suggests you must consider how sensitive your system is to errors in the implementation

You don't have to know how to calculate the condition number

Summary

- Following this topic, you now
  - Understand that solving a system of linear equations may result in significant errors
  - Understand that the Gaussian elimination algorithm with partial pivoting reduces the effect of such errors
  - Have seen the Jacobi method, where we approximate a solution to a system of linear equations using iteration
  - Are aware of the condition number of a matrix
    - You do not need to know how to calculate the condition number

References

[1] https://en.wikipedia.org/wiki/Gaussian_elimination

[2] https://en.wikipedia.org/wiki/Pivot_element

[3] https://en.wikipedia.org/wiki/Jacobi_method

[4] https://en.wikipedia.org/wiki/Condition_number

Acknowledgments

Tazik Shahjahan for pointing out typos.

Hassaan Ali Qazi for suggesting showing the images of the unit sphere under the two matrices, one ill-conditioned, the other non-invertible.

Colophon

These slides were prepared using the Cambria typeface. Mathematical equations use Times New Roman, and source code is presented using `Consolas`. Mathematical equations are prepared in MathType by Design Science, Inc.

Examples may be formulated and checked using Maple by Maplesoft, Inc.

The photographs of flowers and a monarch butter appearing on the title slide and accenting the top of each other slide were taken at the Royal Botanical Gardens in October of 2017 by Douglas Wilhelm Harder. Please see

https://www.rbg.ca/

for more information.

Disclaimer

These slides are provided for the ECE 204 *Numerical methods* course taught at the University of Waterloo. The material in it reflects the author's best judgment in light of the information available to them at the time of preparation. Any reliance on these course slides by any party for any other purpose are the responsibility of such parties. The authors accept no responsibility for damages, if any, suffered by any party as a result of decisions made or actions based on these course slides for any other purpose than that for which it was intended.