

EE 521/ECE 582 – Analysis of Power systems

Class #5 – September 8, 2022

*Dr. Noel N. Schulz
Edmund O. Schweitzer III Chair in
Power Apparatus and Systems
Chief Scientist Joint Appointment, PNNL
Co-Director, PNNL/WSU Advanced Grid Institute (AGI)
Washington State University Pullman
Noel.Schulz@wsu.edu EME 35
509-335-0980 (o) and 509-336-5522 (c)*

- Here are my office hours via zoom and in-person as possible. I will let you know when I am in each location
- **Tuesdays 4:30-5:30 pm (after class)**
- **Wednesdays 4-5 pm**
- **Fridays 1:30-2:30 pm**
- Below is the zoom information for the semester! I will also set other times to go over your code individually.
- Join from PC, Mac, Linux, iOS, or Android: <https://wsu.zoom.us/j/8237216735> (Links to an external site.)
- Meeting ID: 823 721 6735

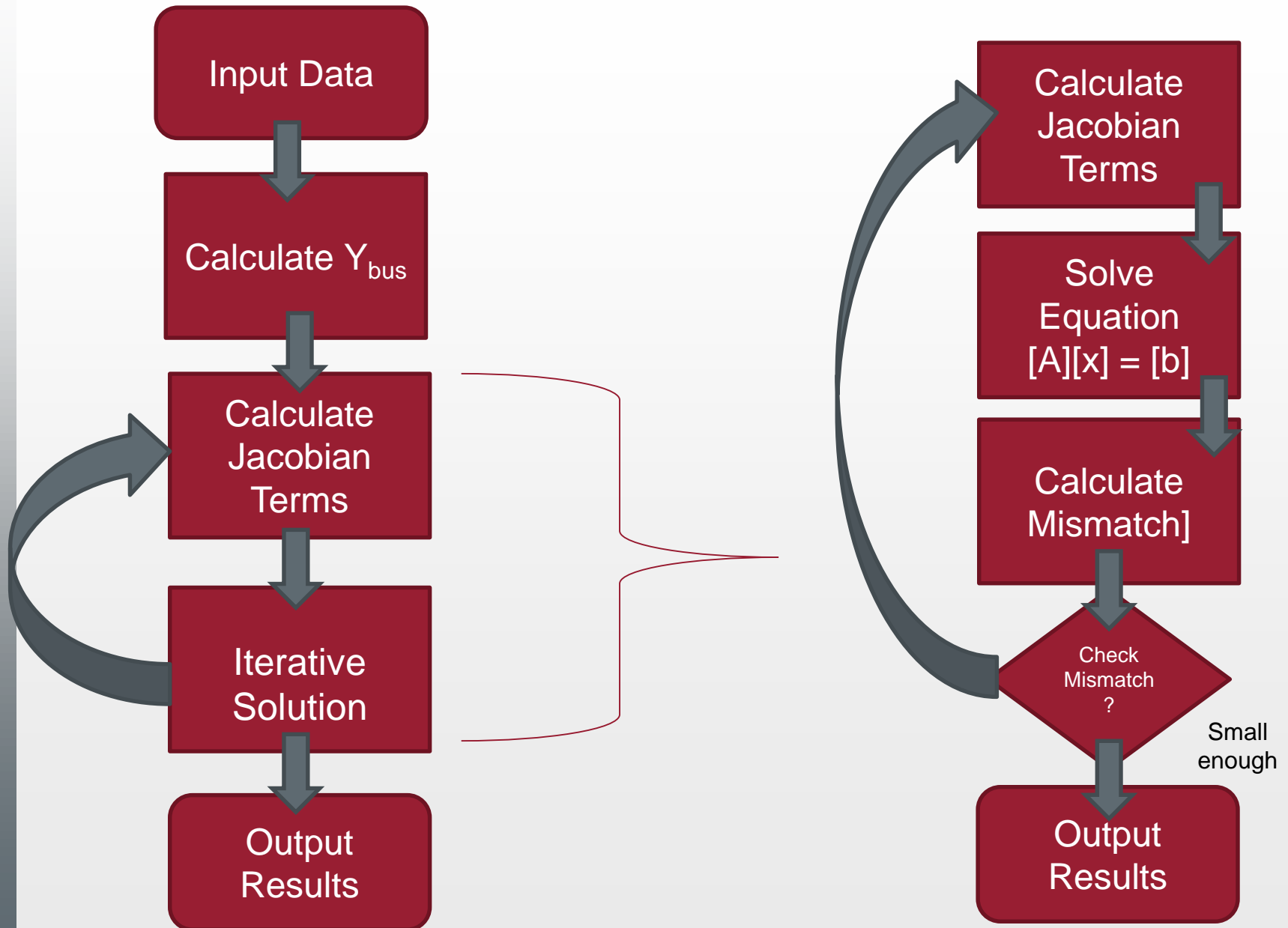
Programming Projects: (80%)

1. Power Flow Project (20%)

- A. Y_{bus} , Jacobian matrices – Newton Raphson
- B. LU Factorization, Backward-Forward substitution for NR
- C. Fast Decoupled
- D. Q-limits, transformer taps

- Using IEEE 14 bus system as given, and I will provide a slightly modified version after you confirm the 14-bus system.
- Projects build on each other, adding different features
- In addition to getting the right answers, for each program we'll set up a session for you to describe your code to me.
- Copying code from a colleague or from on-line resources is not permitted.

Newton-Raphson and Parts



Solving the Matrix once you calculate it

$$\begin{array}{c}
 \textcircled{2} \quad \textcircled{3} \quad \textcircled{4} \quad \textcircled{2} \quad \textcircled{3} \\
 \begin{array}{c}
 \textcircled{2} \\
 \textcircled{3} \\
 \textcircled{4} \\
 \textcircled{2} \\
 \textcircled{3}
 \end{array}
 \left[\begin{array}{ccc|cc}
 45.443 & 0 & -26.365 & 8.882 & 0 \\
 0 & 41.269 & -15.421 & 0 & 8.133 \\
 -26.365 & -15.421 & 41.786 & -5.273 & -3.084 \\
 \hline
 -9.089 & 0 & 5.273 & 44.229 & 0 \\
 0 & -8.254 & 3.084 & 0 & 40.459
 \end{array} \right]
 \begin{bmatrix}
 \Delta\delta_2 \\
 \Delta\delta_3 \\
 \Delta\delta_4 \\
 \hline
 \frac{\Delta|V_2|}{|V_2|} \\
 \hline
 \frac{\Delta|V_3|}{|V_3|}
 \end{bmatrix}
 =
 \begin{bmatrix}
 -1.597 \\
 -1.940 \\
 \hline
 2.213 \\
 \hline
 -0.447 \\
 -0.835
 \end{bmatrix}
 \end{array}$$

An elementary row operation is one of the following:

1. Interchange any two rows of the matrix
2. Multiply any row by a constant
3. Take a linear combination of rows and add it to another row

Gaussian Elimination for solving $Ax=b$

is the process of using a series of elementary row operations to convert:

$$\left[\begin{array}{cccc|c} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} & b_n \end{array} \right] \xrightarrow{\quad} \left[\begin{array}{cccc|c} 1 & 0 & \cdots & 0 & x_1^* \\ 0 & 1 & \cdots & 0 & x_2^* \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & x_n^* \end{array} \right]$$

Identity matrix

solution

An elementary row operation is one of the following:

1. Interchange any two rows of the matrix
2. Multiply any row by a constant
3. Take a linear combination of rows and add it to another row

Basic Approach to Gaussian Elimination

Upper Triangularize (i.e. zero out all elements below the diagonal) using elementary row operations (EROs)

$$\left[\begin{array}{cccc|c} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} & b_n \end{array} \right] \xrightarrow{\text{(EROs)}} \left[\begin{array}{cccc|c} 1 & a'_{12} & \cdots & a'_{1n} & b_1^* \\ 0 & 1 & \cdots & a'_{2n} & b_2^* \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & b_n^* \end{array} \right]$$

All zeros below diagonal

Then back substitute to find solutions

$$\text{ERO}(4) \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \begin{array}{l} \text{Scale diagonal } a_{II} \text{ to 1} \\ \text{(in this case ERO}(4) = \text{identity since } a_{II} \text{ is already 1)} \end{array}$$

Update A by EROs 1 through 4

$$\begin{array}{cccccc} \text{ERO}(4) & \text{ERO}(3) & \text{ERO}(2) & \text{ERO}(1) & A & A' \end{array}$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -9 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -4 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -2 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 3 & 4 & 8 \\ 2 & 1 & 2 & 3 \\ 4 & 3 & 5 & 8 \\ 9 & 2 & 7 & 4 \end{bmatrix} = \begin{bmatrix} 1 & 3 & 4 & 8 \\ 0 & -5 & -6 & -13 \\ 0 & -9 & -11 & -24 \\ 0 & -25 & -29 & -68 \end{bmatrix}$$

All zeros

Update A' by EROs 5 through 7

$$\begin{array}{ccccc}
 \text{ERO(7)} & & \text{ERO(6)} & & \text{ERO(5)} & & A' & & A'' \\
 \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -\frac{1}{5} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} & \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -\frac{25}{5} & 0 & 1 \end{bmatrix} & \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & -\frac{9}{5} & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} & \begin{bmatrix} 1 & 3 & 4 & 8 \\ 0 & -5 & -6 & -13 \\ 0 & -9 & -11 & -24 \\ 0 & -25 & -29 & -68 \end{bmatrix} & = & \begin{bmatrix} 1 & 3 & 4 & 8 \\ 0 & 1 & 1.2 & 2.6 \\ 0 & 0 & -0.2 & -0.6 \\ 0 & 0 & 1 & -3 \end{bmatrix}
 \end{array}$$

All zeros

Now eliminate third column

$$\begin{array}{ccc}
 \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 5 & 1 \end{bmatrix} & \text{And diagonalize} & \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -5 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} & \xrightarrow{\text{Green Arrow}} & \begin{bmatrix} 1 & 3 & 4 & 8 \\ 0 & 1 & 1.2 & 2.6 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & -6 \end{bmatrix} \\
 \text{ERO(8)} & & \text{ERO(9)} & & A'''
 \end{array}$$

$$\begin{array}{ccc}
 \begin{array}{c} A''' \\ \left[\begin{array}{cccc} 1 & 3 & 4 & 8 \\ 0 & 1 & 1.2 & 2.6 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & -6 \end{array} \right] \end{array} & \begin{array}{c} \text{Normalize to 1} \\ \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -\frac{1}{6} \end{array} \right] \end{array} & \begin{array}{c} \xrightarrow{\hspace{1cm}} \\ \left[\begin{array}{cccc} 1 & 3 & 4 & 8 \\ 0 & 1 & 1.2 & 2.6 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 1 \end{array} \right] \end{array} \\
 & \text{ERO(10)} &
 \end{array}$$

Multiply b by EROs 1 through 10:

$$\text{ERO(10)ERO(9)ERO(8)...ERO(1)}b = b'''' = \begin{bmatrix} 1 \\ 0.2 \\ 6 \\ 1.5 \end{bmatrix}$$

Now we can use back substitution to find x_1 , x_2 , x_3 , and x_4

$$\begin{bmatrix} 1 & 3 & 4 & 8 \\ 0 & 1 & 1.2 & 2.6 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 1 \\ 0.2 \\ 6 \\ 1.5 \end{bmatrix}$$

$$x_4 = 1.5$$

$$x_3 = 6 - 3(1.5) = 1.5$$

$$x_2 = 0.2 - 2.6(1.5) - 1.2(1.5) = -5.5$$

$$x_1 = 1 - 8(1.5) - 4(1.5) - 3(-5.5) = -0.5$$



$$\begin{bmatrix} -0.5 \\ -5.5 \\ 1.5 \\ 1.5 \end{bmatrix} \\ x^*$$

LU Factorization

LU Factorization is an extension of Gaussian elimination in that the forward triangularization process is formalized such that the matrix A is factorized into an upper triangular matrix (U) and a lower triangular matrix (L) such that

$$A = LU$$

So how does this help us solve
 $Ax=b$?

Note that $Ax=b$ implies that $LUx=b$. If we introduce a “dummy” variable y such that $y=Ux$, then

$$Ax = b$$

$$LUx = b$$

$$Ly = b$$



Since L is lower triangular, then it is easy to solve for y !

This is forward substitution.

And once we have y , then we can solve $Ux=y$, where U is upper triangular.

This is backward substitution.

Let $Q = L + U - I$

$$= \begin{bmatrix} l_{11} & u_{12} & u_{13} & \cdots & u_{1n} \\ l_{21} & l_{22} & u_{23} & \cdots & u_{2n} \\ l_{31} & l_{32} & l_{33} & \cdots & u_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ l_{n1} & l_{n2} & l_{n3} & \cdots & l_{nn} \end{bmatrix}$$

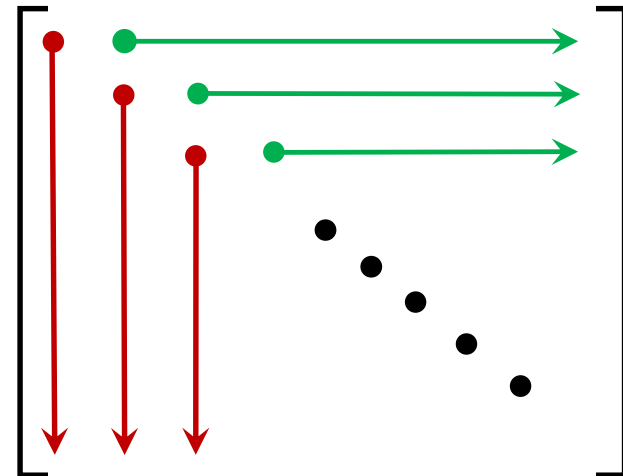
L

U

Note that U has ones (1) on the diagonal

Q will be calculated first by column
and then by row

Q
=



Crout's algorithm for computing LU from A

1. Initialize Q to the zero matrix. Let $j=1$.
2. Complete the j^{th} column of Q (j^{th} column of L) as

$$q_{kj} = a_{kj} - \sum_{i=1}^{j-1} q_{ki}q_{ij} \quad \text{for } k = j, \dots, n$$

3. If $j=n$, then stop.
4. Assuming that $q_{jj} \neq 0$, complete the j^{th} row of Q (j^{th} row of U) as

$$q_{jk} = \frac{1}{q_{jj}} \left(a_{jk} - \sum_{i=1}^{j-1} q_{ji}q_{ik} \right) \quad \text{for } k = j+1, \dots, n$$

5. Set $j=j+1$. Go to step 2.

Note that Q overwrites A , thus they can both be stored in the same matrix!

$$Q = \begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} & q_{15} & q_{16} & \cdots & q_{1n} \\ q_{21} & q_{22} & q_{23} & q_{24} & q_{25} & q_{26} & \cdots & q_{2n} \\ q_{31} & q_{32} & q_{33} & q_{34} & q_{35} & q_{36} & \cdots & q_{3n} \\ q_{41} & q_{42} & q_{43} & q_{44} & q_{45} & q_{46} & \cdots & q_{4n} \\ q_{51} & q_{52} & q_{53} & q_{54} & \boxed{} & & & \\ q_{61} & q_{62} & q_{63} & q_{64} & & & & \\ \vdots & \vdots & \vdots & \vdots & & & & \\ q_{n1} & q_{n2} & q_{n3} & q_{n4} & & & & \end{bmatrix}$$

Example: To find q_{55} :

$$q_{55} = a_{55} - \sum_{i=1}^4 q_{5i} q_{i5}$$

$$= a_{55} - (q_{51}q_{15} + q_{52}q_{25} + q_{53}q_{35} + q_{54}q_{45})$$

$$Q = \begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} & q_{15} & q_{16} & \cdots & q_{1n} \\ q_{21} & q_{22} & q_{23} & q_{24} & q_{25} & q_{26} & \cdots & q_{2n} \\ q_{31} & q_{32} & q_{33} & q_{34} & q_{35} & q_{36} & \cdots & q_{3n} \\ q_{41} & q_{42} & q_{43} & q_{44} & q_{45} & q_{46} & \cdots & q_{4n} \\ q_{51} & q_{52} & q_{53} & q_{54} & q_{55} & & & \\ q_{61} & q_{62} & q_{63} & q_{64} & \boxed{} & & & \\ \vdots & \vdots & \vdots & \vdots & & & & \\ q_{n1} & q_{n2} & q_{n3} & q_{n4} & & & & \end{bmatrix}$$

Similarly: To find q_{65} :

$$q_{65} = a_{65} - \sum_{i=1}^4 q_{6i} q_{i5}$$

$$= a_{65} - (q_{61}q_{15} + q_{62}q_{25} + q_{63}q_{35} + q_{64}q_{45})$$

Example: Find the LU factors of the A matrix

$$A = \begin{bmatrix} 1 & 3 & 4 & 8 \\ 2 & 1 & 2 & 3 \\ 4 & 3 & 5 & 8 \\ 9 & 2 & 7 & 4 \end{bmatrix}$$

Let $j=1$, then from $q_{kj} = a_{kj} - \sum_{i=1}^{j-1} q_{ki}q_{ij}$ we get $q_{k1} = a_{k1}$
and the first column of Q is just the first column
of A :

For the first row of Q , we use $q_{jk} = \frac{1}{q_{jj}} \left(a_{jk} - \sum_{i=1}^{j-1} q_{ji}q_{ik} \right)$

from which $q_{1k} = \frac{a_{1k}}{q_{11}}$

$$Q = \begin{bmatrix} 1 & 3 & 4 & 8 \\ 2 & 1 & 2 & 3 \\ 4 & 3 & 5 & 8 \\ 9 & 2 & 7 & 4 \end{bmatrix}$$

$$q_{12} = \frac{3}{1}$$

$$q_{13} = \frac{4}{1}$$

$$q_{14} = \frac{8}{1}$$

$$Q = \begin{bmatrix} 1 & \textcircled{3} & 4 & 8 \\ \textcircled{2} & & & \\ \textcircled{4} & & & \\ \textcircled{9} & & & \end{bmatrix}$$

Second column ($j = 2$):

$$q_{k2} = a_{k2} - \sum_{i=1}^1 q_{ki} q_{i2} = a_{k2} - q_{k1} q_{12}$$

$$q_{22} = a_{22} - q_{21} q_{12} = 1 - (2)(3) = -5$$

$$q_{32} = a_{32} - q_{31} q_{12} = 3 - (4)(3) = -9$$

$$q_{42} = a_{42} - q_{41} q_{12} = 2 - (9)(3) = -25$$

Second row:

$$Q = \begin{bmatrix} 1 & 3 & \textcircled{4} & \textcircled{8} \\ \textcircled{2} & -5 & & \\ 4 & -9 & & \\ 9 & -25 & & \end{bmatrix}$$

$$q_{2k} = \frac{1}{q_{22}} \left(a_{2k} - \sum_{i=1}^1 q_{2i} q_{ik} \right) = \frac{1}{q_{22}} (a_{2k} - q_{21} q_{1k})$$

$$q_{23} = \frac{1}{q_{22}} (a_{23} - q_{21} q_{13}) = \frac{1}{-5} (2 - (2)(4)) = 1.2$$

$$q_{24} = \frac{1}{q_{22}} (a_{24} - q_{21} q_{14}) = \frac{1}{-5} (3 - (2)(8)) = 2.6$$

$$Q = \begin{bmatrix} 1 & 3 & 4 & 8 \\ 2 & -5 & 1.2 & 2.6 \\ 4 & -9 & & \\ 9 & -25 & & \end{bmatrix}$$

Third column ($j = 3$):

$$q_{k3} = a_{k3} - \sum_{i=1}^2 q_{ki} q_{i3} = a_{k3} - (q_{k1} q_{13} + q_{k2} q_{23})$$

$$q_{33} = a_{33} - (q_{31} q_{13} + q_{32} q_{23}) \\ = 5 - ((4)(4) + (-9)(1.2)) = -0.2$$

$$q_{43} = a_{43} - (q_{41} q_{13} + q_{42} q_{23}) \\ = 7 - ((9)(4) + (-25)(1.2)) = 1$$

$$Q = \begin{bmatrix} 1 & 3 & 4 & 8 \\ 2 & -5 & 1.2 & 2.6 \\ 4 & -9 & -0.2 & \\ 9 & -25 & 1 & \end{bmatrix}$$

Third row:

$$q_{3k} = \frac{1}{q_{33}} \left(a_{3k} - \sum_{i=1}^2 q_{3i} q_{ik} \right) = \frac{1}{q_{33}} (a_{3k} - (q_{31} q_{1k} + q_{32} q_{2k}))$$

$$q_{34} = \frac{1}{q_{33}} (a_{34} - (q_{31} q_{14} + q_{32} q_{24})) \\ = \frac{1}{-0.2} (8 - ((4)(8) + (-9)(2.6))) = 3$$

Fourth column ($j = 4$)

$$Q = \begin{bmatrix} 1 & 3 & 4 & 8 \\ 2 & -5 & 1.2 & 2.6 \\ 4 & -9 & -0.2 & 3 \\ 9 & -25 & 1 & -6 \end{bmatrix}$$

$$q_{kj} = a_{kj} - \sum_{i=1}^{j-1} q_{ki} q_{ij}$$

$$q_{44} = a_{44} - (q_{41}q_{14} + q_{42}q_{24} + q_{43}q_{34})$$

$$= 4 - ((9)(8) + (-25)(2.6) + (3)(1)) = -6$$

Check: $LU=A$?

$$LU = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 2 & -5 & 0 & 0 \\ 4 & -9 & -0.2 & 0 \\ 9 & -25 & 1 & -6 \end{bmatrix} \begin{bmatrix} 1 & 3 & 4 & 8 \\ 0 & 1 & 1.2 & 2.6 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 3 & 4 & 8 \\ 2 & 1 & 2 & 3 \\ 4 & 3 & 5 & 8 \\ 9 & 2 & 7 & 4 \end{bmatrix} \quad \checkmark$$

Solve
$$\begin{bmatrix} 1 & 3 & 4 & 8 \\ 2 & 1 & 2 & 3 \\ 4 & 3 & 5 & 8 \\ 9 & 2 & 7 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

1. Solve $Ly = b$ using Forward substitution

$$y_k = \frac{1}{q_{kk}} \left(b_k - \sum_{j=1}^{k-1} q_{kj} y_j \right) \text{ for } k = 1, \dots, n$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 2 & -5 & 0 & 0 \\ 4 & -9 & -0.2 & 0 \\ 9 & -25 & 1 & -6 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

2. Solve $Ux=y$ using Backward substitution

$$x_k = y_k - \sum_{j=k+1}^{n-1} q_{kj} x_j \text{ for } k = n, n-1, \dots, 1$$

$$y_1 = 1 \quad y_2 = \frac{1}{-5} (1 - (2)(1)) = 0.2$$

$$y_3 = \frac{1}{-0.2} (1 - ((4)(1) + (-9)(0.2))) = 6$$

$$y_4 = \frac{1}{-6} (1 - ((9)(1) + (-25)(0.2) + (1)(6))) = 1.5$$

$$\begin{bmatrix} 1 & 3 & 4 & 8 \\ 0 & 1 & 1.2 & 2.6 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 1 \\ 0.2 \\ 6 \\ 1.5 \end{bmatrix}$$

Note that this is exactly the same backward substitution that we had for Gaussian Elimination!

$$x_4 = 1.5 \quad x_3 = 1.5 \quad x_2 = -5.5 \quad x_1 = -0.5$$


What is zero?

Challenges Solving Problems with Computers

What is good enough when solving?

Now do the magnitude relationships affect our results?

Partial Pivoting for LU factorization

Solve $\begin{bmatrix} 10^{-10} & 1 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 5 \end{bmatrix}$ 

We can estimate that the solution to this system will be:

$$x_2 \approx 1$$

$$x_1 \approx \frac{1}{2}(5 - (1)(1)) = 2$$

Find LU factors:

$$Q = \begin{bmatrix} 10^{-10} & 10^{10} \\ 2 & 1 - (2)(10^{10}) \end{bmatrix} \approx \begin{bmatrix} 10^{-10} & 10^{10} \\ 2 & -2 \times 10^{10} \end{bmatrix}$$

Solve for y :

$$\begin{bmatrix} 10^{-10} & 0 \\ 2 & -2 \times 10^{10} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 5 \end{bmatrix} \longrightarrow \begin{aligned} y_1 &= \frac{1}{10^{-10}} = 10^{10} \\ y_2 &= \frac{(5 - 2 \times 10^{10})}{-2 \times 10^{10}} \approx 1 \end{aligned}$$

Solve for x :

$$\begin{bmatrix} 1 & 10^{10} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \approx \begin{bmatrix} 10^{10} \\ 1 \end{bmatrix} \longrightarrow \begin{aligned} x_2 &\approx 1 & x_1 &\approx 10^{10} - 10^{10} = 0 \end{aligned}$$

This cannot be correct!

It should be $x_1 \approx 2$

What happened?

The problem is that 10^{-10} is too close to zero for most computers (and human) accuracy, essentially leading to a “divide by zero” problem

We can fix this error by rearranging the equations:
$$\begin{bmatrix} 2 & 1 \\ 10^{-10} & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 5 \\ 1 \end{bmatrix}$$

The LU factors are:

$$Q = \begin{bmatrix} 2 & 0.5 \\ 10^{-10} & 1 - (0.5)(10^{-10}) \end{bmatrix} \approx \begin{bmatrix} 2 & 0.5 \\ 10^{-10} & 1 \end{bmatrix}$$

Resulting in:

$$\begin{bmatrix} 2 & 0 \\ 10^{-10} & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 5 \\ 1 \end{bmatrix} \quad \begin{aligned} y_1 &= 2.5 \\ y_2 &= 1 - 2.5 \times 10^{-10} \approx 1 \end{aligned}$$

Then:

$$\begin{bmatrix} 1 & 0.5 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \approx \begin{bmatrix} 2.5 \\ 1 \end{bmatrix} \quad \begin{aligned} x_2 &= 1 \\ x_1 &= 2.5 - 0.5 = 2 \end{aligned}$$

Much better!

We can introduce a pivoting strategy to move the largest element to the diagonal to avoid any potential divide by zero problems

Partial Pivoting Strategy

1. At the j^{th} step of the LU factorization, choose the k^{th} row as the exchange row such that

$$|q_{jj}| = \max |q_{kj}| \text{ for } k = j, \dots, n$$

2. Exchange rows and update A , P , and Q correspondingly



What is P ?

P is a **permutation matrix** that reflects any row interchanges that have occurred during partial pivoting. P starts as the identity matrix.

$$P' = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & \boxed{0} & 0 & 0 & \boxed{1} & \cdots & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & \boxed{1} & 0 & 0 & \boxed{0} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 1 \end{bmatrix} \quad \begin{array}{c} \curvearrowright \\ \text{Swap rows} \end{array}$$

Repeat previous example with partial pivoting.

Calculate third row and fourth column of Q :

$$Q = \begin{bmatrix} 9.000 & 0.2222 & 0.7778 & 0.9999 \\ 2.000 & 0.5556 & 1.1600 & 2.2222 \\ 4.000 & 2.1111 & -0.5600 & -0.8571 \\ 1.000 & 0.5556 & -0.2000 & 0.4286 \end{bmatrix}$$

This is the largest element in the second column, so no pivoting required.

$$A = \begin{bmatrix} 9 & 2 & 4 & 8 \\ 2 & 3 & 2 & 8 \\ 4 & 3 & 5 & 8 \\ 1 & 3 & 2 & 8 \end{bmatrix}$$

$$P = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

How is P used?

Start with $Ax = b$

Premultiply both sides by P : $PAx = Pb$

Substitute: $LU = PA$

$$LUx = \underbrace{Pb}$$

Solve: $Ly = Pb$

← This is just a reordered b vector

Then: $Ux = y$ (same as usual. The order of x remains unchanged.)

Complete Pivoting for LU factorization

In some cases, complete pivoting is used in which the largest element remaining (rather than in just the column) is pivoted to the diagonal element.

This requires both a column and row pivot.

In this case, two permutation matrices are developed such that $P_1AP_2 = LU$

Then (as before) $Ax = b$ Now define $P_2z = x$, then $P_1AP_2z = P_1b$

$$P_1Ax = P_1b$$

$$LUz = P_1b$$

Solve for z , then $x = P_2z$.

The vector z has the same values as x , just in a different order.

The action P_2z reorders the values back to the original order.

Gauss' algorithm for computing LU from A

1. Initialize Q to the zero matrix. Let $j=1$.
2. Complete the j^{th} column of Q (j^{th} column of L) to the j^{th} column of the reduced matrix $A^{(j)}$, where $A^{(1)}=A$ and

$$q_{kj} = a_{kj}^j \quad \text{for } k = j, \dots, n$$

3. If $j=n$, then stop.
4. Assuming that $q_{jj} \neq 0$, set the j^{th} row of Q (j^{th} row of U) as

$$q_{jk} = \frac{a_{jk}^{(j)}}{q_{jj}} \quad \text{for } k = j+1, \dots, n$$

5. Update $A^{(j+1)}$ from $A^{(j)}$ as:

$$a_{ik}^{(j+1)} = a_{ik}^{(j)} - q_{ij}q_{jk} \quad \text{for } i = j+1, \dots, n, \text{ and } k = j+1, \dots, n$$

6. Set $j=j+1$. Go to 2.

Stationary Iterative methods for solving $Ax=b$:

Iterative methods generate a sequence of approximations to x^* from an initial guess (i.e. $x^0, x^1, x^2, \dots x^k$) such that $x^k \rightarrow x^*$ as $k \rightarrow \infty$.

If convergent¹, these methods are accurate to within a user-defined convergence criterion.

Examples include:

1. Gauss Jacobi
2. Gauss Seidel
3. Successive Over-Relaxation

¹ we'll discuss convergence later

Relaxation methods for solving $Ax=b$:

Start with $Ax = b$

Define a matrix M such that: $Ax = b + Mx - Mx$

Rearranging: $Mx = (M - A)x + b$

Adding iteration indices: $Mx^{k+1} = (M - A)x^k + b \quad k = 1, \dots, \infty$

The choice of M determines if, and how fast, $Ax - b \rightarrow 0$.

When $\|Ax^{(k+1)} - b\| < \varepsilon$, for some small ε , then the iteration is said to have **converged**.

Jacobi

Let $M = D$ (the diagonal elements of A), then

$$\begin{aligned} Dx^{k+1} &= (D - A)x^k + b \\ x^{k+1} &= -D^{-1}((A - D)x^k - b) \\ &= M_J x^k + D^{-1}b \end{aligned}$$

Or in scalar form:
$$x_i^{k+1} = -\sum_{j \neq i}^n \left(\frac{a_{ij}}{a_{ii}} \right) x_j^k + \frac{b_i}{a_{ii}} \quad 1 \leq i \leq n, \quad k \geq 0$$

This method is attractive for parallel processing applications because each x_i^{k+1} depends only previous values of x^k . Thus each i^{th} element can be calculated independently.

Solve
$$\begin{bmatrix} -10 & 2 & 3 & 6 \\ 0 & -9 & 1 & 4 \\ 2 & 6 & -12 & 2 \\ 3 & 1 & 0 & -8 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

using the Jacobi method.

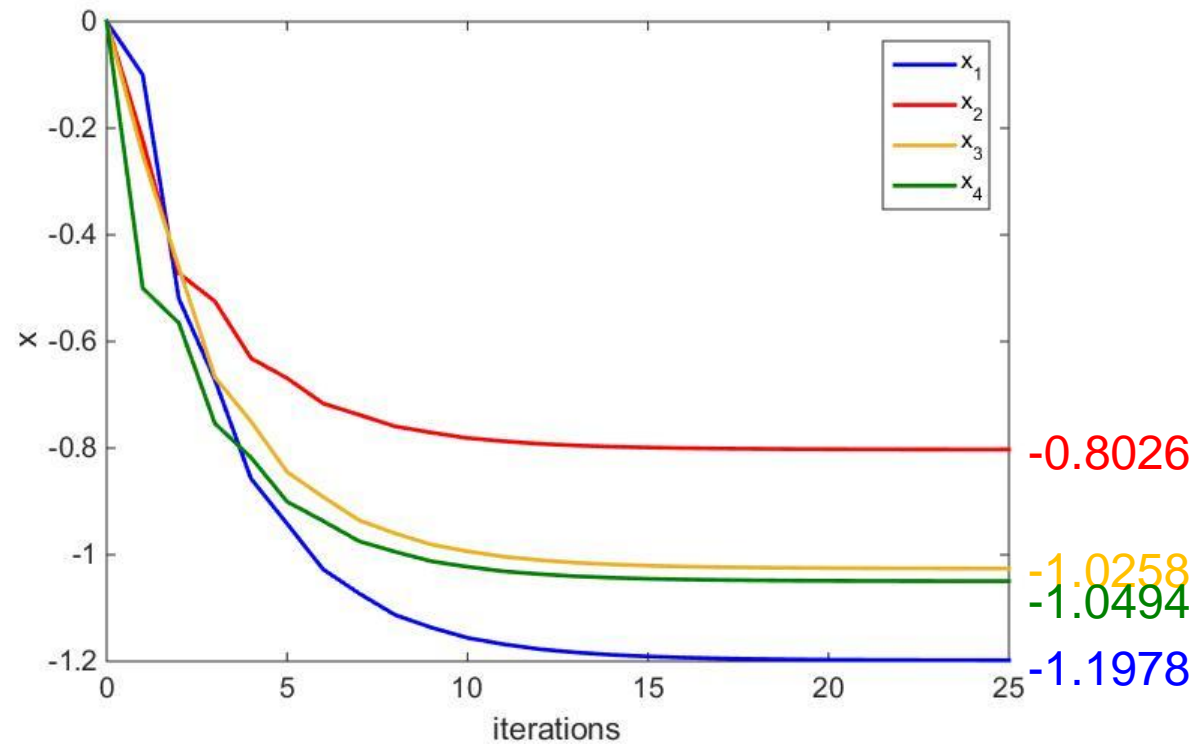
$$x_i^{k+1} = -\sum_{j \neq i}^n \left(\frac{a_{ij}}{a_{ii}} \right) x_j^k + \frac{b_i}{a_{ii}} \quad 1 \leq i \leq n, \quad k \geq 0$$

Set up the equations:

$$x_1^{k+1} = \frac{1}{10} (2x_2^k + 3x_3^k + 6x_4^k - 1)$$
$$x_2^{k+1} = \frac{1}{9} (0x_1^k + x_3^k + 4x_4^k - 2)$$
$$x_3^{k+1} = \frac{1}{12} (2x_1^k + 6x_2^k + 2x_4^k - 3)$$
$$x_4^{k+1} = \frac{1}{8} (3x_1^k + x_2^k + 0x_3^k - 4)$$

Jacobi

k	x_1	x_2	x_3	x_4
1	0.0000	0.0000	0.0000	0.0000
2	-0.1000	-0.2222	-0.2500	-0.5000
3	-0.5194	-0.4722	-0.4611	-0.5653
4	-0.6719	-0.5247	-0.6669	-0.7538
5	-0.8573	-0.6314	-0.7500	-0.8176
6	-0.9418	-0.6689	-0.8448	-0.9004
7	-1.0275	-0.7163	-0.8915	-0.9368
8	-1.0728	-0.7376	-0.9355	-0.9748
9	-1.1131	-0.7594	-0.9601	-0.9945
10	-1.1366	-0.7709	-0.9810	-1.0123
11	-1.1559	-0.7811	-0.9936	-1.0226
12	-1.1679	-0.7871	-1.0037	-1.0311
13	-1.1772	-0.7920	-1.0100	-1.0363
14	-1.1832	-0.7950	-1.0149	-1.0404
15	-1.1877	-0.7974	-1.0181	-1.0431
16	-1.1908	-0.7989	-1.0205	-1.0451
17	-1.1930	-0.8001	-1.0221	-1.0464
18	-1.1945	-0.8009	-1.0233	-1.0474
19	-1.1956	-0.8014	-1.0241	-1.0480
20	-1.1963	-0.8018	-1.0247	-1.0485
21	-1.1969	-0.8021	-1.0250	-1.0489
22	-1.1972	-0.8023	-1.0253	-1.0491
23	-1.1975	-0.8024	-1.0255	-1.0492
24	-1.1977	-0.8025	-1.0257	-1.0494
25	-1.1978	-0.8026	-1.0258	-1.0494



Solve
$$\begin{bmatrix} -10 & 2 & 3 & 6 \\ 0 & -9 & 1 & 4 \\ 2 & 6 & -12 & 2 \\ 3 & 1 & 0 & -8 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$
 using the Gauss-Seidel method.

$$x_i^{k+1} = -\sum_{j=1}^{i-1} \left(\frac{a_{ij}}{a_{ii}} \right) x_j^{k+1} - \sum_{j=i+1}^n \left(\frac{a_{ij}}{a_{ii}} \right) x_j^k + \frac{b_i}{a_{ii}} \quad 1 \leq i \leq n, \quad k \geq 0$$

Set up the equations:

$$x_1^{k+1} = \frac{1}{10} (2x_2^k + 3x_3^k + 6x_4^k - 1)$$

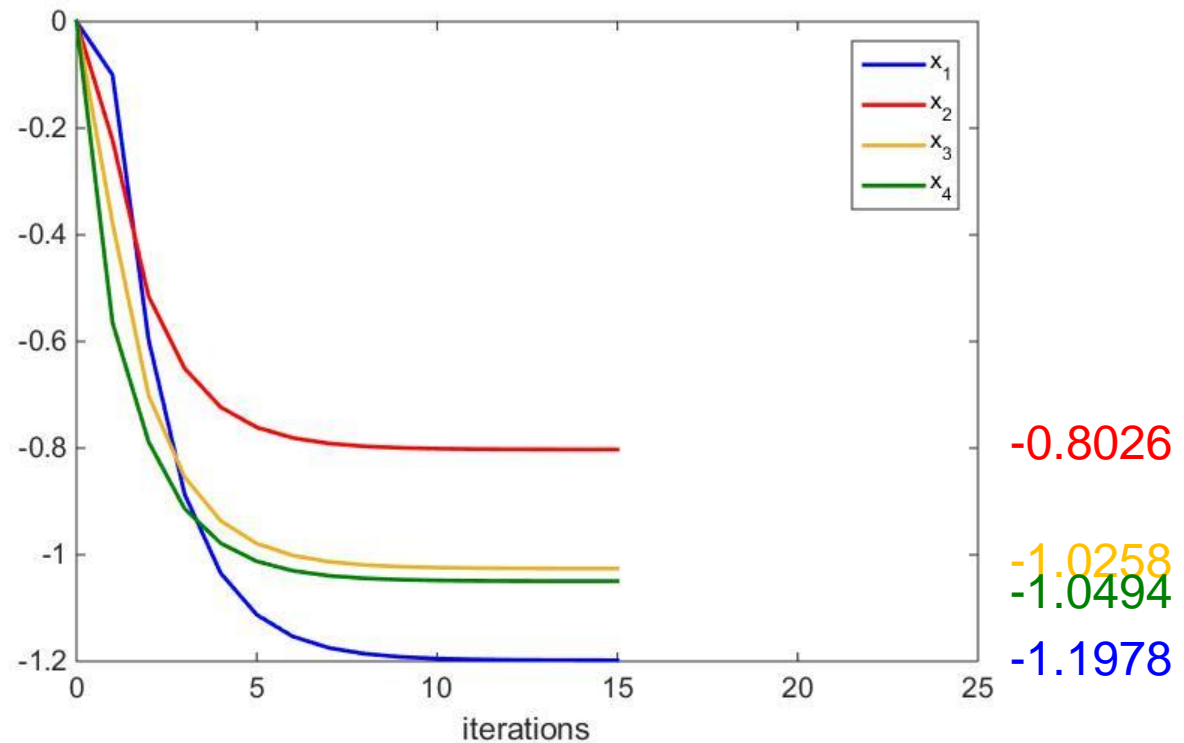
$$x_2^{k+1} = \frac{1}{9} (0x_1^{k+1} + x_3^k + 4x_4^k - 2)$$

$$x_3^{k+1} = \frac{1}{12} (2x_1^{k+1} + 6x_2^{k+1} + 2x_4^k - 3)$$

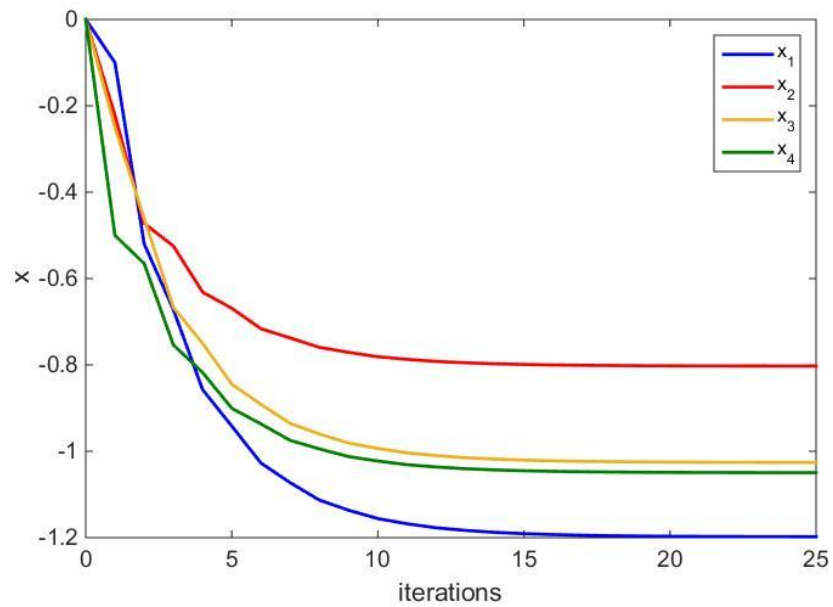
$$x_4^{k+1} = \frac{1}{8} (3x_1^{k+1} + x_2^{k+1} + 0x_3^{k+1} - 4)$$

Gauss-Seidel

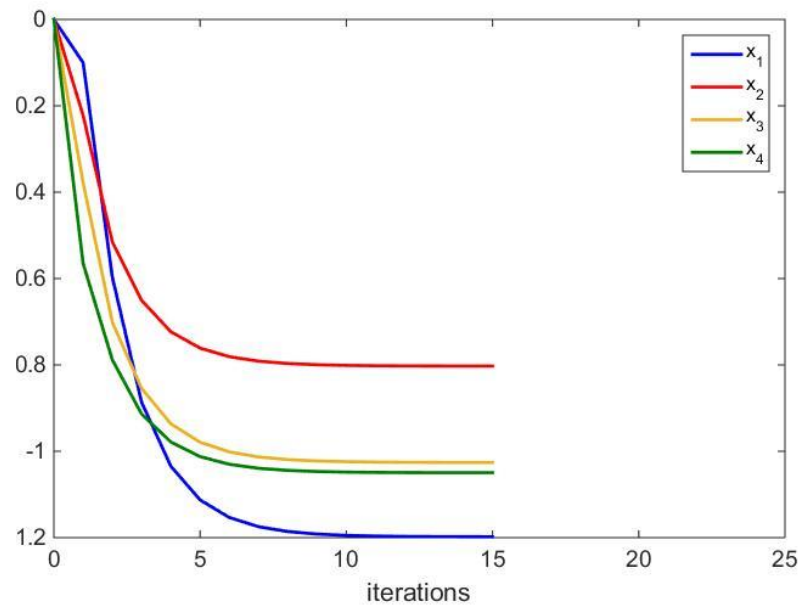
k	x_1	x_2	x_3	x_4
1	0.0000	0.0000	0.0000	0.0000
2	-0.1000	-0.2222	-0.3778	-0.5653
3	-0.5969	-0.5154	-0.7014	-0.7883
4	-0.8865	-0.6505	-0.8544	-0.9137
5	-1.0347	-0.7233	-0.9364	-0.9784
6	-1.1126	-0.7611	-0.9791	-1.0124
7	-1.1534	-0.7809	-1.0014	-1.0301
8	-1.1747	-0.7913	-1.0131	-1.0394
9	-1.1859	-0.7968	-1.0193	-1.0443
10	-1.1917	-0.7996	-1.0225	-1.0468
11	-1.1948	-0.8011	-1.0241	-1.0482
12	-1.1964	-0.8019	-1.0250	-1.0489
13	-1.1972	-0.8023	-1.0255	-1.0492
14	-1.1976	-0.8025	-1.0257	-1.0494
15	-1.1979	-0.8026	-1.0259	-1.0495
16	-1.1980	-0.8027	-1.0259	-1.0496



Jacobi



Gauss-Seidel



The Gauss Seidel method took fewer iterations to converge than did the Gauss Jacobi method.

Is it possible to predict if, and how fast, a method may converge?

Consider the matrices M_J and M_{GS} . If all of the eigenvalues of M_J or M_{GS} lie within the unit circle in the complex plane, then the iteration will converge for any initial guess.

$$M_J \triangleq -D^{-1}(L + U) \qquad M_{GS} \triangleq -(L + D)^{-1}U$$

$$M_J = \begin{bmatrix} 0 & -0.2000 & -0.3000 & -0.6000 \\ 0 & 0 & -0.1111 & -0.4444 \\ -0.1667 & -0.5000 & 0 & -0.1667 \\ -0.3750 & -0.1250 & 0 & 0 \end{bmatrix} \longrightarrow \text{eig}(M_J) = \begin{bmatrix} -0.7029 \\ 0.5679 \\ 0.0675 + j0.1636 \\ 0.0675 - j0.1636 \end{bmatrix} < \text{unit circle}$$

Convergence guaranteed!

$$M_{GS} = \begin{bmatrix} 0 & -0.2000 & -0.3000 & -0.6000 \\ 0 & 0 & -0.1111 & -0.4444 \\ 0 & -0.0333 & -0.1056 & -0.4889 \\ 0 & -0.0750 & -0.1264 & -0.2806 \end{bmatrix} \longrightarrow \text{eig}(M_{GS}) = \begin{bmatrix} 0 \\ -0.5234 \\ 0.0233 \\ 0.1140 \end{bmatrix} < \text{unit circle}$$

Convergence guaranteed!

Consider now the Jacobi method applied to the following system of equations:

$$\begin{bmatrix} 1 & 3 & 4 & 8 \\ 2 & 1 & 2 & 3 \\ 4 & 3 & 5 & 8 \\ 9 & 2 & 7 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad (\text{used previously in LU factorization example})$$

$$\begin{aligned} M_J &= -D^{-1}(L+U) \\ &= \begin{bmatrix} 0.00 & -3.00 & -4.00 & -8.00 \\ -2.00 & 0.00 & -2.00 & -3.00 \\ -0.80 & -0.60 & -0.00 & -1.60 \\ -2.25 & -0.50 & -1.75 & -0.00 \end{bmatrix} \end{aligned}$$

$$\text{eig}(M_J) = \begin{bmatrix} -6.6212 \\ 4.3574 \\ 1.2072 \\ 1.0566 \end{bmatrix} > \text{unit circle} \quad \text{Iterations will not converge}$$

k	x_1	x_2	x_3	x_4
1	0	0	0	0
2	1.0000	1.0000	0.2000	0.2500
3	-4.8000	-2.1500	-1.6000	-2.8500
4	36.6500	22.3500	9.8900	14.9250
5	-225.0100	-136.8550	-66.4100	-110.6950


Not converging!

Successive Overrelaxation

The Gauss-Seidel iteration $x^{k+1} = -(L + D)^{-1}((A - L - D)x^k - b)$

can be rewritten as: $x^{k+1} = x^k - D^{-1}(Lx^{k+1} + (D + U)x^k - b)$

Upper triangular part of A
(not the U from LU
factorization)

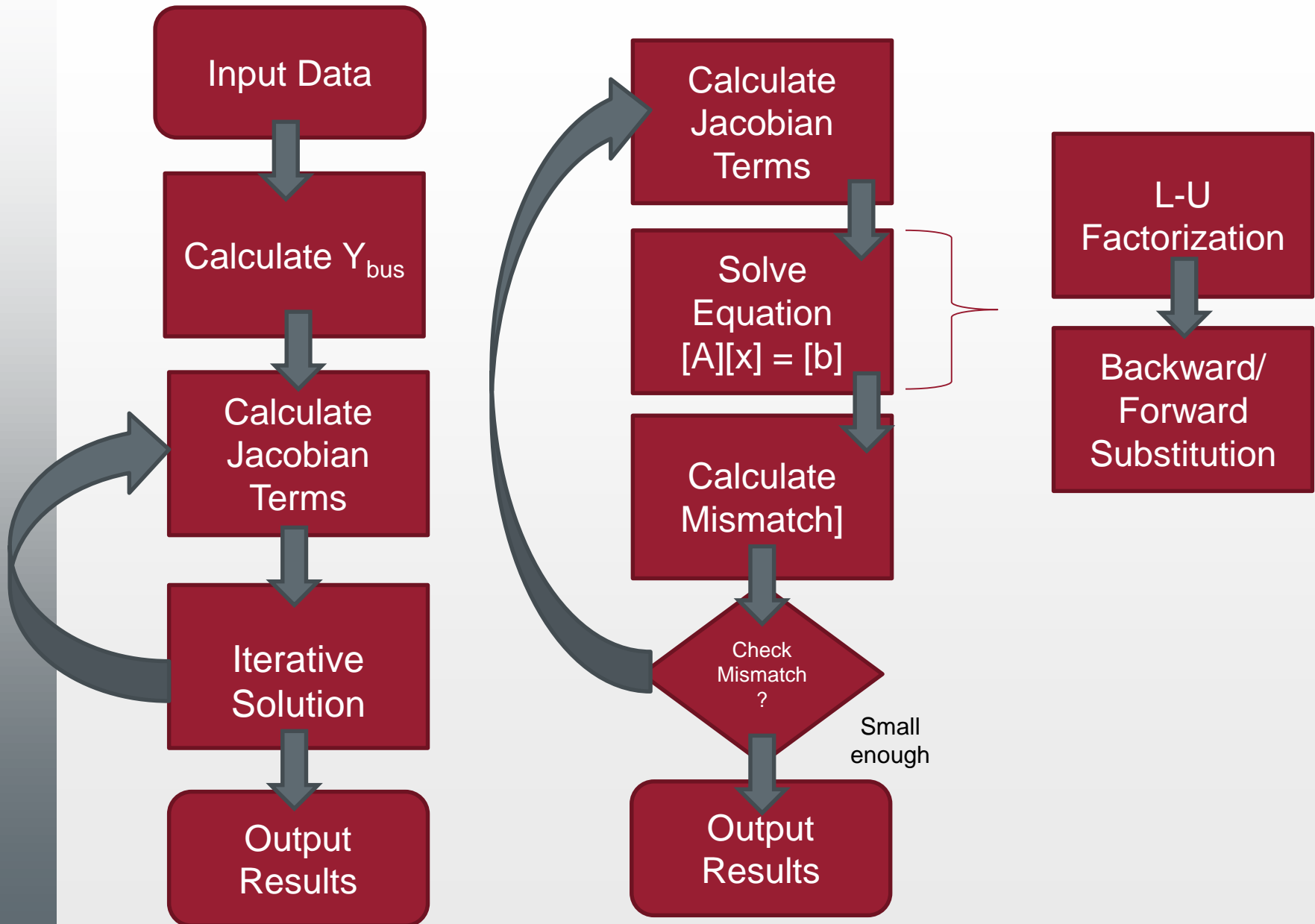


Add a weighting factor $0 < \omega < 2$: $x^{k+1} = x^k - \omega D^{-1}(Lx^{k+1} + (D + U)x^k - b)$

If $\omega = 1$, then the method reduces to the Gauss-Seidel.

For large systems, this method can potentially converge much more rapidly than either Jacobi or Gauss-Seidel, but it is difficult to find an optimal value of ω

Newton-Raphson and Parts



Programming Projects: (80%)

1. Power Flow Project (20%)

- A. Y_{bus} , Jacobian matrices – Newton Raphson
- B. LU Factorization, Backward-Forward substitution for NR
- C. Fast Decoupled
- D. Q-limits, transformer taps

- Using IEEE 14 bus system as given, and I will provide a slightly modified version after you confirm the 14-bus system.
- Projects build on each other, adding different features
- In addition to getting the right answers, for each program we'll set up a session for you to describe your code to me.
- Copying code from a colleague or from on-line resources is not permitted.

Reminder

- Here are my office hours via zoom and in-person as possible. I will let you know when I am in each location
- **Tuesdays 4:30-5:30 pm (after class)**
- **Wednesdays 4-5 pm**
- **Fridays 1:30-2:30 pm**
- Below is the zoom information for the semester! I will also set other times to go over your code individually.
- Join from PC, Mac, Linux, iOS, or Android: <https://wsu.zoom.us/j/8237216735> (Links to an external site.)
- Meeting ID: 823 721 6735

Announcements

- Review examples from Chapter 2
- Read Sections 3.1 and 3.2
- Working on MATLAB Program
 - Input the CDF file – Start with Data in Program
 - Start working on building the Ybus
 - Calculate Jacobian Matrix Values
 - Work on LU Decomposition