• Solve \( A \cdot x = b \) (matrix equation) numerically.

• Naive Gaussian elimination and Gaussian elimination with *partial pivoting*.

• Iterative methods of solving \( A \cdot x = b \).

• Practice 1: Gaussian-elimination.

• Practice 2: Check \( < r_1 r_2 > \neq 0 \) and \( < r^2 > = 0 \) from RNG.

---

\(^a\)Changing the order of the rows.

\(^b\)Jacobi and Gauss-Seidel method. How come it converges.
Consider a self-consistent charged particle system

- The kinetic model has two ways of solving: the Lagrangian particle (PIC) model and Eulerian Vlasov model.

- For example, electrostatic plasma dynamics can be given by
  \[
  \begin{align*}
  \dot{x} &= v \\
  \dot{v} &= \left(\frac{m_j}{q_j}\right)E.
  \end{align*}
  \]

- Alternatively by solving both Vlasov equation\(^a\)
  \[
  \frac{\partial f_j}{\partial t} + v \cdot \nabla f_j + a \cdot \frac{\partial f_j}{\partial v} = 0.
  \]
  with acceleration \(a = (m_j/q_j)E\)

- Either case, Poisson equation is solved
  \[
  \nabla \cdot E = \nabla \cdot (-\nabla \Phi) = q\rho(x)/\varepsilon_0
  \]
  where \(\rho(x, t) = \sum_j \int d^3v f_j(x, v, t)\).

\(^a\)Continuity equation in the 6D phase space.
• Now, the Poisson equation in 1D limit is given by an ordinary differential equation

\[ \frac{d^2 \Phi}{dx^2} = -q\rho(x)/\epsilon_0 \]  \hspace{1cm} (1)

With two boundary conditions, we can solve the Poisson equation.

• Equation (1) can be written by numerical differentiation

\[ \frac{\Phi_{i-1} - 2\Phi_i + \Phi_{i+1}}{h^2} = -\bar{\rho}_i, \]  \hspace{1cm} (2)

where the subscript \( i \) is the index of the grid points. BC, for example, \( \Phi(0) = 0 \) and \( \Phi(1) = 0 \) within the domain \( 0 \leq x \leq 1 \).
Let us exercise solving Eq.(1) with the matrix equation.

\[
\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & -2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & -2 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & -2 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & -2 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & -2 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -2 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
\end{pmatrix}
\begin{pmatrix}
\Phi_0 \\
\Phi_1 \\
\Phi_i \\
\Phi_{i-1} \\
\Phi_i \\
\Phi_{i+1} \\
\Phi_{n-1} \\
\Phi_n \\
\end{pmatrix}
= 
\begin{pmatrix}
0 \\
h^2 \rho_1 \\
\Phi_i \\
h^2 \rho_{i-1} \\
h^2 \rho_i \\
h^2 \rho_{i+1} \\
\Phi_{n-1} \\
h^2 \rho_{n-1} \\
0 \\
\end{pmatrix}
\]
One way to solve $A \cdot x = b$ is by the Gaussian elimination

- Let us consider

$$
\begin{pmatrix}
6 & -2 & 2 & 4 \\
12 & -8 & 6 & 10 \\
3 & -13 & 9 & 3 \\
-6 & 4 & 1 & -18 \\
\end{pmatrix}
\cdot
\begin{pmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4 \\
\end{pmatrix} =
\begin{pmatrix}
16 \\
26 \\
-19 \\
-34 \\
\end{pmatrix}
$$

- Subtract 1st row $\times 2$ from 2nd, $\times 1/2$ from 3rd, $\times -1$ from 4th. In this case, the first row is called the **pivot equation**. Then,

$$
\begin{pmatrix}
6 & -2 & 2 & 4 \\
0 & -4 & 2 & 2 \\
0 & -12 & 8 & 1 \\
0 & 2 & 3 & -14 \\
\end{pmatrix}
\cdot
\begin{pmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4 \\
\end{pmatrix} =
\begin{pmatrix}
16 \\
-6 \\
-27 \\
-18 \\
\end{pmatrix}
$$

---

Faithfully recapitulating Sec.7.1 of Cheney and Kincaid.
• Similarly then
\[
\begin{pmatrix}
6 & -2 & 2 & 4 \\
0 & -4 & 2 & 2 \\
0 & 0 & 2 & -5 \\
0 & 0 & 4 & -13
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4
\end{pmatrix}
= 
\begin{pmatrix}
16 \\
-6 \\
-9 \\
-21
\end{pmatrix}
\]

• Finally we obtain an upper triangular matrix.
\[
\begin{pmatrix}
6 & -2 & 2 & 4 \\
0 & -4 & 2 & 2 \\
0 & 0 & 2 & -5 \\
0 & 0 & 0 & -3
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4
\end{pmatrix}
= 
\begin{pmatrix}
16 \\
-6 \\
-9 \\
-3
\end{pmatrix}
\]

• Back substitution gives \( x_4 = 1, x_3 = (-9 + 5x_4)/2 \) and so on. We obtain \((x_1, x_2, x_3, x_4) = (3, 1, -2, 1)\).
• The procedure we took for forward elimination was, for $1 \leq j \leq n$

\[ a_{ij} \leftarrow a_{ij} - \frac{a_{i1}}{a_{11}} a_{1j} \]
\[ b_i \leftarrow b_i - \frac{a_{i1}}{a_{11}} b_1, \]

for $2 \leq j \leq n$

\[ a_{ij} \leftarrow a_{ij} - \frac{a_{i2}}{a_{22}} a_{2j} \]
\[ b_i \leftarrow b_i - \frac{a_{i2}}{a_{22}} b_2. \]

• For the $k \leq j \leq n$ pivot, it can be generalized by,

\[ a_{ij} \leftarrow a_{ij} - \frac{a_{ik}}{a_{kk}} a_{kj} \]
\[ b_i \leftarrow b_i - \frac{a_{ik}}{a_{kk}} b_k. \]

• The back substitution is given by “$x_n = b_n/a_{nn}$”,

“$x_{n-1} = (b_{n-1} - a_{n-1,n} x_n)/a_{n-1,n-1}$”, and so on.

\[ x_i = \frac{1}{a_{ii}} \left( b_i - \sum_{j=i+1}^{n} a_{ij} x_j \right). \]
An example naive Gaussian elimination fails

• Let us consider a simple example

\[
\begin{pmatrix}
0 & 1 \\
1 & 1
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2
\end{pmatrix}
=
\begin{pmatrix}
1 \\
2
\end{pmatrix}.
\]

Despite we know the solutions are \( x_1 = 1 \) and \( x_2 = 1 \), previous method fails.

• Now consider \( \epsilon \ll 1 \)

\[
\begin{pmatrix}
\epsilon & 1 \\
1 & 1
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2
\end{pmatrix}
=
\begin{pmatrix}
1 \\
2
\end{pmatrix},
\]

\[
\begin{pmatrix}
1 & 1/\epsilon \\
0 & 1-1/\epsilon
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2
\end{pmatrix}
=
\begin{pmatrix}
1/\epsilon \\
2-1/\epsilon
\end{pmatrix}.
\]

Back substitution will give

\[
x_2 = \frac{2 - 1/\epsilon}{1-1/\epsilon}, \quad x_1 = \frac{1}{\epsilon} - \frac{1}{\epsilon}x_2.
\]

• If the order is flipped, it works ok. The key is the relative magnitude.

\[
\begin{pmatrix}
1 & 1 \\
0 & 1
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2
\end{pmatrix}
=
\begin{pmatrix}
2 \\
1
\end{pmatrix}.
\]
**GE with partial pivoting is demonstrated**

- Find the largest component at each row: \( L = (1, 2, 3, 4) \) for book keeping.

\[
\begin{pmatrix}
3 & -13 & 9 & 3 \\
-6 & 4 & 1 & -18 \\
6 & -2 & 2 & 4 \\
12 & -8 & 6 & 10
\end{pmatrix}
\cdot
\begin{pmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4
\end{pmatrix}
=
\begin{pmatrix}
-19 \\
-34 \\
16 \\
26
\end{pmatrix}
\]

A scale vector \( s = (13, 18, 6, 12) \) is introduced. Find pivot equation

\[
\frac{a_{i,1}}{s_i} = \left( \frac{3}{13}, \frac{6}{18}, \frac{6}{6}, \frac{12}{12} \right)
\]

And now \( L = (3, 2, 1, 4) \) : exchanged 3 and 1

- Subtract 3rd row from 1st, 2nd, and 4th, then:

\[
\begin{pmatrix}
0 & -12 & 8 & 1 \\
0 & 2 & 3 & -14 \\
6 & -2 & 2 & 4 \\
0 & -4 & 2 & 2
\end{pmatrix}
\cdot
\begin{pmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4
\end{pmatrix}
=
\begin{pmatrix}
-27 \\
-18 \\
16 \\
-6
\end{pmatrix}
\]

\[
\frac{a_{i,2}}{s_i} = \left( \frac{2}{18}, \frac{12}{13}, \frac{4}{12} \right)
\]

and for next, \( L = (3, 1, 2, 4) \) exchanged 1 and 2
• Similarly, while \( \mathbf{L} = (3, 1, 2, 4) \)

\[
\begin{pmatrix}
0 & -12 & 8 & 1 \\
0 & 0 & 13/3 & -83/6 \\
6 & -2 & 2 & 4 \\
0 & 0 & -2/3 & 5/3
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4
\end{pmatrix}
=
\begin{pmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4
\end{pmatrix}
= \begin{pmatrix}
-27 \\
-45/2 \\
16 \\
3
\end{pmatrix}
\]

• Finally we obtain an upper triangular equivalent

\[
\begin{pmatrix}
0 & -12 & 8 & 1 \\
0 & 0 & 13/3 & -83/6 \\
6 & -2 & 2 & 4 \\
0 & 0 & 0 & -6/13
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4
\end{pmatrix}
=
\begin{pmatrix}
-27 \\
-45/2 \\
16 \\
-6/13
\end{pmatrix}
\]

• Back substitution 4th, 2nd, 1st, and 3rd gives \( x_4 = 1, x_3 = -2, x_2 = 1, \) and \( x_1 = 3. \)
The pivoting scheme is summarized

- At the beginning, a scale vector \( s = (s_1, s_2, ..., s_n) \) with

\[
s_i = \max_{1 \leq j \leq n} |a_{ij}| \text{ for } 1 \leq i \leq n
\]

- At the same time, set an index vector \( \mathbf{l} = (l_1, l_2, ..., l_n) = (1, 2, ..., n) \) at the beginning. Select \( j \) to be the index for the largest value of

\[
\frac{a_{l_i,j}}{s_{l_i}} \text{ for } 1 \leq i \leq n
\]

and change \( l_j \) with \( l_1 \). Use multipliers \( a_{l_i,1}/a_{l_1,1} \) to eliminate first column in other rows. Note that only entries in \( \mathbf{l} \) are interchanged and not the equations.

- In the second step, check the ratios

\[
\frac{a_{l_i,2}}{s_{l_i}} \text{ for } 2 \leq i \leq n
\]

interchange \( l_j \) and \( l_2 \), and use multipliers \( a_{l_i,2}/a_{l_2,2} \).
• Similarly at the k-th step, check the ratios

\[
\frac{a_{l_i,k}}{s_{l_i}} \text{ for } k \leq i \leq n
\]

interchange \(l_j\) and \(l_k\), and use multipliers \(a_{l_i,k}/a_{l_k,k}\).
Iterative method is another way of solving \[ A \cdot x = b \]

- For

\[
A = \begin{pmatrix}
  a_{11} & \ldots & \ldots & \ldots & a_{1n} \\
  \ldots & \ldots & \ldots & \ldots & \ldots \\
  \ldots & a_{i,i-1} & a_{i,i} & a_{i,i+1} & \ldots \\
  \ldots & \ldots & \ldots & \ldots & \ldots \\
  a_{n1} & \ldots & \ldots & \ldots & a_{nn}
\end{pmatrix}
\]

\[
x = \begin{pmatrix}
  x_1 \\
  \ldots \\
  x_i \\
  \ldots \\
  x_n
\end{pmatrix}
\]

\[
b = \begin{pmatrix}
  b_1 \\
  \ldots \\
  b_i \\
  \ldots \\
  b_n
\end{pmatrix}
\]

The \(i\)-th row reads

\[a_{i1}x_1 + a_{i2}x_1 + \ldots + a_{ii}x_i + \ldots + a_{in}x_n = b_i.\]

If we single out the diagonal component, we have

\[a_{ii}x_i = b_i - \sum_{j=1}^{i-1} a_{ij}x_j - \sum_{j=i+1}^{n} a_{ij}x_j.\]
Now the concept of iteration comes into play.

\[ x_i^{(\text{new})} = (b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(\text{old})} - \sum_{j=i+1}^{n} a_{ij}x_j^{(\text{old})})/a_{ii} \]

This is called the Jacobi method.

Gauss-Seidel method:

\[ x_i^{(\text{new})} = (b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(\text{new})} - \sum_{j=i+1}^{n} a_{ij}x_j^{(\text{old})})/a_{ii} \]

or

\[ x_i^{(k+1)} = (b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k)})/a_{ii} \]

where \( k \) represents the iteration step.

We start from an initial guess. But how come such an ansatz (guess) converges?
The procedure can be designed to converge

- Iterative method produces a sequence of approximate solutions $x^{(0)}, x^{(1)}, x^{(2)}, \ldots$. The numerical procedure is designed so that they converge to the actual solution. The process can stop when sufficient precision has been reached.

- For $A \cdot x = b$, taking a nonsingular matrix $Q$, we can write

  $$Q \cdot x = (Q - A) \cdot x + b$$

  What is done previously is formally given by$^a$

  $$Q \cdot x^{(k+1)} = (Q - A) \cdot x^{(k)} + b$$

  Then at the limit of $k \to \infty$,

  $$Q \cdot x^{(\ast)} = (Q - A) \cdot x^{(\ast)} + b$$

  which leads to

  $$A \cdot x^\ast = b.$$  

$^a$With $Q$ being the diagonal components in the previous examples.
Matrix $Q$ should be chosen so that the sequence of $x$ converges independent of the initial choice.

$$x^{(k+1)} = Q^{-1} (Q - A) \cdot x^{(k)} + Q^{-1} b$$

or

$$x^{(k+1)} - x = (I - Q^{-1} A) x^{(k)} + Q^{-1} b - x$$

$$= (I - Q^{-1} A) x^{(k)} - (I - Q^{-1} A) x$$

$$= (I - Q^{-1} A) (x^{(k)} - x).$$

With the error vector $e^{(k)} \equiv x^{(k)} - x$, we can write

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

Roughly speaking $e^{(k+1)}$ will be “smaller” than $e^{(k)}$ if $(I - Q^{-1} A)$ is “small”.
• An example of Jacobi method.

Let us exercise \( x^{(k+1)} = (I - Q^{-1}A) \cdot x^{(k)} + Q^{-1}b \). Take,

\[
A = \begin{pmatrix} 2 & -1 & 0 \\ -1 & 3 & -1 \\ 0 & -1 & 2 \end{pmatrix}, \quad b = \begin{pmatrix} 1 \\ 8 \\ -5 \end{pmatrix}
\]

Choose \( Q \) as the diagonal components of \( A \):

\[
Q = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 2 \end{pmatrix}, \quad Q^{-1} = \begin{pmatrix} 1/2 & 0 & 0 \\ 0 & 1/3 & 0 \\ 0 & 0 & 1/2 \end{pmatrix}, \quad Q^{-1}A = \begin{pmatrix} 1 & -1/2 & 0 \\ -1/3 & 1 & -1/3 \\ 0 & -1/2 & 1 \end{pmatrix}
\]

\[
I - Q^{-1}A = \begin{pmatrix} 0 & 1/2 & 0 \\ 1/3 & 0 & 1/3 \\ 0 & 1/2 & 0 \end{pmatrix}, \quad Q^{-1}b = \begin{pmatrix} 1/2 \\ 8/3 \\ -5/2 \end{pmatrix}
\]

Then

\[
x^{(0)} = (0.0, 0.0, 0.0) \\
x^{(1)} = (0.5, 2.67, -2.5) \\
\quad \cdots \\
x^{(21)} = (2.0, 3.0, -1.0)
\]
Convergence theorems for the iterative methods

• For the sequence $x^k$ obtained by the iteration to converge, no matter what starting point is selected, it is necessary and sufficient that all eigenvalues of $I - Q^{-1}A$ lie in the open unit disk $|z| < 1$ in the complex plane.

• All the eigenvalues of the matrix $I - Q^{-1}A$ needs to be smaller than unity:
  \[
  \rho(I - Q^{-1}A) < 1.
  \]
  Here, $\rho$ is the spectral radius. For the example of Jacobi method above, we have
  \[
  \det | (I - Q^{-1}A) - \lambda I | = -\lambda^3 + \lambda/6 + \lambda/6 = 0
  \]
  The eigen values are $\lambda = 0, \pm \sqrt{1/3} \simeq \pm 0.5774$ thus the iteration succeeds.

• If $A$ is diagonally dominant, then the Jacobi and Gauss-Seidel methods converge for any $x^{(0)}$:
  \[
  \sum |a_{i,j}|/a_{ii} < 1.
  \]
Vector norms can be used for the checking the convergence

- $L_1$ vector norm:
  $$||x||_1 = \sum_{i=1}^{n} |x_i|$$

- $L_2$ vector norm:
  $$||x||_2 = \left( \sum_{i=1}^{n} x_i^2 \right)^{1/2}$$

- $L_\infty$ vector norm:
  $$||x||_\infty = \max_{1 \leq i \leq n} |x_i|$$
1D Poisson equation can be solved by the iteration methods

• Numerical discretization of 1D Poisson reads

\[ \Phi_{i-1} - 2\Phi_i + \Phi_{i+1} = -h^2 \bar{\rho}_i, \]

where the subscript \( i \) is the index of the grid points. \(^a\)

• As before, Jacobi method reads

\[ \Phi_i^{(\text{new})} = (1/2) \left( \Phi_{i-1}^{(\text{old})} + \Phi_{i+1}^{(\text{old})} + h^2 \bar{\rho}_i \right), \]

and Gauss-Seidel method reads

\[ \Phi_i^{(\text{new})} = (1/2) \left( \Phi_{i-1}^{(\text{new})} + \Phi_{i+1}^{(\text{old})} + h^2 \bar{\rho}_i \right), \]

You can directly solve Eqs.(1) and (2) or use matrix. Note that Eqs.(1) and (2) are equivalent to solving the tri-diagonal band matrix equation.

\(^a\)Boundary conditions, for example, take \( \Phi(0) = 0 \) and \( \Phi(1) = 0 \) within the domain \( 0 \leq x \leq 1. \)
As a reminder, the matrix equation for 1D Poisson solve is given by

\[
\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & -2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & -2 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & -2 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & -2 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{pmatrix}
\cdot
\begin{pmatrix}
\Phi_0 \\
\Phi_1 \\
\Phi_{i-1} \\
\Phi_i \\
\Phi_{i+1} \\
\Phi_{n-1} \\
\Phi_n \\
\end{pmatrix}
= 
\begin{pmatrix}
0 \\
h^2 \rho_1 \\
\vdots \\
h^2 \rho_{i-1} \\
h^2 \rho_i \\
\vdots \\
h^2 \rho_{i+1} \\
\vdots \\
h^2 \rho_{n-1} \\
0 \\
\end{pmatrix}
\]

The components multiplied by zero’s are practically of no use (note the computational memory for the storage).
Summary of today’s discussions

- Solved $A \cdot x = b$ (matrix equation) numerically by Naive Gaussian elimination and Gaussian elimination with partial pivoting.

- Iterative methods of solving $A \cdot x = b$.

- Discussed an application to 1D Poisson problem.

- Thursday and Next week: continue $A \cdot x = b$ solve, e.g. Tridiagonal and banded matrix, LU decomposition, and SOR.