Like chapter 5 we solve $Ax = b$ when $A$ is square and nonsingular.

- When $A$ is large, Gaussian elimination is expensive and time consuming.
- Matrix multiplication $Av$ is cheap and fast.
7.1 The need for iterative methods

- Gaussian Elimination introduces fill-in, zeros in L and U where A had a zero. Much fill in is expensive.
- Sometimes we need not solve system exactly.
- Sometimes we have a good approximate guess.
- Sometimes the matrix A is not given but matrix products Av are given.
7.1 The need for iterative methods

Do not let anything above fool you into believing that Gaussian elimination, LU factorization, and Cholesky factorization are not important. They are still the most commonly used methods to solve systems $Ax = b$. 
7.1 The need for iterative methods

- Poisson equation: \(- \left( \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) = g(x, y)\) on unit square.

- Dirichlet boundary conditions:
  \(u(x, 0) = u(x, 1) = u(0, y) = u(1, y) = 0\).

- Discretizing: \(4u_{i,j} - u_{i+1,j} - u_{i-1,j} - u_{i,j+1} - u_{i,j-1} = b_{i,j}\).
7.1 The need for iterative methods

Express discretized Poisson equation: \( Ax = b \) when \( n = 3 \).

\[
A = \begin{bmatrix}
J & -I & 0 \\
-I & J & -I \\
0 & -I & J \\
\end{bmatrix}.
\]

\[
J = \begin{bmatrix}
4 & -1 & 0 \\
-1 & 4 & -1 \\
0 & -1 & 4 \\
\end{bmatrix}.
\]
Several methods solve problems like the above $Ax = b$ by transforming it into a fixed point problem.

- Rewrite $Ax = b$ as $0 = f(x) = Ax - b$.
- Transform $0 = f(x)$ to $x = g(x)$.
- Find the FPI sequence $\vec{x}_{k+1} = g(\vec{x}_k)$. 
7.2 Stationary iteration and relaxation methods.

- Split $A = M - N$.
- Rewrite $Ax = b$ as $Mx = Nx + b$.
- Rewrite as FPI:
  \[ x_{k+1} = M^{-1}Nx_k + M^{-1}b = x_k + M^{-1}(b - Ax_k). \]
- $g(x) = x + M^{-1}(b - Ax)$. 
7.2 Stationary iteration and relaxation methods.

- How to split $A = M - N$? There are many ways. We will study three.
- Jacobi method: $M = D$, the diagonal of $A$.
- Gauss-Seidel method: $M = E$, the lower triangular part of $A$.
- Successive over-relaxation (SOR) is a mix of Jacobi and Gauss-Seidel.
7.2 Stationary iteration and relaxation methods.

\[ r_k = b - A x_k \] is the residual after the k-th FPI iteration.

- The Jacobi method: \[ x_{k+1} = x_k + D^{-1} r_k. \]
- and then the Gauss-Seidel: \[ x_{k+1} = x_k + E^{-1} r_k. \]
- SOR: \[ x_{k+1} = x_k + \omega ((1 - \omega) D + \omega E)^{-1} r_k. \]
Example $A = \begin{bmatrix} 3 & -1 \\ -1 & 2 \end{bmatrix}$ and $b = \begin{bmatrix} 5 \\ 4 \end{bmatrix}$. Compute $\vec{x}_1$ and $\vec{x}_2$ when $x_0 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ for the Jacobi method and then for the Gauss-Seidel method.
Example $A = \begin{bmatrix} 7 & 3 & 1 \\ -3 & 10 & 2 \\ 1 & 7 & -15 \end{bmatrix}$ and $b = \begin{bmatrix} 3 \\ 4 \\ 2 \end{bmatrix}$. Compute $x_1$ when $x_0$ is the zero vector in three dimensions for the Jacobi method, and then for the Gauss-Seidel method.
7.3 Convergence of Stationary Methods.

Consider the iterative method

\[ x_{k+1} = x_k + M^{-1}r_k \]

when \( r_k \) is the kth residual. Then the method converges if and only if the spectral radius \( \rho(T) \) of the corresponding iteration matrix \( T = I - M^{-1}A \) satisfies

\[ \rho(T) < 1. \]

The small \( \rho(T) \) the faster the convergence.
We will not cover the material from sections 7.5 "Conjugate Gradient" and 7.6 Krylov subspaces in this course.
EXERCISE

Given the system of equations

\begin{align*}
3x_1 + x_2 - x_3 &= 3 \\
x_1 - 4x_2 + 2x_3 &= -1 \\
-2x_1 - x_2 + 5x_3 &= 2
\end{align*}

- Use Jacobi to compute $\vec{x}_1$ by hand when $\vec{x}_0 = (0, 0, 0)$.
- Use Gauss-Seidel to compute $\vec{x}_1$ by hand when $\vec{x}_0 = (0, 0, 0)$. 
EXERCISE

Given the system of equations

\[3x_1 + x_2 - x_3 = 3\]
\[x_1 - 4x_2 + 2x_3 = -1\]
\[-2x_1 - x_2 + 5x_3 = 2\]

- Use Jacobi to compute \(\vec{x}_7\) using numpy when \(\vec{x}_0 = (0, 0, 0)\). Round your answer to five decimal places. Guess if this FPI converges. To what? Is it the solution to \(Ax = b\)?
- Use Gauss-Seidel to compute \(\vec{x}_7\) using numpy when \(\vec{x}_0 = (0, 0, 0)\). Round your answer to five decimal places.
- Use SOR with \(\omega = 1.2\) to compute \(\vec{x}_4\) using numpy when \(\vec{x}_0 = (0, 0, 0)\). Round your answer to five decimal places.
EXERCISE

Given the system of equations

\[
\begin{align*}
3x_1 + x_2 - x_3 &= 3 \\
x_1 - 4x_2 + 2x_3 &= -1 \\
-2x_1 - x_2 + 5x_3 &= 2
\end{align*}
\]

- Here is yet another FPI \( x = g(x) = (I - A)x + b \).
- Verify that a fixed point of this FPI is a solution to \( Ax = b \).
- Find \( \vec{x}_1 \) and \( \text{vec}x_7 \) when \( \vec{x}_0 = (0, 0, 0) \). (write your own python code to find \( \vec{x}_7 \)).
- What is happening with this FPI sequence?
Given the system of equations

\begin{align*}
3x_1 + x_2 - x_3 &= 3 \\
x_1 - 4x_2 + 2x_3 &= -1 \\
-2x_1 - x_2 + 5x_3 &= 2
\end{align*}

- **Jacobi:** \( \vec{x}_0 = (0, 0, 0), \vec{x}_1 = (1, .25, .4), \ldots, \vec{x}_7 = (1.00038, 1.00122, .99985) \). Converges to \((1, 1, 1)\) which is indeed the solution.

- **Gauss-Seidel:** \( \vec{x}_0 = (0, 0, 0), \vec{x}_1 = (1, 0.5, 0.9), \ldots, \vec{x}_7 = (1.00015, .99997, 1.00005) \).

- **SOR:** \( \vec{x}_0 = (0, 0, 0), \vec{x}_4 = (1.01776, 1.01520, 1.01154) \).
SOLUTIONS

Given the system of equations

\[
\begin{align*}
3x_1 + x_2 - x_3 &= 3 \\
x_1 - 4x_2 + 2x_3 &= -1 \\
-2x_1 - x_2 + 5x_3 &= 2
\end{align*}
\]

▶ Here is yet another FPI \( x = g(x) = (I - A)x + b \).
▶ Verify that a fixed point of this FPI is a solution to \( Ax = b \). Solve for \( Ax \) in \( x = x - Ax + b \). You’ll get \( Ax = b \).
▶ Find \( \vec{x}_1 = (3, -1, 2) \) and \( \text{vec}x_7 = (6069, -40300, -5665) \) when \( \vec{x}_0 = (0, 0, 0) \).
▶ What is happening with this FPI sequence? ANSWER: diverging.