Lecture 9: Iterative methods

CSC 338: Numerical Methods

Ray Wu

University of Toronto

March 22, 2023

Outline

- Motivation and a representative problem
- Stationary methods
- Convergence of stationary methods
- Gradient-based methods (conjugate gradient)

The need for iterative methods

What is wrong with a direct method (LU with partial pivoting)?

- LU may introduce fill-in for sparse matrices, and destroy the sparsity.
- LU can't solve the linear system to an arbitrary degree of accuracy more efficiently.
- LU can't make use of an educated guess of the solution. This arises frequently in time-dependent problems and is known as a warm start.

The representative problem

- We have looked at one-dimensional problems. However, for applications in our real world, we often require two or three dimensional modelling.
- Example two-dimensional PDE (Poisson Equation):

$$-\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) = g(x, y) \tag{1}$$

  for some given function g.
- Additionally, let us assume that the domain is a square ($x, y \in [0, 1]$) and that the boundary values of u on the square is 0 ($u(x, 0) = u(x, 1) = u(0, y) = u(1, y) = 0$).

Discretization

- Poission Equation:

$$-\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) = g(x, y) \tag{2}$$

- Discretization with finite differences:

$$(2u_{i,j} - u_{i-1,j} - u_{i+1,j}) + (2u_{i,j} - u_{i,j-1} - u_{i,j+1}) = h^2 g_{i,j} \tag{3}$$

- These are linear equations, and combining them all gives rise to a linear system of equations $Au = b$.
- Notation:
  - $u_{i,j}$ is shorthand for $u(ih, jh)$.
  - Assumed equal number of gridpoints in both dimensions.

What does the matrix look like?

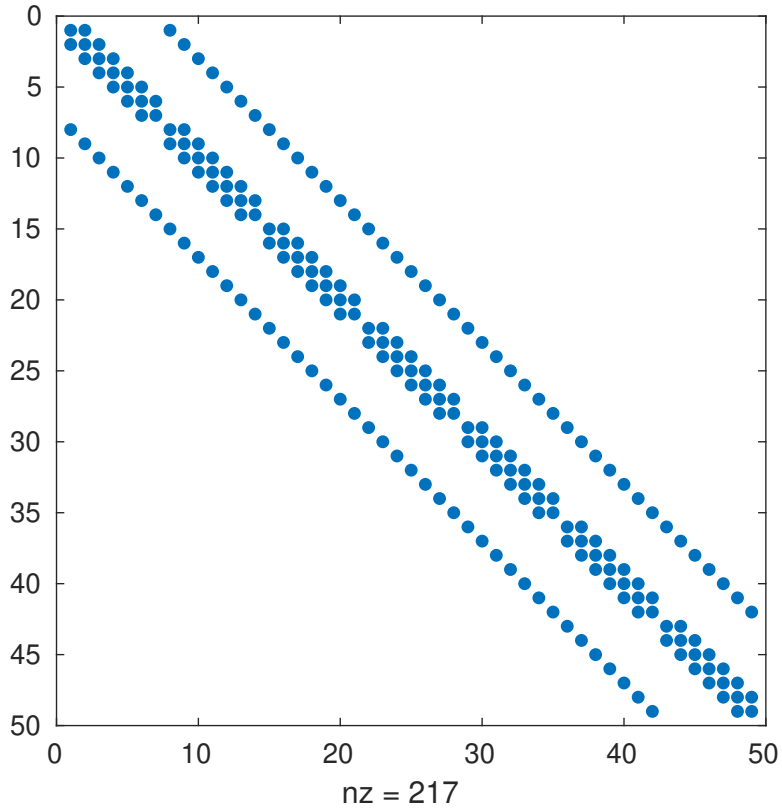

Figure 1: The matrix arising from the discretization of the two-dimensional problem. $\mathcal{O}(n^2)$ entries are stored ($n = 8$).

Constructing d-dimensional discretizations

- The easiest way to construct these matrices is using the **kronecker product**
- Denoted as `kron(A, B)` in Matlab, or $\otimes$.
- Defined by the following:

$$A \otimes B = a_{i,j}B. \tag{4}$$

- To construct the two dimensional discretization, let T_2 be the size n matrix that computes the 2nd derivative.
- Then,

$$A = T_2 \otimes I_n + I_n \otimes T_2 \tag{5}$$

- Three-dimensional case is similar.
- Must be careful about which dimension is which for nonsymmetric problems!

Direct Methods

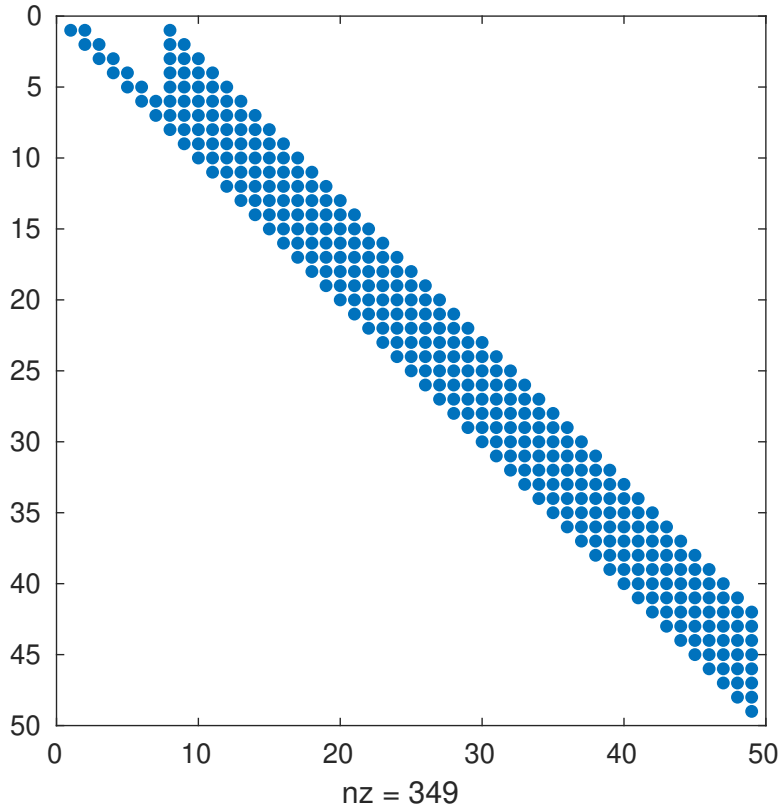

Figure 2: The Cholesky factor of the matrix arising from the discretization of the two-dimensional problem. $\mathcal{O}(n^3)$ entries are stored ($n = 8$).

Digression: algorithmic efficiency

- What is the computational complexity of evaluating a polynomial

$$p_n(x) = \sum c_i x^i \tag{6}$$

- The simple way that you were taught in math class:
  - For each $i = 0, 1, 2, \ldots, n$, calculate x^i
  - Calculate $c_i x^i$, then sum together.
  - Computational complexity: $\mathcal{O}(n^2)$.
- Consider problem in terms of input size/output size.
- Input size is $n + 2$, output size is 1.
- Computational complexity is lower-bounded by input/output size.

Algorithmic efficiency continued

- The computational complexity is lower-bounded by input size $\mathcal{O}(n)$.
- Naive algorithm is $\mathcal{O}(n^2)$.
- We can consider avoiding computing unnecessary powers of x.
- Consider instead Horner's method

$$p_n(x) = \sum c_i x^i = c_0 + x(c_1 + x(c_2 + \cdots + x(c_{n-1} + xc_n))) \quad (7)$$

- Computational complexity: n multiplications and n additions, $\mathcal{O}(n)$.
- Optimal because input size is $\mathcal{O}(n)$, and we must read the input.

Another example of algorithmic efficiency

- Matrix Multiplication of two n by n matrices: Naive method from the definition of matrix multiplication is $\mathcal{O}(n^3)$.
  - i.e. compute the matrix product as the dot product of the corresponding row and columns, each entry is $\mathcal{O}(n)$ computational complexity to compute and we have n^2 entries.
- Input size: $2n^2$.
- This means that the straightforward matrix multiplication method is not optimal.
- Strassen algorithm (1969): $\mathcal{O}(n^{\log_2(7)}) \approx \mathcal{O}(n^{2.81})$.
  `https://en.wikipedia.org/wiki/Strassen_algorithm`
- Open question in theoretical computer science
- Duan, Wu, Zhou (2022): $\mathcal{O}(n^{2.37188})$.
  `https://arxiv.org/pdf/2210.10173.pdf`

Going back to our example

- One dimension: n entries, computational complexity is $\mathcal{O}(n)$, optimal.
- Two dimensions: n^2 entries, computational complexity is $\mathcal{O}(n^4)$.
- Iterative methods can be used to reduce this computational complexity.

Splitting Methods

- Consider a splitting of a matrix $A = M - N$.
- Then, $Ax = b$ is equivalent to $Mx = Nx + b$.
- Fixed-point iteration results in

$$x_{k+1} = M^{-1}(Nx_k + b) = x_k + M^{-1}(b - Ax_k) = x_k + M^{-1}r_k \quad (8)$$

- Choice of M leads to different iterative schemes.
- Of course, need to choose M that is easily invertible.

Two simple direct methods

- Jacobi method: choose $M = D$, leading to

$$x_{k+1} = x_k + D^{-1} r_k \tag{9}$$

- D is the diagonal matrix with the same diagonal entries as A.
- Gauss-Seidel: choose $M = E$, leading to

$$x_{k+1} = x_k + E^{-1} r_k \tag{10}$$

- E is the lower-triangular part of A.
- Note that D and E are both easily invertible.
- The more advanced methods SOR (successive over-relaxation) build upon these methods.

Convergence of stationary methods

- It is useful to consider the general form:

$$x_{k+1} = x_k + M^{-1} r_k \tag{11}$$

- At each iteration k the residual is r_k, and the error is $A^{-1} r_k$.
- Write

$$x_{k+1} = M^{-1} b + (I - M^{-1} A) x_k$$
$$x = M^{-1} b + (I - M^{-1} A) x$$

- Take the difference to get

$$e_{k+1} = (I - M^{-1} A) e_k = T e_k \tag{12}$$

- T is called the iteration matrix and the method converges if $\rho(T) < 1$.
- ρ denotes the spectral radius
- The rate of convergence is $-\log_{10}(\rho(T))$.

Gradient based methods

- Assume that A is SPD.
  - The matrix from Poission's equation satisfies this assumption.
- Solving $Ax = b$ is equivalent to minimizing

$$\phi(x) = \frac{1}{2} x^T A x - b^T x \tag{13}$$

- Update iterate with $x_{k+1} = x_k + \alpha_k p_k$.
- Gradient descent: Choose $p_k = r_k$.

Exact Line-search

- How to choose α_k?
- Greedy approach: minimize the function

$$\phi(x_k + \alpha_k r_k) = \frac{1}{2}(x_k + \alpha_k r_k)^T A(x_k + \alpha_k r_k) - b^T(x_k + \alpha_k r_k) \quad (14)$$

- Find the minimizer by differentiating with respect to α and setting to zero. Choose

$$\alpha_k = \frac{r_k^T r_k}{r_k^T A r_k} \quad (15)$$

Issues with Gradient Descent

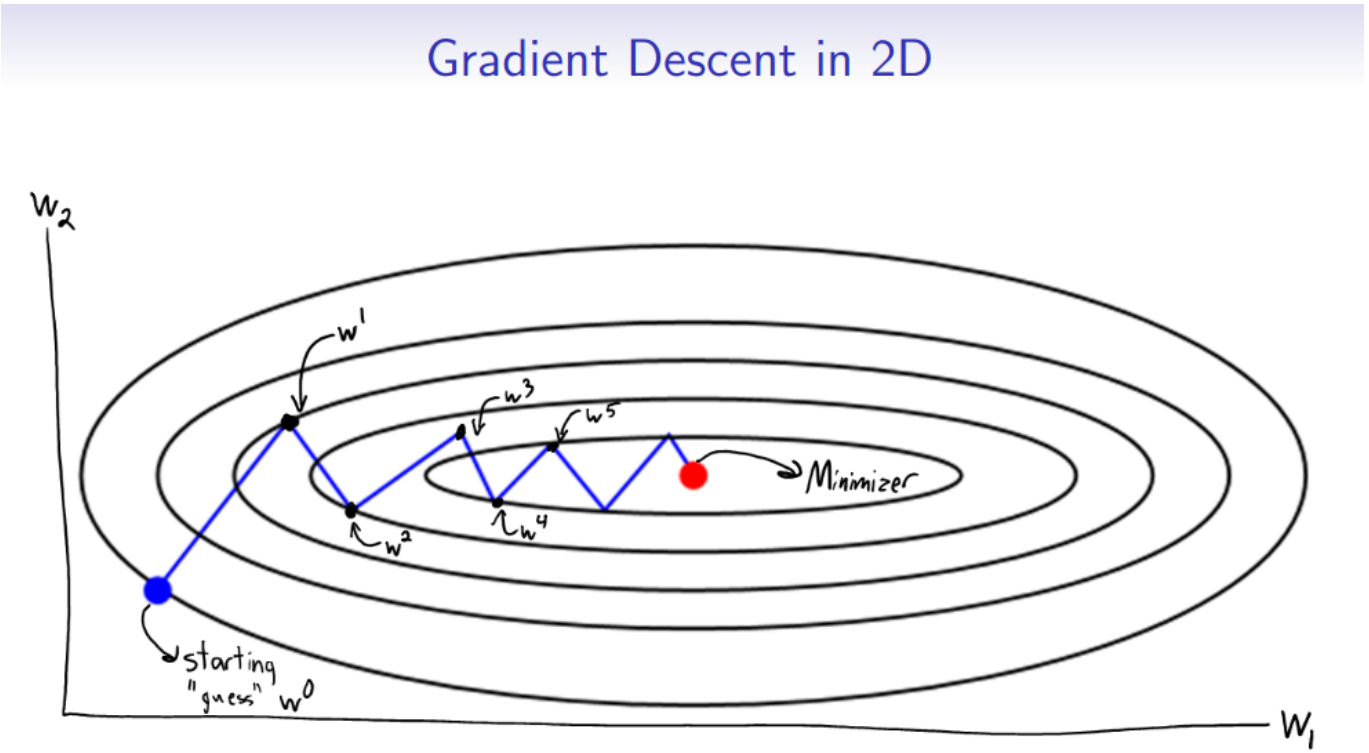

Figure 3: Convergence of Gradient Descent. Note that the search directions are orthogonal to each other. Figure from
`https://www.cs.ubc.ca/~schmidtm/Courses/540-W19/L4.pdf`

Conjugate directions

- What if, once we search a direction, we are done with it forever?
- Conjugate directions: Two vectors p_j and p_k are A-conjugate if

$$p_j^T A p_k = 0 \tag{16}$$

- Energy norm:

$$\|x\|_A = \sqrt{x^T A x} \tag{17}$$

  assuming that A is positive definite.
- Choosing conjugate directions allows us to search in one direction and be done with it. Algorithm:
  - Find n conjugate directions
  - Solve in each direction.

Conjugate gradient

- First search direction: nothing changes.
- Every subsequent direction: enforce A-conjugate condition. Set

$$p_{k+1} = r_{k+1} + \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k} p_k \tag{18}$$

- The stepsize α_k is chosen to be the same as in gradient descent:

$$\alpha_k = \frac{r_k^T r_k}{p_k^T A p_k} \tag{19}$$

- We can show that the residuals are all orthogonal to each other.
- We can show that the search directions are all A-orthogonal to each other.

More on Conjugate gradient

- Conjugate gradient is both an iterative method and a direct method, since it is guaranteed to terminate in n iterations.
  - Additionally, if the matrix has $m < n$ distinct eigenvalues, then the number of iterations is at most m.
- However, generally we consider Conjugate Gradient to be an iterative method.
- Only $\mathcal{O}(\sqrt{\kappa(A)})$ iterations are required to reduce the error $\|e_k\|_A$ by a fixed amount, with an error bound given by

$$\|e_k\|_A \leq 2\left(\frac{\sqrt{\kappa(A)}-1}{\sqrt{\kappa(A)}+1}\right)^k \|e_0\|_A \tag{20}$$

- This is in contrast to gradient descent, where $\mathcal{O}(\kappa(A))$ iterations are required to reduce the norm by a fixed amount.

Computational complexity of conjugate gradient

- Recall that for a one-dimensional problem, the condition number of A increases like $\mathcal{O}(n^2)$.
- The same thing can be shown for a two-dimensional problem. Hence, the number of iterations is $\mathcal{O}(n)$.
- How much does each iteration cost? Computational complexity of matrix-vector multiplication is $\mathcal{O}(n^2)$.
- Improved efficiency.

Final comments on iterative methods

- Conjugate gradient method is for symmetric positive definite matrices. For general matrices
  - biconjugate gradient method (bicg)
  - generalized mean residual (gmres)
- Preconditioning: computing a matrix P such that $P^{-1}A$ has a reduced condition number.
- Implementation: in these iterative methods (gradient descent, conjugate gradient, biconjugate gradient, generalized mean residual), the matrix itself is not necessary to construct.
- We only use the matrix as an operator to a vector.
- Hence, in certain applications we do not construct the matrix, but rather write a function that takes input x and outputs Ax.
- Can lead to improved efficiency.