TMA4205 - AUTUMN 2012

DISCRETIZATION OF THE POISSON EQUATION

We shall study the linear system that results from the discretization of the Poisson problem with finite difference methods. We investigate some direct numerical methods that can used to solve the system. Examples include the Gauss elimination (or LU factorization), the Cholesky factorization and diagonalization methods. The latter will require a knowledge of the eigenvalues and eigenvectors of the constituent matrix operators.

1. The 1-dimension problem

We consider the boundary-value problem (BVP)

$$-\frac{\mathrm{d}^2 u}{\mathrm{d}\, x^2} = f \quad \text{in} \quad \Omega \tag{1}$$

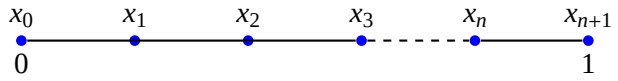
$$u = g_D \quad \text{on} \quad \partial\Omega \tag{2}$$

For simplicity, we consider homogeneous Dirichlet boundary conditions ($g_D = 0$) and choose $\Omega = (0, 1)$. That is, we consider the BVP

$$-\frac{\mathrm{d}^2 u}{\mathrm{d}\, x^2} = f, \quad 0 < x < 1, \tag{3}$$

$$u(0) = u(1) = 0. \tag{4}$$

Given the uniform grid below



with nodes (points)

$$x_i = ih, \quad i = 0, \ldots, n+1, \quad h = \frac{1}{n+1},$$

we will approximate the unknown function u at the n discrete nodes $x_1, \ldots, x_n$ such that

$$u(x_i) \approx \mathsf{u}_i, \quad i = 1, \ldots, n$$

while

$$u(x_0) = u(0) = \mathsf{u}_0, \quad u(x_{n+1}) = u(1) = \mathsf{u}_{n+1}.$$

We shall also write

$$f(x_i) = \mathsf{f}_i, \quad i = 1, \ldots, n.$$

The second derivative in (3) is approximated at each point (node) x_i by the finite difference formula

$$\left.\frac{\mathrm{d}^2 u}{\mathrm{d}x^2}\right|_{x_i} \approx \frac{\mathsf{u}_{i-1} - 2\mathsf{u}_i + \mathsf{u}_{i+1}}{h^2}, \quad i = 1, \ldots n.$$

We use this formula to approximate (3) by

$$-\frac{\mathsf{u}_{i-1} - 2\mathsf{u}_i + \mathsf{u}_{i+1}}{h^2} = \mathsf{f}_i, \quad i = 1, \ldots n.$$

This yields a linear system

$$-\mathsf{u}_{i-1} + 2\mathsf{u}_i - \mathsf{u}_{i+1} = h^2 \mathsf{f}_i, \quad i = 1, \ldots n, \tag{5}$$

of n equations in n unknowns. In matrix-vector form, we get

$$\begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{bmatrix} \begin{bmatrix} \mathsf{u}_1 \\ \\ \vdots \\ \\ \mathsf{u}_n \end{bmatrix} = h^2 \begin{bmatrix} \mathsf{f}_1 \\ \\ \vdots \\ \\ \mathsf{f}_2 \end{bmatrix}$$

or simply

$$A\mathsf{u} = \mathsf{b}, \tag{6}$$

where

$$A = \begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{bmatrix}, \quad \mathsf{u} = \begin{bmatrix} \mathsf{u}_1 \\ \\ \vdots \\ \\ \mathsf{u}_n \end{bmatrix}, \quad \mathsf{b} = h^2 \begin{bmatrix} \mathsf{f}_1 \\ \\ \vdots \\ \\ \mathsf{f}_n \end{bmatrix}.$$

We observe that A is symmetric and positive-definite (SPD).

Exercise. Use Gerschgorin's theorem to obtain bounds for the eigenvalues of A.

The system (6) can be solved using direct methods based on the following matrix factorizations

i) Gaussian elimination or LU factorization. Total cost $\approx 2n^3/3$ [1]flops.
ii) Cholesky factorization (since A is SPD). Total cost $\approx n^3/3$ flops.
iii) Eigenvalue decomposition (diagonalization). Since A is normal, the decomposition is unitary. Once A has been diagonalized, we only require $\approx (4n^2 + n)$ flops to solve for u.

Suppose the diagonalization yields

$$A = Q\Lambda Q^T$$

[1]floating point operations

where $\Lambda = diag(\lambda_1, \ldots, \lambda_n)$ with $\lambda_m, \quad m = 1, \ldots, n$ the eigenvalues of A, and $Q = [v_1|\cdots|v_n]$ contains the eigenvectors and is *unitary*. Substituting in (6) we get

$$Q\Lambda Q^T \mathsf{u} = \mathsf{b}, \tag{7}$$

and by applying the change of variables

$$\tilde{\mathsf{u}} := Q^T \mathsf{u}, \quad \tilde{\mathsf{b}} := Q^T \mathsf{b}$$

we get the simpler system

$$\Lambda \tilde{\mathsf{u}} = \tilde{\mathsf{b}},$$
$$\text{i.e.} \quad \lambda_i \tilde{\mathsf{u}}_i = \tilde{\mathsf{b}}_i, \quad i = 1, \ldots, n.$$

Thus given the λ_i and Q we proceed in three simple steps:

Step 1: Compute $\tilde{\mathsf{b}}$

$$\tilde{\mathsf{b}} = Q^T \mathsf{b}.$$

Step 2: Compute $\tilde{\mathsf{u}}$

$$\tilde{\mathsf{u}}_i = \tilde{\mathsf{b}}_i / \lambda_i, \quad i = 1, \ldots, n.$$

Step 3: Compute u

$$\mathsf{u} = Q\tilde{\mathsf{u}}.$$

Thus the total cost for solving the diagonalized system (7) is given by

$$\begin{aligned} \text{Cost} &= 2n^2 + n + 2n^2 \qquad \text{(Step 1 + Step 2 + Step 3)} \\ &= (4n^2 + n) \text{ flops.} \end{aligned}$$

1.1. **Diagonalization.** The matrix A is tridiagonal and toeplitz. So its eigenvalues can be obtained in closed form as[2]

$$\lambda_m = 2 - 2\cos\frac{m\pi}{n+1} = 4\sin^2\frac{m\pi}{2(n+1)}, \quad m = 1, \ldots, n.$$

The eigenvalue problem of the continuous Laplace operator in 1D

$$\begin{cases} -u'' = \lambda u \quad \text{in} \quad (0,1) \\ u(0) = u(1) = 0 \end{cases}$$

is known to have eigenfunctions $\varphi_m = \sin \pi m x$ for $n = 1, 2, \ldots$. It can be shown that the eigenvectors of the discrete Laplace operator A are given by values of the continuous eigenfunctions φ_m *sampled* at the discrete nodes $x_i, \quad i = 1, \ldots, n$. That is,

$$v_m = \begin{bmatrix} v_{m,1} \\ \vdots \\ v_{m,n} \end{bmatrix} = \begin{bmatrix} \varphi_m(x_1) \\ \vdots \\ \varphi_m(x_n) \end{bmatrix} = \begin{bmatrix} \sin \pi m x_1 \\ \vdots \\ \sin \pi m x_n \end{bmatrix}$$

satisfies $Av_m = \lambda_m v_m, \quad m = 1, \ldots, n$.

Exercise. Show that $Av_m = \lambda_m v_m, \quad m = 1, \ldots, n$, where v_m, λ_m are given as above.

[2] see notes on eigenvalues of toeplitz matrices on the course webpage

2. The 2-dimensional problem

Here we consider the boundary-value problem (BVP)

$$-\Delta u = f \quad \text{in} \quad \Omega \tag{8}$$
$$u = g_D \quad \text{on} \quad \partial\Omega \tag{9}$$

where Δ denotes the Laplace operator, defined for $u = u(x, y)$ by

$$\Delta u := \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = u_{xx} + u_{yy}$$

Again, for simplicity's sake, we consider homogeneous Dirichlet boundary conditions ($g_D = 0$) and choose $\Omega = (0, 1) \times (0, 1)$. That is, we consider the BVP

$$-(u_{xx} + u_{yy}) = f \quad \text{in} \quad \Omega \tag{10}$$
$$u = 0 \quad \text{on} \quad \partial\Omega \tag{11}$$

We will approximate $u = u(x, y)$ on nodes (points) in a uniform rectangular mesh

$$x_i = ih,\ y_j = jh, \quad i, j = 0, \ldots, n+1, \quad h = \frac{1}{n+1},$$

such that

$$u(x_i, y_j) \approx U_{i,j}, \quad i, j = 1, \ldots, n$$

while

$$\begin{aligned} u(x_0, y_j) = u(0, y_j) &= U_{0,j}, & j &= 0, \ldots, n+1 \\ u(x_{n+1}, y_j) = u(1, y_j) &= U_{n+1,j}, & j &= 0, \ldots, n+1 \\ u(x_i, y_0) = u(x_i, 0) &= U_{i,0}, & i &= 0, \ldots, n+1 \\ u(x_i, y_{n+1}) = u(x_i, 1) &= U_{i,n+1}, & i &= 0, \ldots, n+1 \end{aligned}$$

are given from the boundary conditions.

We apply finite differences to approximate the second derivatives as follows:

$$u_{xx}|_{(x_i, y_j)} \approx \frac{U_{i-1,j} - 2U_{i,j} + U_{i+1,j}}{h^2}$$

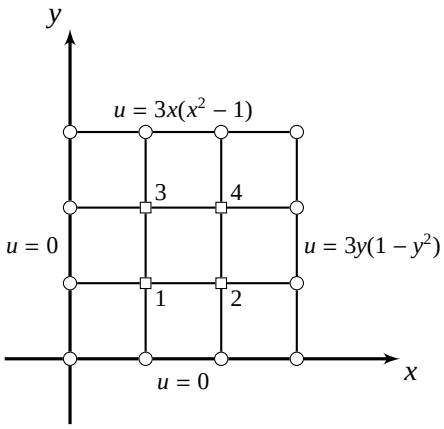
$$u_{yy}|_{(x_i, y_j)} \approx \frac{U_{i,j-1} - 2U_{i,j} + U_{i,j+1}}{h^2}$$

Substituting these formulas in (10) we obtain the linear system

$$4U_{i,j} - U_{i-1,j} - U_{i+1,j} - U_{i,j-1} - U_{i,j+1} = h^2 F_{i,j}, \quad i, j = 1, \ldots, n \tag{12}$$

where $F_{i,j} = f(x_i, y_j)$. The system consists of $N = n^2$ equations in N unknowns.

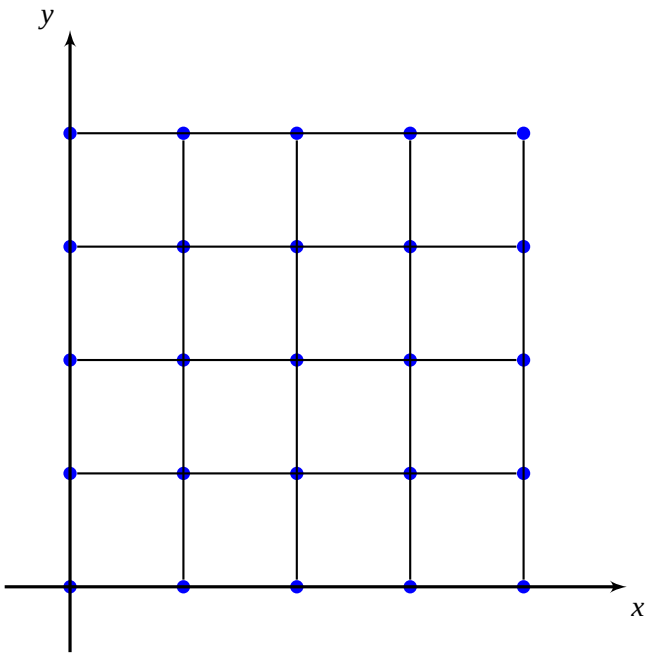
Example 1. *Consider the discretization of the Poisson equation (10) with $f(x, y) = y - x$ on the uniform grid given below (with the indicated Dirichlet boundary conditions).*

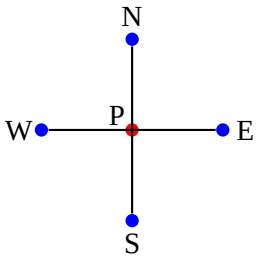


The Poisson problem in 2D:
$f = y - x$, $h = \frac{1}{3}$; BC: $u(x,0) = u(0,y) = 0$, $u(x,1) = 3x(x^2-1)$, $u(1,y) = 3y(1-y^2)$, $\Omega = (0,1)\times(0,1)$.

Applying (12) on each of the interior nodes (square nodes, labelled 1, 2, 3, 4*) and collecting the boundary contributions to the right-hand side we obtain the* $n^2 \times n^2$ *(with* $n = 2$*) linear system*

$$\begin{bmatrix} 4 & -1 & -1 & 0 \\ -1 & 4 & 0 & -1 \\ -1 & 0 & 4 & -1 \\ 0 & -1 & -1 & 4 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} 0 \\ \frac{2}{3} \\ -\frac{2}{3} \\ 0 \end{bmatrix}.$$





We can use a 5-point stencil (see Figure above) to write down each of the equations in (12) as follows:

$$U_P - U_W - U_E - U_S - U_N = F_P, \qquad \text{for each grid point } P. \tag{13}$$

The resulting matrix system is of the (general) form

$$A\mathsf{u} = \mathsf{b} \tag{14}$$

where

$$\begin{aligned} \mathsf{b} &= h^2\mathsf{f} + \mathsf{b}_0; \\ \mathsf{b}_0 &= \text{ boundary contributions from the left-hand side of (13).} \end{aligned}$$

Meanwhile

$$\mathsf{u} = \begin{bmatrix} U_{11} \\ \vdots \\ U_{n1} \\ \\ \vdots \\ \\ U_{1n} \\ \vdots \\ U_{nn} \end{bmatrix}, \quad \mathsf{f} = \begin{bmatrix} F_{11} \\ \vdots \\ F_{n1} \\ \\ \vdots \\ \\ F_{1n} \\ \vdots \\ F_{nn} \end{bmatrix},$$

and A is a block-tridiagonal matrix of the form

$$A = \begin{bmatrix} S & -I & & & \\ -I & S & -I & & \\ & \ddots & \ddots & \ddots & \\ & & -I & S & -I \\ & & & -I & S \end{bmatrix}, \quad \text{with} \quad S = \begin{bmatrix} 4 & -1 & & & \\ -1 & 4 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 4 & -1 \\ & & & -1 & 4 \end{bmatrix}$$

and I is the $n \times n$ identity matrix. So A is an $n^2 \times n^2$ matrix while u, b are each $n^2 \times 1$ vectors.

We observe that A is symmetric and positive-definite (SPD). The system can be solved using

i) Gaussian elimination or LU factorization. Total cost $\approx 2N^3/3$ flops.
ii) Cholesky factorization (since A is SPD). Total cost $\approx N^3/3$ flops. Exploiting the sparsity of A reduces cost to $\approx N^2 + N^{3/2}$.
iii) Eigenvalue decomposition (diagonalization). We would show that this would lead to a fast Poisson solver, several orders of magnitude faster than the LU or Cholesky factorization approaches.

2.1. **Diagonalization and Fast Poisson solvers in 2D.** From (12) we have that

$$4U_{i,j} - U_{i-1,j} - U_{i+1,j} - U_{i,j-1} - U_{i,j+1} = h^2 F_{i,j}, \quad i, j = 1, \ldots, n, \tag{15}$$

where the unknowns are stored in the $n \times n$ matrix U. In the 1D case we obtain the discrete linear operator

$$T = \begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{bmatrix}.$$

The left-hand side of (15) can be split into two as follows:

$$\begin{aligned} & 4U_{i,j} - U_{i-1,j} - U_{i+1,j} - U_{i,j-1} - U_{i,j+1} \\ &= \underbrace{\left(-U_{i-1,j} + 2U_{i,j} - U_{i+1,j}\right)}_{(TU)_{i,j}} + \underbrace{\left(-U_{i,j-1} + 2U_{i,j} - U_{i,j+1}\right)}_{(UT)_{i,j}} \end{aligned}$$

Thus we can write (15) in matrix-matrix form as

$$TU + UT = G := h^2 F \tag{16}$$

This matrix-matrix form has an advantage over the matrix-vector form (14) in terms of the storage requirements. In the latter we need $O(N^2) = O(n^4)$ space in memory to store the matrix A, and additional $O(N) = O(n^2)$ to store the right-hand side b. [Recall that the dimension of A is $N \times N$, where $N = n^2$]. This makes a total memory space requirement of order $O(n^4 + n^2)$. On the other hand, the representation in (16) only requires $O(3n^2)$ memory space to store the three $n \times n$ matrices U, T, G.

From Section 1, we know that the tridiagonal matrix T is unitarily diagonalizable such that

$$T = Q\Lambda Q^T, \quad \Lambda = diag(\lambda_1, \ldots, \lambda_n), \quad Q^T Q = I$$

since T is symmetric (hence normal). The eigenvalues $\lambda_1, \ldots, \lambda_n$ are real (since T is symmetric) and positive (since T is positive-definite). We have also explained how to explicitly compute Q and Λ.

Substituting the factorization of T into (16) we obtain

$$Q\Lambda Q^T U + UQ\Lambda Q^T = G.$$

Pre-multiplying and post-multiplying both sides by Q^T and Q respectively, yields

$$\Lambda \underbrace{Q^T U Q}_{\tilde{U}} + \underbrace{Q^T U Q}_{\tilde{U}} \Lambda = \underbrace{Q^T G Q}_{\tilde{G}}$$

i.e.

$$\begin{aligned} & \Lambda\tilde{U} + \tilde{U}\Lambda = \tilde{G} \\ \implies \quad & \lambda_i \tilde{U}_{i,j} + \tilde{U}_{i,j}\lambda_j = \tilde{G}_{i,j} \\ \implies \quad & \tilde{U}_{i,j} = \frac{\tilde{G}_{i,j}}{\lambda_i + \lambda_j}, \quad i, j = 1, \ldots, n. \end{aligned}$$

This can be done in three steps (given λ_i and Q)

Step 1: Compute

$$\tilde{G} = Q^T G Q.$$

Cost = $2n^3 + 2n^3$ (2 matrix-matrix products).

Step 2: Compute the entries of $\tilde{U}$ using

$$\tilde{U}_{i,j} = \frac{\tilde{G}_{i,j}}{\lambda_i + \lambda_j}, \quad i, j = 1, \ldots, n.$$

Cost = n^2 (adds) + n^2 (divides) = $2n^2$ flops.

Step 3: Compute U using

$$U = Q\tilde{U}Q^T.$$

Cost = $2n^3 + 2n^3$ flops (2 matrix-matrix products).

Total cost = $4n^3 + 2n^2 + 4n^3 = 8n^3 + 2n^2$ flops. Using FFT, the cost of matrix-matrix products reduces to $O(n^2 \log n)$, with $n^2 \log n \ll n^3$ for large n. This brings the total cost down to $O(2n^2 + n^2 \log n)$. The overall cost of this three-step procedure is only about $O(N^{3/2})$ which far lower than $O(N^3)$ the cost for Gaussian elimination.

We shall see in subsequent lectures that iterative methods have greater advantages over the direct methods mentioned here, both in terms computational speed and memory storage.