Math 471. Numerical methods

Chapter 8&9. Elliptic PDEs in 1D & 2D:
Boundary Value Problems

Overlap §8.1, 8.2, 9.1 of Bradie

§ 8.1 Boundary value problem in 1D

The (time-dependent) heat equation in 1D

$$\text{PDE}: \frac{1}{k(x)} u_t = u_{xx} + \alpha(x) u_x + \beta(x) u + Q(x,t)$$

$$\text{Boubadry Conditions}: u(0,t) = a, \quad u(L,t) = b$$

$$\text{Initial Condition}: u(x,0) = f(x)$$

By setting $u_t = 0$, we arrive at an ODE discribing the equilibrium, "steady-state" solution

$$u_{xx} + \alpha(x) u_x + \beta(x) u + Q(x,t), \qquad \text{with } u(0) = a, \quad u(L) = b. \tag{1}$$

Here, $\alpha(x)$, $\beta(x)$ are all given functions.

Question: how do we interpretate equation (1) in such a way that the computer can properly handle the derivatives, etc. and eventually find a numerical solution to (1) — at least on the approximate level?

The process is called **discretization**, at the center of which is the use of Finite Difference in approximating derivatives.

$$\begin{cases} \text{domain } x \in [0,L] \to \text{discrete domain } x_i = ih \text{ for } 0 \le i \le n, \text{ where } h = x_{i+1} - x_i = \dfrac{L}{n} \\ \text{function } Q(x) \to \text{vector } \vec{q} = (Q(x_1), Q(x_2), ..., Q(x_{n-1}))^T \\ \qquad\qquad \stackrel{def}{=} (Q_1, Q_2, ..., Q_{n-1})^T \\ \text{Define } \{\alpha_i\},\ \{\beta_i\} \text{ similarly} \\ \text{unknown function } u(x) \to \text{unknown vector } \vec{v} = (u(x_1), u(x_2), ..., u(x_{n-1}))^T \\ \qquad\qquad \stackrel{def}{=} (u_1, u_2, ..., u_{n-1})^T \end{cases}$$

To approximate 1st derivative, we have 3 formulas, named after the stencils they use

$$u'(x_i) \approx \begin{cases} \text{backward difference: } D^- v_i = D^- u\big|_{x=x_i} \\ \qquad\qquad\qquad\qquad \stackrel{def}{=} \dfrac{1}{h}(u(x_i) - u(x_{i-1}) = \dfrac{1}{h}(v_i - v_{i-1}) \\ \text{forward difference: } D^+ v_i \stackrel{def}{=} \dfrac{1}{h}(v_{i+1} - v_i) \\ \text{centered difference: } D^0 v_i \stackrel{def}{=} \dfrac{1}{2h}(v_{i+1} - v_i) \end{cases}$$

A study of the truncation errors $|u'(x_i) - D^- v_i|$ etc (using Taylor series) reveals that the centered difference is of order $O(h^2)$ accuracy while the other 2 are only of $O(h)$. So, we will adopt $D^0 v_i$ to approximate $u'(x_i)$.

For the second derivative, there is an ideal candidate of $O(h^2)$ truncation error, called the "2nd order centered difference"

$$u'(x_i) \approx D^+ D^- v_i = \frac{1}{h^2}(v_{i-1} - 2v_i + v_{i+1}).$$

It is essentially a composition of D^+ and D^-.

Note. Here, we make strict distinction in notations of the exact solution $u(x)$ and the approximate solution $\vec{v}$. The approximation occurs in using finite difference to approximate derivatives.

So, equation (1) has a discretized version at *each and every* x_i for $1 \le i \le n-1$

$$D^+ D^- v_i + \alpha_i D^0 v_i + \beta_i v_i + Q_i = 0,\ i = 1, 2, ..., n-1 \text{ with BC } v_0 = v_n = 0$$

which is essentially a system of linear equations of the unknown $\vec{v} = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_{n-1} \end{pmatrix}$. From the discussion above, we see that

$$\text{This discretization introduces truncation error at order } O(h^2) \tag{2}$$

Such source of error should be strictly distinguished from the error incurred in solving the linear system itself.

With more details, the i-th equation of the system is

$$\frac{1}{h^2}(v_{i-1} - 2v_i + v_{i+1}) + \frac{\alpha_i}{2h}(v_{i+1} - v_{i-1}) + \beta_i v_i + Q_i = 0 \tag{3}$$

Notice that, if we set $i = 1$ in the above equation, the entry v_0 appears and should be replaced by its value a as given in the boundary conditions. This way, we treat v_0 as given. Similar treatment for the case $i = n - 1$ and v_n.

In order to rewrite (3) in matrix-vector notations $A\vec{v} = \vec{b}$, we write the coefficients of $v_i,\ v_{i-1},\ v_{i+1}$ in (3),

$$\begin{aligned} v_i &: -\frac{2}{h^2} + \beta_i \\ v_{i-1} &: \frac{1}{h^2} - \frac{1}{2h}\alpha_i \\ v_{i+1} &: \frac{1}{h^2} + \frac{1}{2h}\alpha_i \end{aligned}$$

So, the coef of v_i enters the (i, i) position on the *diagonal* of A, coef of v_{i-1} in the $(i, i-1)$ position and coef of v_{i+1} in the $(i, i+1)$ position

$$\begin{pmatrix} -\frac{2}{h^2} + \beta_1 & \frac{1}{h^2} + \frac{1}{2h}\alpha_1 & & & & & \\ \frac{1}{h^2} - \frac{1}{2h}\alpha_2 & -\frac{2}{h^2} + \beta_2 & \frac{1}{h^2} + \frac{1}{2h}\alpha_2 & & & & \\ & \ddots & \ddots & \ddots & & & \\ & & \frac{1}{h^2} - \frac{1}{2h}\alpha_i & -\frac{2}{h^2} + \beta_i & \frac{1}{h^2} + \frac{1}{2h}\alpha_i & & \\ & & & \ddots & \ddots & \ddots & \\ & & & & \frac{1}{h^2} - \frac{1}{2h}\alpha_{n-1} & -\frac{2}{h^2} + \beta_{n-1} \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_{i-1} \\ v_i \\ v_{i+1} \\ \vdots \\ v_{n-1} \end{pmatrix}$$

$$= \begin{pmatrix} Q_1 \\ Q_2 \\ \vdots \\ Q_i \\ \vdots \\ Q_{n-1} \end{pmatrix} - \begin{pmatrix} \left(\frac{1}{h^2} - \frac{1}{2h}\alpha_1\right) * a \\ 0 \\ \vdots \\ 0 \\ \vdots \\ \left(\frac{1}{h^2} + \frac{1}{2h}\alpha_{n-1}\right) * b \end{pmatrix} \tag{4}$$

We see that, in the first equation, what could've been virtually placed to the left of $A_{1,1}$ as coef of v_0 is moved to the right. So is the value of $v_0 = a$. Same thing for the last equation.

Solving the tridiagonal system (4).

With the methods learned in Chapter 3,

- Direct methods of LU decomposition. For a tri-diagonal system, the most efficient way is the Thomas algorithm with operation count $O(n)$. $A = LU$ where A is the matrix on the left side of (4) and L and U are both bi-diagonal. Then, solving $A\vec{v} = \vec{b}$ costs $O(n)$ operations.
  A major draw back is that the condition number of A grows like $O(n^2)$, which makes direct methods sensitive to round-off error at large n's. In fact, a more accurate definition of condition number is

$$\kappa(A) = \rho(A)\rho(A^{-1})$$

  since the spectral radius $\rho(A)$ is a more reliable quantification than $\|A\|$ in terms of the amplification ability of A. Then, one can show that for eigenvalues of A in our case: $\lambda_1 < \lambda_2 < ... < \lambda_n < 0$ and

$$\lambda_1 \approx -4n^2 + C, \qquad \lambda_n \approx -C$$

$$\implies \rho(A) = |\lambda_1| \approx 4n^2 - C.$$

  But the eigenvalues of A^{-1} are just $\lambda_n^{-1} < \lambda_{n-1}^{-1} < ... < \lambda_1^{-1} < 0$ which leads to

$$\rho(A^{-1}) = \left|\frac{1}{\lambda_n}\right| \approx 1/C$$

  Therefore, the condition number

$$\kappa(A) \approx O(n^2)$$

- Iterative methods, on the other hand, are generally more stable and less sensitive to round-off errors. Any error (including $\vec{x}_k - \vec{x}_\infty$ and round-off error) in the iteration $\vec{x}_{k+1} = B\vec{x}_k + \vec{c}$ is being amplified only by the iteration matrix B. So as long as $\rho(B) < 1$, errors from all sources decay like $(\rho(B))^k$.
  Is operation count still $O(n)$, comparable to direct LU methods?

  In our case, A is sparse and tridiagonal. It is easy to check that the Jacobi method costs $O(n)$ in each iteration, and so does the Guass-Sidel method. So, the total op. ct. is $O(n) * m$ where m is the number of iterations performed. This brings us to the next question ...
  How many iterations needed for certain accuracy?

The answer relies on the convergence analysis learned in Chapter 3. The key formula is

$$\lim_{k\to\infty} \frac{\|e_{k+1}\|}{\|e_k\|} = \rho(B).$$

Using a very similar approach as for the condition number above, one can show that

$$\rho(B_{jac}) \approx 1 - \frac{C}{n^2}.$$

Moreover, A is tri-diagonal and negative-definite if it is formulated in a better way so that $A = A'$. By a theorem from Chapter 3,

$$\rho(B_{GS}) = \rho(B_{Jac})^2 \approx 1 - \frac{C}{n^2} \leq \rho(B_{Jac})$$

and for the SOR method at an optimal choice of ω^*,

$$\rho(B_{SOR(\omega^*)}) \approx 1 - \frac{C}{n} \ll \rho(B_{GS}).$$

So the SOR method converges faster than the other two for large n's. A little analysis will show that at the k-th iteration

$$\|e_k\| \approx \|e_0\|(1 - \frac{C}{n})^k = \|e_0\| k \exp(\ln(1 - \frac{C}{n})) \approx \|e_0\| \exp(-\frac{Ck}{n})$$

§8.2 Other boundary conditions Here, we only briefly mention some cases that involve derivatives of u as part of the boundary data.

$$\begin{aligned} \text{von Neumann type :} \quad & u'(0) = a, \;\; u'(L) = b \\ \text{mixed type :} \quad & k_1 u(0) - k_2 u'(0) = a, \;\; k_3 u(L) + k_4 u'(L) = b \\ \text{different types at} \quad & x = 0 \text{ and } x = L \\ & \ldots\ldots \end{aligned}$$

Take $u'(0) = a$ for example. Now $v_0 = u(0)$ is no longer given and should be included as part of the unknown vector

$$\vec{v} = \begin{pmatrix} v_0 \\ v_1 \\ \vdots \end{pmatrix}$$

To adapt this change, we add one more row at the top and one more column at the left of A in System (4). So the equation at x_1, in the second row of the new system, becomes the same as the generic case (3) with $i = 1$,

$$\frac{1}{h^2}(v_0 - 2v_1 + v_2) + \frac{\alpha_1}{2h}(v_2 - v_0) + \beta_1 v_1 + Q_1 = 0.$$

The equation at x_0, in the first row of the new system, should reflect the boundary condition $u'(0) = a$. To this end, we approximate $u'(0)$ using forward difference because the value of $v_{-1} \approx u(x_{-1})$ is beyond the physical domain of this problem.

$$a = u'(0) \approx D^+ v_0 = \frac{1}{h}(v_1 - v_0).$$

Realize this equation in A and we see the first row of A is simply

$$-\frac{1}{h} \quad \frac{1}{h} \quad 0....0$$

which still preserves the tridiagonal pattern!

The problem with D^+ is that the truncation error becomes $O(h)$, degrading the overall $O(h^2)$ truncation error that we make an effort to preserve (see (2)). So, an alternative approach is to use higher precision versions of D^+. This will typically involve more stencils. If v_0, v_1, v_2 are used,

$$u'(0) \approx \frac{-3v_0 + 4v_1 - v_2}{2h}.$$

The truncation error is studied using Taylor series and turns out to be $O(h^2)$. Another way to improve accuracy is to use a "ghost point" v_{-1} in the centered difference

$$u'(0) \approx \frac{v_1 - v_{-1}}{2h}.$$

The value of v_{-1} comes from extrapolation of $u(x)$. (details skipped)

§9.1 Elliptic PDEs in 2D

Consider a 1×1 plate at thermal equilibrium

$$u_{xx} + u_{yy} = f(x, y), \qquad \text{for } x \in [0, 1],\ y \in [0, 1]$$

subject to Direchlet boundary conditions $u(0, y) = u(1, y) = u(x, 0) = u(x, 1) = 0$. Here, $f(x, y)$ is a given source term. How to numerically solve this problem?

• Step 1. Disretization.

We start with discretizing the equation. As before, use central difference for the second derivatives.

$$u_{xx} \approx D_x^- D_x^+ u(x,y) = \frac{1}{h^2}\left(u(x+h,y) - 2u(x,y) + u(x-h,y)\right) \tag{5}$$

$$u_{yy} \approx D_y^- D_y^+ u(x,y) = \frac{1}{h^2}\left(u(x,y-h) - 2u(x,y) + u(x,y-h)\right) \tag{6}$$

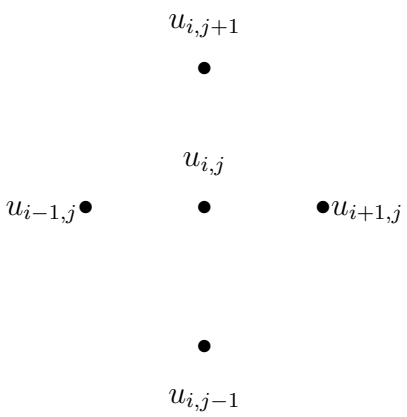
Now, by partioning the square domain into a $n \times n$ grid with $h = 1/n$,

$$(x,y) = (ih, jh), \quad \text{for} \quad i = 0,1,...,n, \quad j = 0,1,...,n$$

we represent $u(x,y)$ with dicretized values

$$u_{i,j} \sim u(ih, jh).$$

Here, $u_{i,j}$ is an approximate value for $u(ih,jh)$ (and therefore the $\sim$ sign).



The above cartoon is a visualization of a 5-point stencil around point $(x,y) = (ih,jh)$. Thus, rewrite the central difference in (5), (6) as

$$u_{xx}(ih,jh) \approx \frac{1}{h^2}\left(u_{i+1,j} - 2u_{i,j} + u_{i-1,j}\right) \tag{7}$$

$$u_{yy}(ih,jh) \approx \frac{1}{h^2}\left(u_{i,j-1} - 2u_{i,j} + u_{i,j-1}\right) \tag{8}$$

• Step 2. Write a structured linear system.

By (7), (8), the discretized equations are

$$\frac{1}{h^2}\left(u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j}\right) = f_{i,j} = f(ih,jh), \quad \text{for } i = 1,...,n-1, \quad j = 1,...,n-1 \tag{9}$$

subject to boundary conditions $u_{0,j} = u_{n,j} = u_{j,0} = u_{j,n} = 0$. Here, there are $(n-1)^2$ unknowns $u_{i,j}$ for $i = 1, ..., n-1, \quad j = 1, ..., n-1$. Note that only the values of $u_{i,j}$ at the interior points are unknown.

To transfer (9) into a linear system $A\vec{x} = \vec{b}$, we need an unknown vector $\vec{x}$ with $(n-1)^2$ entries representing all the $u_{i,j}$ and we need a coefficient matrix A of dimension $(n-1)^2$-by-$(n-1)^2$. The ordering of $u_{i,j}$ inside $\vec{x}$ is not unique. Let's use such an order: fill in the first $n-1$ entries of $\vec{x}$ with $\vec{u}_1 = (u_{1,1}, u_{2,1}, u_{3,1}...u_{n-1,1})^T$ and then fill in the second $n-1$ entries of $\vec{x}$ with $\vec{u}_2 = (u_{1,2}, u_{2,2}, u_{3,2}...u_{n-1,2})^T$

$$
\begin{array}{lcccc}
 & \bullet & \bullet & ... & \bullet \\
\vec{u}_k \rightarrow & u_{1,k} & u_{2,k} & ... & u_{n-1,k} \\
 & \vdots & \vdots & & \vdots \\
 & \bullet & \bullet & ... & \bullet \\
\vec{u}_2 \rightarrow & u_{1,2} & u_{2,2} & ... & u_{n-1,2} \\
 & \bullet & \bullet & ... & \bullet \\
\vec{u}_1 \rightarrow & u_{1,1} & u_{2,1} & ... & u_{n-1,1}
\end{array}
$$

Assemble all the $\vec{u}_k$'s as

$$\vec{x} = \begin{pmatrix} \vec{u}_1 \\ \vec{u}_2 \\ \vdots \\ \vec{u}_{n-1} \end{pmatrix} \quad \text{where, zooming in,} \quad \vec{u}_j = \begin{pmatrix} u_{1,j} \\ u_{2,j} \\ \vdots \\ u_{n-1,j} \end{pmatrix}.$$

Likewise assemble the RHS of (9) into vector $\vec{b} = \begin{pmatrix} \vec{f}_1 \\ \vec{f}_2 \\ \vdots \\ \vec{f}_{n-1} \end{pmatrix}$.

Coefficient matrix A comes from the LHS of (9). It has the following **block** tri-diagonal structure

$$A = \frac{1}{h^2} \begin{pmatrix} T & I & & & & \\ I & T & I & & & \\ & I & T & I & & \\ & & \ddots & \ddots & \ddots & \\ & & & I & T & I \\ & & & & I & T \end{pmatrix} \tag{10}$$

where each block I is the $(n-1)$-by-$(n-1)$ identity matrix and each block T is an $(n-1)$-by-$(n-1)$ tri-diangonal matrix

$$T = \begin{pmatrix} -4 & 1 & & & & & \\ 1 & -4 & 1 & & & & \\ & 1 & -4 & 1 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & \ddots & \ddots & \ddots & \\ & & & & 1 & -4 & 1 \\ & & & & & 1 & -4 \end{pmatrix}$$

Why such a structure? Roughly speaking, the diagonal part of A corresponds to $-4u_{i,j}$ in (9). The j-th T block (minus the diagonal) in the $j-th$ "block-row" of A corresponds to the x direction $u_{i+1,j} + u_{i-1,j}$ in (9) since $u_{i+1,j},\ u_{i,j},\ u_{i-1,j}$ are ordered next to each other in the j-th block of $\vec{x}$. The two I blocks in the j-th "block-row" of A correspond to the y direction $u_{i,j+1} + u_{i,j-1}$ in (9) since $u_{i,j+1},\ u_{i,j},\ u_{i,j-1}$ are place in $\vec{x}$ exactly $(n-1)$ entries apart from each other in 3 different blocks.

• Step 3. Solve $A\vec{x} = \vec{b}$ using direct methods or iterative methods. It is recommended that, even for moderate size n, iterative methods be used. The issue with direct methods, as discussed in Chapter 8, is worsened in 2D and 3D.