03_elliptic_equations

December 10, 2025

[4]:
```
%matplotlib inline
from __future__ import print_function
import numpy
import matplotlib.pyplot as plt
```

1 Finite Difference Methods for 2D Elliptic PDEs

1.1 Overview

In today's lecture, we will discuss the **5-point Laplacian** for the numerical solution of the **Poisson equation**, and introduce the **9-point Laplacian**. These notes are based largely on the book by Randall LeVeque entitled *Finite Difference Methods for Ordinary and Partial Differential Equations* (SIAM, Philadelphia, PA, 2007). Other interesting reference is *Numerical Solution of Differential Equations* by Z. Li, Z. Qiao and T. Tang.

1.2 Introduction

A constant-coefficient elliptic equation in $\mathbb{R}^2$ has the form

$$Au_{xx} + Bu_{xy} + Cu_{yy} + Du_x + Eu_y + Fu = f,$$

where the coefficients A, B and C satisfy

$$B^2 - 4AC < 0.$$

In this section of the course, we will consider the **Poisson problem**

$$u_{xx} + u_{yy} = f,$$

and the special case $f = 0$ which reduces to **Laplace's equation**,

$$u_{xx} + u_{yy} = 0.$$

1.3 The 5-point Laplacian

Consider the Poisson equation with **Dirichlet BC**

$$\begin{aligned} u_{xx} + u_{yy} &= f(x,y), \quad (x,y) \in \Omega = (a,b) \times (c,d), \\ u(x,y) &= g(x,y), \qquad\qquad\qquad (x,y) \in \partial\Omega. \end{aligned}$$

A finite difference approximation can be obtained through the following procedure:

- **Step 1: Generate a grid.**

For instance,

$$\begin{aligned} x_i &= a + ih_x, \quad i = 0,1,2,\dots,m+1, \quad h_x = \frac{b-a}{m+1}, \\ y_j &= c + jh_y, \quad j = 0,1,2,\dots,n+1, \quad h_y = \frac{d-c}{n+1}. \end{aligned}$$

- **Step 2: Define a grid function.**

$$U_{ij} \approx u(x_i, y_j), \quad i = 0,1,\dots,m+1, \quad j = 0,1,\dots,n+1$$

  where U_{ij} is the numerical approximation on the grid to the unknown exact solution $u(x_i, y_j)$.

- **Step 3: FD approximation of the Laplace operator.**

  Approximate the partial derivatives at grid points with finite difference formulas. For example, if we adopt the three-point central finite difference formula for second-order partial derivatives in the x- and y-directions, then

$$\frac{1}{h_x^2}\left[u(x_{i-1},y_j) - 2u(x_i,y_j) + u(x_{i+1},y_j)\right] + \frac{1}{h_y^2}\left[u(x_i,y_{j-1}) - 2u(x_i,y_j) + u(x_i,y_{j+1})\right] = f(x_i,y_j) + \tau(x_i,y_j),$$

  which leads to the numerical scheme

$$\frac{1}{h_x^2}\left[U_{i-1,j} - 2U_{ij} + U_{i+1,j}\right] + \frac{1}{h_y^2}\left[U_{i,j-1} - 2U_{ij} + U_{i,j+1}\right] = F_{ij},$$

  for $i = 1,\dots,m; j = 1,\dots,n$.

- **Step 4: Assemble collocation matrix.**

  For simplicity, let's assume here that $(b-a) = (d-c)$ and $m = n$, so $h_x = h_y = h$. Then, the numerical scheme above can be replaced by

$$\frac{1}{h^2}\left[U_{i-1,j} + U_{i+1,j} + U_{i,j-1} + U_{i,j+1} - 4U_{ij}\right] = F_{ij}, \quad i,j = 1,2,\dots,m.$$

  If we collect all these equations together into a matrix equation,

$$\mathbf{AU} = \mathbf{F},$$

  it will yield an $m^2 \times m^2$ sparse matrix $\mathbf{A}$, with at most five nonzero entries per row and at least $m^2 - 5$ elements that are zero. Defining the vector of unknowns along the bottom row, $U_{11}, U_{21}, U_{31}, \dots, U_{m1}$, followed by the unknowns in the second row, $U_{12}, U_{22}, U_{32}, \dots, U_{m2}$, and so on, i.e.

$$\mathbf{U} = \begin{pmatrix} U^{[1]} \\ U^{[2]} \\ \vdots \\ U^{[m]} \end{pmatrix}, \quad \text{where } U^{[j]} = \begin{pmatrix} U_{1j} \\ U_{2j} \\ \vdots \\ U_{mj} \end{pmatrix},$$

leads to

$$\mathbf{A} = \frac{1}{h^2}\begin{pmatrix} \mathbf{B} & \mathbf{I} & & & \\ \mathbf{I} & \mathbf{B} & \mathbf{I} & & \\ & \mathbf{I} & \ddots & \ddots & \\ & & \ddots & \mathbf{B} & \mathbf{I} \\ & & & \mathbf{I} & \mathbf{B} \end{pmatrix},$$

which is an $m \times m$ block tridiagonal matrix in which each block $\mathbf{B}$ or $\mathbf{I}$ is itself an $m \times m$ matrix,

$$\mathbf{B} = \begin{pmatrix} -4 & 1 & & & \\ 1 & -4 & 1 & & \\ & 1 & \ddots & \ddots & \\ & & \ddots & -4 & 1 \\ & & & 1 & -4 \end{pmatrix},$$

and $\mathbf{I}$ is the $m \times m$ identity matrix.

- **Step 5: Solve linear system.**

  Two fundamentally different approaches can be followed: using a **direct method**, such as MATLAB's **backslash** (which is based on Gaussian elimination); or an **iterative method**, which will be discussed in the next lecture.

1.3.1 Task 1: Poisson Solver in 2D

Implement a Python function to solve the **Poisson problem** on a square domain with homogeneous Dirichlet boundary conditions, using the **5-point finite difference Laplacian**:

$$\nabla^2 u = -2\sin(x)\sin(y), \qquad (x, y) \in [0, 2\pi] \times [0, 2\pi],$$

with

$$u(x, 0) = 0, \quad u(x, 2\pi) = 0, \quad u(0, y) = 0, \quad u(2\pi, y) = 0.$$

(a) Implement the solver Write a Python function:

```
def poisson(m):
    """
    Solve the 2D Poisson equation:
         ²u = -2 sin(x) sin(y),    (x,y)   [0,2 ] × [0,2 ],
        u = 0 on the boundary.

    Parameters
    ----------
    m : int
        Number of interior grid points in each direction.

    Returns
    -------
    X, Y : 2D ndarrays
```

```
        Meshgrid of all grid points including boundaries.
    U : 2D ndarray
        Numerical solution at all grid points.
    """
    # Step 1: Discretize domain [0, 2] × [0, 2]
    # Step 2: Build sparse matrix A for the 5-point Laplacian
    # Step 3: Assemble RHS vector with f(x,y) = -2 sin(x) sin(y)
    # Step 4: Solve linear system AU = F
    # Step 5: Reconstruct solution including boundary values

    return X, Y, U
```

(b) Verify numerical convergence

1. Compute the solution for increasing values of `m` (e.g. `m = 8, 16, 32, 64`).
2. Compare with the exact solution

$$u(x,y) = \sin(x)\sin(y),$$

   and compute the error in ℓ^2 and ℓ^∞ norms.

3. Plot the errors vs. h (grid spacing) in a log-log plot, and verify that the scheme is **second-order accurate**.

1.4 Accuracy and Stability

We can of course (and should) ask the same questions as with the one-dimensional case, namely whether our scheme will converge. To do this we need to consider the LTE and the stability of our method. We know that the LTE is defined as

$$\tau_{ij} = \frac{1}{h^2}(u(x_{i-1},y_j) + u(x_{i+1},y_j) + u(x_i,y_{j-1}) + u(x_i,y_{j+1}) - 4u(x_i,y_j)) - f(x_i,y_j)$$

To compute this expression we need the Taylor series in each direction.

For the x-direction:

$$u(x_{i+1},y_j) = u(x_i,y_j) + hu(x_i,y_j)_x + \frac{h^2}{2}u(x_i,y_j)_{xx} + \frac{h^3}{6}u(x_i,y_j)_{xxx} + \frac{h^4}{24}u(x_i,y_j)_{xxxx} + \mathcal{O}(h^5)$$
$$u(x_{i-1},y_j) = u(x_i,y_j) - hu(x_i,y_j)_x + \frac{h^2}{2}u(x_i,y_j)_{xx} - \frac{\Delta^3}{6}u(x_i,y_j)_{xxx} + \frac{h^4}{24}u(x_i,y_j)_{xxxx} + \mathcal{O}(h^5)$$

For the y-direction:

$$u(x_i,y_{j+1}) = u(x_i,y_j) + \Delta y u(x_i,y_j)_y + \frac{\Delta y^2}{2}u(x_i,y_j)_{yy} + \frac{\Delta y^3}{6}u(x_i,y_j)_{yyy} + \frac{\Delta y^4}{24}u(x_i,y_j)_{yyyy} + \mathcal{O}(\Delta y^5)$$
$$u(x_i,y_{j-1}) = u(x_i,y_j) - \Delta y u(x_i,y_j)_y + \frac{\Delta y^2}{2}u(x_i,y_j)_{yy} - \frac{\Delta y^3}{6}u(x_i,y_j)_{yyy} + \frac{\Delta y^4}{24}u(x_i,y_j)_{yyyy} + \mathcal{O}(\Delta y^5)$$

Also using the Taylor expansions in the y-direction we can write the LTE as

$$\tau_{ij} = \frac{1}{12}h^2(u_{xxxx} + u_{yyyy}) + \mathcal{O}(h^4).$$

The linear system for the LTE then has the form

$$A_h E_h = -\tau_h$$

where now A is the discretization we wrote before. Note that the ordering of the equations does not matter when considering the error. For the stability in the 2-norm we again can consider the eigenvalues of the system above. The eigenvalues are

$$\lambda_{pq} = \frac{2}{h^2}((\cos(p\pi h) - 1) + (\cos(q\pi h) - 1))$$

with corresponding eigenvectors

$$v_{ij}^{p,q} = \sin(p\pi ih)\sin(q\pi jh).$$

Since the eigenvalues are strictly negative (A is in fact negative definite) the closest one to the origin is

$$\lambda_{11} = -2\pi^2 + \mathcal{O}(h^2)$$

leading to the spectral radius

$$\rho((A^h)^{-1}) = \frac{1}{\lambda_{11}} \approx -\frac{1}{2\pi^2}.$$

We can use this bound on A^{-1} then to show stability and hence convergence of the discretization. A similar and useful quantity to consider is the *condition number* of the matrix A. Recall that this can be defined as

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

In the 2-norm we already know some information about A^{-1} but we can use our expressions from above to also find the spectral radius of A. The largest eigenvalue there is

$$\lambda_{mm} \approx -\frac{8}{h^2}$$

leading to the condition number

$$\kappa_2(A) = \frac{4}{\pi^2 h^2} = \mathcal{O}\left(\frac{1}{h^2}\right)$$

This matrix therefore becomes more ill-conditioned as $h \to 0$.

1.4.1 Task 2 – Eigenvalues of the 5-Point Laplacian

Write a Python function to compute and plot the eigenvalues of the 5-point finite difference Laplacian on a square grid.

```python
def laplacian_eigenvalues(m):
    """
    Compute eigenvalues of the 5-point Laplacian on a square m x m grid.

    Parameters
    ----------
    m : int
        Number of interior points in each spatial direction.

    Returns
    -------
    eig_vals : ndarray
        Array of eigenvalues of size m**2.
    """
```

```python
eig_vals = laplacian_eigenvalues(m=10)
```

1.5 The 9-point Laplacian

In addition to the 5-point Laplacian, denoted in the following by $\nabla_5^2 U_{ij}$, another possible approximation is the 9-point Laplacian

$$\nabla_9^2 U_{ij} = \frac{1}{6h^2}\left[4U_{i-1,j} + 4U_{i+1,j} + 4U_{i,j-1} + 4U_{i,j+1} + U_{i-1,j-1} + U_{i-1,j+1} + U_{i+1,j-1} + U_{i+1,j+1} - 20U_{ij}\right].$$

After applying this to the true solution $u(x, y)$ and expanding it in Taylor series, we obtain

$$\nabla_9^2 u(x_i, y_j) = \nabla^2 u(x_i, y_j) + \frac{1}{12}h^2(u_{xxxx} + 2u_{xxyy} + u_{yyyy}) + O(h^4).$$

Although the approximation is $O(h^2)$ accurate, observe that

$$u_{xxxx} + 2u_{xxyy} + u_{yyyy} = \nabla^4 u = \nabla^2(\nabla^2 u) = \nabla^2 f.$$

Therefore, like in deferred corrections, it is possible to obtain a fourth-order accurate method of the form

$$\nabla_9^2 U_{ij} = F_{ij}$$

by defining

$$F_{ij} = f(x_i, y_j) + \frac{h^2}{12}\nabla^2 f(x_i, y_j)$$

for an arbitrary smooth function $f(x,y)$. If we only know data values at the grid points F_{ij}, but the underlying function $f(x,y)$ is sufficiently smooth, then we can still achieve fourth-order accuracy by using

$$F_{ij} = F_{ij} + \frac{h^2}{12}\nabla_5^2 F_{ij}.$$

We have introduced deliberately an $O(h^2)$ error into the right-hand side of the equation that is chosen to cancel the $O(h^2)$ part of the local truncation error. This type of "**tricks**" are often used in developing numerical methods.

1.6 Polar coordinates

Consider the **Laplace equation** on the **unit disk**,

$$\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial u}{\partial r}\right) + \frac{1}{r^2}\frac{\partial^2 u}{\partial \theta^2} = 0, \quad 0 < r < 1, ; 0 < \theta < 2\pi, \;\; u(r,\theta) = u(r,\theta+2\pi), \quad 0 \le r \le 1, \;\; |u(r,\theta)| < \infty, \quad \forall (r,\theta) \in$$

We can use FD approximations to solve this problem defined in polar coordinates. The approach is very similar, we just need to consider the following:

- **Step 1: Generate a grid.** For $0 < R_1 \le r \le R_2$ and $0 \le \theta \le \theta_{\max}$, where the origin is not in the domain of interest, using a uniform grid in the polar coordinates:

$$r_i = R_1 + i\Delta r, \quad i = 0,1,\dots,m+1, \quad \Delta r = \frac{R_2 - R_1}{m+1}, \;\; \theta_j \;\; = \theta_1 + j\Delta\theta, j \;\; = 0,1,\dots,n+1, \Delta\theta \;\; = \frac{\theta_{\max} - \theta_1}{n+1}.$$

- **Step 2: FD approximation of the Laplace operator.** The discretized finite difference equation (in conservative form) is

$$\frac{1}{r_i}\frac{1}{\Delta r^2}\left[r_{i-1/2}U_{i-1,j} - (r_{i-1/2} + r_{i+1/2})U_{ij} + r_{i+1/2}U_{i+1,j}\right] + \frac{1}{r_i^2}\frac{1}{\Delta\theta^2}\left[U_{i,j-1} - 2U_{ij} + U_{i,j+1}\right] = f(r_i,\theta_j),$$

where again U_{ij} is an approximation to the solution $u(r_i,\theta_j)$.

1.6.1 Treating the Polar Singularity

If the origin is within the domain and $0 \le \theta < 2\pi$, we have a periodic boundary condition in the θ direction (i.e., $u(r,\theta) = u(r,\theta+2\pi)$), but in the radial ($r$) direction the origin $R_1 = 0$ needs special attention. The PDE is singular at $r = 0$, which is called a **pole singularity**.

The main new feature of polar coordinates is the condition that must be imposed at this point. It is important to realize that any difficulties that arise at the origin are only a result of the choice of coordinate system and are not reflected in the continuous function $u(r,\theta)$.

One approach to derive a condition at the origin is to integrate $\nabla^2 u = f$ over a disk D of radius ϵ, obtaining

$$\iint_D f\, r\, dr\, d\theta = \iint_D \nabla \cdot (\nabla u)\, r\, dr\, d\theta = \int_0^{2\pi} \epsilon \frac{\partial u}{\partial r}\Big|_D \, d\theta,$$

after applying the divergence theorem. Now choose $\epsilon = \Delta r/2$ and approximate this relation by

$$f(0)\,\pi \left(\frac{\Delta r}{2}\right)^2 = \sum_{j=1}^{n} \frac{U_{1j} - U_0}{\Delta r}, \frac{\Delta r}{2}, \Delta\theta.$$

Note that since U_{0j} is independent of j, we have called this value U_0. Finally, solving for U_0 leads to the searched condition

$$U_0 = \frac{1}{n}\sum_{j=1}^{n} U_{1j} - f(0)\left(\frac{\Delta r}{2}\right)^2.$$

1.7 Exercises

Exercise 1. Consider the **9-point Laplacian approximation**.

- (a) Obtain its **local truncation error** by using Taylor expansions.
- (b) Modify your 2D Poisson solver to use the 9-point finite difference Laplacian instead of the standard 5-point stencil

---

Exercise 2. Consider the **Laplace equation**

$$\Delta u + \lambda u = 0, \quad 0 < x, y < 1, \;\; u = 0, \quad \text{on the boundaries.}$$

- (a) Show that the **eigenvalues and eigenvectors** for this problem are

$$\lambda_{k,l} = \pi^2(l^2 + k^2), \quad l, k = 1, 2, \dots,$$
$$u_{k,l}(x, y) = \sin(k\pi x)\sin(l\pi y).$$

- (b) Show that the standard central finite difference scheme using the five-point stencil is stable for the **Poisson equation**.

Hint: The eigenvectors for $\mathbf{A}^h\mathbf{U}^h = \mathbf{F}$ *(eigen-grid functions) are*

$$v_{p,q}^{ij} = \sin(p\pi h)\sin(q\pi h), \quad i, j, p, q = 1, 2, \dots, m.$$

Moreover, remember that

$$|(\mathbf{A}^h)^{-1}|_2 = \frac{1}{\min|\lambda_i(\mathbf{A}^h)|}.$$

Exercise 3. Write down the **coefficient matrix** of the finite difference method using the standard central five-point stencil for the Poisson equation defined on the rectangle $[a, b] \times [c, d]$. Take $m = n = 3$ and assume a **Dirichlet boundary condition** at $x = a$, $y = c$ and $y = d$, and a **Neumann boundary condition** $\partial u/\partial n = g(y)$ at $x = b$. Use the **ghost point method** to deal with the Neumann boundary condition.

Exercise 4. Write a solver for the **Laplace equation**

$$\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial u}{\partial r}\right) + \frac{1}{r^2}\frac{\partial^2 u}{\partial \theta^2} = 0, \quad 0 < r < 1,\, 0 < \theta < 2\pi,\, u(r,\theta) = u(r, \theta + 2\pi), \quad 0 \leq r \leq 1,\, |u(r,\theta)| < \infty, \quad \forall (r,\theta) \in \Omega,$$

As a check, consider a numerical test to analyze the **convergence**, i.e. choose $g(\theta)$ as you wish and do a **grid refinement study**.

Remember that the general solution to this problem is

$$u(r,\theta) = \sum_{n=0}^{\infty} A_n \cos(n\theta) r^n + \sum_{n=1}^{\infty} B_n \sin(n\theta) r^n.$$

```
[ ]:
```