2 2D Elliptic PDEs

2.0.1 2D PDEs

There are three classes of second order linear PDEs: elliptic, parabolic and hyperbolic. Classification is based on their characteristics, with elliptic PDEs having no real characteristics, parabolic having 1 and hyperbolic having 2. Elliptic equations require only boundary values to solve, while parabolic and hyperbolic PDEs also require initial conditions and are often called *evolution equations*.

Example 2.1.

$-\nabla^2 u = f,$	The Poisson equation, elliptic,
$\nabla^2 u = u_t,$	The diffusion equation, parabolic,
$\nabla^2 u = u_{tt},$	The wave equation, hyperbolic.

We will first consider (a subclass of) the elliptic PDEs:

$$-\nabla \cdot (D(x,y)\nabla u) + q(x,y)u = f(x,y),$$

where $D > 0$ and $q \geq 0$ to satisfy ellipticity.

The simplest example satisfying these conditions, where $q = 0$ and D is constant, is the Poisson equation, seen above.

For simple domains (e.g. rectangles) we have many choices of discretisation: finite differences, finite element methods, spectral collocation (e.g. Chebyshev), spline collocation (e.g. piecewise cubics in C^1), 'fast Poisson solvers' (only for the Poisson equation), and multipole methods (dating to about 1988). But once the domain becomes difficult, finite element methods "reign supreme".

We will also find that as the size of the linear system gets bigger, we will need to consider alternatives to "\" in MATLAB.

2.1 Elliptic PDEs: Finite Difference

Ref: Leveque Ch. 3

Consider the equation

$$-\nabla(D(x,y)\nabla u) = f(x,y), \quad x,y, \in \Omega.$$

If we have Dirichlet boundary conditions then:

$$u = g_D(x,y), \quad \text{on } \partial\Omega.$$

If we have Neumann boundary conditions then:

$$\mathbf{q}\cdot\hat{\mathbf{n}} = g_N(x,y), \quad \text{on } \partial\Omega,$$

where $\mathbf{q} = -D\nabla u$ is the flux at the boundary as $\frac{\partial u}{\partial n} \equiv \hat{\mathbf{n}}\cdot\nabla u$.

Finally Robin boundary conditions are:

$$\alpha u + \beta\mathbf{q}\cdot\hat{\mathbf{u}} = g_R, \quad \text{on } \partial\Omega.$$

2.1.1 Discretising the domain

First we need to define the mesh inside Ω. Immediately there are two questions to answer:

- How to handle the curved boundaries?
- How to define the mesh in 2D?

We will avoid these questions for the moment by considering a rectangle aligned with the axes. Then the use of a tensor product mesh $\{x_i\} \otimes \{y_i\}$, which gives a set of mesh points

$$\{x_i, y_i\}, \quad i = 0, \ldots, m+1 \quad j = 0 \ldots m+1,$$

with $\Delta x_i = x_{i+1} - x_i$, $\Delta y_j = y_{j+1} - y_j$ in general. If we have a (square) uniform mesh then $h_i = k_i = h = k$.

Let's simplify to the case $D(x, y) = 1$ and consider Dirichlet BCs.

Then there are $(m+2)^2$ mesh points, m^2 internal mesh values $u_{ij} = u(x_i, y_j)$ and $4m + 4$ boundary points (which are known if given Dirichlet boundary conditions). So we need m^2 equations for the unknowns u_{ij}.

Using central differences for $-\nabla^2 = -\delta_{xx} - \delta_{yy}$, we obtain a 5 point stencil.

Figure 5: five point stencil

$$
\begin{aligned}
-\nabla^2 u_{ij} &\approx -\frac{1}{\Delta x^2}(u_W - 2u_C + u_E) - \frac{1}{\Delta y^2}(u_N - 2u_C + u_S) \\
&\overset{\Delta x=\Delta y=h}{=} -\frac{1}{h^2}(u_W + u_E + u_S + u_N - 4u_C) + O(h^2) = f_C.
\end{aligned}
$$

Therefore for each internal mesh point (m^2 equations).

$$
-\frac{1}{h^2}(u_{i-1,j} + u_{i+1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{ij}) = f_{ij}, i, j = 1, \ldots, m.
$$

2.1.2 Ordering unknowns:

There are many possible orderings, each gives a different linear system which can affect the efficiency of the solution (when using direct methods).

We will use the natural ordering

Figure 6: natural ordering

which defines a mapping from the 2D index set i, j to the 1D index p, $p = (j-1)m + i$.

Other possible orderings are possible, for example the red/black (checkerboard) ordering.

Figure 7: Red/black ordering

Given an ordering and a five point stencil, we obtain a linear system for mesh values u_{ij}.

Example 2.2. Consider a 6×6 mesh, so $i, j = 0, \ldots, 5$. $m = 3$ so there are 9 internal mesh points and 16 boundary conditions.

Figure 8: mesh

On each internal mesh point $u_W + u_E + u_N + u_S - 4u_C = h^2 f_C$.

$$
\begin{aligned}
p = 1, (1,1) &: \frac{1}{h^2}\left(-4u_1 + u_2 + u_4 + u_{0,1} + u_{1,0}\right) = f_{1,1} \\
p = 2, (2,1) &: \frac{1}{h^2}\left(u_1 - 4u_2 + u_3 + u_5 + u_{2,0}\right) = f_{2,1} \\
p = 3, (3,1) &: \frac{1}{h^2}\left(u_2 - 4u_3 + u_6 + u_{3,0} + u_{4,1}\right) = f_{3,1} \\
p = 4, (1,2) &: \frac{1}{h^2}\left(u_1 - 4u_4 + u_5 + u_7 + u_{0,2}\right) = f_{1,2} \\
p = 5, (2,2) &: \frac{1}{h^2}\left(u_2 + u_4 - 4u_5 + u_6 + u_8\right) = f_{2,2} \\
p = 6, (3,2) &: \frac{1}{h^2}\left(u_3 + u_5 - 4u_6 + u_9 + u_{4,2}\right) = f_{3,2} \\
p = 7, (1,3) &: \frac{1}{h^2}\left(u_4 - 4u_7 + u_8 + u_{0,3} + u_{1,4}\right) = f_{1,3} \\
p = 8, (2,3) &: \frac{1}{h^2}\left(u_5 + u_7 - 4u_8 + u_9 + u_{2,4}\right) = f_{2,3} \\
p = 9, (3,3) &: \frac{1}{h^2}\left(u_6 + u_8 - 4u_9 + u_{3,4} + u_{4,3}\right) = f_{3,3}.
\end{aligned}
$$

where red denotes the known boundary conditions.

We now move the known Dirichlet data to the RHS to get

$$
\begin{aligned}
p=1,(1,1):&\frac{1}{h^2}\left(-4u_1+u_2+u_4\right)=f_{1,1}-\frac{u_{0,1}+u_{1,0}}{h^2}\\
p=2,(2,1):&\frac{1}{h^2}\left(u_1-4u_2+u_3+u_5\right)=f_{2,1}-\frac{u_{2,0}}{h^2}\\
p=3,(3,1):&\frac{1}{h^2}\left(u_2-4u_3+u_6\right)=f_{31}-\frac{u_{30}+u_{4,1}}{h^2}\\
p=4,(1,2):&\frac{1}{h^2}\left(u_1-4u_4+u_5+u_7\right)=f_{1,2}-\frac{u_{0,2}}{h^2}\\
p=5,(2,2):&\frac{1}{h^2}\left(u_2+u_4-4u_5+u_6+u_8\right)=f_{2,2}\\
p=6,(3,2):&\frac{1}{h^2}\left(u_3+u_5-4u_6+u_9\right)=f_{3,2}-\frac{u_{4,2}}{h^2}\\
p=7,(1,3):&\frac{1}{h^2}\left(u_4-4u_7+u_8\right)=f_{1,3}-\frac{u_{0,3}+u_{1,4}}{h^2}\\
p=8,(2,3):&\frac{1}{h^2}\left(u_5+u_7-4u_8+u_9\right)=f_{2,3}-\frac{u_{2,4}}{h^2}\\
p=9,(3,3):&\frac{1}{h^2}\left(u_6+u_8-4u_9\right)=f_{3,3}-\frac{u_{3,4}+u_{4,3}}{h^2}.
\end{aligned}
$$

i.e. a linear system of the form $-\frac{1}{h^2}A\mathbf{u}=\frac{1}{h^2}\mathbf{b}+\mathbf{f}$ where A is a block tridiagonal matrix (also called A_5 due to the 5 point stencil)

$$
A=-\frac{1}{h^2}\begin{pmatrix}T & I & 0\\ I & T & I\\ 0 & I & T\end{pmatrix},
$$

where I is the 3×3 identity matrix, and

$$
T=\begin{pmatrix}-4 & 1 & 0\\ 1 & -4 & 1\\ 0 & 1 & -4\end{pmatrix},
$$

is a 3×3 tridiagonal matrix.

$$
\mathbf{b}=\begin{pmatrix}u_{0,1}+u_{1,0}\\ u_{2,0}\\ \vdots\\ \vdots\\ u_{3,4}+u_{4,3}\end{pmatrix},
$$

and $\mathbf{f}$ is as given by the RHS of the PDE evaluated at the mesh points $f_p = f(x_i, y_j)$.

A is symmetric, block tridiagonal, banded with band-width m, and sparse. (There are 74 non zero entries in the 256 entries of the matrix.) In general the sparsity goes to $\frac{5}{N}$ as $N \to \infty$ where $N = m^2$.

So as we can see, we would like to use sparse matrix storage to only store the non-zero entries of the matrix. Otherwise if we have a 1000×1000 mesh, then $N = m^2 = 10^6$ and $A = (10^6)^2 = 10^{12}$ entries, which is about 8 terabytes!. Type in `help sparse` in MATLAB to see uses of sparse storage in `speye` and `spdiags`. For the moment we will assume we'll solve this system using \ which uses direct methods for sparse matrices, but later we'll discuss alternatives.

2.1.3 Error Analysis

As before, the local truncation error is the residual when the exact solution is put into the finite difference formula.

For a five point stencil, if we do the algebra (as in the 1D case) we get

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

so the method is consistent of order 2.

Again, as in 1 dimension, the error solves the equation $A\mathbf{E} = -\tau$ so if we can bound A^{-1} in some norm we have a stability result.

Example 2.3. Leveque §3.4 proves $\|A^{-1}\|_2 < \frac{1}{2\pi^2}$. He does this by knowing the eigenvalues of A since if A is symmetric $\|A\|_2 = \max|\lambda_i|$. This stability bound proves the method is stable, so the five point stencil method for the Poisson equation is convergent of order 2.

How convergence depends on the smoothness of f and the boundary conditions is 'difficult' and beyond the scope of the course (see Iserles for details).

2.1.4 2D Example

Example 2.4. Run the example `delsquare`. Note this is just a copy of an inbuilt MATLAB example, see `showdemo delsqdemo`. Experiment with different domain shapes, how does the node ordering affect the structure of the linear system?

This code solves the problem

$$\nabla^2 = -1,$$

on various domains with $u = 0$ on the boundaries. It uses the following commands: `numgrid, numgrid(R,n), delsq, nnz, spy`. Look up what they all do.

Note: the commands

```
A5 = gallery('poisson', n)
```

and

```
A5 = delsq(numgrid('S',n))
```

produce the same matrix (check this for yourselves!)

Recap of FD Methods

- Easy for simple geometry e.g. rectangles.
- Convergence theory can be tricky.
- Produces large sparse matrices, even in 2D.

2.1.5 Other topics

- There are many nice properties of the finite difference method that rely on the particular self-adjoint elliptic form (e.g. A is symmetric positive definite).

  Once we add the extra convection term, $\propto \nabla u$, we lose lots of nice properties and things become harder to prove, and it is also harder to converge fast.

- Curved Regions also make finite difference methods messy, but possible.

  This has recently become more fashionable e.g. 'Immersed Interface Method' Ref: Li & Leveque 1994.

- Nonlinear elliptic PDEs can be tackled by quasi-linearisation or by Newton's method and its variants (these are used in the PDE toolbox).

- For easy problems ($-\nabla^2 u = f$) on a square there are special 'fast Poisson solvers' that don't generalize to the variable coefficient case, or other coordinate systems.

- Higher order methods:

  1. For $-\nabla^2 u = f$, there is a 4th order method (Ref: Collatz) that is not too difficult.
     At the internal mesh points apply a 9-pt stencil:

$$\begin{aligned}\nabla_9^2 u_{ij} =& \frac{1}{6h^2}[4u_{i-1,j} + 4u_{i+1,j} + 4u_{i,j-1} + 4u_{i,j+1} + \\ & u_{i-1,j-1} + u_{i-1,j+1} + u_{i+1,j-1} + u_{i+1,j+1} - 20u_{i,j}].\end{aligned}$$

     Then (it can be shown that)

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

     which doesn't seem to help.
     But then

$$\begin{aligned}\tau_{ij} =& \frac{1}{12}h^2\nabla^2(\nabla^2 u) + O(h^4) \\ =& -\frac{1}{12}h^2\nabla^2 f + O(h^4).\end{aligned}$$

     Therefore, if we know f analytically, we can evaluate the first term of τ_{ij}.

Figure 9: 9 pt stencil

Example 2.5. For the special case $f = 0$ then error is $O(h^4)$, $\nabla_9^2 u = 0 \to A_9\mathbf{u} = \mathbf{b}$, $\tau = O(h^4)$.

Back to Poisson's equation: since

$$-\nabla_9^2 u = f + \frac{1}{12}h^2\nabla^2 f + O(h^4),$$

then if we solve the *different problem*:

$$-\nabla_9^2 u = f + \frac{1}{12}h^2\nabla^2 f = \tilde{f},$$

then

$$-\nabla^2 u + \frac{1}{12}h^2\nabla^2\tilde{f} + O(h^4) = \tilde{f},$$

therefore

$$-\nabla^2 u + \frac{1}{12}h^2\nabla^2 f + O(h^4) = f + \frac{1}{12}h^2\nabla^2 f,$$

so

$$-\nabla^2 u + O(h^4) = f,$$

i.e. we've solved $-\nabla^2 u = f$ to $O(h^4)$.

This also works even if we don't know f analytically, but just at grid points. Just form

$$\begin{aligned}\tilde{f}_{ij} =& f_{ij} + \frac{1}{12}h^2\nabla_5^2 f_{ij} \\ =& f_{ij} + \frac{1}{12}h^2(\nabla^2 f + O(h^2)) \\ =& f_{ij} + \frac{1}{12}h^2\nabla^2 f_{ij} + O(h^4),\end{aligned}$$

so we still get a 4th order method.

2. **Method of deferred corrections**

   This is a more general method that can be applied to more equations. (See Ref: Ascher, Mattheij, Russel)

   Solve $A_5\hat{\mathbf{u}} = \mathbf{f}$ which gives $\mathbf{u}$ with an error $O(h^2)$. We know

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

   Then, since $A_5\mathbf{u} - \mathbf{f} = \tau$ and $A_5\hat{\mathbf{u}} - \mathbf{f} = \mathbf{0}$,

$$A_5(\hat{\mathbf{u}} - \mathbf{u}) = -\tau,$$

   so

$$A_5\mathbf{E} = -\tau.$$

   We now attempt to use $\hat{\mathbf{u}}$ to estimate τ. For example we can use central differences to get u_{xxxx} and u_{yyyy} to $O(h^2)$. Then we obtain

$$\hat{\tau} = \tau + O(h^4),$$

   and solve

$$A_5\mathbf{E} = -\hat{\tau},$$

   then update

$$\tilde{\mathbf{u}} = \hat{\mathbf{u}} - \mathbf{E},$$

   which has improved our estimate to $O(h^4)$.