DISCRETIZATION OF PARTIAL DIFFERENTIAL EQUATIONS

Goal: to show how partial differential lead to sparse linear systems

• See Chap. 2 of text
• Finite difference methods
• Finite elements
• Assembled and unassembled finite element matrices
Why study discretized PDEs?

- Still the most important source of sparse linear systems
- Will help understand the structures of the problem and their connections with “meshes” in 2-D or 3-D space
- Also: iterative methods are often formulated for the PDE directly – instead of a discretized (sparse) system.
A typical numerical simulation

- Physical Problem
  - Nonlinear PDEs
    - Discretization
      - Linearization (Newton)
        - Sequence of Sparse Linear Systems \( Ax = b \)
Example: discretized Poisson equation

- Common Partial Differential Equation (PDE):

\[
\frac{\partial^2 u}{\partial x_1^2} + \frac{\partial^2 u}{\partial x_2^2} = f, \text{ for } x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \text{ in } \Omega
\]

where \( \Omega = \) bounded, open domain in \( \mathbb{R}^2 \)

+ boundary conditions:

- Dirichlet: \( u(x) = \phi(x) \)
- Neumann: \( \frac{\partial u}{\partial \vec{n}}(x) = 0 \)
- Cauchy: \( \frac{\partial u}{\partial \vec{n}} + \alpha(x)u = \gamma \)
\[ \Delta = \frac{\partial^2}{\partial x_1^2} + \frac{\partial^2}{\partial x_2^2} \] is the Laplace operator or Laplacean

How to approximate the problem?

Answer: discretize, i.e., replace continuum with discrete set.

Then approximate Laplacean using this discretization

Many types of discretizations. Will briefly cover Finite differences and finite elements.

**Finite Differences: Basic approximations**

Formulas derived from Taylor series expansion:

\[ u(x + h) = u(x) + h \frac{du}{dx} + \frac{h^2 d^2u}{2 dx^2} + \frac{h^3 d^3u}{6 dx^3} + \frac{h^4 d^4u}{24 dx^4}(\xi) \]
Discretization of PDEs - Basic approximations

Simplest scheme: forward difference

\[
\frac{du}{dx} = \frac{u(x + h) - u(x)}{h} - \frac{hd^2u(x)}{2} + O(h^2)
\]

≈ \frac{u(x + h) - u(x)}{h}

Centered differences for second derivative:

\[
\frac{d^2u(x)}{dx^2} = \frac{u(x + h) - 2u(x) + u(x - h)}{h^2} - \frac{h^2d^4u(\xi)}{12} dx^4,
\]

where \(\xi_- \leq \xi \leq \xi_+\).
Notation:

\[ \delta^+ u(x) = u(x + h) - u(x) \]
\[ \delta^- u(x) = u(x) - u(x - h) \]

Operations of the type:

\[ \frac{d}{dx} \left[ a(x) \frac{d}{dx} \right] \]

are very common [in-homogeneous media].

The following is a second order approximation:

\[ \frac{d}{dx} \left[ a(x) \frac{du}{dx} \right] = \frac{1}{h^2} \delta^+ \left( a_{i - \frac{1}{2}} \delta^- u \right) + O(h^2) \]
\[ \approx \frac{a_{i + \frac{1}{2}}(u_{i+1} - u_i) - a_{i - \frac{1}{2}}(u_i - u_{i-1})}{h^2} \]

Show that \( \delta^+ \left( a_{i - \frac{1}{2}} \delta^- u \right) = \delta^- \left( a_{i + \frac{1}{2}} \delta^+ u \right) \)
Finite Differences for 2-D Problems

Consider the simple problem,

\[- \left( \frac{\partial^2 u}{\partial x_1^2} + \frac{\partial^2 u}{\partial x_2^2} \right) = f \quad \text{in } \Omega \]

\[u = 0 \quad \text{on } \Gamma \]

\(\Omega = \text{rectangle } (0, l_1) \times (0, l_2) \) and \(\Gamma\) its boundary.

Discretize uniformly:

\[x_{1,i} = i \times h_1 \quad i = 0, \ldots, n_1 + 1 \quad h_1 = \frac{l_1}{n_1 + 1} \]

\[x_{2,j} = j \times h_2 \quad j = 0, \ldots, n_2 + 1 \quad h_2 = \frac{l_2}{n_2 + 1} \]
Finite Difference Scheme for the Laplacean

- Using centered differences for both the $\frac{\partial^2}{\partial x_1^2}$ and $\frac{\partial^2}{\partial x_2^2}$ terms - with mesh sizes $h_1 = h_2 = h$:

$$\Delta u(x) \approx \frac{1}{h^2} \left[ u(x_1 + h, x_2) + u(x_1 - h, x_2) + u(x_1, x_2 + h) + u(x_1, x_2 - h) - 4u(x_1, x_2) \right]$$

The 5-point ‘stencil:’

```
\begin{array}{c}
1 \\
\end{array}
```

```
\begin{array}{c}
1 \\
\end{array}
```

```
\begin{array}{c}
1 \\
\end{array}
```

```
\begin{array}{c}
1 \\
\end{array}
```

```
\begin{array}{c}
1 \\
\end{array}
```

\begin{equation}
\Delta u(x) \approx \frac{1}{h^2} \left[ u(x_1 + h, x_2) + u(x_1 - h, x_2) + u(x_1, x_2 + h) + u(x_1, x_2 - h) - 4u(x_1, x_2) \right]
\end{equation}
The resulting matrix has the following block structure:

\[
A = \frac{1}{h^2} \begin{bmatrix}
    B & -I \\
    -I & B \\
    -I & -I
\end{bmatrix}
\]

With

\[
B = \begin{bmatrix}
    4 & -1 & & & \\
    -1 & 4 & -1 & & \\
    -1 & -1 & 4 & -1 & \\
    -1 & -1 & -1 & 4 & -1 \\
    -1 & -1 & -1 & -1 & 4
\end{bmatrix}.
\]
**Finite Elements: a quick overview**

**Background:** Green's formula

\[
\int_{\Omega} \nabla v \cdot \nabla u \, dx = - \int_{\Omega} v \Delta u \, dx + \int_{\Gamma} v \frac{\partial u}{\partial \vec{n}} \, ds.
\]

- \( \nabla = \text{gradient operator}. \) In 2-D:
  \[
  \nabla u = \begin{pmatrix} \frac{\partial u}{\partial x_1} \\ \frac{\partial u}{\partial x_2} \end{pmatrix},
  \]

- The dot indicates a dot product of two vectors.
- \( \Delta u = \text{Laplacean of } u \)
- \( \vec{n} \) is the unit vector that is normal to \( \Gamma \) and directed outwards.
Frechet derivative:

\[
\frac{\partial u}{\partial \vec{v}}(x) = \lim_{h \to 0} \frac{u(x + h\vec{v}) - u(x)}{h}
\]

Green’s formula generalizes the usual formula for integration by parts

Define

\[
a(u, v) \equiv \int_{\Omega} \nabla u \cdot \nabla v \, dx = \int_{\Omega} \left( \frac{\partial u}{\partial x_1} \frac{\partial v}{\partial x_1} + \frac{\partial u}{\partial x_2} \frac{\partial v}{\partial x_2} \right) \, dx
\]

\[
(f, v) \equiv \int_{\Omega} f v \, dx.
\]

Denote:

\[
(u, v) = \int_{\Omega} u(x)v(x) \, dx,
\]
With Dirichlet BC, the integral on the boundary in Green's formula vanishes →

$$a(u, v) = - (\Delta u, v).$$

Weak formulation of the original problem: select a subspace of reference $V$ of $L^2$ and then solve

Find $u \in V$ such that $a(u, v) = (f, v)$, $\forall v \in V$

Finite Element method solves this weak problem...

... by discretization
The original domain is approximated by the union $\Omega_h$ of $m$ triangles $K_i$.

**Triangulation of $\Omega$**:

$$\Omega_h = \bigcup_{i=1}^{m} K_i.$$ 

Some restrictions on angles, edges, etc..

$$V_h = \{ \phi \mid \phi|_{\Omega_h} \in C^0, \quad \phi|_{\Gamma_h} = 0, \quad \phi|_{K_j} \text{ linear } \forall \ j \}$$

- $C^0 = \text{set of continuous functions}$
- $\phi|_X = \text{restriction of } \phi \text{ to the subset } X$
- Let $x_j, j = 1, \ldots, n$, be the nodes of the triangulation
Can define a (unique) 'hat' function $\phi_j$ in $V_h$ associated with each $x_j$ s.t.:

$$\phi_j(x_i) = \delta_{ij} = \begin{cases} 1 & \text{if } x_i = x_j \\ 0 & \text{if } x_i \neq x_j \end{cases}.$$  

Each function $u$ of $V_h$ can be expressed as

$$u(x) = \sum_{i=1}^{n} \xi_i \phi_i(x). \quad (*)$$

The finite element approximation consists of writing the Galerkin condition for functions in $V_h$:

Find $u \in V_h$ such that $a(u, v) = (f, v), \ \forall \ v \in V_h$

Express $u$ in the basis $\{\phi_i\}$ (see $*$), then substitute above
Result: the linear system

\[ \sum_{j=1}^{n} \alpha_{ij} \xi_i = \beta_i \]

where

\[ \alpha_{ij} = a(\phi_i, \phi_j), \quad \beta_i = (f, \phi_i). \]

The above equations form a linear system of equations

\[ Ax = b \]

\[ A \text{ is Symmetric Positive Definite} \]

Prove it
The Assembly Process: Illustration

If triangle $K \notin$ support domains of both $\phi_i$ and $\phi_j$ then $a_K(\phi_i, \phi_j) = 0$

If triangle $K \in$ *both* nonzero domains of $\phi_i$ and $\phi_j$ then $a_K(\phi_i, \phi_j) \neq 0$

So: $a_K(\phi_i, \phi_j) \neq 0$ iff $i \in \{k, l, m\}$ and $j \in \{k, l, m\}$. 

2-17 Chap 2 – discr
The Assembly Process

A simple finite element mesh and the pattern of the corresponding assembled matrix.
Element matrices $A^{[e]}$, $e = 1, \ldots, 4$ for FEM mesh shown above

- Each element contributes a $3 \times 3$ submatrix $A^{[e]}$ (spread out)
- Can use the matrix in un-assembled form - To multiply a vector by $A$ for example we can do

$$y = Ax = \sum_{e=1}^{nel} A^{[e]}x = \sum_{e=1}^{nel} P_e A_{Ke} (P_e^T x).$$
Can be computed using the element matrices $A_{Ke}$ - no need to assemble

The product $P^T_e x$ gathers $x$ data associated with the $e$-element into a 3-vector consistent with the ordering of the matrix $A_{Ke}$.

Advantage: some simplification in process

Disadvantage: cost (memory + computations).
Resources: A few matlab scripts

- These (and others) will be posted in the matlab folder of class web-site

```matlab
>> help fd3d
    function A = fd3d(nx,ny,nz,alpx,alpy,alpz,dshift)
    NOTE nx and ny must be > 1 -- nz can be == 1. 5- or 7-point block-Diffusion/conv. matrix. with

- A stripped-down version is `lap2D(nx,ny)`

```matlab
>> help mark
    [A] = mark(m)
    generates a Markov chain matrix for a random walk on a triangular grid. A is sparse of size n=m*(m+1)/2
```
Explore A few useful matlab functions

* kron

* gplot for plotting graphs

* reshape for going from say 1-D to 2-D or 3-D arrays

Write a script to generate a 9-point discretization of the Laplacean.
The PDE toolbox provides functions for setting up and solving a PDE of the form

\[ m \frac{\partial^2 u}{\partial t^2} + d \frac{\partial u}{\partial t} - \Delta(c \nabla u) + au = f \]

- `model=createmodel()`. Initiates the class 'model'
- `geometryFromEdges(model,...)` Creates the geometry.
- `pdegplot(model,...)` plots the geometry
- `applyBoundaryCondition(model,...)` Applies boundary conditions
- `specifyCoefficients(model,...)` Sets coeff.s \( m, d, c, a, f \) above
• `generateMesh(model,...)` Generates the mesh
• `results = pdesolve(model,...)` solves the PDE
• `pdeplot(model,...)` plots solution

➤ Also `assembleFEMatrices(model,...)` assembles the FEM problem, [returns $K$ and $M$ in a structure]

❗ Follow the example in the documentation and get an understanding of the functions that are called.