CS 450: Numerical Analysis\(^1\)
Partial Differential Equations

University of Illinois at Urbana-Champaign

\(^1\)These slides have been drafted by Edgar Solomonik as lecture templates and supplementary material for the book “Scientific Computing: An Introductory Survey” by Michael T. Heath (slides).
Partial Differential Equations

- *Partial differential equations (PDEs)* describe physical laws and other continuous phenomena:

\[
\frac{\partial u}{\partial t} = -a(t, x)\frac{\partial u}{\partial x}
\]

where \( \frac{\partial u}{\partial t} = \partial u/\partial t \) and \( \frac{\partial u}{\partial x} = \partial u/\partial x \).
Types of PDEs

- Some of the most important PDEs are second order:

  - The discriminant determines the canonical form of second-order PDEs:
Characteristic Curves

A characteristic of a PDE is a level curve in the solution:

More generally, characteristic curves describe curves in the solution field $u(t, x)$ that correspond to solutions of ODEs, e.g. for $u_t = -a(t, x)u_x$ with $u(0, x) = u_0(x)$,
Method of Lines

Semidiscrete methods obtain an approximation to the PDE by solving a system of ODEs. Consider the heat equation,

\[ u_t = cu_{xx} \text{ on } 0 \leq x \leq 1, \quad u(0, x) = f(x), u(t, 0) = u(t, 1) = 0. \]

This method of lines often yields a stiff ODE:
Semidiscrete Collocation

- Instead of finite-differences, we can express $u(t, x)$ in a spatial basis $\phi_1(x), \ldots, \phi_n(x)$ with time-dependent coefficients $\alpha_1(t), \ldots, \alpha_n(t)$:

- For the heat equation $u_t = cu_{xx}$, we obtain a linear constant-coefficient vector ODE:
Fully Discrete Methods

- Generally, both time and space dimensions are discretized, either by applying an ODE solver to a semidiscrete method or using finite differences.
  - *Again consider the heat equation* $u_t = cu_{xx}$ *and discretize so* $u_i^{(k)} \approx u(t_k, x_i)$,

  *This iterative scheme corresponds to a 3-point stencil,*
Implicit Fully Discrete Methods

- Using Euler’s method for the heat equation, stability requirement is

- This step-size restriction on stability can be circumvented by use of implicit time-stepper, such as backward Euler,

- Using the trapezoid method to solve the ODE we obtain the second-order Crank-Nicolson method,
Convergence and Stability

- **Lax Equivalence Theorem**: consistency + stability = convergence
  - Consistency means that the local truncation error goes to zero, and is easy to verify by Taylor expansions.
  - Stability implies that the approximate solution at any time $t$ must remain bounded.
  - Together these conditions are necessary and sufficient for convergence.

- Stability can be ascertained by spectral or Fourier analysis:
  - In the method of lines, we saw that the eigenvalues of the resulting ODE define the stability region.
  - Fourier analysis decomposes the solution into a sum of harmonic functions and bounds their amplitudes.
CFL Condition

- The domain of dependence of a PDE for a given point \((t, x)\) is the portion of the problem domain influencing this point through the PDE:

- *The Courant, Friedrichs, and Lewy (CFL) condition* states that for an explicit finite-differencing scheme to be stable for a hyperbolic PDE, it is necessary that the domain of the dependence of the PDE must be contained in the domain of dependence of the scheme:
We now turn our focus to time-independent PDEs as exemplified by the **Helmholtz equation**:

\[ u_{xx} + u_{yy} + \lambda u = f(x, y) \]

We discretize as before, but no longer perform time stepping:
Finite-Differencing for Poisson

Consider the Poisson equation with equispaced mesh-points on $[0, 1]$:
Multidimensional Finite Elements

- There are many ways to define localized basis functions, for example in the 2D FEM method:

Source: Comsol Multiphysics Cyclopedia

Sparse Linear Systems

- Finite-difference and finite-element methods for time-independent PDEs give rise to sparse linear systems:
  - typified by the 2D Laplace equation, where for both finite differences and FEM,

- Direct methods apply LU or other factorization to $A$, while iterative methods refine $x$ by minimizing $r = Ax - b$, e.g. via Krylov subspace methods.
Direct Methods for Sparse Linear Systems

- It helps to think of $A$ as the adjacency matrix of graph $G = (V, E)$ where $V = \{1, \ldots, n\}$ and $a_{ij} \neq 0$ if and only if $(i, j) \in E$:

- Factorizing the $l$th row/column in Gaussian elimination corresponds to removing node $i$, with nonzeros (new edges) introduces for each $k, l$ such that $(i, k)$ and $(i, l)$ are in the graph.
Vertex Orderings for Direct Methods

- Select the node of minimum degree at each step of factorization:

- Graph partitioning also serves to bound fill, remove vertex separator \( S \subset V \) so that \( V \setminus S = V_1 \cup \cdots \cup V_k \) become disconnected, then order \( V_1, \ldots, V_k, S \):

- Nested dissection ordering partitions graph into halves recursively, ordering each separator last.
Sparse Iterative Methods

- Sparse iterative methods avoid overhead of fill in sparse direct factorization. *Matrix splitting* methods provide the most basic iterative methods:
Sparse Iterative Methods

- The *Jacobi method* is the simplest iterative solver:

- The Jacobi method converges if $A$ is strictly row-diagonally-dominant:
Gauss-Seidel Method

▶ The Jacobi method takes weighted sums of $x^{(k)}$ to produce each entry of $x^{(k+1)}$, while Gauss-Seidel uses the latest available values, i.e. to compute $x^{(k+1)}_i$ it uses a weighted sum of

$$x^{(k+1)}_1, \ldots, x^{(k+1)}_{i-1}, x^{(k)}_i, \ldots, x^{(k)}_n.$$ 

▶ Gauss-Seidel provides somewhat better convergence than Jacobi:
Successive Over-Relaxation

The *successive over-relaxation* (SOR) method seeks to improve the spectral radius achieved by Gauss-Seidel, by choosing

\[ M = \frac{1}{\omega} D + L, \quad N = \left( \frac{1}{\omega} - 1 \right) D - U \]

The parameter \( \omega \) in SOR controls the ‘step-size’ of the iterative method:
Conjugate Gradient

- The solution to $Ax = b$ when $A$ is symmetric positive definite is the minima of the quadratic optimization problem,

\[ \min_x x^T Ax - x^T b \]

- Conjugate gradient works by picking $A$-orthogonal descent directions

- The convergence rate of CG is linear with coefficient $\frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1}$:
Preconditioning techniques choose matrix $M \approx A$ that is easy to invert and solve a modified linear system with an equivalent solution to $Ax = b$,

$$M^{-1}Ax = M^{-1}b$$

$M$ is chosen to be an effective approximation to $A$ with a simple structure: