Lecture 20: Initial Boundary Value Problems

Today:

• *Finite difference methods* applied to *initial boundary value problems* (IBVPs)
  • *Punchline*: approximate solution to an *IBVP* by combining our FD methods for BVPs and IVPs
Where are we up to now?

Last several weeks.
(A) We developed finite difference methods (one-step and multi-step) methods for solving initial value problems.
(B) We developed a suite of methods (finite difference, global spectral, and finite element) for boundary value problems.
(C) We learned how to characterize the error for these methods and how to implement them.

This week. Finite difference methods for IBVPs — PDEs that depend on time and space
• Use an FD method to discretize the PDE in space → gives an IVP
• Solve the resulting IBVP with an FD method

This approach is called the method of lines.
This discretization approach is called the **method of lines**

Discretizing the IBVP in space to get an IVP, and then solving that using your favorite time stepping method is called the **method of lines**

Why? Schematically:

After discretizing in space, the solution at each spatial point evolves “along its own line” in time
Reminder: what is an IBVP?

Before talking about solving IBVPs, let’s remind ourselves what an IBVP is.

An IBVP is a *partial differential equation* that depends on *space and time*. Most of the key engineering phenomena we care about are governed by an IBVP:

- The Navier-Stokes equations
- Governing equations for structural deformation (Euler Bernoulli beams, plates & shells, etc.)
- ...

Solving these equations numerically is an active area of research, so we will focus on a canonical equation. The building blocks we develop here form the heart of methods developed for these more complex systems!

The canonical IBVP we will consider is the *heat equation*

\[
\frac{\partial u}{\partial t} = \kappa \frac{\partial^2 u}{\partial x^2} + g(x, t), \quad 0 \leq t \leq T, \quad a \leq x \leq b
\]

where \(\kappa\) is the heat diffusivity and \(g(x, t)\) is a prescribed forcing.

We need both *boundary conditions* and *initial conditions* to specify the solution

- \(u(x, t = 0) = \eta(x)\) (Prescribed initial condition; \(\eta(x)\) is given)
- \(u(x = a, t) = g_a(t)\) (Prescribed boundary conditions; \(g_a(t)\) and \(g_b(t)\) are given)
- \(u(x = b, t) = g_b(t)\)

These are Dirichlet boundary conditions. Other options (Neumann, mixed) are possible!
Discretizing the IBVP in space

Just as for BVPs, we will use a uniformly distributed set of points:

\[ x_j = a + \frac{(b - a)(j - 1)}{n}, \quad j = 1, \ldots, n + 1 \]

Approximate the solution as a linear combination of locally defined basis functions

\[ u(x, t) \approx \sum_{i=j-p/2}^{j+p/2} u_i(t) L_i^{(j)}(x) \]  
\[ L_k^{(j)}(x) = \prod_{m=j-p/2}^{j+p/2} \frac{x - x_m}{x_k - x_m} \quad k = j - p/2, \ldots, j, \ldots, j + p/2. \]

Plug the approximation (2) into the PDE (1), and evaluate at \( x_j \) to get

\[ \sum_{i=j-p/2}^{j+p/2} \dot{u}_i(t) L_i^{(j)}(x_j) = \kappa \sum_{i=j-p/2}^{j+p/2} u_i(t) \left. \frac{d^2 L_i^{(j)}}{dx^2} \right|_{x=x_j} + g(x_j, t) \quad (j = 2, \ldots, n) \]

Remember that \( L_i(x_j) = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases} \)

We can say more about this when \( p = 2 \ldots \)
Simplifying the \textit{spatially discrete} PDE for $p = 2$

\[ \implies \dot{u}_j(t) = \kappa \sum_{i=j-p/2}^{i+p/2} u_i(t) \frac{d^2 L_i}{dx^2} \bigg|_{x=x_j} + g(x_j, t) \quad (j = 2, \ldots, n) \]

We showed in Lecture 14, slide 10 that this term is
\[ \frac{1}{\Delta x^2}(u_{j-1}(t) - 2u_j(t) + u_{j+1}(t)) \]

This gives an ODE in time for each $j = 2, \ldots, n$. Can aggregate into a linear system of IVPs:

\[
\begin{bmatrix}
    \ddot{u}_2 \\
    \ddot{u}_3 \\
    \vdots \\
    \ddot{u}_{n-1} \\
    \ddot{u}_n
\end{bmatrix}
= \frac{\kappa}{\Delta x^2}
\begin{bmatrix}
    -2 & 1 & 0 & \cdots & 0 \\
    1 & -2 & 1 & \cdots & 0 \\
    \vdots & \vdots & \ddots & \ddots & \vdots \\
    0 & 0 & \cdots & -2 & 1 \\
    1 & -2 & 1
\end{bmatrix}
\begin{bmatrix}
    u_2 \\
    u_3 \\
    \vdots \\
    u_{n-1} \\
    u_n
\end{bmatrix}
+ \begin{bmatrix}
    g(x_2, t) + \frac{\kappa g_{u_2}(t)}{\Delta x^2} \\
    g(x_3, t) \\
    \vdots \\
    g(x_{n-1}, t) \\
    g(x_n, t) + \frac{\kappa g_{u_n}(t)}{\Delta x^2}
\end{bmatrix}
\]

These two rows are different because of the BCs. Take the first row:

\[ \dot{u}_2 = \frac{\kappa}{\Delta x^2}(u_1 - 2u_2 + u_3) + g(x_2, t) \]

Using the BC, $u_1 = u(x_1, t) = g_a(t)$

Similar arguments apply to the last row.
Spatially discretizing leads to an IVP

Can write equation (3) more succinctly as \( \dot{u} = Au + g \)

But this is just an IVP in the desired first order form, with \( f(u, t) = Au + g(t) \)

Solve using your favorite time stepping method (backward Euler, trapezoid method, ...)

To implement, we need to construct an initial condition for \( u \)

Remember from slide 3 that the IC for the IBVP is \( u(x, t = 0) = \eta(x) \), where \( \eta \) is a function given to us.

Then our IC for the spatially discrete \( u \) can be constructed by evaluating at the discretization points \( x_2, \ldots, x_n \):

\[
   u(t = 0) = \begin{bmatrix}
   \eta(x_2) \\
   \eta(x_3) \\
   \vdots \\
   \eta(x_{n-1}) \\
   \eta(x_n)
   \end{bmatrix}
\]
This discretization approach is called the **method of lines**

Discretizing the IBVP in space to get an IVP, and then solving that using your favorite time stepping method is called the **method of lines**

Why? Schematically:

After discretizing in space, the solution at each spatial point evolves “along its own line” in time