Computational Science:
Computational Methods in Engineering

Time-Domain Finite-Difference Method

Outline

• Introduction
• Time-Domain Solution of the Heat Equation
• Time-Domain Solution of Maxwell’s Equations
Introduction

Basic Approach

**Formulation**
1. Identify governing equation
2. Approximate equation using finite-differences
3. Solve finite-difference equation for the function at the future time-value
4. Collect constants into update coefficients.
5. Write final update equation.

**Implementation**
1. Dashboard
2. Calculate grid
3. Build device on grid
4. Calculate update coefficients
5. Initialize unknown function(s)
6. Main loop – iterate over time
7. Post process data
8. Visualize data
Notes

• Time-domain FDM does not require linear algebra.
• Scales nearly linearly
• Excellent technique for very large scale simulations
• Excellent for simulating transient response
• Excellent for incorporating nonlinear devices and materials
• Excellent technique for visualizing motion and waves
• Excellent technique for learning about devices

Transient Vs. Steady-State

STEADY-STATE RESPONSE

TRANSIENT RESPONSE
Time-Domain Solution of the Heat Equation

Define Problem

A long bar of length $L$ is a temperature $0 \degree C$. At time $t = 0$, a $350 \degree C$ heat source is applied to the far end of the bar while the other end is held at $0 \degree C$. Calculate and show how the temperature evolves with time over the length of the bar if the thermal diffusivity $\alpha(x)$ is

$$\alpha(x) = 25 + 50 \sin \left( \frac{\pi x}{L} \right) \quad 0 \leq x \leq L$$
The governing equation is the time-domain form of the heat equation.

\[
\frac{\partial T}{\partial t} - \alpha \nabla^2 T = 0
\]

\(T(x)\) \(\equiv\) temperature
\(\alpha(x)\) \(\equiv\) thermal diffusivity
\(t\) \(\equiv\) time
\(\nabla\) \(\equiv\) del operator

<table>
<thead>
<tr>
<th>Material</th>
<th>(\alpha) (m(^2)/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metal</td>
<td>(10^4)</td>
</tr>
<tr>
<td>Water</td>
<td>(10^7)</td>
</tr>
<tr>
<td>Air</td>
<td>(10^5)</td>
</tr>
<tr>
<td>Nylon</td>
<td>(10^7)</td>
</tr>
<tr>
<td>Wood</td>
<td>(10^7)</td>
</tr>
</tbody>
</table>

Reduce to 1D

\[
\frac{\partial T}{\partial t} - \alpha \nabla^2 T = 0
\]

\[
\frac{\partial T(x,t)}{\partial t} - \alpha(x) \left[ \frac{\partial^2 T(x,t)}{\partial x^2} + \frac{\partial^2 T(x,t)}{\partial y^2} + \frac{\partial^2 T(x,t)}{\partial z^2} \right] = 0
\]

\[
\frac{\partial T(x,t)}{\partial t} - \alpha(x) \frac{\partial^2 T(x,t)}{\partial x^2} = 0
\]
Approximate with Finite-Differences

The governing equation is
\[ \frac{\partial T(x,t)}{\partial t} - \alpha(x) \frac{\partial^2 T(x,t)}{\partial x^2} = 0 \]

Make a guess at the finite-difference approximation
\[ \frac{T(x,t + \Delta t) - T(x,t)}{\Delta t} - \alpha(x) \frac{T(x + \Delta x, t) - 2T(x,t) + T(x - \Delta x, t)}{(\Delta x)^2} = 0 \]

Not all finite-difference approximations in this equation exist at the same instant in time.

Fixing the Finite-Difference Equation (1 of 2)

Solution 1 – Interpret the time finite-difference as a backward finite-difference.
\[ \frac{T(x,t + \Delta t) - T(x,t)}{\Delta t} - \alpha(x) \frac{T(x + \Delta x, t) - 2T(x,t) + T(x - \Delta x, t)}{(\Delta x)^2} = 0 \]

Solution 2 – Interpret the time finite-difference as a forward finite-difference.
\[ \frac{T(x,t) - T(x,t - \Delta t)}{\Delta t} - \alpha(x) \frac{T(x + \Delta x, t) - 2T(x,t) + T(x - \Delta x, t)}{(\Delta x)^2} = 0 \]
Fixing the Finite-Difference Equation (2 of 2)

Solution 3 – Interpolate the second term in the equation to exist at \( t + \Delta t/2 \).

\[
\frac{T(x, t + \Delta t) - T(x, t)}{\Delta t} = \alpha(x) \frac{T(x + \Delta x, t + \Delta t) - 2T(x, t + \Delta t) + T(x - \Delta x, t + \Delta t)}{(\Delta x)^2} + \alpha(x) \frac{T(x + \Delta x, t) - 2T(x, t) + T(x - \Delta x, t)}{(\Delta x)^2} = 0
\]

This is called the Crank-Nicholson scheme.

Very often, Solution 1 or 2 can be used successfully, but the Crank-Nicholson scheme is unconditionally stable.

Solve for Temperature at Future Step

Proceed with Solution 1 because it is the simplest, but not necessarily the most accurate or stable.

\[
\frac{T(x, t + \Delta t) - T(x, t)}{\Delta t} = \alpha(x) \frac{T(x + \Delta x, t) - 2T(x, t) + T(x - \Delta x, t)}{(\Delta x)^2} = 0
\]

Write this equation in terms of array indices where \( k \) is the time step (i.e. \( t = k\Delta t \)).

\[
\frac{T_{i+1}^k - T_i^k}{\Delta t} - \alpha \frac{T_{i+1}^k - 2T_i^k + T_{i-1}^k}{(\Delta x)^2} = 0
\]

Solve for temperature at the future time step \( T_i^{k+1} \).

\[
T_i^{k+1} = T_i^k + \frac{\alpha \Delta t}{(\Delta x)^2} \left( T_{i+1}^k - 2T_i^k + T_{i-1}^k \right)
\]
Write Update Equation

The update equation is

$$T_i^{k+1} = T_i^k + c_i \left( T_{i+1}^k - 2T_i^k + T_{i-1}^k \right)$$

When performing the update, calculate the new values based only on old values. Do not mix the two! It is usually necessary to store the old and the new values in two different arrays to do this.

The update coefficient is

$$c_i = \frac{\alpha_i \Delta t}{(\Delta x)^2}$$

The update coefficients $c_i$ do not change with time so they should be calculated before the main time loop to improve speed and efficiency.

Stability Condition (1 of 2)

The finite-difference equation couples only the immediately adjacent points on the grid. Therefore, it is impossible for numbers to propagate across the grid faster than one cell $\Delta x$ in on time step $\Delta t$. This defines the numerical speed $v_{\text{FDM}}$ as

$$v_{\text{FDM}} \approx \frac{\Delta x}{\Delta t}$$

The speed $v_{\text{heat}}$ at which a physical heat front would diffuse through a 1D medium is

$$v_{\text{heat}} \approx \frac{\alpha_i}{\Delta x}$$
Stability Condition (2 of 2)

In order for the simulation to be stable, the numerical parameters must be chosen so that speed of the heat front cannot is not faster than is numerically possible on the grid.

\[ v_{\text{heat}} < \frac{1}{2} v_{\text{FDM}} \]

A factor of \( \frac{1}{2} \) was included as a safety margin.

Substituting in the expressions for speed gives

\[ \frac{\alpha_i}{\Delta x} < \frac{1}{2} \frac{\Delta x}{\Delta t} \]

Solving this for \( \Delta t \) gives the stability condition.

\[ \Delta t < \frac{(\Delta x)^2}{2\alpha_i} \]

Revised Algorithm

1. Initialize MATLAB
2. Dashboard – define all simulation parameters
3. Calculate grid: \( N_x \) and \( dx \).
4. Build device on grid: \( \alpha \)
5. Calculate a stable time step (and number of iterations): \( \Delta t \) and \( NT \)
6. Calculate update coefficients: \( [c_1, c_2, c_3, \ldots, c_N] \)
7. Initialize arrays: \( T, T_2, \ldots, \)
8. Main loop – iterate over time \( t \)
   a. Update \( T \), all the way across grid (loop over \( x \))
   b. Enforce boundary conditions
   c. Record intermediate results (if needed)
   d. Visualize intermediate results (if needed)
9. Post process results
10. Visualize results
11. Finished!
Time-Domain Solution of Maxwell’s Equations

\[ \nabla \times \vec{E}(t) = -\mu \frac{\partial \vec{H}(t)}{\partial t} \]

A circulating \( \vec{E} \) field induces a change in the \( \vec{H} \) field at the center of circulation in proportion to the permeability.

\[ \nabla \times \vec{H}(t) = \varepsilon \frac{\partial \vec{E}(t)}{\partial t} \]

A circulating \( \vec{H} \) field induces a change in the \( \vec{E} \) field at the center of circulation in proportion to the permittivity.
Fields are Staggered in Both Space and Time

\[ \nabla \times \bar{E}(t) = -\mu \frac{\partial \bar{H}(t)}{\partial t} \]

\[ \nabla \times \bar{H}(t) = \varepsilon \frac{\partial \bar{E}(t)}{\partial t} \]

\[ \nabla \times \bar{E} \bigg|_{t} \approx -\mu \frac{\bar{H} \bigg|_{t+\Delta t/2} - \bar{H} \bigg|_{t-\Delta t/2}}{\Delta t} \]

\[ \nabla \times \bar{H} \bigg|_{t+\Delta t/2} \approx \varepsilon \frac{\bar{E} \bigg|_{t+\Delta t/2} - \bar{E} \bigg|_{t}}{\Delta t} \]

Update Equations

\[ \nabla \times \bar{E} \bigg|_{t} = -\mu \frac{\bar{H} \bigg|_{t+\Delta t/2} - \bar{H} \bigg|_{t-\Delta t/2}}{\Delta t} \]

\[ \nabla \times \bar{H} \bigg|_{t+\Delta t/2} = \bar{H} \bigg|_{t-\Delta t/2} - \frac{\Delta t}{\mu} (\nabla \times \bar{E}) \bigg|_{t} \]

\[ \nabla \times \bar{E} \bigg|_{t+\Delta t} = \varepsilon \frac{\bar{E} \bigg|_{t+\Delta t} - \bar{E} \bigg|_{t}}{\Delta t} \]

\[ \nabla \times \bar{H} \bigg|_{t+\Delta t} = \bar{E} \bigg|_{t} + \frac{\Delta t}{\varepsilon} (\nabla \times \bar{H}) \bigg|_{t+\Delta t/2} \]
Courant Stability Condition

Due to how the update equations are formulated, a disturbance cannot travel more than one grid cell in one time step.

The time step must be made small enough that a physical wave cannot outpace a numerical wave.

\[ \Delta t < \frac{1}{c_0 \sqrt{\left(\frac{1}{(\Delta x)^2} + \frac{1}{(\Delta y)^2} + \frac{1}{(\Delta z)^2}\right)}} \approx \frac{n_{\text{min}} \Delta_{\text{min}}}{2c_0} \]

Sequence of Code Development

*Step 1 – Basic FDTD Algorithm*

- Basic update equations
Sequence of Code Development

**Step 2 – Add Simple Soft Source**

- Basic update equations
- Add a soft source

Sequence of Code Development

**Step 3 – Add Absorbing Boundary**

- Basic update equations
- Add a soft source
- Add perfect boundary condition
Sequence of Code Development

**Step 4 – Add TF/SF Source**

- Basic update equations
- Add a soft source
- Add perfectly absorbing boundary condition
- Incorporate TF/SF “one-way” source

**Step 5 – Move Source and Add T & R**

- Basic update equations
- Add a soft source
- Add perfect boundary condition
- Incorporate TF/SF “one-way” source
- Move position of source
- Calculate transmittance and reflectance
Sequence of Code Development

**Step 6 – Add Device (Algorithm Done)**

- Basic update equations
- Add a soft source
- Add perfect boundary condition
- Incorporate TF/SF “one-way” source
- Move position of source
- Calculate transmittance and reflectance
- Add a real device

---

Summary of Code Development Sequence

- **Step 1** – Implement basic FDTD algorithm
- **Step 2** – Add the source
- **Step 3** – Add absorbing boundary
- **Step 4** – Add “one-way” source
- **Step 5** – Calculate transmittance and reflectance
- **Step 6** – Add a device
Movie of Simple Hard Source

Movie of Simple Soft Source
Movie of TF/SF Soft Source

Calculating Transmission & Reflection
Block Diagram of 1D FDTD

Animation of Numerical Dispersion

Simulation with near-zero numerical dispersion...

Simulation with strong numerical dispersion...
2D Code Development Sequence

Step 1 – Basic Update Equations

Step 2 of 1000

Dirichlet Boundary Condition

Step 2 of 1000

Periodic Boundary Condition

2D Code Development Sequence

Step 2 – Periodic Boundary Conditions
2D Code Development Sequence

*Step 2 – Perfectly Matched Layer*

![Diagram showing Perfectly Matched Layer](image)

2D Code Development Sequence

*Step 4 – TF/SF Source*

![Diagram showing TF/SF Source](image)
2D Code Development Sequence

**Step 5 – Transmission & Reflection**

![Diagram](image1)

**Step 6 – Simulate Device**

![Diagram](image2)
Summary of 2D Code Development Sequence

Step 1 – Basic Update + Dirichlet
Step 2 – Basic Update + Periodic BC
Step 3 – Add PML
Step 4 – TF/SF

Step 5 – Calculate Response
Step 6 – Add a Device and Benchmark

Real FDTD Simulation