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
Define Problem

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

$$\alpha(x) = 25 + 50 \sin \left( \frac{\pi x}{L} \right) \quad 0 \leq x \leq L$$
Governing Equation

The governing equation is the time-domain form of the heat equation.

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

- \( T(x) \equiv \text{temperature} \)
- \( \alpha(x) \equiv \text{thermal diffusivity} \)
- \( t \equiv \text{time} \)
- \( \nabla \equiv \text{del operator} \)

<table>
<thead>
<tr>
<th>Material</th>
<th>( \alpha )</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

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

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

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

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 – We interpolate the second term in the equation to exist at \( t + \Delta t/2 \).

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

This is called the Crank-Nicholson scheme.

Very often, we can get away with using Solution 1 or 2, but the Crank-Nicholson scheme is unconditionally stable.

Solve for Future Value

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

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

We now write this equation in terms of array indices.

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

Solve for future value of \( T \).

\[
T_{i}^{k+1} = T_{i}^{k} + \frac{\alpha \Delta t}{(\Delta x)^2} \left[ T_{i+1}^{k} - 2T_{i}^{k} + T_{i-1}^{k} \right]
\]
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! We will have 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 do not change with time so they should be calculated before the main time loop.

Stability Condition

The finite-difference equations couples only adjacent points on the grid. Thus, it is numerically impossible for a number to travel farther than one grid cell in one time step.

We need to make sure that the speed of the heat front is slower than what is numerically possible.

The stability condition that ensures this is

$$ \frac{\alpha_i \Delta t}{(\Delta x)^2} < \frac{1}{2} $$

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

Typically we calculate the time step based on this stability condition.
Revised Algorithm

1. Initialize MATLAB
2. Dashboard – define all simulation parameters
3. Calculate grid: \( N_x \) and \( d_x \).
4. Build device on grid: \( \alpha \)
5. Calculate a stable time step (and number of iterations): \( dt \) and \( NT \)
6. Calculate update coefficients: \([c_1, c_2, c_3, \ldots, c_N]\)
7. Initialize arrays: \( T, T_2, \text{etc.} \)
8. Main loop – iterate over time \( t \)
   a. Update \( T_i \) 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
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

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

\[ \nabla \times \vec{E}(t) \approx -\mu \frac{\vec{H}(t_{i+\Delta t/2}) - \vec{H}(t_{i-\Delta t/2})}{\Delta t} \]

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

\[ \nabla \times \vec{H}(t) \approx \varepsilon \frac{\vec{E}(t_{i+\Delta t}) - \vec{E}(t_{i})}{\Delta t} \]
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 perfect boundary condition
- Incorporate TF/SF “one-way” source
Sequence of Code Development

**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

FDTD Simulation

Reflection Response

Transmission Response

Block Diagram of 1D FDTD

Compute Grid Resolution

$\Delta t = \min \left( \frac{\Delta x}{c}, \frac{\Delta y}{c}, \frac{\Delta z}{c} \right)$

$N = \text{round} \left( \frac{d}{\Delta t} \right)$

Build Device

Klystron and Waveguide

Compute Time Step

$\Delta t = \frac{\Delta x}{c} / (2\pi)$

Compute Source

$g(t) = \exp \left( -\frac{(t-\tau)^2}{\tau^2} \right)$

Compute Update Coefficients

$\eta_x = \frac{\epsilon_x}{\epsilon_0}$

$\eta_y = \frac{\mu_y}{\mu_0}$

$\eta_z = \frac{\epsilon_z}{\epsilon_0}$

Initialize Fields

$E_0 = 1$

Initialize Boundary Terms

$A_L = 0, A_R = 0, A_0 = 0, A_1 = 0$

Update $H$ from $E$

$R_{L+} = R_{L+} + \kappa_c (E_{R+} - E_{L-}) / \Delta z \quad k = \pm 1$

$R_{L-} = R_{L-} + \kappa_c (E_{R-} - E_{L+}) / \Delta z \quad k = \pm 1$

Record $H$ Field Boundary Term

$A_L = h_0, A_R = h_1, A_0 = h_0$

Update $E$ from $H$

$E_{\Delta z+} = E_{\Delta z-} + \kappa_c (H_{\Delta z+} - H_{\Delta z-}) / \Delta z \quad k = \pm 1$

$E_{\Delta z-} = E_{\Delta z+} + \kappa_c (H_{\Delta z+} - H_{\Delta z-}) / \Delta z \quad k = \pm 1$

Record $E$ Field Boundary Term

$A_L = \eta_x A_0 - \eta_x h_0, A_R = \eta_x A_1 - \eta_x h_1, A_0 = \eta_x h_0, A_1 = \eta_x h_1$

Inject Source

$E_0 = \eta_x p(t)$

Visualize Fields

Includes:

- Basic FDTD engine
- Dirichlet BC's
- Calculate source parameters
- Simple soft source
- Perfectly absorbing BC's

Excludes:

- TEO source
- Fourier transforms
- Reflectance/transmittance
- Calculate grid parameters
- Incorporate device
Animation of Numerical Dispersion

Simulation with near-zero numerical dispersion...

Simulation with strong numerical dispersion...

2D Code Development Sequence
Step 1 – Basic Update Equations
2D Code Development Sequence
Step 2 – Periodic Boundary Conditions

Periodic Boundary Condition

2D Code Development Sequence
Step 2 – Perfectly Matched Layer

Perfectly Matched Layer
2D Code Development Sequence

Step 4 – TF/SF Source

Step 5 – Transmission & Reflection
2D Code Development Sequence

Step 6 – Simulate Device

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