Course Instructor
Dr. Raymond C. Rumpf
Office: A-337
Phone: (915) 747-6958
E-Mail: rcrumpf@utep.edu

Computational Methods in EE

Topic 6b – Finite-Difference Approximations

EE 4386/5301 Computational Methods in EE

Outline

Computational Methods in EE

- What are finite-difference approximations?
- Polynomial technique
  - Description of the method
  - Examples
  - Implementation in MATLAB
  - Examples using MATLAB

Computational
Methods in EE

What are Finite-Difference Approximations?

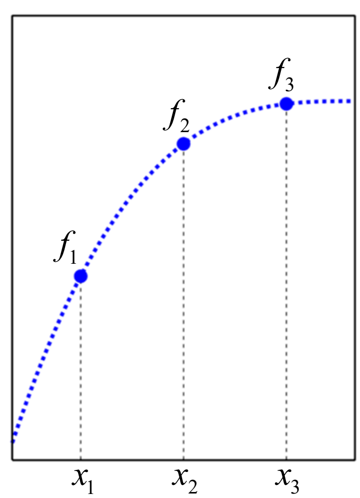
What are Finite-Difference Approximations? (1 of 3)

Computational
Methods in EE

Very often in science and engineering we must calculate a derivative.

When we are processing data from measurements or simulations, there may not be an analytical equation to work with symbolically.

Typically, we only know the function at discrete points.



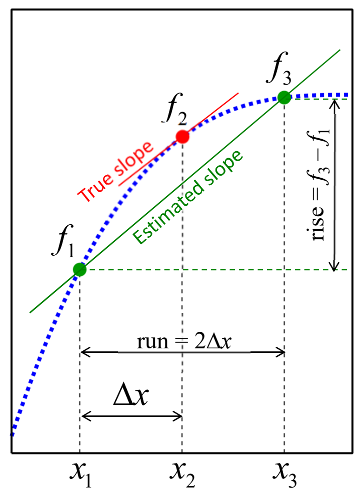
What are Finite-Difference Approximations? (2 of 3)

Computational Methods in EE

Suppose we wish to numerically calculate the first-order derivative at x_2.

The first-order derivative is slope. We can estimate the slope as rise÷run using information from surrounding points.

$$f'(x_2) \approx \frac{\text{rise}}{\text{run}} = \frac{f_3 - f_1}{2\Delta x}$$



What are Finite-Difference Approximations? (3 of 3)

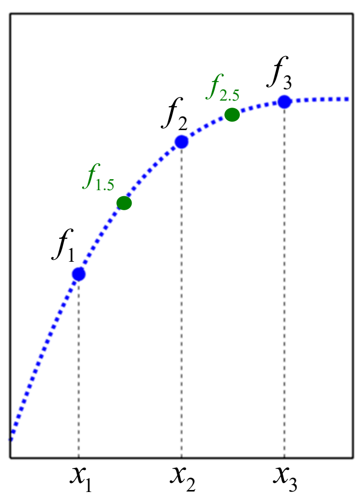
Computational Methods in EE

We can estimate the derivative at the midpoint between data points.

$$f'(x_{1.5}) = \frac{f_2 - f_1}{\Delta x} \qquad f'(x_{2.5}) = \frac{f_3 - f_2}{\Delta x}$$

The second-order derivative is the slope of the slope.

$$f''(x_2) = \frac{f'(x_{2.5}) - f'(x_{1.5})}{\Delta x} = \frac{\frac{f_3 - f_2}{\Delta x} - \frac{f_2 - f_1}{\Delta x}}{\Delta x} = \frac{f_3 - 2f_2 + f_1}{\Delta x^2}$$



Our First Two Finite-Difference Approximations

Computational Methods in EE

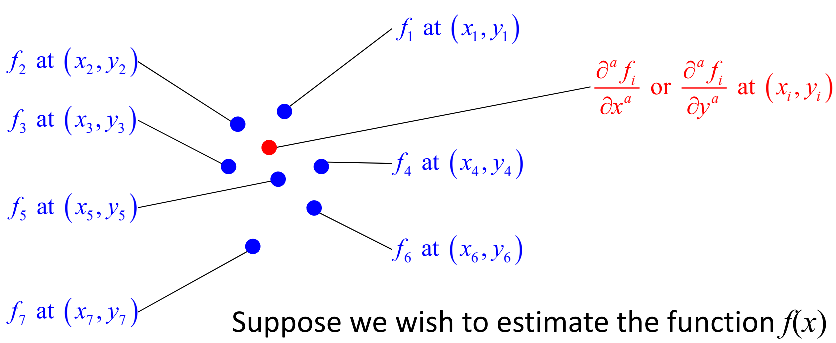
We already derived two finite-difference approximations

$$f'(x_2) \approx \frac{f_3 - f_1}{2\Delta x}$$

$$f''(x_2) \approx \frac{f_3 - 2f_2 + f_1}{\Delta x^2}$$

General Concept of Finite-Difference Approximations (1 of 2)

Computational Methods in EE



Suppose we wish to estimate the function $f(x)$ or one of its derivatives at location (x_i, y_i).

$$\frac{\partial^a f_i}{\partial x^a} \text{ or } \frac{\partial^a f_i}{\partial y^a} = a_1 f_1 + a_2 f_2 + a_3 f_3 + a_4 f_4 + a_5 f_5 + a_6 f_6 + a_7 f_7$$

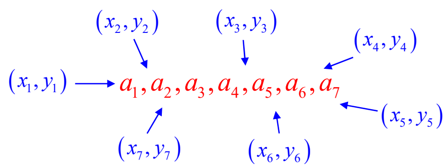
It is always possible to estimate this from a linear sum of the function values at surrounding points.

General Concept of Finite-Difference Approximations (2 of 2)

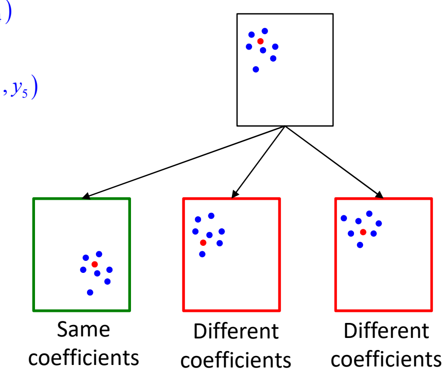
Computational Methods in EE

The trick is, how do we calculate the coefficients a_n?

These are a function of the positions of the points.



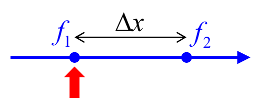
Finite-difference coefficients depend only on the relative position of the points. They do not depend on the absolute positions.



Types of Finite-Differences

Computational Methods in EE

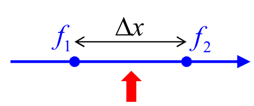
$$\frac{df_1}{dx} \approx \frac{f_2 - f_1}{\Delta x}$$



Forward Finite-Difference

Reaches ahead to use data in the forward direction.

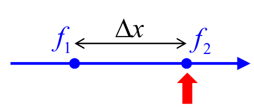
$$\frac{df_{1.5}}{dx} \approx \frac{f_2 - f_1}{\Delta x}$$



Central Finite-Difference

Reaches symmetrically to use data in both directions for highest accuracy.

$$\frac{df_2}{dx} \approx \frac{f_2 - f_1}{\Delta x}$$



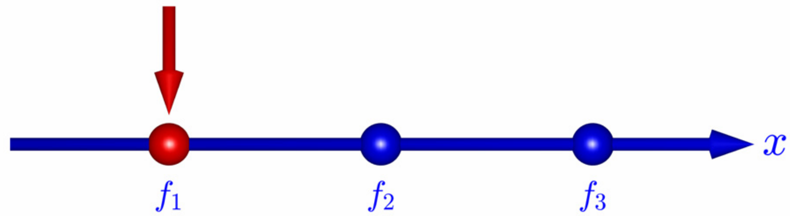
Backward Finite-Difference

Reaches behind to use data in the backward direction.

Continuum of Finite-Difference Approximations (1 of 2)

Computational Methods in EE

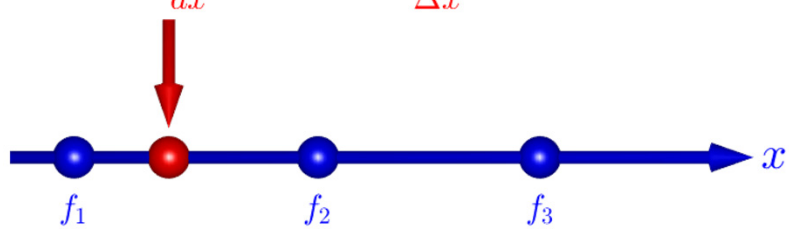
$$\frac{df_{1.0}}{dx} \simeq \frac{-1.5f_1 + 2.0f_2 - 0.5f_3}{\Delta x}$$



Continuum of Finite-Difference Approximations (2 of 2)

Computational Methods in EE

$$\frac{df_{1.0}}{dx} \simeq \frac{-1.0f_1 + 1.1f_2 - 0.1f_3}{\Delta x}$$

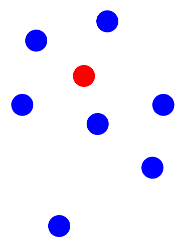


Two Key Considerations

Computational Methods in EE

1. The position of the points from which the finite-difference approximation is calculated. More closely spaced points improves accuracy, but typically leads to more computations.

2. The location of the point where the finite-difference is being evaluated. We typically want to be as centered as possible for best accuracy.



Computational Methods in EE

Polynomial Technique for Deriving Finite-Difference Approximations

Concept of Using Polynomials

Computational Methods in EE

We can fit an Nth order polynomial given $N+1$ points.

$$f(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_N x^N$$

After we have performed the curve fit, we can interpolate the function or any of its derivatives from the polynomial.

$$
\begin{aligned}
f(x) &= a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + \cdots + a_N x^N \\
f'(x) &= a_1 + 2a_2 x + 3a_3 x^2 + 4a_4 x^3 + \cdots + N a_N x^{N-1} \\
f''(x) &= 2a_2 + 6a_3 x + 12 a_4 x^2 + \cdots + N(N-1) a_N x^{N-2} \\
f'''(x) &= 6a_3 + 24 a_4 x + \cdots + N(N-1)(N-2) a_N x^{N-3} \\
&\vdots
\end{aligned}
$$

Easiest Point for Evaluating $f(x)$

Computational Methods in EE

Recall the equations we will use to evaluate $f(x)$ or one of its derivatives:

$$
\begin{aligned}
f(x) &= a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + \cdots + a_N x^N \\
f'(x) &= a_1 + 2a_2 x + 3a_3 x^2 + 4a_4 x^3 + \cdots + N a_N x^{N-1} \\
f''(x) &= 2a_2 + 6a_3 x + 12 a_4 x^2 + \cdots + N(N-1) a_N x^{N-2}
\end{aligned}
$$

These are most easily evaluated at $x = 0$ because the above equations reduce to

$$
\begin{aligned}
f(0) &= a_0 \\
f'(0) &= a_1 \\
f''(0) &= 2a_2 \\
f'''(0) &= 6a_3 \\
&\vdots
\end{aligned}
$$

How can we make this happen every time?

How to Make Any Point Easy

Computational Methods in EE

Now suppose we wish to evaluate $f(x)$ or one of its derivatives at the general point $x = x_{\text{fd}}$.

To do this, we shift our x-axis by x_{fd} before fitting the polynomial.

Recall that the finite-difference coefficients depend only on the relative position of the points. An offset will not affect their values.

Now we write our polynomial at each shifted point.

$$
\begin{aligned}
f(\tilde{x}_1) &= a_0 + a_1\tilde{x}_1 + a_2\tilde{x}_1^2 + \cdots + a_N\tilde{x}_1^N \\
f(\tilde{x}_2) &= a_0 + a_1\tilde{x}_2 + a_2\tilde{x}_2^2 + \cdots + a_N\tilde{x}_2^N \\
&\vdots \\
f(\tilde{x}_{N+1}) &= a_0 + a_1\tilde{x}_{N+1} + a_2\tilde{x}_{N+1}^2 + \cdots + a_N\tilde{x}_{N+1}^N
\end{aligned}
$$

$$\tilde{x}_n = x_n - x_{\text{fd}}$$

In our shifted coordinate system, our finite-difference is being evaluated at $\tilde{x} = 0$.

$$f(0) = a_0 \qquad f'(0) = a_1 \qquad f''(0) = 2a_2 \qquad f'''(0) = 6a_3 \qquad \cdots$$

Four-Step Procedure to Derive Finite-Difference Approximations

Computational Methods in EE

Step 1 – Identify set of points x_1, x_2, …, to x_N from which to derive a finite-difference approximation.

Step 2 – Shift coordinates so that $\tilde{x} = 0$ corresponds to where you wish to approximate the function or one of its derivatives.

$$\tilde{x}_i = x_i - x_{\text{fd}}$$

Step 3 – Fit shifted points to a polynomial.

$$f(\tilde{x}) = a_0 + a_1\tilde{x} + a_2\tilde{x}^2 + \cdots + a_N\tilde{x}^N$$

Step 4 – Write finite-difference approximation directly from one of the derivatives of the polynomial.

$$
\begin{aligned}
f(0) &= a_0 \\
f'(0) &= a_1 \\
f''(0) &= 2a_2 \\
&\vdots
\end{aligned}
$$

Step 1 – Choose x Coordinates

Computational Methods in EE

Identify the x-coordinates of the points from which you wish to approximate a derivate.

Store these in a column

$$[x] = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{N+1} \end{bmatrix}$$

Step 2 – Shift x-Axis

Computational Methods in EE

Shift the function across the x-axis until $\tilde{x} = 0$ corresponds to the point where you wish to approximate the derivative.

$$\frac{d^a}{dx^a} f\left(x = x_{\text{fd}}\right) = \frac{d^a}{dx^a} f\left(\tilde{x} = 0\right)$$

Subtract x_{fd} from the column vector $[x]$ to shift coordinates.

$$[\tilde{x}] = [x] - x_{\text{fd}} = \begin{bmatrix} x_1 - x_{\text{fd}} \\ x_2 - x_{\text{fd}} \\ \vdots \\ x_{N+1} - x_{\text{fd}} \end{bmatrix} = \begin{bmatrix} \tilde{x}_1 \\ \tilde{x}_2 \\ \vdots \\ \tilde{x}_{N+1} \end{bmatrix}$$

Step 3 – Fit Points to Polynomial (1 of 3)

Computational Methods in EE

Use the column vector $[\tilde{x}]$ to build matrix $[\tilde{X}]$.

$$[\tilde{X}] = \begin{bmatrix} [\tilde{x}]^0 & [\tilde{x}]^1 & \cdots & [\tilde{x}]^N \end{bmatrix}$$

$$= \begin{bmatrix} 1 & \tilde{x}_1 & \cdots & \tilde{x}_1^N \\ 1 & \tilde{x}_2 & \cdots & \tilde{x}_2^N \\ \vdots & \vdots & \ddots & \\ 1 & \tilde{x}_{N+1} & & \tilde{x}_{N+1}^N \end{bmatrix} = \begin{bmatrix} 1 & x_1 - x_{\text{fd}} & \cdots & (x_1 - x_{\text{fd}})^N \\ 1 & x_2 - x_{\text{fd}} & \cdots & (x_2 - x_{\text{fd}})^N \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{N+1} - x_{\text{fd}} & \cdots & (x_{N+1} - x_{\text{fd}})^N \end{bmatrix}$$

Insert 1's instead of $[\tilde{x}]^0$.

Step 3 – Fit Points to Polynomial (2 of 3)

Computational Methods in EE

Invert the matrix $[\tilde{X}]$.

$$[\tilde{Y}] = [\tilde{X}]^{-1} = \begin{bmatrix} \tilde{y}_{11} & \tilde{y}_{12} & \cdots & \tilde{y}_{1,N+1} \\ \tilde{y}_{21} & \tilde{y}_{22} & \cdots & \tilde{y}_{2,N+1} \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{y}_{N+1,1} & \tilde{y}_{N+1,2} & \cdots & \tilde{y}_{N+1,N+1} \end{bmatrix}$$

Step 3 – Fit Points to Polynomial (3 of 3)

Computational Methods in EE

Calculate the polynomial coefficients.

$$[a]=\left[\tilde{X}\right]^{-1}[f]=\left[\tilde{Y}\right][f] \quad \rightarrow \quad \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_N \end{bmatrix} = \begin{bmatrix} \tilde{y}_{11} & \tilde{y}_{12} & \tilde{y}_{13} & \cdots & \tilde{y}_{1,N+1} \\ \tilde{y}_{21} & \tilde{y}_{22} & \tilde{y}_{23} & \cdots & \tilde{y}_{2,N+1} \\ \tilde{y}_{31} & \tilde{y}_{32} & \tilde{y}_{33} & \cdots & \tilde{y}_{3,N+1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \tilde{y}_{N+1,1} & \tilde{y}_{N+1,2} & \tilde{y}_{N+1,3} & \cdots & \tilde{y}_{N+1,N+1} \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_{N+1} \end{bmatrix}$$

$$\begin{aligned} a_0 &= \tilde{y}_{11}f_1 + \tilde{y}_{12}f_2 + \tilde{y}_{13}f_3 + \cdots + \tilde{y}_{1,N+1}f_{N+1} \\ a_1 &= \tilde{y}_{21}f_1 + \tilde{y}_{22}f_2 + \tilde{y}_{23}f_3 + \cdots + \tilde{y}_{2,N+1}f_{N+1} \\ a_2 &= \tilde{y}_{31}f_1 + \tilde{y}_{32}f_2 + \tilde{y}_{33}f_3 + \cdots + \tilde{y}_{3,N+1}f_{N+1} \\ &\vdots \\ a_N &= \tilde{y}_{N+1,1}f_1 + \tilde{y}_{N+1,2}f_2 + \tilde{y}_{N+1,3}f_3 + \cdots + \tilde{y}_{1,N+1}f_{N+1} \end{aligned}$$

At this point, f_1 to f_{N+1} will be symbolic.

Step 4 – Write Finite-Difference Approximation

Computational Methods in EE

Recall how we interpolate the function or one of it's derivatives given our polynomial…

$$\begin{aligned} f(\tilde{x}=0) &= a_0 \\ \frac{d}{dx}f(\tilde{x}=0) &= a_1 \\ \frac{d^2}{dx^2}f(\tilde{x}=0) &= 2a_2 \\ &\vdots \end{aligned}$$

$$\begin{aligned} a_0 &= \tilde{y}_{11}f_1 + \tilde{y}_{12}f_2 + \tilde{y}_{13}f_3 + \cdots + \tilde{y}_{1,N+1}f_{N+1} \\ a_1 &= \tilde{y}_{21}f_1 + \tilde{y}_{22}f_2 + \tilde{y}_{23}f_3 + \cdots + \tilde{y}_{2,N+1}f_{N+1} \\ a_2 &= \tilde{y}_{31}f_1 + \tilde{y}_{32}f_2 + \tilde{y}_{33}f_3 + \cdots + \tilde{y}_{3,N+1}f_{N+1} \\ &\vdots \\ a_N &= \tilde{y}_{N+1,1}f_1 + \tilde{y}_{N+1,2}f_2 + \tilde{y}_{N+1,3}f_3 + \cdots + \tilde{y}_{1,N+1}f_{N+1} \end{aligned}$$

$$\begin{aligned} f(\tilde{x}=0) &= \tilde{y}_{11}f_1 + \tilde{y}_{12}f_2 + \tilde{y}_{13}f_3 + \cdots + \tilde{y}_{1,N+1}f_{N+1} \\ \frac{d}{dx}f(\tilde{x}=0) &= \tilde{y}_{21}f_1 + \tilde{y}_{22}f_2 + \tilde{y}_{23}f_3 + \cdots + \tilde{y}_{2,N+1}f_{N+1} \\ \frac{d^2}{dx^2}f(\tilde{x}=0) &= 2\tilde{y}_{N+1,1}f_1 + 2\tilde{y}_{N+1,2}f_2 + 2\tilde{y}_{N+1,3}f_3 + \cdots + 2\tilde{y}_{1,N+1}f_{N+1} \end{aligned}$$

The rows of $\left[\tilde{Y}\right]$ are essentially our finite-difference coefficients.

Examples Using Polynomial Technique

Example #1

Derive first-order and second-order finite-difference approximations that span across three points. The approximations should be evaluated at the midpoint.

$$[\tilde{x}] = \begin{bmatrix} -h \\ 0 \\ h \end{bmatrix} \qquad [\tilde{X}] = \begin{bmatrix} [\tilde{x}]^0 & [\tilde{x}]^1 & [\tilde{x}]^2 \end{bmatrix} = \begin{bmatrix} 1 & -h & h^2 \\ 1 & 0 & 0 \\ 1 & h & h^2 \end{bmatrix} \qquad [\tilde{Y}] = [\tilde{X}]^{-1} = \begin{bmatrix} 0 & 1 & 0 \\ -1/2h & 0 & 1/2h \\ 1/2h^2 & -1/h^2 & 1/2h^2 \end{bmatrix}$$

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ -1/2h & 0 & 1/2h \\ 1/2h^2 & -1/h^2 & 1/2h^2 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} \Rightarrow \begin{aligned} a_0 &= 0 \cdot f_1 + 1 \cdot f_2 + 0 \cdot f_3 = f_2 \\ a_1 &= (-1/2h) \cdot f_1 + 0 \cdot f_2 + (1/2h) \cdot f_3 = \frac{-f_1 + f_3}{2h} \\ a_2 &= (1/2h^2) \cdot f_1 + (-1/h^2) \cdot f_2 + (1/2h^2) \cdot f_3 = \frac{f_1 - 2f_2 + f_3}{2h^2} \end{aligned}$$

$$f(x_{\text{fd}}) = a_0 = f_2$$

$$\frac{df(x_{\text{fd}})}{dx} = a_1 = \frac{f_3 - f_1}{2h}$$

$$\frac{d^2 f(x_{\text{fd}})}{dx^2} = 2a_2 = \frac{f_1 - 2f_2 + f_3}{h^2}$$

Example #2

Computational Methods in EE

Derive first-order and second-order finite-difference approximations that span across three points. The approximations should be evaluated at the first point.

$$[\tilde{x}] = \begin{bmatrix} 0 \\ h \\ 2h \end{bmatrix} \qquad \left[\tilde{X}\right] = \left[[\tilde{x}]^0 \quad [\tilde{x}]^1 \quad [\tilde{x}]^2\right] = \begin{bmatrix} 1 & 0 & 0 \\ 1 & h & h^2 \\ 1 & 2h & (2h)^2 \end{bmatrix} \qquad \left[\tilde{Y}\right] = \left[\tilde{X}\right]^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ -3/(2h) & 2/h & -1/2h \\ 1/(2h^2) & -1/h^2 & 1/(2h^2) \end{bmatrix}$$

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ -3/(2h) & 2/h & -1/2h \\ 1/(2h^2) & -1/h^2 & 1/(2h^2) \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} \Rightarrow \begin{aligned} a_0 &= 1\cdot f_1 + 0\cdot f_2 + 0\cdot f_3 = f_1 \\ a_1 &= (-3/2h)\cdot f_1 + (2/h)\cdot f_2 + (-1/2h)\cdot f_3 = \frac{-1.5f_1 + 2f - 0.5f_3}{h} \\ a_2 &= (1/2h^2)\cdot f_1 + (-1/h^2)\cdot f_2 + (1/2h^2)\cdot f_3 = \frac{f_1 - 2f_2 + f_3}{2h^2} \end{aligned}$$

$$f(x_{\text{fd}}) = a_0 = f_1$$

$$\frac{df(x_{\text{fd}})}{dx} = a_1 = \frac{-1.5f_1 + 2f - 0.5f_3}{h}$$

$$\frac{d^2 f(x_{\text{fd}})}{dx^2} = 2a_2 = \frac{f_1 - 2f_2 + f_3}{h^2}$$

Example #3 – Higher Order Accuracy (1 of 2)

Computational Methods in EE

Let's evaluate some derivatives at the midpoint of four discrete points.

$$[\tilde{x}] = \begin{bmatrix} -3h/2 \\ -h/2 \\ h/2 \\ 3h/2 \end{bmatrix} \qquad \left[\tilde{X}\right] = \left[[\tilde{x}]^0 \quad [\tilde{x}]^1 \quad [\tilde{x}]^2 \quad [\tilde{x}]^3\right] = \begin{bmatrix} 1 & -\frac{3h}{2} & \frac{9h^2}{4} & -\frac{27h^3}{8} \\ 1 & -\frac{h}{2} & \frac{h^2}{4} & -\frac{h^3}{8} \\ 1 & \frac{h}{2} & \frac{h^2}{4} & \frac{h^3}{8} \\ 1 & \frac{3h}{2} & \frac{9h^2}{4} & \frac{27h^3}{8} \end{bmatrix}$$

$$\left[\tilde{Y}\right] = \left[\tilde{X}\right]^{-1} = \begin{bmatrix} -\frac{1}{16} & \frac{9}{16} & \frac{9}{16} & -\frac{1}{16} \\ \frac{1}{24h} & -\frac{9}{8h} & \frac{9}{8h} & -\frac{1}{24h} \\ \frac{1}{4h^2} & -\frac{1}{4h^2} & -\frac{1}{4h^2} & \frac{1}{4h^2} \\ -\frac{1}{6h^3} & \frac{1}{2h^3} & -\frac{1}{2h^3} & \frac{1}{6h^3} \end{bmatrix}$$

Example #3 – Higher Order Accuracy (2 of 2)

Computational Methods in EE

The coefficients are then

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \underbrace{\begin{bmatrix} -\frac{1}{16} & \frac{9}{16} & \frac{9}{16} & -\frac{1}{16} \\ \frac{1}{24h} & -\frac{9}{8h} & \frac{9}{8h} & -\frac{1}{24h} \\ \frac{1}{4h^2} & -\frac{1}{4h^2} & -\frac{1}{4h^2} & \frac{1}{4h^2} \\ -\frac{1}{6h^3} & \frac{1}{2h^3} & -\frac{1}{2h^3} & \frac{1}{6h^3} \end{bmatrix}}_{[\tilde{Y}]} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix} \Rightarrow$$

$$a_0 = -\frac{1}{16} f_1 + \frac{9}{16} f_2 + \frac{9}{16} f_3 - \frac{1}{16} f_4$$
$$a_1 = \frac{1}{24h} f_1 - \frac{9}{8h} f_2 + \frac{9}{8h} f_3 - \frac{1}{24h} f_4$$
$$a_2 = \frac{1}{4h^2} f_1 - \frac{1}{4h^2} f_2 - \frac{1}{4h^2} f_3 + \frac{1}{4h^2} f_4$$
$$a_3 = -\frac{1}{6h^3} f_1 + \frac{1}{2h^3} f_2 - \frac{1}{2h^3} f_3 + \frac{1}{6h^3} f_4$$

$$f(x_{2.5}) = a_0 = \frac{-f_1 + 9f_2 + 9f_3 - f_4}{16}$$
$$\frac{df(x_{2.5})}{dx} = a_1 = \frac{f_1 - 27f_2 + 27f_3 - f_4}{24\Delta x}$$
$$\frac{d^2 f(x_{2.5})}{dx^2} = 2a_2 = \frac{f_1 - f_2 - f_3 + f_4}{2(\Delta x)^2}$$

Topic 6b -- Finite-difference approximations 29

Computational Methods in EE

Implementing the Polynomial Technique Using MATLAB

Topic 6b -- Finite-difference approximations 30

General Form of the Polynomial Fit

Computational Methods in EE

We have so far derived finite-difference approximations symbolically.
What if we want 6th-order accurate finite-differences?
This is unreasonable to do symbolically.

Recall our matrix equation representing the polynomial written at each discrete point. It always had the following form where the w's were just numerical constants. The h's were symbolic.

$$\begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_{N+1} \end{bmatrix} = \begin{bmatrix} 1 & w_{12}h & w_{13}h^2 & \cdots & w_{1N}h^N \\ 1 & w_{22}h & w_{23}h^2 & & w_{2N}h^N \\ & \vdots & & \ddots & \\ 1 & w_{N+1,2}h & w_{N+1,3}h^2 & & w_{N+1,N}h^N \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_N \end{bmatrix}$$

Topic 6b -- Finite-difference approximations Slide 31

Factor Out Symbolic Term h

Computational Methods in EE

Now, we are able to separate the w terms from the h terms.

$$\begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_{N+1} \end{bmatrix} = \begin{bmatrix} 1 & w_{12} & w_{13} & \cdots & w_{1N} \\ 1 & w_{22} & w_{23} & & w_{2N} \\ & \vdots & & \ddots & \\ 1 & w_{N+1,2} & w_{N+1,3} & & w_{N+1,N} \end{bmatrix} \begin{bmatrix} 1 & & & & \\ & h & & & \\ & & h^2 & & \\ & & & \ddots & \\ & & & & h^N \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_N \end{bmatrix}$$

We were able to put numbers to all of these coefficients. This is a fully numerical matrix. It does not contain any symbolic variables.

These are our symbolic variables.

Hint: $[W] = [\tilde{X}]$ when $h = 1$

So we build $[W]$ by building $[\tilde{X}]$ and pretending $h = 1$.

Topic 6b -- Finite-difference approximations Slide 32

Solve Matrix Equation for $[a]$

Computational Methods in EE

Solving our matrix equation for $[a]$ gives

$$\begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} = \begin{bmatrix} 1 & & & & \\ & h & & & \\ & & h^2 & & \\ & & & \ddots & \\ & & & & h^N \end{bmatrix}^{-1} \begin{bmatrix} 1 & w_{12} & w_{13} & \cdots & w_{1N} \\ 1 & w_{22} & w_{23} & & w_{2N} \\ \vdots & & & \ddots & \\ 1 & w_{N+1,2} & w_{N+1,3} & & w_{N+1,N} \end{bmatrix}^{-1} \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_{N+1} \end{bmatrix}$$

$$\downarrow \qquad [V] = [W]^{-1}$$

$$\begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} = \begin{bmatrix} 1 & & & & \\ & \frac{1}{h} & & & \\ & & \frac{1}{h^2} & & \\ & & & \ddots & \\ & & & & \frac{1}{h^N} \end{bmatrix} \begin{bmatrix} v_{11} & v_{12} & v_{13} & \cdots & v_{1N} \\ v_{21} & v_{22} & v_{23} & & v_{2N} \\ & \vdots & & \ddots & \\ v_{N+1,1} & v_{N+1,2} & v_{N+1,3} & & v_{N+1,N} \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_{N+1} \end{bmatrix}$$

The key aspect here is that $[W]$ will be completely numerical so it is easily inverted using MATLAB. This accommodates large matrices and avoids symbolic manipulation.

Topic 6b -- Finite-difference approximations | Slide 33

Incorporate Symbolic h Again

Computational Methods in EE

Next we reincorporate symbolic h by multiplying our matrices.

$$\begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} = \begin{bmatrix} 1 & & & & \\ & \frac{1}{h} & & & \\ & & \frac{1}{h^2} & & \\ & & & \ddots & \\ & & & & \frac{1}{h^N} \end{bmatrix} \begin{bmatrix} v_{11} & v_{12} & v_{13} & \cdots & v_{1N} \\ v_{21} & v_{22} & v_{23} & & v_{2N} \\ & \vdots & & \ddots & \\ v_{N+1,1} & v_{N+1,2} & v_{N+1,3} & & v_{N+1,N} \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_{N+1} \end{bmatrix}$$

$$\downarrow$$

$$\begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} = \begin{bmatrix} 1 \cdot v_{11} & 1 \cdot v_{12} & 1 \cdot v_{13} & \cdots & 1 \cdot v_{1N} \\ \frac{1}{h} \cdot v_{21} & \frac{1}{h} \cdot v_{22} & \frac{1}{h} \cdot v_{23} & & \frac{1}{h} \cdot v_{2N} \\ & \vdots & & \ddots & \\ \frac{1}{h^N} \cdot v_{N+1,1} & \frac{1}{h^N} \cdot v_{N+1,2} & \frac{1}{h^N} \cdot v_{N+1,3} & & \frac{1}{h^N} \cdot v_{N+1,N} \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_{N+1} \end{bmatrix}$$

Topic 6b -- Finite-difference approximations | Slide 34

Extract Polynomial Coefficients

Computational Methods in EE

Next, we read off the polynomial coefficients from our matrix equation.

$$\begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} = \begin{bmatrix} 1\cdot v_{11} & 1\cdot v_{12} & 1\cdot v_{13} & \cdots & 1\cdot v_{1N} \\ \frac{1}{h}\cdot v_{21} & \frac{1}{h}\cdot v_{22} & \frac{1}{h}\cdot v_{23} & & \frac{1}{h}\cdot v_{2N} \\ & \vdots & & \ddots & \\ \frac{1}{h^N}\cdot v_{N+1,1} & \frac{1}{h^N}\cdot v_{N+1,2} & \frac{1}{h^N}\cdot v_{N+1,3} & & \frac{1}{h^N}\cdot v_{N+1,N} \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_{N+1} \end{bmatrix}$$

$$\downarrow$$

$$a_0 = v_{11}f_1 + v_{12}f_2 + \cdots + v_{1N}f_{N+1}$$
$$a_1 = \frac{v_{21}f_1 + v_{22}f_2 + \cdots + v_{2N}f_{N+1}}{h}$$
$$\vdots$$
$$a_N = \frac{v_{N+1,1}f_1 + v_{N+1,2}f_2 + \cdots + v_{N+1,N}f_{N+1}}{h^N}$$

Write Finite-Difference Approximations

Computational Methods in EE

Last, we write our finite-difference approximations from the polynomial coefficients.

$$f = a_0 = v_{11}f_1 + v_{12}f_2 + \cdots + v_{1N}f_{N+1}$$
$$\frac{df}{dx} = a_1 = \frac{v_{21}f_1 + v_{22}f_2 + \cdots + v_{2N}f_{N+1}}{h}$$
$$\frac{d^2 f}{dx^2} = 2a_2 = 2\frac{v_{31}f_1 + v_{32}f_2 + \cdots + v_{3N}f_{N+1}}{h^2}$$

Staring at these equations long enough, we realize that the v_{ij} coefficients can be determined completely numerically. We just have to remember to divide by h^α and perhaps multiply the finite-difference expression by a constant.

Computational Methods in EE

MATLAB Examples

Example #4 – 6th Order Accurate Finite-Differences (1 of 2)

Computational Methods in EE

1. Here we need seven points to calculate seven polynomial coefficients.

$$[\tilde{x}] = [-3h \quad -2h \quad -h \quad 0 \quad h \quad 2h \quad 3h]^T$$

2. To build the $[W]$ matrix, think $h = 1$ for now.

$$[\hat{\tilde{x}}] = [-3 \quad -2 \quad -1 \quad 0 \quad 1 \quad 2 \quad 3]^T$$

$$[W] = \begin{bmatrix} [\hat{\tilde{x}}]^0 & [\hat{\tilde{x}}]^1 & [\hat{\tilde{x}}]^2 & [\hat{\tilde{x}}]^3 & [\hat{\tilde{x}}]^4 & [\hat{\tilde{x}}]^5 & [\hat{\tilde{x}}]^6 \end{bmatrix} = \begin{bmatrix} 1 & -3 & 9 & -27 & 81 & -243 & 729 \\ 1 & -2 & 4 & -8 & 16 & -32 & 64 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 8 & 16 & 32 & 64 \\ 1 & 3 & 9 & 27 & 81 & 243 & 729 \end{bmatrix}$$

Example #4 – 6th Order Accurate Finite-Differences (2 of 2)

Computational Methods in EE

3. Invert $[W]$.

$$[V]=[W]^{-1}=\begin{bmatrix} 0 & 0 & 0 & 1.0000 & 0 & 0 & 0 \\ -0.0167 & 0.1500 & -0.7500 & -0.0000 & 0.7500 & -0.1500 & 0.0167 \\ 0.0056 & -0.0750 & 0.7500 & -1.3611 & 0.7500 & -0.0750 & 0.0056 \\ 0.0208 & -0.1667 & 0.2708 & 0.0000 & -0.2708 & 0.1667 & -0.0208 \\ -0.0069 & 0.0833 & -0.2708 & 0.3889 & -0.2708 & 0.0833 & -0.0069 \\ -0.0042 & 0.0167 & -0.0208 & -0.0000 & 0.0208 & -0.0167 & 0.0042 \\ 0.0014 & -0.0083 & 0.0208 & -0.0278 & 0.0208 & -0.0083 & 0.0014 \end{bmatrix}$$

4. Write the finite-difference approximations, remembering to incorporate the symbolic h's back in.

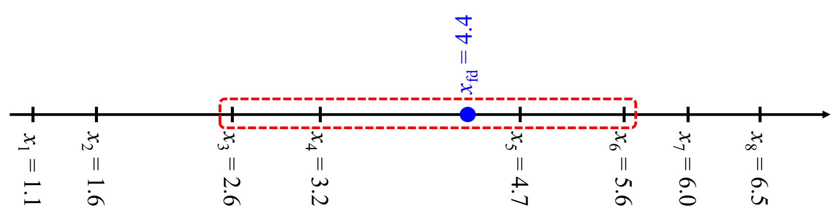
$$f \approx a_0 = \frac{0\cdot f_1 + 0\cdot f_2 + 0\cdot f_3 + 1\cdot f_4 + 0\cdot f_5 - 0\cdot f_6 + 0\cdot f_7}{1}$$

$$\frac{\partial f}{\partial x} \approx a_1 = \frac{-0.0167 f_1 + 0.15 f_2 - 0.75 f_3 - 0\cdot f_4 + 0.75 f_5 - 0.15 f_6 + 0.0167 f_7}{h}$$

$$\frac{\partial^2 f}{\partial x^2} \approx 2a_2 = \frac{2\cdot 0.0056 f_1 - 2\cdot 0.0750 f_2 + 2\cdot 0.7500 f_3 - 2\cdot 1.3611 f_4 + 2\cdot 0.7500 f_5 - 2\cdot 0.0750 f_6 + 2\cdot 0.0056 f_7}{h^2}$$

Example #5 – Nonuniform Grid (1 of 3)

Computational Methods in EE



Derive the finite-difference equations for first- and second-order derivatives at the point x_{fd}.

1. Choose range of points.

$$[x] = [2.6 \quad 3.2 \quad 4.7 \quad 5.6]^T$$

Example #5 – Nonuniform Grid (2 of 3)

Computational Methods in EE

2. Shift x axis.

$$\left[\tilde{x}\right] = \left[x\right] - x_{\text{fd}}$$
$$= \left[2.6 \quad 3.2 \quad 4.7 \quad 5.6\right]^T - 4.4$$
$$= \left[-1.8 \quad -1.2 \quad 0.3 \quad 1.2\right]^T$$

3. Build $\left[\tilde{X}\right]$ matrix.

$$\left[\tilde{X}\right] = \left[\left[\hat{\tilde{x}}\right]^0 \quad \left[\hat{\tilde{x}}\right]^1 \quad \left[\hat{\tilde{x}}\right]^2 \quad \left[\hat{\tilde{x}}\right]^3\right] = \begin{bmatrix} 1 & -1.8 & 3.24 & -5.832 \\ 1 & -1.2 & 1.44 & -1.728 \\ 1 & 0.3 & 0.09 & 0.027 \\ 1 & 1.2 & 1.44 & 1.728 \end{bmatrix}$$

Note: We are not calling this the $[W]$ matrix because we did not have to factor out a symbolic h term.

Example #5 – Nonuniform Grid (3 of 3)

Computational Methods in EE

4. Invert $\left[\tilde{X}\right]$ matrix.

$$\left[\tilde{Y}\right] = \left[\tilde{X}\right]^{-1} = \begin{bmatrix} -0.1143 & 0.3000 & 0.9143 & -0.1000 \\ 0.3810 & -1.0833 & 0.5079 & 0.1944 \\ 0.0794 & 0.1389 & -0.6349 & 0.4167 \\ -0.2646 & 0.4630 & -0.3527 & 0.1543 \end{bmatrix}$$

5. Write the finite-difference approximations directly from the rows of $\left[\tilde{Y}\right]$.

$$f \approx a_0 = -0.1143 f_1 + 0.3 f_2 + 0.9143 f_3 - 0.1 f_4$$
$$\frac{\partial f}{\partial x} \approx a_1 = 0.3810 f_1 - 1.0833 f_2 + 0.5079 f_3 + 0.1944 f_4$$
$$\frac{\partial^2 f}{\partial x^2} \approx 2a_2 = 2 \cdot 0.0794 f_1 + 2 \cdot 0.1389 f_2 - 2 \cdot 0.6349 f_3 + 2 \cdot 0.4167 f_4$$

Note: The symbolic variable h did not appear in $[\tilde{x}]$ so it does not need to be incorporated here.