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

Computational Methods in EE

Topic 7d – Multi-Variable & 2D FDM

EE 4386/5301 Computational Methods in EE

Outline

Computational Methods in EE

- Multi-variable FDM
- 2D FDM on collocated grid
- 2D FDM on staggered grid
- Calculating 1D derivative matrices from 2D derivative matrices

Computational Methods in EE

Multi-Variable FDM

Multi-Variable Problems

Computational Methods in EE

$$\frac{\partial f(x)}{\partial x} = ag(x) \qquad \Rightarrow \qquad \frac{f(x+\Delta x) - f(x-\Delta x)}{2\Delta x} = ag(x)$$

$$\frac{\partial g(x)}{\partial x} = bf(x) \qquad \qquad \frac{g(x+\Delta x) - g(x-\Delta x)}{2\Delta x} = bf(x)$$

This formulation will work, but it is less accurate than is possible because the finite-difference approximations are spanning across the grid farther than necessary.

Tighter Finite-Difference Approximations

Computational Methods in EE

$$\frac{\partial f(x)}{\partial x} = ag(x) \qquad \frac{\partial g(x)}{\partial x} = bf(x)$$

$$\Rightarrow$$

$$\frac{f\left(x+\frac{\Delta x}{2}\right) - f\left(x-\frac{\Delta x}{2}\right)}{\Delta x} = ag(x)$$

$$\frac{g\left(x+\frac{\Delta x}{2}\right) - g\left(x-\frac{\Delta x}{2}\right)}{\Delta x} = bf(x)$$

This formulation will not work because it requires us to know the value of $f(x)$ and $g(x)$ at midpoints. These do not exist.

2D FDM 5

Use Terms Only Where They are Defined

Computational Methods in EE

$$\frac{\partial f(x)}{\partial x} = ag(x) \qquad \frac{\partial g(x)}{\partial x} = bf(x)$$

$$\Rightarrow$$

$$\frac{f(x+\Delta x) - f(x)}{\Delta x} = ag(x)$$

$$\frac{g(x+\Delta x) - g(x)}{\Delta x} = bf(x)$$

We have an important rule:

All terms in a finite-difference equation must exist at the same point.

We are violating this rule because the finite-differences on the left exist at the midpoints while the terms on the right do not.

2D FDM 6

Adopt a Staggered Grid

Computational Methods in EE

$$\frac{\partial f(x)}{\partial x} = ag(x) \qquad \frac{f(x+\Delta x)-f(x)}{\Delta x} = ag\left(x+\frac{\Delta x}{2}\right)$$

$$\frac{\partial g(x)}{\partial x} = bf(x) \qquad \frac{g\left(x+\frac{\Delta x}{2}\right)-g\left(x-\frac{\Delta x}{2}\right)}{\Delta x} = bf(x)$$

This works! We have "tighter" finite-difference approximations and each term exists at the same point.

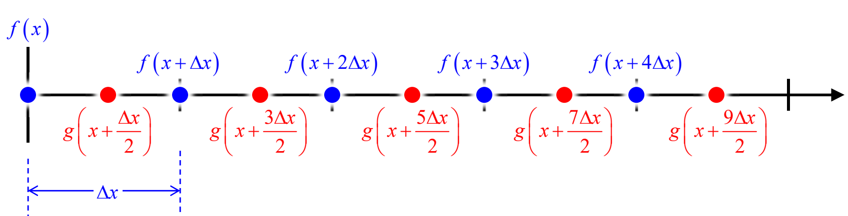
The only drawback is that we need to remember that $f(x)$ and $g(x)$ will be stored at physically different points.

1D Staggered Grid

Computational Methods in EE

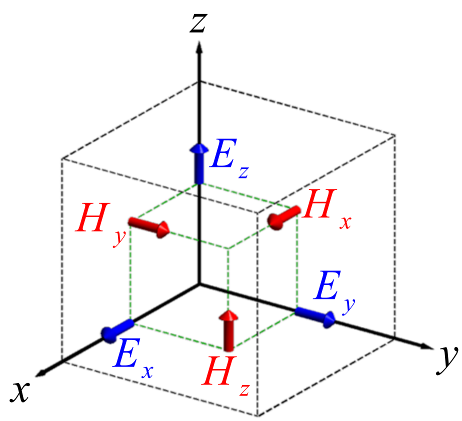
$$\frac{\partial f(x)}{\partial x} = ag(x) \qquad \frac{f(x+\Delta x)-f(x)}{\Delta x} = ag\left(x+\frac{\Delta x}{2}\right)$$

$$\frac{\partial g(x)}{\partial x} = bf(x) \qquad \frac{g\left(x+\frac{\Delta x}{2}\right)-g\left(x-\frac{\Delta x}{2}\right)}{\Delta x} = bf(x)$$



Staggering Functions in Three Dimensions

Computational Methods in EE



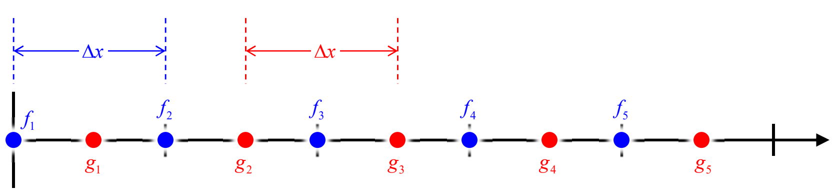
K. S. Yee, "Numerical solution of the initial boundary value problems involving Maxwell's equations in isotropic media," IEEE Trans. Microwave Theory and Techniques, vol. 44, pp. 61–69, 1966.

1D Staggered Grid in Terms of Array Indices

Computational Methods in EE

$$\frac{\partial f(x)}{\partial x} = ag(x) \qquad \Rightarrow \qquad \frac{f_{i+1} - f_i}{\Delta x} = ag_i$$

$$\frac{\partial g(x)}{\partial x} = bf(x) \qquad \qquad \frac{g_i - g_{i-1}}{\Delta x} = bf_i$$



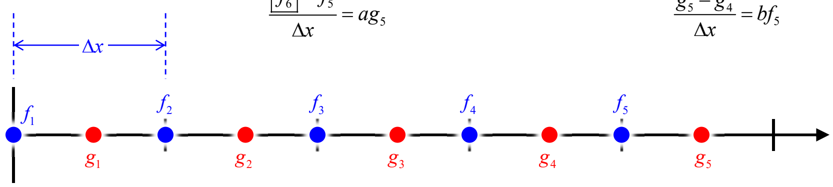
Derivative Matrices for Staggered Grids (1 of 2)

Computational Methods in EE

We write both of our finite-difference equations at each point on the grid.

$$\frac{f_{i+1}-f_i}{\Delta x}=ag_i \quad \Rightarrow \quad \begin{aligned} \frac{f_2-f_1}{\Delta x}&=ag_1\\ \frac{f_3-f_2}{\Delta x}&=ag_2\\ \frac{f_4-f_3}{\Delta x}&=ag_3\\ \frac{f_5-f_4}{\Delta x}&=ag_4\\ \frac{\boxed{f_6}-f_5}{\Delta x}&=ag_5 \end{aligned}$$

$$\frac{g_i-g_{i-1}}{\Delta x}=bf_i \quad \Rightarrow \quad \begin{aligned} \frac{g_1-\boxed{g_0}}{\Delta x}&=bf_1\\ \frac{g_2-g_1}{\Delta x}&=bf_2\\ \frac{g_3-g_2}{\Delta x}&=bf_3\\ \frac{g_4-g_3}{\Delta x}&=bf_4\\ \frac{g_5-g_4}{\Delta x}&=bf_5 \end{aligned}$$



Derivative Matrices for Staggered Grids (2 of 2)

Computational Methods in EE

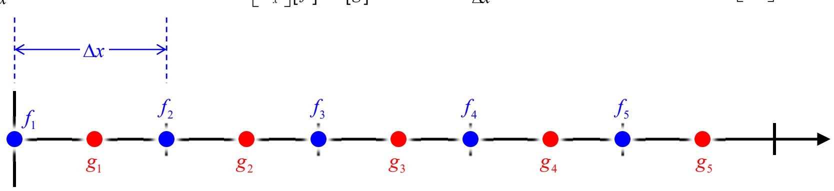
Notice the boundary condition fixes are incorporated at opposite sides of the grid for each derivative matrix.

$$\begin{aligned} \frac{f_2-f_1}{\Delta x}&=ag_1\\ \frac{f_3-f_2}{\Delta x}&=ag_2\\ \frac{f_4-f_3}{\Delta x}&=ag_3\\ \frac{f_5-f_4}{\Delta x}&=ag_4\\ \frac{\boxed{f_6}-f_5}{\Delta x}&=ag_5 \end{aligned} \quad \Rightarrow \quad \frac{1}{\Delta x}\begin{bmatrix} -1 & 1 & 0 & 0 & 0\\ 0 & -1 & 1 & 0 & 0\\ 0 & 0 & -1 & 1 & 0\\ 0 & 0 & 0 & -1 & 1\\ 0 & 0 & 0 & 0 & -1 \end{bmatrix}\begin{bmatrix} f_1\\ f_2\\ f_3\\ f_4\\ f_5 \end{bmatrix}=a\begin{bmatrix} g_1\\ g_2\\ g_3\\ g_4\\ g_5 \end{bmatrix}$$

$$\left[D_x^f\right]\left[f\right]=a\left[g\right]$$

$$\begin{aligned} \frac{g_1-\boxed{g_0}}{\Delta x}&=bf_1\\ \frac{g_2-g_1}{\Delta x}&=bf_2\\ \frac{g_3-g_2}{\Delta x}&=bf_3\\ \frac{g_4-g_3}{\Delta x}&=bf_4\\ \frac{g_5-g_4}{\Delta x}&=bf_5 \end{aligned} \quad \Rightarrow \quad \frac{1}{\Delta x}\begin{bmatrix} 1 & 0 & 0 & 0 & 0\\ -1 & 1 & 0 & 0 & 0\\ 0 & -1 & 1 & 0 & 0\\ 0 & 0 & -1 & 1 & 0\\ 0 & 0 & 0 & -1 & 1 \end{bmatrix}\begin{bmatrix} g_1\\ g_2\\ g_3\\ g_4\\ g_5 \end{bmatrix}=b\begin{bmatrix} f_1\\ f_2\\ f_3\\ f_4\\ f_5 \end{bmatrix}$$

$$\left[D_x^g\right]\left[g\right]=b\left[f\right]$$



Final Matrix Equations

Computational Methods in EE

Here is what we have done so far

$$\frac{\partial f(x)}{\partial x} = ag(x) \qquad \frac{\partial g(x)}{\partial x} = bf(x) \quad \Rightarrow \quad \left[D_x^f\right][f] = a[g] \qquad \left[D_x^g\right][g] = b[f]$$

With practice, this step will be obvious and not require any intermediate work for you.

We have two options for solving this system of matrix equations.

Solve Simultaneously

$$\begin{bmatrix} D_x^f & -a \\ -b & D_x^g \end{bmatrix} \begin{bmatrix} f \\ g \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Solve Individually

$$\left(\left[D_x^g\right]\left[D_x^f\right] - ab[I]\right)[f] = [0]$$

$$[g] = \frac{1}{a}\left[D_x^f\right][f]$$

Second-Order Derivative Matrix?

Computational Methods in EE

An interesting thing happened on the last slide.

$$\left(\left[D_x^g\right]\left[D_x^f\right] - ab[I]\right)[f] = [0]$$

Recall last time we tried to build a second-order derivate matrix? It did not work well.

$$\mathbf{D}_x^{(1)}\mathbf{D}_x^{(1)} = \frac{1}{(2\Delta x)^2}\begin{bmatrix} -1 & 0 & 1 & 0 & 0 \\ 0 & -2 & 0 & 1 & 0 \\ 1 & 0 & -2 & 0 & 1 \\ 0 & 1 & 0 & -2 & 0 \\ 0 & 0 & 1 & 0 & -1 \end{bmatrix} \qquad \mathbf{D}_x^{(2)} = \frac{1}{(\Delta x)^2}\begin{bmatrix} -2 & 1 & 0 & 0 & 0 \\ 1 & -2 & 1 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 \\ 0 & 0 & 1 & -2 & 1 \\ 0 & 0 & 0 & 1 & -2 \end{bmatrix}$$

So what is $\left[D_x^g\right]\left[D_x^f\right]$?

Perfect!

$$\left[D_x^g\right]\left[D_x^f\right] = \frac{1}{\Delta x}\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & -1 & 1 \end{bmatrix} \cdot \frac{1}{\Delta x}\begin{bmatrix} -1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 0 & -1 \end{bmatrix} = \frac{1}{(\Delta x)^2}\begin{bmatrix} -2 & 1 & 0 & 0 & 0 \\ 1 & -2 & 1 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 \\ 0 & 0 & 1 & -2 & 1 \\ 0 & 0 & 0 & 1 & -2 \end{bmatrix}$$

$\left[D_x^f\right]$ and $\left[D_x^g\right]$ are Related

Computational Methods in EE

After observing our two derivative matrices,

$$\left[D_x^f\right]=\frac{1}{\Delta x}\begin{bmatrix}-1 & 1 & 0 & 0 & 0\\ 0 & -1 & 1 & 0 & 0\\ 0 & 0 & -1 & 1 & 0\\ 0 & 0 & 0 & -1 & 1\\ 0 & 0 & 0 & 0 & -1\end{bmatrix} \qquad \left[D_x^g\right]=\frac{1}{\Delta x}\begin{bmatrix}1 & 0 & 0 & 0 & 0\\ -1 & 1 & 0 & 0 & 0\\ 0 & -1 & 1 & 0 & 0\\ 0 & 0 & -1 & 1 & 0\\ 0 & 0 & 0 & -1 & 1\end{bmatrix}$$

We see they are related through

$$\left[D_x^g\right]=-\left[D_x^f\right]^H$$

This holds for most of the comment boundary conditions.

Confirmed BC's
- Dirichlet
- Periodic
- Floquet

Exceptions
- Neumann
- ??

Final Notes

Computational Methods in EE

- It is usually best to form matrix equations early in the formulation process.
- We usually only ever need first-order derivative matrices. If we stagger our functions, we can just multiply the first-order derivative matrices to get any order we wish.
- For most boundary conditions, the derivative matrices are related through

$$\left[D_x^g\right]=-\left[D_x^f\right]^H$$

This means we actually only need to build one derivative matrix.

- Even more magic happens wh we have expressions like

$$\frac{\partial}{\partial x}\left[a(x)\frac{\partial}{\partial y}\right] \quad \rightarrow \quad \left[D_x^g\right]\left[A\right]\left[D_x^f\right]$$

This is all we have to do! ☺

Example (1 of 5)

Computational Methods in EE

Solve the following coupled differential equation in the interval $1 < x < 10$.

$$\frac{df(x)}{dx} = 2g(x)$$
$$\frac{dg(x)}{dx} = -f(x)$$
$$f(0) = 1 \qquad f(10) = 0$$

2D FDM 17

Example (2 of 5)

Computational Methods in EE

Formulation

Step 1 – Write equations in matrix form

$$\begin{aligned}\frac{df(x)}{dx} &= 2g(x) \\ \frac{dg(x)}{dx} &= -f(x)\end{aligned} \quad \rightarrow \quad \begin{aligned}\left[D_x^f\right][f] &= 2[g] \\ \left[D_x^g\right][g] &= -[f]\end{aligned}$$

Step 2 – Solve first equation for $[g]$

$$[g] = 0.5\left[D_x^f\right][f]$$

Step 3 – Substitute this expression for $[g]$ into second equation and simplify.

$$\left[D_x^g\right][g] = -[f]$$
$$\left[D_x^g\right]\left(0.5\left[D_x^f\right][f]\right) = -[f]$$
$$\left[D_x^g\right]\left[D_x^f\right][f] + 2[f] = [0]$$
$$\left(\left[D_x^g\right]\left[D_x^f\right] + 2[I]\right)[f] = [0]$$

Step 4 – Write final matrix equation in standard form.

$$[A][f] = [0]$$
$$[A] = \left[D_x^g\right]\left[D_x^f\right] + 2[I]$$

2D FDM 18

Example (3 of 5)

Computational Methods in EE

Implementation

Step 1 – Initialize MATLAB

```
% Lecture7c_example1.m

% INITIALIZE MATLAB
close all;
clc;
clear all;

% OPEN FIGURE WINDOW
fig = figure('Color','w');

% DEFINE PROBLEM PARAMETERS
xa = 0;
xb = 10;
fa = 1;
fb = 0;
Nx = 1000;
```

Step 2 – Calculate Grid

```
% COMPUTE GRID
dx = (xb - xa)/(Nx - 1);
x  = linspace(xa,xb,Nx);
```

Step 3 – Build Derivative Matrices

```
% BUILD DERIVATIVE MATRICES
DFX = sparse(Nx,Nx);
DFX = spdiags(-ones(Nx,1),0,DFX);
DFX = spdiags(+ones(Nx,1),1,DFX);
DFX = DFX / dx;

DGX = - DFX';
```

2D FDM 19

Example (4 of 5)

Computational Methods in EE

Implementation

Step 4 – Build Standard Matrix Equation

```
% BUILD A (DIRICHLET)
I = speye(Nx,Nx);
A = DGX*DFX + 2*I;
```

Step 5 – Incorporate Boundary Values

```
% INCORPORATE BOUNDARY VALUES
b = zeros(Nx,1);

A(1,:) = 0;
A(1,1) = 1;
b(1)   = fa;

A(Nx,:)  = 0;
A(Nx,Nx) = 1;
b(Nx)    = fb;
```

Step 6 – Solve Matrix Equation

```
% SOLVE MATRIX EQUATION
f = A\b;
g = 0.5*DFX*f;
```

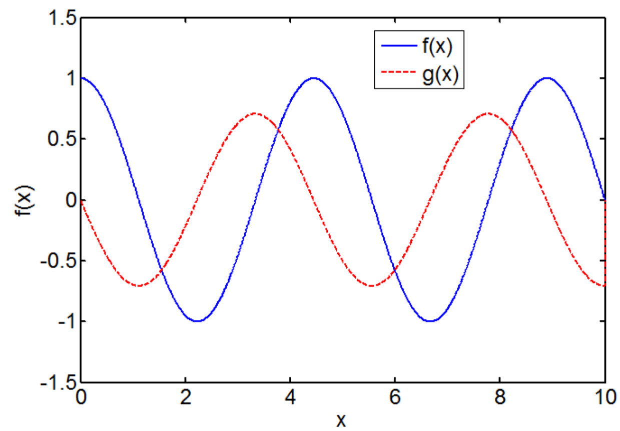
Step 7 – Visualize Results

```
% VISUALIZE RESULTS
h = plot(x,f,'-b','LineWidth',2); hold
on;
plot(x,g,'-r','LineWidth',2); hold off;
h2 = get(h,'Parent');
set(h2,'LineWidth',2,'FontSize',18);
xlabel('x');
ylabel('f(x)');
```

2D FDM 20

Example (5 of 5)

Computational Methods in EE



Computational Methods in EE

Finite-Difference Method in Two Dimensions

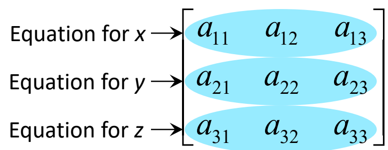
Interpretation of Matrices

Computational Methods in EE

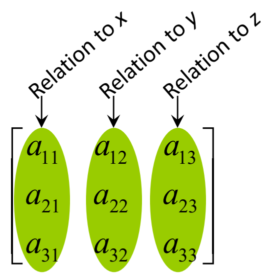
$$a_{11}x + a_{12}y + a_{13}z = b_1$$
$$a_{21}x + a_{22}y + a_{23}z = b_2$$
$$a_{31}x + a_{32}y + a_{33}z = b_3$$

$$\Rightarrow \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

EQUATION FOR…



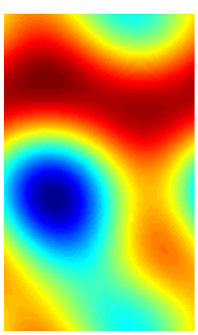
RELATION TO…



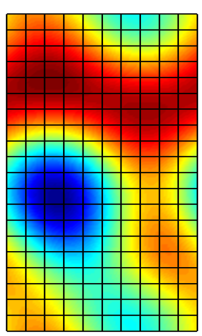
Representing Functions on a Grid

Computational Methods in EE

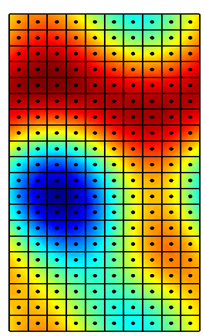
Example physical (continuous) 2D function



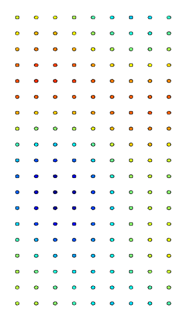
A grid is constructed by dividing space into discrete cells



Function is known only at discrete points



Representation of what is actually stored in memory

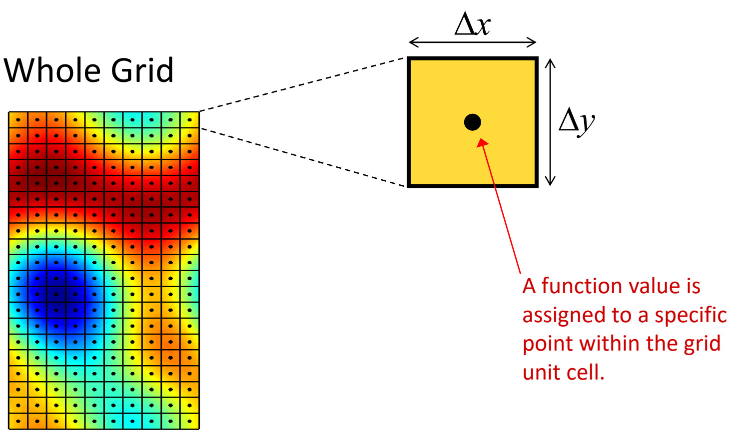


Grid Unit Cell

Computational
Methods in EE

A Single Unit Cell

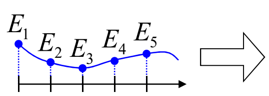
Whole Grid



Functions are Put Into Column Vectors

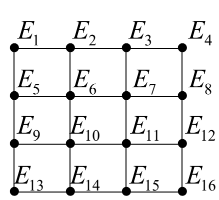
Computational
Methods in EE

1-D Systems



$$\mathbf{E} = \begin{bmatrix} E_1 \\ E_2 \\ E_3 \\ E_4 \\ E_5 \end{bmatrix}$$

2-D Systems

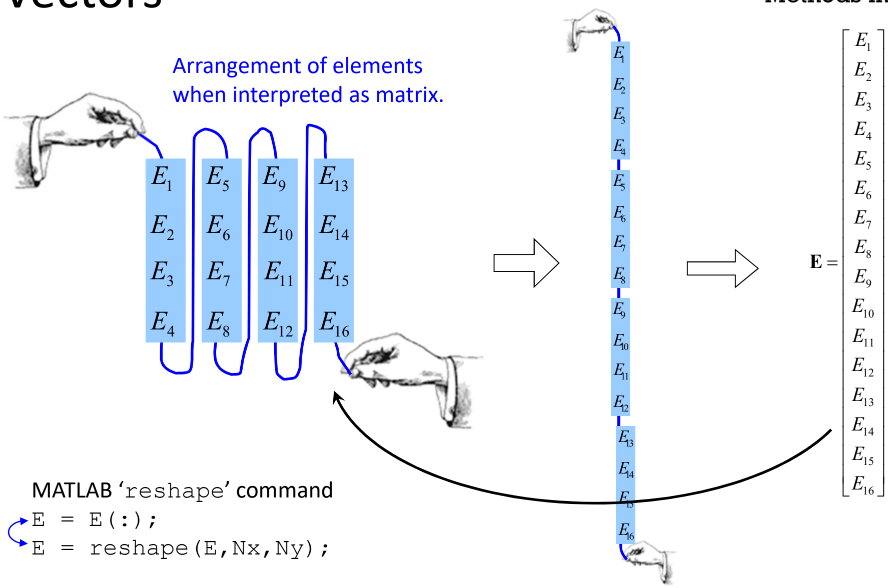


Arrangement of elements when interpreted as a 2D array.

$$\mathbf{E} = \begin{bmatrix} E_1 \\ E_2 \\ E_3 \\ E_4 \\ E_5 \\ E_6 \\ E_7 \\ E_8 \\ E_9 \\ E_{10} \\ E_{11} \\ E_{12} \\ E_{13} \\ E_{14} \\ E_{15} \\ E_{16} \end{bmatrix}$$

Putting Functions into Column Vectors

Computational Methods in EE



MATLAB ‘reshape’ command

```
E = E(:);
E = reshape(E,Nx,Ny);
```


Computational Methods in EE

2D FDM on Collocated Grid

Finite-Difference Approximations on a Two-Dimensional Grid

Computational Methods in EE

Derivatives in the x direction

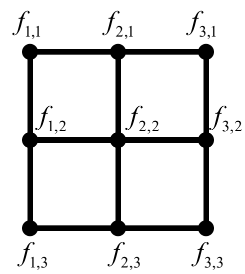
$$\frac{\partial f_{i,j}}{\partial x} \cong \frac{f_{i+1,j} - f_{i-1,j}}{2\Delta x}$$

$$\frac{\partial^2 f_{i,j}}{\partial x^2} \cong \frac{f_{i+1,j} - 2f_{i,j} + f_{i-1,j}}{(\Delta x)^2}$$

Derivatives in the y direction

$$\frac{\partial f_{i,j}}{\partial y} \cong \frac{f_{i,j+1} - f_{i,j-1}}{2\Delta y}$$

$$\frac{\partial^2 f_{i,j}}{\partial y^2} \cong \frac{f_{i,j+1} - 2f_{i,j} + f_{i,j-1}}{(\Delta y)^2}$$



$[D_x]$ on a 3×3 Grid

Computational Methods in EE

$$\frac{1}{2\Delta x}\left[\quad ? \quad\right]\begin{bmatrix} f_{1,1} \\ f_{2,1} \\ f_{3,1} \\ f_{1,2} \\ f_{2,2} \\ f_{3,2} \\ f_{1,3} \\ f_{2,3} \\ f_{3,3} \end{bmatrix} = \frac{1}{2\Delta x}\begin{bmatrix} f_{2,1} - f_{0,1} \\ f_{3,1} - f_{1,1} \\ f_{4,1} - f_{2,1} \\ f_{2,2} - f_{0,2} \\ f_{3,2} - f_{1,2} \\ f_{4,2} - f_{2,2} \\ f_{2,3} - f_{0,3} \\ f_{3,3} - f_{1,3} \\ f_{4,3} - f_{2,3} \end{bmatrix}$$

$\left[D_x\right]$ on a 3×3 Grid

Computational Methods in EE

$$\frac{1}{2\Delta x}\begin{bmatrix} 0 & 1 & & & & & & & \\ -1 & 0 & 1 & & & & & & \\ & -1 & 0 & 0 & & & & & \\ & & 0 & 0 & 1 & & & & \\ & & & -1 & 0 & 1 & & & \\ & & & & -1 & 0 & 0 & & \\ & & & & & 0 & 0 & 1 & \\ & & & & & & -1 & 0 & 1 \\ & & & & & & & -1 & 0 \end{bmatrix}\begin{bmatrix} f_{1,1} \\ f_{2,1} \\ f_{3,1} \\ f_{1,2} \\ f_{2,2} \\ f_{3,2} \\ f_{1,3} \\ f_{2,3} \\ f_{3,3} \end{bmatrix} = \frac{1}{2\Delta x}\begin{bmatrix} f_{2,1} - f_{0,1} \\ f_{3,1} - f_{1,1} \\ f_{4,1} - f_{2,1} \\ f_{2,2} - f_{0,2} \\ f_{3,2} - f_{1,2} \\ f_{4,2} - f_{2,2} \\ f_{2,3} - f_{0,3} \\ f_{3,3} - f_{1,3} \\ f_{4,3} - f_{2,3} \end{bmatrix}$$

$\left[D_y\right]$ on a 3×3 Grid

Computational Methods in EE

$$\frac{1}{2\Delta y}\begin{bmatrix} & & \\ & ? & \\ & & \end{bmatrix}\begin{bmatrix} f_{1,1} \\ f_{2,1} \\ f_{3,1} \\ f_{1,2} \\ f_{2,2} \\ f_{3,2} \\ f_{1,3} \\ f_{2,3} \\ f_{3,3} \end{bmatrix} = \frac{1}{2\Delta y}\begin{bmatrix} f_{1,2} - f_{1,0} \\ f_{2,2} - f_{2,0} \\ f_{3,2} - f_{3,0} \\ f_{1,3} - f_{1,1} \\ f_{2,3} - f_{2,1} \\ f_{3,3} - f_{3,1} \\ f_{1,4} - f_{1,2} \\ f_{2,4} - f_{2,2} \\ f_{3,4} - f_{3,2} \end{bmatrix}$$

$\left[D_y\right]$ on a 3×3 Grid

Computational Methods in EE

$$\frac{1}{2\Delta y}\begin{bmatrix} & & & 1 & & & & & \\ & & & & 1 & & & & \\ & & & & & 1 & & & \\ -1 & & & & & & 1 & & \\ & -1 & & & & & & 1 & \\ & & -1 & & & & & & 1 \\ & & & -1 & & & & & \\ & & & & -1 & & & & \\ & & & & & -1 & & & \end{bmatrix}\begin{bmatrix} f_{1,1} \\ f_{2,1} \\ f_{3,1} \\ f_{1,2} \\ f_{2,2} \\ f_{3,2} \\ f_{1,3} \\ f_{2,3} \\ f_{3,3} \end{bmatrix} = \frac{1}{2\Delta y}\begin{bmatrix} f_{1,2} - f_{1,0} \\ f_{2,2} - f_{2,0} \\ f_{3,2} - f_{3,0} \\ f_{1,3} - f_{1,1} \\ f_{2,3} - f_{2,1} \\ f_{3,3} - f_{3,1} \\ f_{1,4} - f_{1,2} \\ f_{2,4} - f_{2,2} \\ f_{3,4} - f_{3,2} \end{bmatrix}$$

Computational Methods in EE

2D FDM on Staggered Grid

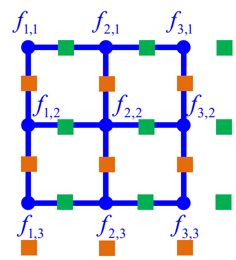
Finite-Difference Approximations on a Two-Dimensional Grid

Computational Methods in EE

Derivatives of $f(x)$

$$\frac{\partial f_{i,j}}{\partial x} \cong \frac{f_{i+1,j} - f_{i,j}}{\Delta x}$$

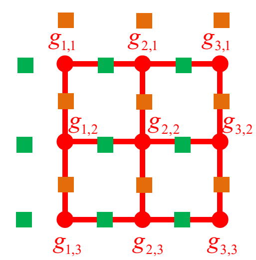
$$\frac{\partial f_{i,j}}{\partial y} \cong \frac{f_{i,j+1} - f_{i,j}}{\Delta y}$$



Derivatives of $g(x)$

$$\frac{\partial g_{i,j}}{\partial x} \cong \frac{g_{i,j} - g_{i-1,j}}{\Delta x}$$

$$\frac{\partial g_{i,j}}{\partial y} \cong \frac{g_{i,j} - g_{i,j-1}}{\Delta y}$$



Derivative Operators on 3x3 Grid Using Dirichlet Boundary Conditions

Computational Methods in EE

$$\mathbf{D}_x^f = \frac{1}{\Delta_x}\begin{bmatrix} -1 & 1 & & & & & & & \\ & -1 & 1 & & & & & & \\ & & -1 & 0 & & & & & \\ & & & -1 & 1 & & & & \\ & & & & -1 & 1 & & & \\ & & & & & -1 & 0 & & \\ & & & & & & -1 & 1 & \\ & & & & & & & -1 & 1 \\ & & & & & & & & -1 \end{bmatrix}$$

Dirichlet boundary conditions; N_x

$$\mathbf{D}_y^f = \frac{1}{\Delta_y}\begin{bmatrix} -1 & 0 & 0 & 1 & & & & & \\ & -1 & 0 & 0 & 1 & & & & \\ & & -1 & 0 & 0 & 1 & & & \\ & & & -1 & 0 & 0 & 1 & & \\ & & & & -1 & 0 & 0 & 1 & \\ & & & & & -1 & 0 & 0 & 1 \\ & & & & & & -1 & 0 & 0 \\ & & & & & & & -1 & 0 \\ & & & & & & & & -1 \end{bmatrix}$$

N_x

$$\mathbf{D}_x^g = \frac{1}{\Delta_x}\begin{bmatrix} 1 & & & & & & & & \\ -1 & 1 & & & & & & & \\ & -1 & 1 & & & & & & \\ & & 0 & 1 & & & & & \\ & & & -1 & 1 & & & & \\ & & & & -1 & 1 & & & \\ & & & & & 0 & 1 & & \\ & & & & & & -1 & 1 & \\ & & & & & & & -1 & 1 \end{bmatrix}$$

Dirichlet boundary conditions; N_x

$$\mathbf{D}_y^g = \frac{1}{\Delta_y}\begin{bmatrix} 1 & & & & & & & & \\ 0 & 1 & & & & & & & \\ 0 & 0 & 1 & & & & & & \\ -1 & 0 & 0 & 1 & & & & & \\ & -1 & 0 & 0 & 1 & & & & \\ & & -1 & 0 & 0 & 1 & & & \\ & & & -1 & 0 & 0 & 1 & & \\ & & & & -1 & 0 & 0 & 1 & \\ & & & & & -1 & 0 & 0 & 1 \end{bmatrix}$$

N_x

Same Operators on 4x4 Grid

Computational Methods in EE

$$\mathbf{D}_x^f = \frac{1}{\Delta x}\begin{bmatrix}
-1 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & +1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & +1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & +1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & +1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & +1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & +1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1
\end{bmatrix}$$

$$\mathbf{D}_y^f = \frac{1}{\Delta y}\begin{bmatrix}
-1 & 0 & 0 & 0 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & 0 & 0 & 0 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 0 & 0 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & 0 & 0 & 0 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & +1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & +1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & +1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & +1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & +1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & +1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & +1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1
\end{bmatrix}$$

$$\mathbf{D}_x^g = \frac{1}{\Delta x}\begin{bmatrix}
+1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & +1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & +1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & +1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & +1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & +1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & +1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & +1
\end{bmatrix}$$

$$\mathbf{D}_y^g = \frac{1}{\Delta y}\begin{bmatrix}
+1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & 0 & 0 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & 0 & 0 & 0 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 0 & 0 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & 0 & 0 & 0 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & +1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & +1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & +1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & +1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & +1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & +1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & +1
\end{bmatrix}$$

Placing Diagonals in Sparse Matrices in MATLAB

Computational Methods in EE

```
M = 6;
Z = sparse(M,M);
d = ones(M,1);
A = spdiags(d,0,Z);
```

⇨

```
A =
[ 1   0   0   0   0   0   ]
[ 0   1   0   0   0   0   ]
[ 0   0   1   0   0   0   ]
[ 0   0   0   1   0   0   ]
[ 0   0   0   0   1   0   ]
[ 0   0   0   0   0   1   ]
```

```
M = 6;
Z = sparse(M,M);
d = ones(M,1);
A = spdiags(-d,-1,Z);
A = spdiags(+d,+1,A);
```

⇨

```
A =
[  0    1    0    0    0    0   ]
[ -1    0    1    0    0    0   ]
[  0   -1    0    1    0    0   ]
[  0    0   -1    0    1    0   ]
[  0    0    0   -1    0    1   ]
[  0    0    0    0   -1    0   ]
```

Relationship Between the Derivative Matrices

Computational Methods in EE

Transpose Operation

$$\left[\mathbf{A}^{\mathrm{T}}\right]_{i,j} = \left[\mathbf{A}\right]_{j,i} \qquad \mathbf{A} = \begin{bmatrix} 1 & 2 \end{bmatrix}, \mathbf{A}^{\mathrm{T}} = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \qquad \mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, \mathbf{A}^{\mathrm{T}} = \begin{bmatrix} 1 & 3 \\ 2 & 4 \end{bmatrix} \qquad \mathbf{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}, \mathbf{A}^{\mathrm{T}} = \begin{bmatrix} a_{11} & a_{21} \\ a_{22} & a_{22} \end{bmatrix}$$

```
AT = transpose(A);        AT = A.';
```

Hermitian (Conjugate) Transpose Operation

$$\left[\mathbf{A}^{\mathrm{H}}\right]_{i,j} = \left[\mathbf{A}\right]^{*}_{j,i} \qquad \mathbf{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}, \mathbf{A}^{\mathrm{H}} = \begin{bmatrix} a^{*}_{11} & a^{*}_{21} \\ a^{*}_{12} & a^{*}_{22} \end{bmatrix} \qquad \mathbf{A} = \begin{bmatrix} (1+4j) & (2-3j) \\ (3-2j) & (4+j) \end{bmatrix}, \mathbf{A}^{\mathrm{H}} = \begin{bmatrix} (1-4j) & (3+2j) \\ (2+3j) & (4-j) \end{bmatrix}$$

```
AT = ctranspose(A);       AT = A';
```

Relationship Between the Derivative Operators

$$\mathbf{D}_x^g = -\left[\mathbf{D}_x^f\right]^H \qquad \mathbf{D}_y^g = -\left[\mathbf{D}_y^f\right]^H$$

```
DGX = -DFX';          DGY = -DFY';
```

This means you only have to construct derivative operators for $f(x)$. The derivative operators for $g(x)$ can be computed directly from the derivative operators for $f(x)$.

This relationship is not valid for some boundary conditions, like Neumann.

Computational Methods in EE

Calculating Derivative Matrices for 1D Grids from Derivative Matrices on 2D Grids

The Rule Using Dirichlet Boundary Conditions

Computational Methods in EE

$$\text{If Nx} = 1, \text{ then } \left[D_x\right] = \left[D_x^2\right] = \cdots = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}$$

$$\text{If Ny} = 1, \text{ then } \left[D_y\right] = \left[D_y^2\right] = \cdots = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}$$

These are all sparse matrices of all zeros.