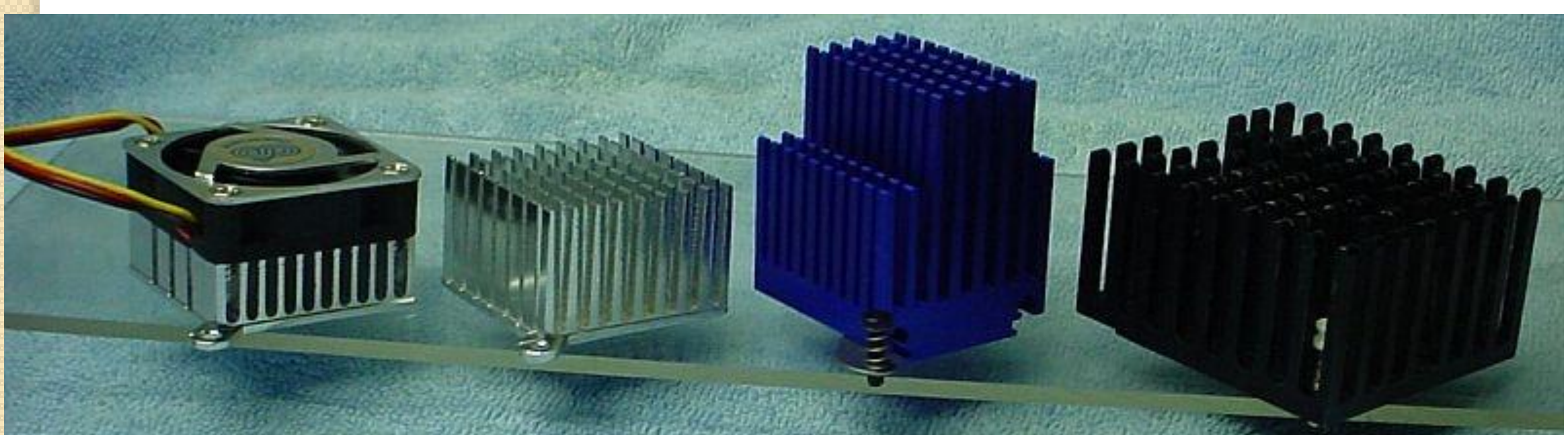
Boundary Value Problems

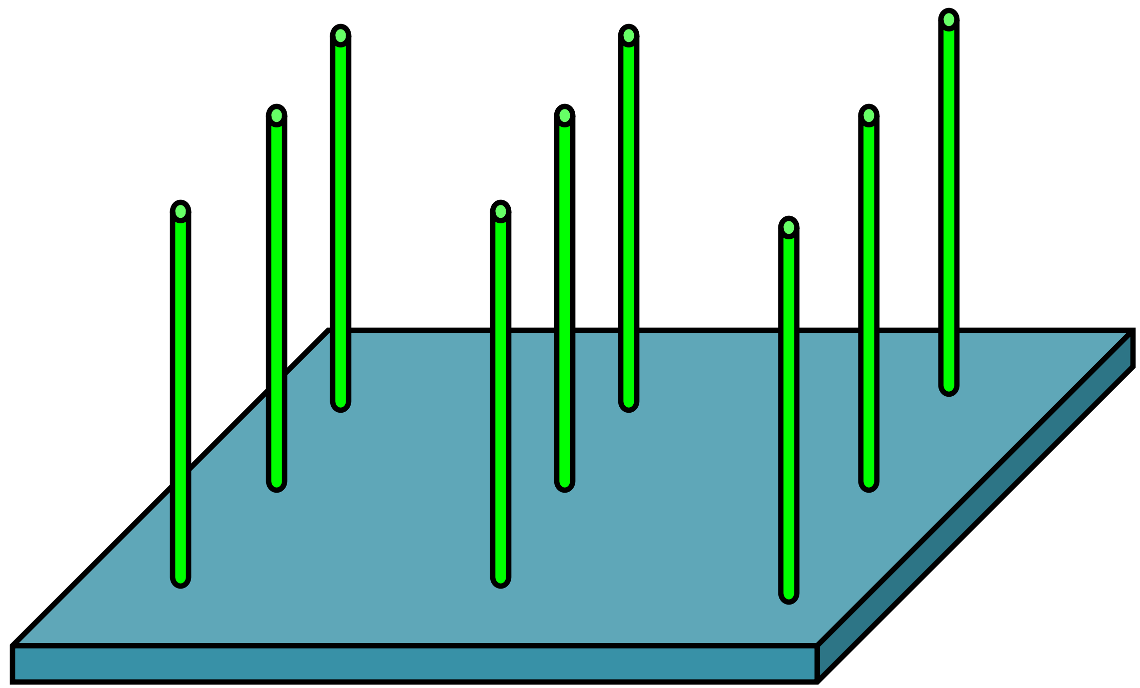
Jake Blanchard
University of Wisconsin - Madison
Spring 2008

Case Study

- We will analyze a cooling configuration for a computer chip
- We increase cooling by adding a number of fins to the surface
- These are high conductivity (aluminum) pins which provide added surface area



The Case - Schematic



The Case - Modeling

- The temperature distribution in the pin is governed by:

$$\boxed{\begin{aligned} &\frac{d^2T}{dx^2} - \frac{hC}{kA}(T - T_f) = 0 \\ &T(0) = 40C \\ &\frac{dT}{dx}\bigg|_{x=L} = 0 \end{aligned}}$$

Finite Difference Techniques

- Used to solve boundary value problems
- We'll look at an example

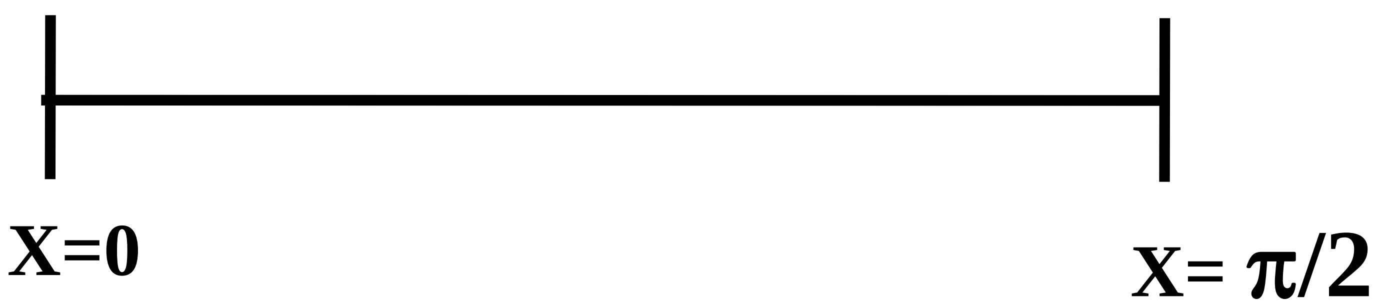
$$\boxed{\frac{d^2y}{dx^2} + y = 1}$$

$$\boxed{\begin{array}{l} y(0) = 1 \\ y(\frac{\pi}{2}) = 0 \end{array}}$$

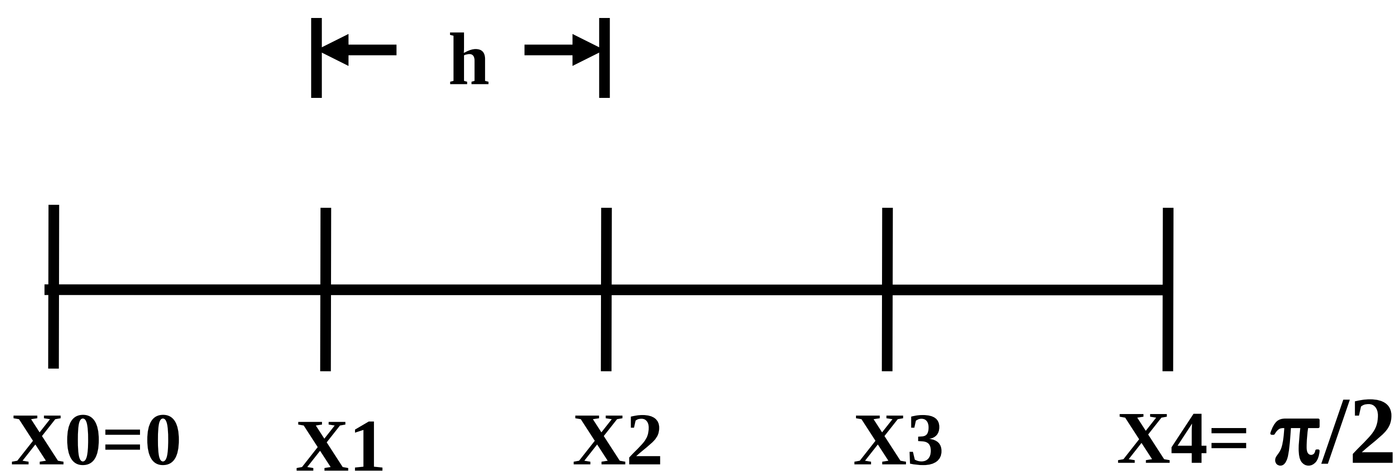
Two Steps

- Divide interval into steps
- Write differential equation in terms of values at these discrete points

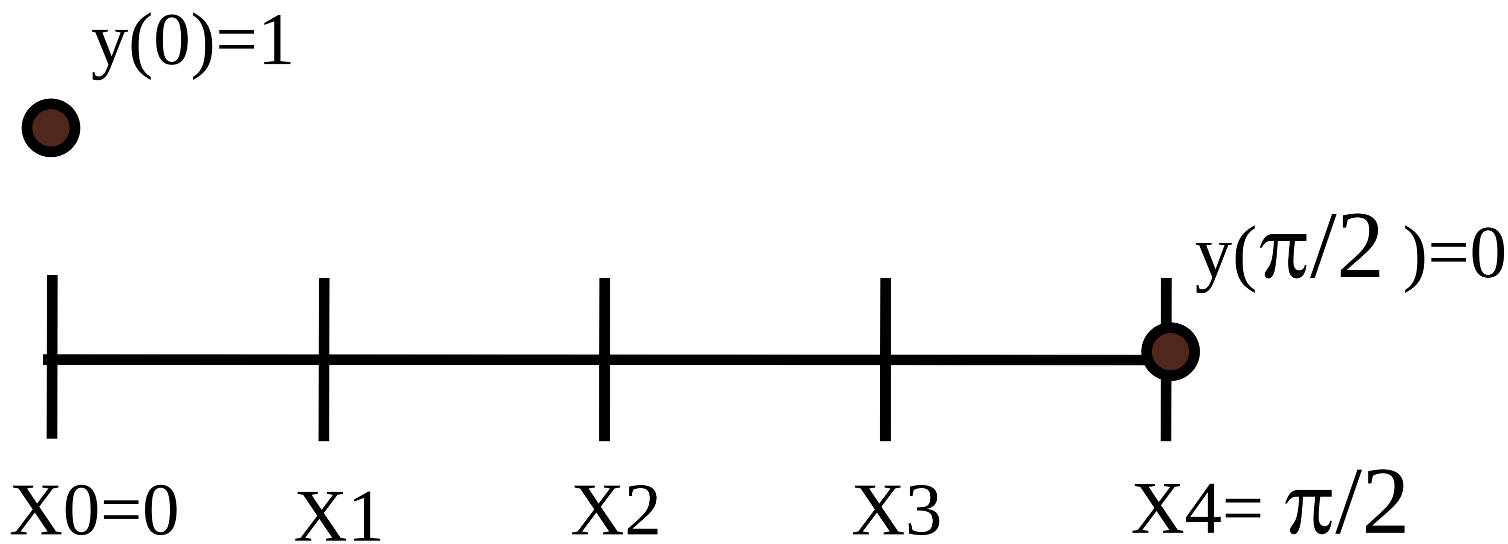
Solution is Desired from x=0 to $\pi/2$



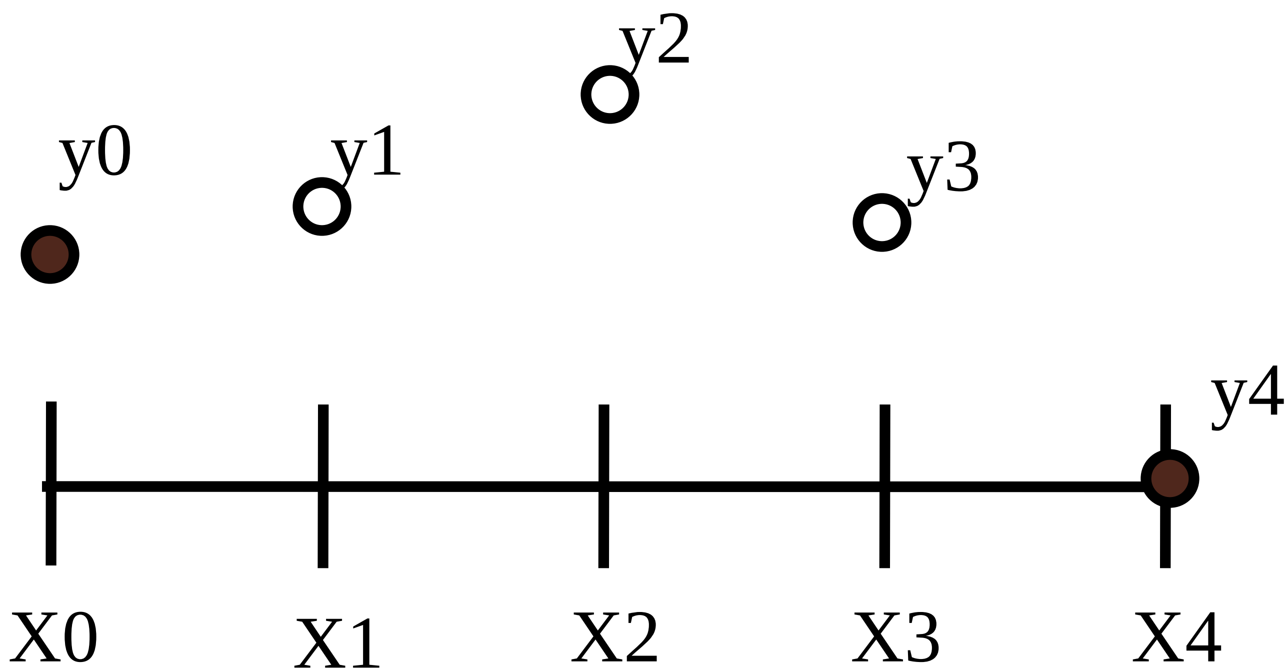
Divide Interval into Pieces



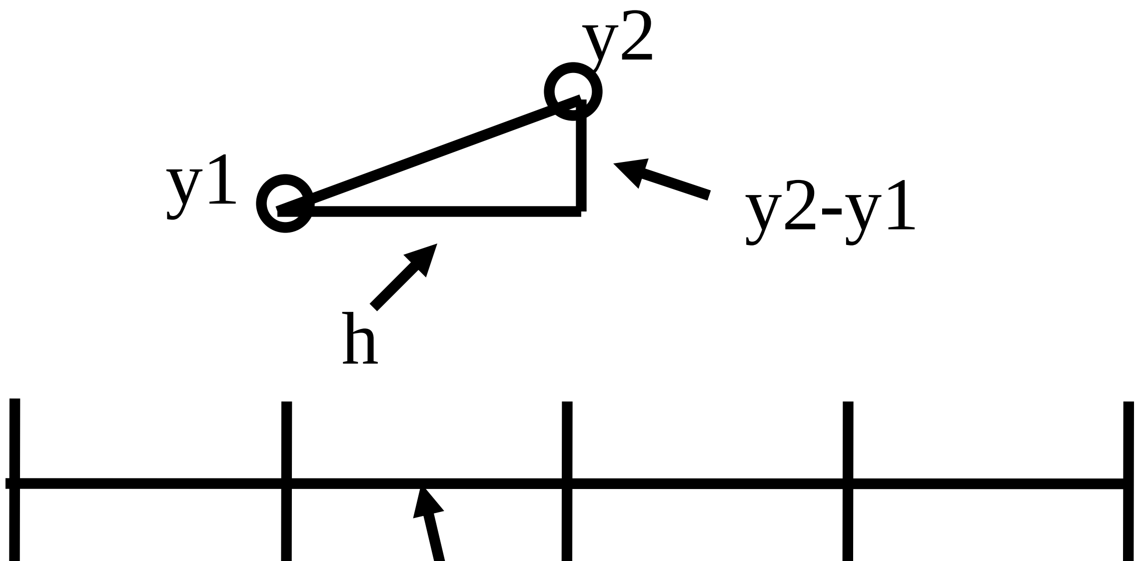
Boundary Values



Calculate Internal Values



Approximations



$$\boxed{\left.\frac{dy}{dx}\right|_{i+1/2} \approx \frac{y_{i+1} - y_i}{h}}$$

Second Derivative

$$\boxed{\left.\frac{d^2 y}{dx^2}\right|_i \approx \frac{y_{i+1} - 2y_i + y_{i-1}}{h^2}}$$

Substitute

$$\frac{d^2y}{dx^2} + y = 1$$

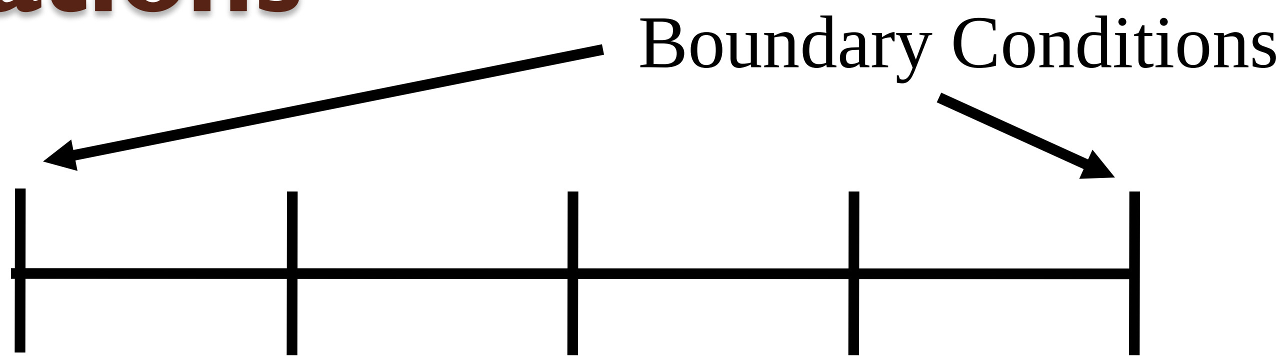
becomes

$$\frac{y_{i-1} - 2y_i + y_{i+1}}{h^2} + y_i = 1$$

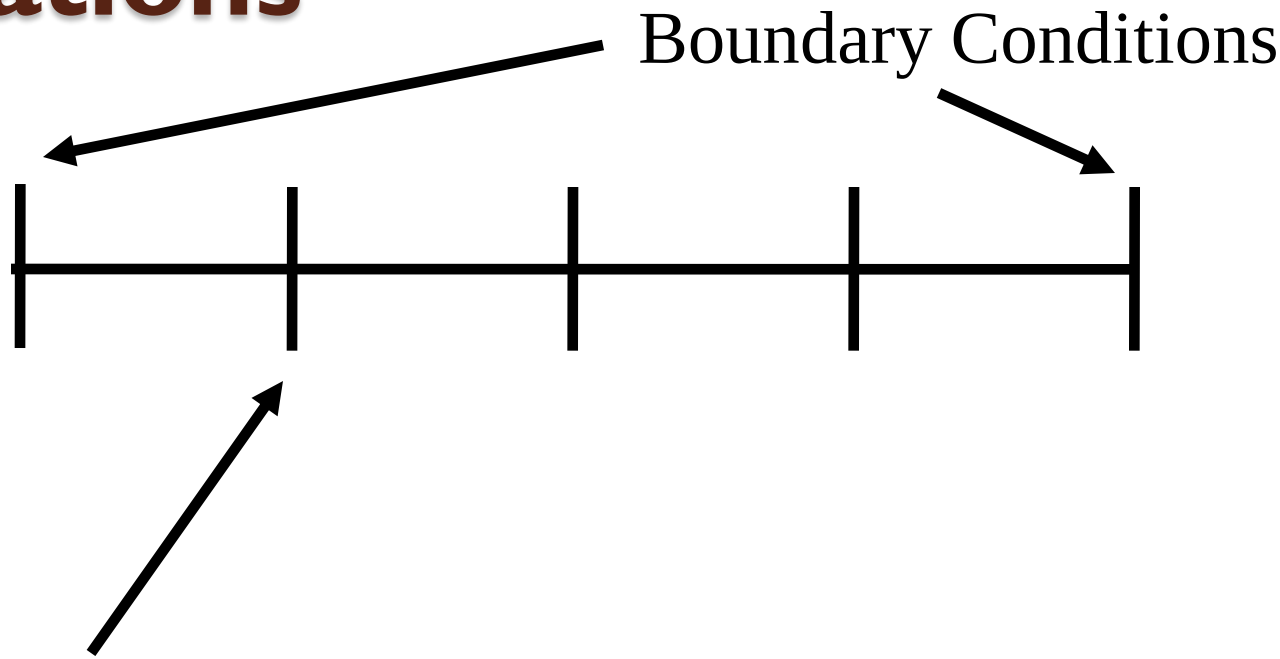
or

$$y_{i-1} - (2 - h^2)y_i + y_{i+1} = h^2$$

Equations

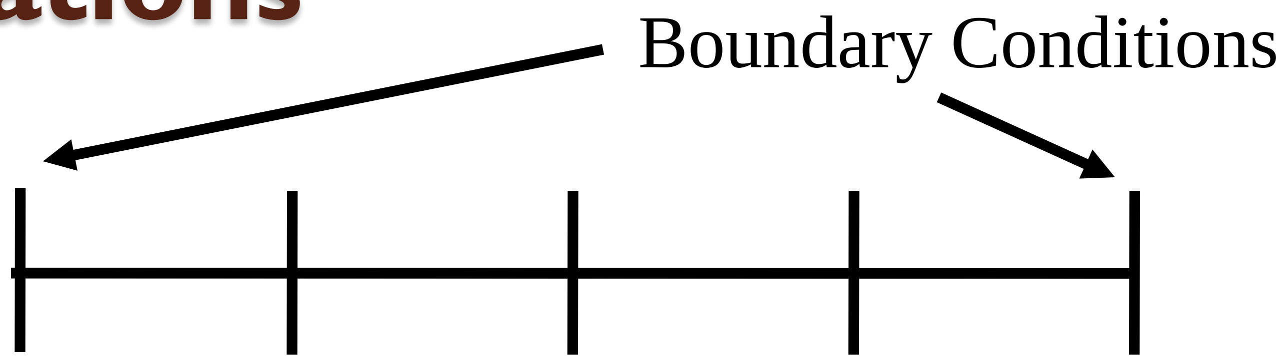


Equations



$$\boxed{y_2 - 2y_1 + y_0 + h^2 y_1 = h^2}$$

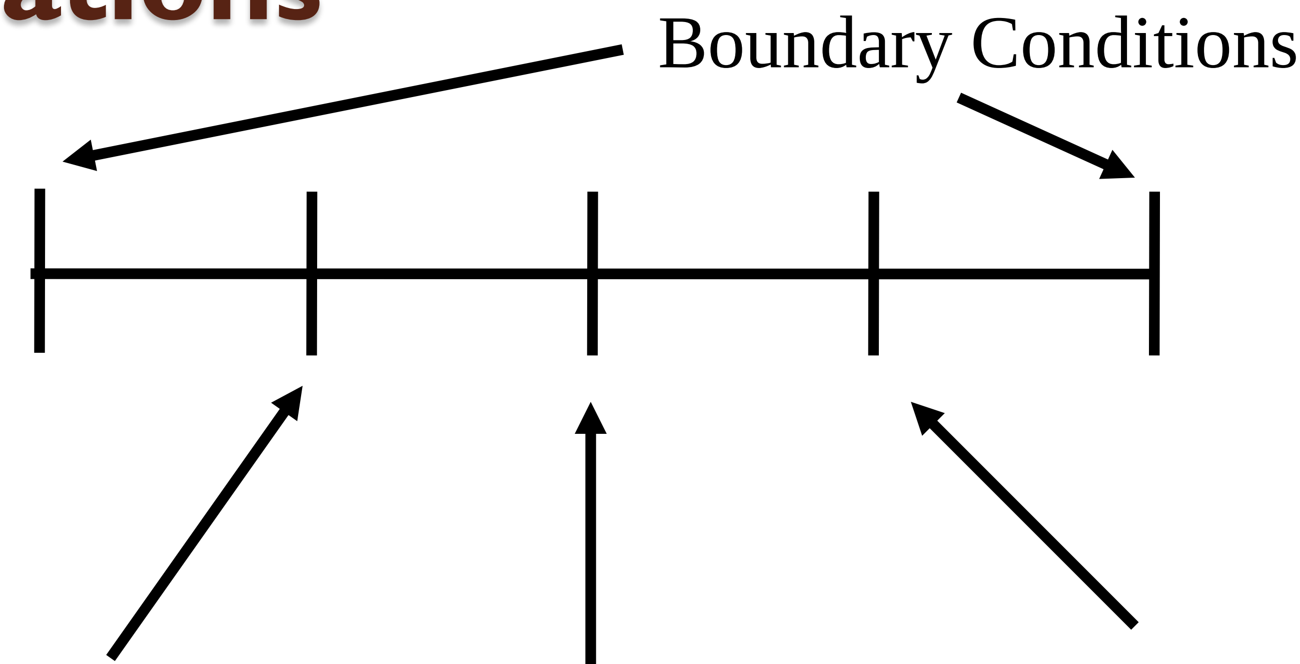
Equations



$$\boxed{y_2 - 2y_1 + y_0 + h^2 y_1 = h^2}$$

$$\boxed{y_3 - 2y_2 + y_1 + h^2 y_2 = h^2}$$

Equations



$$\boxed{y_2 - 2y_1 + y_0 + h^2 y_1 = h^2}$$

$$\boxed{y_4 - 2y_3 + y_2 + h^2 y_3 = h^2}$$

$$\boxed{y_3 - 2y_2 + y_1 + h^2 y_2 = h^2}$$

Using Matlab

$$\boxed{\begin{aligned}
&y_0 = 1\\
&y_0 - (2-h^2)y_1 + y_2 = h^2\\
&y_1 - (2-h^2)y_2 + y_3 = h^2\\
&y_2 - (2-h^2)y_3 + y_4 = h^2\\
&y_4 = 0
\end{aligned}}$$

Convert to matrix and solve

Using Matlab

$$
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
1 & -(2-h^2) & 1 & 0 & 0 \\
0 & 1 & -(2-h^2) & 1 & 0 \\
0 & 0 & 1 & -(2-h^2) & 1 \\
0 & 0 & 0 & 0 & 1
\end{bmatrix}
\begin{Bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \end{Bmatrix}
=
\begin{Bmatrix} 1 \\ h^2 \\ h^2 \\ h^2 \\ 0 \end{Bmatrix}
$$

The Script

```
h=pi/2/4;
A=[1 0 0 0 0; 1 -(2-h^2) 1 0 0; 0 1 -(2-h^2) 1 0;
  ...
   0 0 1 -(2-h^2) 1; 0 0 0 0 1];
b=[1; h^2; h^2; h^2; 0];
x=linspace(0,pi/2,5);
y=A\b;
plot(x,y)
```

What if we use N divisions

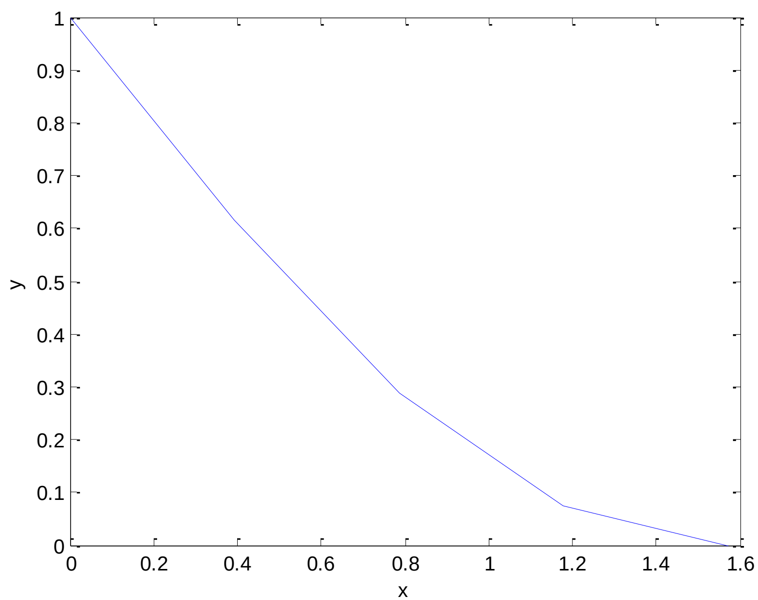
- N divisions, N+1 mesh points
- Matrix is N+1 by N+1

$$\boxed{h = \frac{\pi/2}{N}}$$

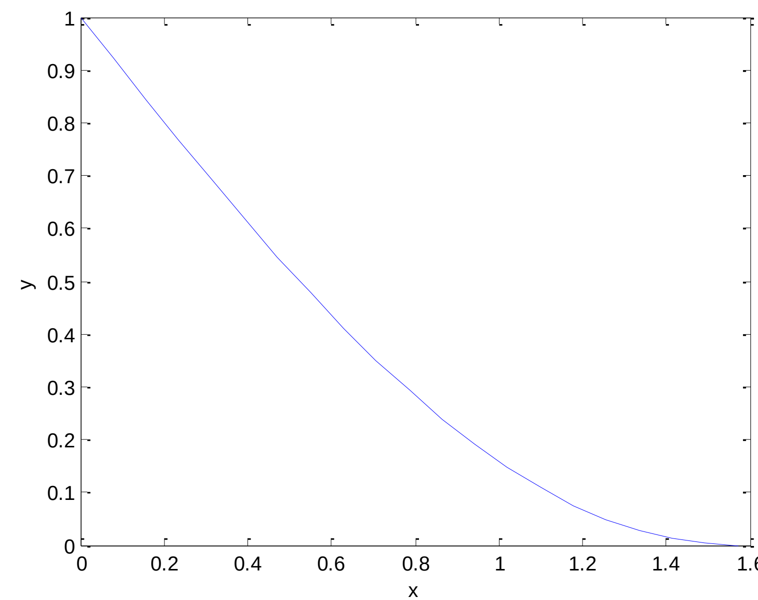
The Script

```
N=4;
h=pi/2/N;
A=-(2-h^2)*eye(N+1);
A(1,1)=1;
A(N+1,N+1)=1;
for i=1:N-1
    A(i+1,i)=1;
    A(i+1,i+2)=1;
end
b=h^2*ones(N+1,1);
b(1)=1;
b(N+1)=0;
x=linspace(0,pi/2,N+1);
y=A\b;
plot(x,y)
```

Results



N=4



N=20

Script Using Diag

- The diag command allows us to put a vector on the diagonal of a matrix
- We can use this to put in the "1's" just off the diagonal in this matrix
- Syntax: **diag(V,K)** - V is the vector, K tells which diagonal to place the vector in

New Script

```
A=-(2-h^2)*eye(N+1)+diag(v,1)+diag(v,-1);
A(1,2)=0;
A(N+1,N)=0;
A(1,1)=1;
A(N+1,N+1)=1;
```

A Built-In Routine

- Matlab includes **bvp4c**
- This carries out finite differences on systems of ODEs
- **SOL = BVP4C(ODEFUN,BCFUN,SOLINIT)**
  - odefun defines ODEs
  - bcfun defines boundary conditions
  - solinit gives mesh (location of points) and guess for solutions (guesses are constant over mesh)

Using bvp4c

- odefun is a function, much like what we used for ode45
- bcfun is a function that provides the boundary conditions at both ends
- solinit created in a call to the bvpinit function and is a vector of guesses for the initial values of the dependent variable

Preparing our Equation

- Let y be variable 1 – y(1)
- Then dy/dx (=z) is variable 2 – y(2)

$$\frac{d^2y}{dx^2} + y = 1$$

$$\frac{dy}{dx} = z = y(2)$$

$$\frac{d^2y}{dx^2} = \frac{dz}{dx} = 1 - y(1)$$

$$y(0) = 1$$

$$y(\frac{\pi}{2}) = 0$$

```
function dydx = bvp4ode(x,y)
dydx = [  y(2)     1-y(1) ];
```

Boundary Conditions

- ya(1) is y(1) at x=a
- ya(2) is y(2) at x=a
- yb(1) is y(1) at x=b
- yb(2) is y(2) at x=b
- In our case, y(1)-1=0 at x=a and y(1)=0 at x=b

```
function res = bvp4bc(ya,yb)
        res = [  ya(1)-1    yb(1) ];
```

Initialization

- How many mesh points? **10**
- Initial guesses for y(1) and y(2)
- Guess y=1, z=-1
- Guess more critical for nonlinear equations

```
xlow=0;
xhigh=pi/2;
solinit =
  bvpinit(linspace(xlow,xhigh,10),[1 -1]);
```

Postprocessing

```
xint = linspace(xlow,xhigh);
Sxint = deval(sol,xint);
plot(xint,Sxint(1,:))
```

The Script

```
function bvp4
xlow=0;  xhigh=pi/2;
solinit = bvpinit(linspace(xlow,xhigh,10),[1 -1]);
sol = bvp4c(@bvp4ode,@bvp4bc,solinit);
xint = linspace(xlow,xhigh);
Sxint = deval(sol,xint);
plot(xint,Sxint(1,:))
% -----------------------------------------------
function dydx = bvp4ode(x,y)
dydx = [  y(2)          1-y(1) ];
% -----------------------------------------------
function res = bvp4bc(ya,yb)
res = [  ya(1)-1          yb(1) ];
```

Things to Change for Different Problems

```
function bvp4
xlow=0;  xhigh=pi/2;
solinit = bvpinit(linspace(xlow,xhigh,10),[1 -1]);
sol = bvp4c(@bvp4ode,@bvp4bc,solinit);
xint = linspace(xlow,xhigh,20);
Sxint = deval(sol,xint);
plot(xint,Sxint(1,:))
% -----------------------------------------
function dydx = bvp4ode(x,y)
dydx = [  y(2)          1-y(1) ];
% -----------------------------------------
function res = bvp4bc(ya,yb)
res = [  ya(1)-1          yb(1) ];
```

Practice

- Download the file **bvpskeleton.m** and modify it to…
- …solve the boundary value problem shown at the right for ε=0.1 and compare to the analytical solution.

$$\varepsilon \frac{d^2 y}{dx^2} - \frac{dy}{dx} = 0$$

$$y(0) = 0$$

$$y(1) = 1$$

$$y_{analytical} = \frac{e^{x/\varepsilon} - 1}{e^{1/\varepsilon} - 1}$$

bvpskeleton.m

```
xlow=???;
xhigh=???;
solinit = bvpinit(linspace(xlow,xhigh,20),[1 0]);
sol = bvp4c(@bvp4ode,@bvp4bc,solinit);
xint = linspace(xlow,xhigh);
Sxint = deval(sol,xint);
eps=0.1;
analyt=(exp(xint/eps)-1)/(exp(1/eps)-1);
plot(xint,Sxint(1,:),xint,analyt,'r')
% ------------------------------------------
function dydx = bvp4ode(x,y)
eps=0.1;
dydx = [  ???? ; ???? ];
% ------------------------------------------
function res = bvp4bc(ya,yb)
res = [  ???? ;   ???? ];
```

What about BCs involving derivatives?

- If we prescribe a derivative at one end, we cannot just place a value in a cell.
- We'll use finite difference techniques to generate a formula
- The formulas work best when "centered", so we will use a different approximation for the first derivative.

Derivative BCs

- Consider a boundary condition of the form dy/dx=0 at x=L
- Finite difference (centered) is:

$$\boxed{\begin{array}{l} \dfrac{dy}{dx} \approx \dfrac{y_{i+1} - y_{i-1}}{2h} = 0 \\ or \\ y_{i+1} = y_{i-1} \end{array}}$$

Derivative BCs

- So at a boundary point on the right we just replace y_{i+1} with y_{i-1} in the formula
- Consider:

$$\boxed{\frac{d^2y}{dx^2} + y = 0}$$

$$\boxed{\begin{aligned} &y(0) = 1 \\ &\frac{dy}{dx}(1) = 0 \end{aligned}}$$

Finite Difference Equation

- We derive a new difference equation

$$\frac{d^2y}{dx^2} + y = 0$$

becomes

$$\frac{y_{i-1} - 2y_i + y_{i+1}}{h^2} + y_i = 0$$

or

$$y_{i-1} - (2 - h^2)y_i + y_{i+1} = 0$$

Derivative BCs

- **The difference equation at the last point is**

$$
\boxed{
\begin{aligned}
& y_{N-1} - \left(2 - h^2\right) y_N + y_{N+1} = 0 \\
& but \\
& y_{N-1} = y_{N+1} \\
& so \\
& 2y_{N-1} - \left(2 - h^2\right) y_N = 0
\end{aligned}
}
$$

Final Matrix

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & -(2-h^2) & 1 & 0 & 0 \\ 0 & 1 & -(2-h^2) & 1 & 0 \\ 0 & 0 & 1 & -(2-h^2) & 1 \\ 0 & 0 & 0 & 2 & -(2-h^2) \end{bmatrix} \begin{Bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \end{Bmatrix} = \begin{Bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{Bmatrix}$$

New Code

```
h=1/4
A=[1 0 0 0 0;
   1 -(2-h^2) 1 0 0;
   0 1 -(2-h^2) 1 0;
   0 0 1 -(2-h^2) 1;
   0 0 0 2 -(2-h^2)]
B=[1; 0; 0; 0; 0]
y=A\B
```

Using bvp4c

- The boundary condition routine allows us to set the derivative of the dependent variable at the boundary

Preparing Equation

$$\frac{d^2y}{dx^2} + y = 0$$

$$\frac{dy}{dx} = z = y(2)$$

$$\frac{d^2y}{dx^2} = \frac{dz}{dx} = -y(1)$$

$$y(0) = 1$$

$$\frac{dy}{dx}(1) = 0$$

$$so$$

$$ya(1) = 1$$

$$yb(2) = 0$$

$$or$$

$$ya(1) - 1 = 0$$

$$yb(2) = 0$$

The Script

```
function bvp5
xlow=0;  xhigh=1;
solinit = bvpinit(linspace(xlow,xhigh,10),[1 -1]);
sol = bvp4c(@bvp5ode,@bvp5bc,solinit);
xint = linspace(xlow,xhigh);
Sxint = deval(sol,xint);
plot(xint,Sxint(1,:))
% -----------------------------------------
function dydx = bvp5ode(x,y)
dydx = [  y(2)          -y(1) ];
% -----------------------------------------
function res = bvp5bc(ya,yb)
res = [  ya(1)-1          yb(2) ];
```

Practice

- Solve the Case Study Problem
- Use L=25 mm, T_f=20 C, and hC/kA=4000 /m^2
- JPB: need a skeleton here

$$\frac{d^2T}{dx^2} - \frac{hC}{kA}(T - T_f) = 0$$

$$T(0) = 40C$$

$$\frac{dT}{dx}\Big|_{x=L} = 0$$

Practice

- A 1 W, 2 Mohm resistor which is 30 mm long has a radius of 1 mm. Determine the peak temperature if the outside surface is held at room temperature.
- Use k=0.1 W/m-K and Q=2.1 MW/m^2

$$\frac{d^2T}{dr^2} + \frac{1}{r}\frac{dT}{dr} + \frac{Q}{k} = 0$$

$$T(R) = 20\,C$$

$$\frac{dT}{dr}(0) = 0$$

Practice

- Repeat the previous problem with convection to external environment.
- Use k=0.1 W/m-K and Q=2.1 MW/m^2
- Also, h=10 W/m^2-K and T_e=20 C

$$\frac{d^2T}{dr^2}+\frac{1}{r}\frac{dT}{dr}+\frac{Q}{k}=0$$

$$h\left(T(R)-T_e\right)=\frac{1}{2}QR$$

$$\frac{dT}{dr}(0)=0$$

Questions?