PROBLEM SET 8.1

Problem 1

$$y'' + y' - y = 0$$

We know the two solutions

$$\begin{aligned} y(0) &= 0 \qquad y'(0) = 1 \rightarrow y(1) = 0.741\,028 \\ y(0) &= 0 \qquad y'(0) = 0 \rightarrow y(1) = 0 \end{aligned}$$

We are looking for u so that

$$y(0) = 0 \qquad y'(0) = u \rightarrow y(1) = 1$$

By linear interpolation

$$u = \frac{1}{0.741\,028} = 1.349\,477 \quad \blacktriangleleft$$

Problem 2

$$y''' + y'' + 2y' = 6$$

The two known solutions are

$$\begin{aligned} y(0) &= 2 \qquad y'(0) = 0 \qquad y''(0) = 1 \rightarrow y(1) = 3.037\,65 \\ y(0) &= 2 \qquad y'(0) = 0 \qquad y''(0) = 0 \rightarrow y(1) = 2.723\,18 \end{aligned}$$

We have to find u so that

$$y(0) = 2 \qquad y'(0) = 0 \qquad y''(0) = u \rightarrow y(1) = 0$$

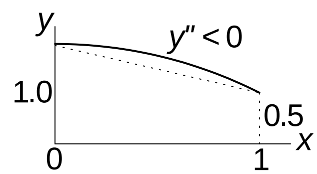
Linear interpolation yields

$$u = -\frac{2.723\,18}{3.037\,65 - 2.723\,18} = -8.659\,59 \quad \blacktriangleleft$$

Problem 3

(a)

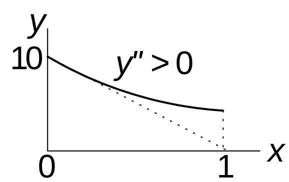
$$y'' = -e^{-y} \qquad y(0) = 1 \qquad y(1) = 0.5$$



Apparently $y'(0) < 0.5$; hence we estimate $y'(0) \approx 0.25$ ◀

(b)

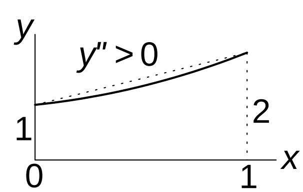
$$y'' = 4y^2 \qquad y(0) = 10 \qquad y'(1) = 0$$



We estimate $y'(0) \approx -10$ ◀

(c)

$$y'' = \cos(xy) \qquad y(0) = 1 \qquad y(1) = 2$$

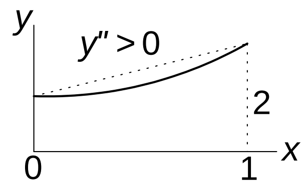


We see that $y'(0) < 1$, so that a reasonable quess is $y'(0) \approx 0.5$ ◀

Problem 4

(a)

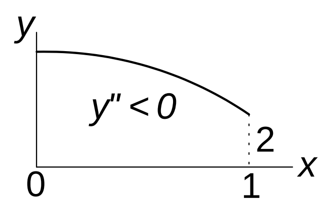
$$y'' = y^2 + xy \qquad y'(0) = 0 \qquad y(1) = 2$$



Estimate $y(0) \approx 1$ ◀

(b)

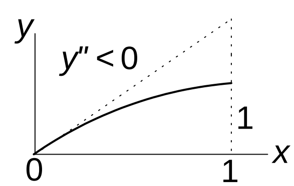
$$y'' = -\frac{2}{x}y' - y^2 \qquad y'(0) = 0 \qquad y(1) = 2$$



Estimate $y(0) \approx 4$ ◀

(c)

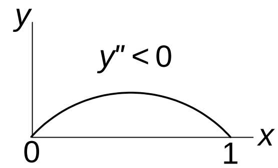
$$y'' = -x(y')^2 \qquad y'(0) = 2 \qquad y(1) = 1$$



Estimate $y(0) = 0$ ◀

Problem 5

$$y''' + 5y''y^2 = 0 \qquad y(0) = 0 \qquad y'(0) = 1 \qquad y(1) = 0$$



Assume that y is parabolic:

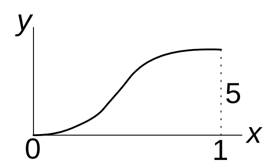
$$y = Cx(1-x) \qquad y' = C(1-2x)$$

But $y'(0) = 1$. Therefore, $C = 1$ and $y'' = -2$. Thus the estimate is

$$y''(0) \approx -2 \blacktriangleleft$$

Problem 6

$$y^{(4)} + 2y'' + y' \sin y = 0$$
$$y(0) = y'(0) = 0 \qquad y(1) = 5 \qquad y'(1) = 0$$



Assume that y is cubic:

$$\begin{aligned} y &= C_1x^2 + C_2x^3 \qquad y' = 2C_1x + 3C_2x^2 \\ y'' &= 2C_1 + 6C_2x \qquad y''' = 6C_2 \end{aligned}$$

The conditions $y(1) = 5$ and $y'(1) = 0$ yield

$$\begin{aligned} C_1 + C_2 &= 5 \\ 2C_1 + 3C_2 &= 0 \end{aligned}$$

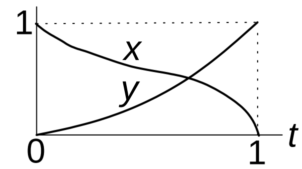
the solution of which is $C_1 = 15$, $C_2 = -10$. Thus the estimates are

$$\begin{aligned} y''(0) &= 2C_1 = 30 \blacktriangleleft \\ y'''(0) &= 6C_2 = -60 \blacktriangleleft \end{aligned}$$

Problem 7

$$\ddot{x} + 2x^2 - y = 0 \qquad \ddot{y} + y^2 - 2x = 1$$

$$x(0) = 1 \qquad x(1) = 0 \qquad y(0) = 0 \qquad y(1) = 1$$



We can obtain the following information about the curvatures from the differential equations and the boundary conditions:

$$\begin{aligned} \text{At } t &= 0\text{:} \qquad \ddot{x} = 1 \qquad \ddot{y} = 0 \\ \text{At } t &= 1\text{:} \qquad \ddot{x} = -2 \qquad \ddot{y} = 3 \end{aligned}$$

This information was used to draw the rough sketches of x and y. From these sketches we estimate that

$$\dot{x}(0) \approx -1 \blacktriangleleft \qquad \dot{y}(0) \approx 0.5 \blacktriangleleft$$

Problem 8

$$y'' = -(1 - 0.2x)\,y \qquad y(0) = 0 \qquad y(\pi/2) = 1$$

The following program is based on the function `shoot2` in Example 8.1. The adaptive Runge-Kutta method (`runKut5`) is used for the intergration.

```
function p8_1_8
% Shooting method for 2nd-order boundary value problem
% in Problem 8, Problem Set 8.1.

xStart = 0; xStop = pi/2; % Range of integration.
h = 0.1;                  % Step size.
freq = 1;                 % Frequency of printout.
u1 = 0.6; u2 = 1.2;       % Trial values of unkown
                          % initial condition u.
x = xStart;
u = ridder(@residual,u1,u2);
[xSol,ySol] = runKut5(@dEqs,x,inCond(u),xStop,h);
printSol(xSol,ySol,freq)

    function F = dEqs(x,y)    % First-order differential
    F = [y(2), -(1-0.2*x)*y(1)^2]; % equations.
    end

    function y = inCond(u)    % Initial conditions (u is
```

```
    y = [0 u];                 % the unknown condition).
    end

    function r = residual(u)  % Boundary residual.
    x = xStart;
    [xSol,ySol] = runKut5(@dEqs,x,inCond(u),xStop,h);
    r = ySol(size(ySol,1),1) - 1;
    end
end
```

```
>>       x             y1             y2
   0.0000e+000   0.0000e+000   7.7880e-001
   1.0000e-001   7.7875e-002   7.7860e-001
   4.9778e-001   3.8477e-001   7.5590e-001
   8.6351e-001   6.4784e-001   6.6960e-001
   1.1987e+000   8.4884e-001   5.1860e-001
   1.5217e+000   9.8517e-001   3.1861e-001
   1.5708e+000   1.0000e+000   2.8516e-001
```

Problem 9

$$y'' = -2y' - 3y^2 \qquad y(0) = 0 \qquad y(2) = -1$$

The following program is based on the function `shoot2` in Example 8.1. The adaptive Runge-Kutta method (`runKut5`) is used for the intergration.

```
function p8_1_9
% Shooting method for 2nd-order boundary value problem
% in Problem 9, Problem Set 8.1.

xStart = 0; xStop = 2;    % Range of integration.
h = 0.1;                  % Step size.
freq = 2;                 % Frequency of printout.
u1 = -1.5; u2 = -0.5;     % Trial values of unkown
                          % initial condition u.
x = xStart;
u = ridder(@residual,u1,u2);
[xSol,ySol] = runKut5(@dEqs,x,inCond(u),xStop,h);
printSol(xSol,ySol,freq)

    function F = dEqs(x,y)    % First-order differential
    F = [y(2) -2*y(2)-3*y(1)^2]; % equations.
    end
```

```
    function y = inCond(u)    % Initial conditions (u is
    y = [0 u];                % the unknown condition).
    end

    function r = residual(u)  % Boundary residual.
    x = xStart;
    [xSol,ySol] = runKut5(@dEqs,x,inCond(u),xStop,h);
    r = ySol(size(ySol,1),1) + 1;
    end
end

>>      x             y1             y2
   0.0000e+000   0.0000e+000  -9.9420e-001
   2.2787e-001  -1.8242e-001  -6.3781e-001
   4.8887e-001  -3.1694e-001  -4.2060e-001
   7.6885e-001  -4.1956e-001  -3.3127e-001
   1.0768e+000  -5.1998e-001  -3.3497e-001
   1.4186e+000  -6.4774e-001  -4.2731e-001
   1.7890e+000  -8.4196e-001  -6.4761e-001
   2.0000e+000  -1.0000e+000  -8.6542e-001
```

Problem 10

$$y'' = -\sin y - 1 \qquad y(0) = y(\pi) = 0$$

The following program is based on the function `shoot2` in Example 8.1. The Bulirsch-Stoer method (`bulStoer`) is used for the intergration.

```
function p8_1_10
% Shooting method for 2nd-order boundary value problem
% in Problem 10, Problem Set 8.1.

xStart = 0; xStop = pi;   % Range of integration.
h = 0.5;                  % Step size.
freq = 1;                 % Frequency of printout.
u1 = 2; u2 = 3;           % Trial values of unkown
                          % initial condition u.
x = xStart;
u = brent(@residual,u1,u2);
[xSol,ySol] = bulStoer(@dEqs,x,inCond(u),xStop,h);
printSol(xSol,ySol,freq)
```

```
    function F = dEqs(x,y)    % First-order differential
    F = [y(2) -sin(y(1))-1]; % equations.
    end

    function y = inCond(u)    % Initial conditions (u is
    y = [0 u];                % the unknown condition).
    end

    function r = residual(u)  % Boundary residual.
    x = xStart;
    [xSol,ySol] = bulStoer(@dEqs,x,inCond(u),xStop,h);
    r = ySol(size(ySol,1),1);
    end
end
```

```
>>      x            y1            y2
   0.0000e+000   0.0000e+000   2.8047e+000
   5.0000e-001   1.2266e+000   2.0219e+000
   1.0000e+000   1.9900e+000   1.0356e+000
   1.5000e+000   2.2767e+000   1.2454e-001
   2.0000e+000   2.1176e+000  -7.6891e-001
   2.5000e+000   1.4931e+000  -1.7422e+000
   3.0000e+000   3.8580e-001  -2.6359e+000
   3.1416e+000   1.8234e-011  -2.8047e+000
```

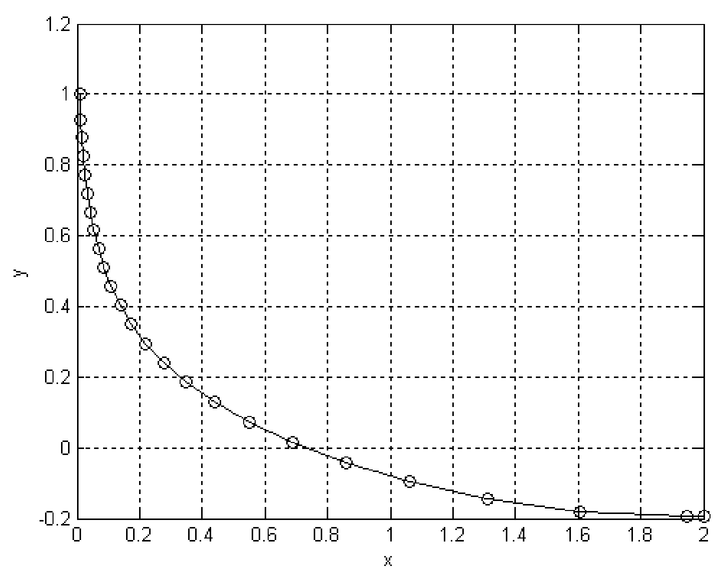
Problem 11

$$y'' = -\frac{1}{x}y' - y \qquad y(0.01) = 1 \qquad y'(2) = 0$$

The following program is based on the function `shoot2` in Example 8.1. The adaptive Runge-Kutta method (`runKut5`) is used for the intergration. Brent's method of root finding is replaced by linear interpolation (`linInterp`).

```
function p8_1_11
% Shooting method for 2nd-order boundary value problem
% in Problem 10, Problem Set 8.1.

xStart = 0.01; xStop = 2; % Range of integration.
h = 0.1;                  % Step size.
freq = 1;                 % Frequency of printout.
u1 = -50; u2 = 0;         % Trial values of unkown
                          % initial condition u.
x = xStart;
```

```
u = linInterp(@residual,u1,u2);
[xSol,ySol] = runKut5(@dEqs,x,inCond(u),xStop,h);
plot(xSol,ySol(:,1),'k-o'); grid on
xlabel('x'); ylabel('y')

    function F = dEqs(x,y)     % First-order differential
    F = [y(2) -y(2)/x-y(1)];  % equations.
    end

    function y = inCond(u)     % Initial conditions (u is
    y = [1 u];                 % the unknown condition).
    end

    function r = residual(u)  % Boundary residual.
    x = xStart;
    [xSol,ySol] = runKut5(@dEqs,x,inCond(u),xStop,h);
    r = ySol(size(ySol,1),2);
    end
end
```



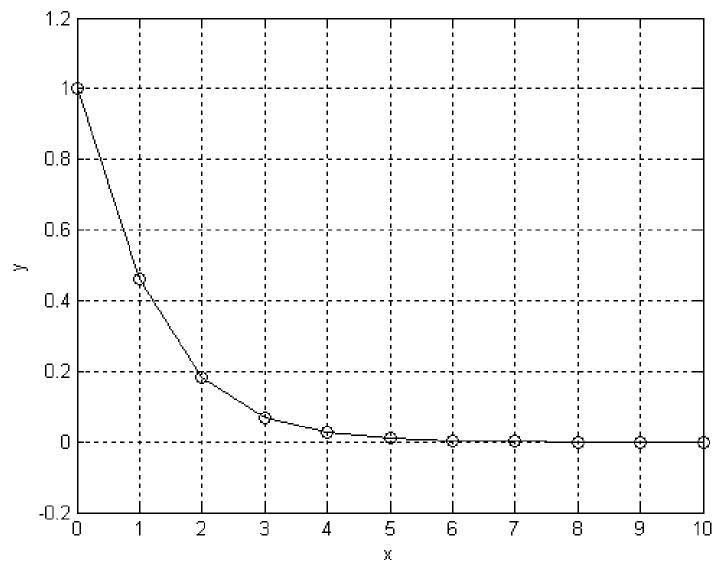
Problem 12

$$y'' = (1 - e^{-x})y \qquad y(0) = 1 \qquad y(\infty) = 0$$

The following program is based on the function `shoot2` in Example 8.1. The

Bulirsch-Stoer method (`bulStoer`) is used for the intergration. Brent's method of root finding is replaced by linear interpolation (`linInterp`).

```
function p8_1_12
% Shooting method for 2nd-order boundary value problem
% in Problem 12, Problem Set 8.1.

xStart = 0; xStop = 10;   % Range of integration.
h = 1;                    % Step size.
freq = 1;                 % Frequency of printout.
u1 = 0; u2 = 2;           % Trial values of unkown
                          % initial condition u.
x = xStart;
u = linInterp(@residual,u1,u2);
[xSol,ySol] = bulStoer(@dEqs,x,inCond(u),xStop,h);
plot(xSol,ySol(:,1),'k-o');grid on
xlabel('x'); ylabel('y')

    function F = dEqs(x,y)    % First-order differential
    F = [y(2) (1-exp(-x))*y(1)]; % equations.
    end

    function y = inCond(u)    % Initial conditions (u is
    y = [1 u];                % the unknown condition).
    end

    function r = residual(u)  % Boundary residual.
    x = xStart;
    [xSol,ySol] = bulStoer(@dEqs,x,inCond(u),xStop,h);
    r = ySol(size(ySol,1),1);
    end
end
```



The results did not change significantly when the program was run with `xStop = 7.5`.

Problem 13

$$
\begin{aligned}
y''' &= -\frac{1}{x}y'' + \frac{1}{x^2}y' + 0.1(y')^3 \\
y(1) &= 0 \qquad y''(1) = 0 \qquad y(2) = 1
\end{aligned}
$$

The following program is based on the function `shoot3` in Example 8.3. We chose the adaptive Runge-Kutta method (`runKut5`) for integration. Because the differential equation is nonlinear, linear interpolation is replaced by Brent's method of root finding (`brent`).

```
function p8_1_13
% Shooting method for 3rd-order boundary value
% problem in Problem 13, Problem Set 8.1.

global XSTART XSTOP H    % Make these params. global.
XSTART = 1; XSTOP = 2;   % Range of integration.
H = 0.2;                 % Step size.
freq = 1;                % Frequency of printout.
u1 = 0; u2 = 2;          % Trial values of unknown
                         % initial condition u.
x = XSTART;
```

```
u = brent(@residual,u1,u2);
[xSol,ySol] = runKut5(@dEqs,x,inCond(u),XSTOP,H);
printSol(xSol,ySol,freq)

function F = dEqs(x,y)   % 1st-order differential eqs.
F = [y(2) y(3) -y(3)/x+y(2)/x^2+0.1*y(2)^3];

function y = inCond(u)   % Initial conditions.
y = [0 u 0];

function r = residual(u) % Boundary residual.
global XSTART XSTOP H
x = XSTART;
[xSol,ySol] = runKut5(@dEqs,x,inCond(u),XSTOP,H);
r = ySol(size(ySol,1),1) - 1;

>>      x             y1            y2            y3
   1.0000e+000   0.0000e+000   9.0111e-001   0.0000e+000
   1.2000e+000   1.8136e-001   9.1753e-001   1.5142e-001
   1.4247e+000   3.9239e-001   9.6436e-001   2.5778e-001
   1.7017e+000   6.7057e-001   1.0487e+000   3.4648e-001
   2.0000e+000   1.0000e+000   1.1635e+000   4.2129e-001
```

Problem 14

$$y''' = -4y'' - 6y' + 10$$
$$y(0) = y''(0) = 0 \qquad y(3) - y'(3) = 5$$

The following program is based on the function `shoot3` in Example 8.3. We chose the adaptive Runge-Kutta method (`runKut5`) for integration.

```
function p8_1_14
% Shooting method for 3rd-order boundary value
% problem in Problem 14, Problem Set 8.1.

xStart = 0; xStop = 3;   % Range of integration.
h = 0.05;                 % Step size.
freq = 2;                % Frequency of printout.
u1 = 0; u2 = 2;          % Trial values of unknown
                         % initial condition u.
x = xStart;
u = linInterp(@residual,u1,u2);
[xSol,ySol] = runKut5(@dEqs,x,inCond(u),xStop,h);
```

```
printSol(xSol,ySol,freq)

    function F = dEqs(x,y)   % 1st-order differential eqs.
    F = [y(2) y(3) -4*y(3)-6*y(2)+10];
    end

    function y = inCond(u)   % Initial conditions.
    y = [0 u 0];
    end

    function r = residual(u) % Boundary residual.
    x = xStart;
    [xSol,ySol] = runKut5(@dEqs,x,inCond(u),xStop,h);
    r = ySol(size(ySol,1),1) - ySol(size(ySol,1),2) - 5;
    end
end

>>    x           y1           y2           y3
  .0000e+000  0.0000e+000  4.1461e+000  0.0000e+000
 1.5953e-001  6.5286e-001  3.9933e+000 -1.7104e+000
 3.7708e-001  1.4721e+000  3.5096e+000 -2.5156e+000
 5.9169e-001  2.1672e+000  2.9726e+000 -2.3919e+000
 8.1038e-001  2.7636e+000  2.5005e+000 -1.8954e+000
 1.0396e+000  3.2922e+000  2.1340e+000 -1.3086e+000
 1.2856e+000  3.7832e+000  1.8802e+000 -7.7971e-001
 1.5563e+000  4.2691e+000  1.7277e+000 -3.7812e-001
 1.8642e+000  4.7879e+000  1.6553e+000 -1.2237e-001
 2.2341e+000  5.3955e+000  1.6375e+000  2.1545e-003
 2.7178e+000  6.1897e+000  1.6485e+000  2.9607e-002
 3.0000e+000  6.6561e+000  1.6561e+000  2.3251e-002
```

Problem 15

$$\begin{aligned} y''' &= -2y'' - \sin y \\ y(-1) &= 0 \qquad y'(-1) = -1 \qquad y'(1) = 1 \end{aligned}$$

The following program is based on the function `shoot3` in Example 8.3. We chose the adaptive Runge-Kutta method (`runKut5`) for integration. Because the differential equation is nonlinear, linear interpolation is replaced by Brent's method of root finding (`brent`).

```
function p8_1_15
```

```
% Shooting method for 3rd-order boundary value
% problem in Problem 15, Problem Set 8.1.

xStart = -1; xStop = 1;  % Range of integration.
h = 0.05;                % Step size.
freq = 2;                % Frequency of printout.
u1 = 2; u2 = 6;          % Trial values of unknown
                         % initial condition u.
x = xStart;
u = brent(@residual,u1,u2);
[xSol,ySol] = runKut5(@dEqs,x,inCond(u),xStop,h);
printSol(xSol,ySol,freq)

    function F = dEqs(x,y)   % 1st-order differential eqs.
    F = [y(2) y(3) -2*y(3)-sin(y(1))];
    end

    function y = inCond(u)   % Initial conditions.
    y = [0 -1 u];
    end

    function r = residual(u) % Boundary residual.
    x = xStart;
    [xSol,ySol] = runKut5(@dEqs,x,inCond(u),xStop,h);
    r = ySol(size(ySol,1),2) - 1;
    end
end
>>    x           y1           y2            y3
-1.0000e+000  0.0000e+000 -1.0000e+000 4.3791e+000
-8.3662e-001 -1.1079e-001 -3.8911e-001  3.1677e+000
-6.0338e-001 -1.2712e-001  2.0434e-001  2.0112e+000
-3.4610e-001 -1.7803e-002  6.1157e-001  1.2176e+000
-5.9201e-002  1.9930e-001  8.7561e-001  6.6531e-001
 2.6601e-001  5.1138e-001  1.0218e+000  2.6144e-001
 6.3828e-001  9.0198e-001  1.0575e+000 -4.9752e-002
 1.0000e+000  1.2763e+000  1.0000e+000 -2.5381e-001
```

Problem 16

$$
\begin{aligned}
y''' &= -2y'' - \sin y \\
y(-1) &= 0 \qquad y(0) = 0 \qquad y(1) = 1
\end{aligned}
$$

We use the Bulirsch-Stoer method for integration because it gives us control over the integration increment (note that the value of y is specified at $x = 0$).

```
function p8_1_16
% Shooting method for 3rd-order boundary value
% problem in Problem 16, Problem Set 8.1.

xStart = -1; xStop = 1;  % Range of integration.
h = 0.2;                 % Step size.
freq = 1;                % Frequency of printout.
u = [-0.5 1];            % Trial values of unknown
                         % initial conditions [u].
x = xStart;
u = newtonRaphson2(@residual,u);
[xSol,ySol] = bulStoer(@dEqs,x,inCond(u),xStop,h);
printSol(xSol,ySol,freq)

    function F = dEqs(x,y)   % 1st-order differential eqs.
    F = [y(2) y(3) -2*y(3)-sin(y(1))];
    end

    function y = inCond(u)   % Initial conditions.
    y = [0 u(1) u(2)];
    end

    function r = residual(u) % Boundary residual.
    x = xStart;
    [xSol,ySol] = bulStoer(@dEqs,x,inCond(u),xStop,h);
    n = size(ySol,1);
    yMiddle = ySol((n+1)/2,1); yEnd = ySol(n,1);
    r = [yMiddle; yEnd-1];
    end
end
```

```
>>    x           y1            y2            y3
-1.0000e+000  0.0000e+000 -1.4947e+000  5.1960e+000
-8.0000e-001 -2.0751e-001 -6.3667e-001  3.5035e+000
-6.0000e-001 -2.7298e-001 -5.4985e-002  2.3895e+000
-4.0000e-001 -2.4164e-001  3.4358e-001  1.6446e+000
-2.0000e-001 -1.4375e-001  6.1838e-001  1.1342e+000
-5.5511e-017  4.4292e-015  8.0705e-001  7.7181e-001
 2.0000e-001  1.7493e-001  9.3326e-001  5.0237e-001
 4.0000e-001  3.7014e-001  1.0119e+000  2.9162e-001
 6.0000e-001  5.7715e-001  1.0525e+000  1.1952e-001
 8.0000e-001  7.8903e-001  1.0615e+000 -2.4616e-002
```

```
 1.0000e+000  1.0000e+000  1.0442e+000 -1.4557e-001
```

Problem 17

$$y^{(4)} = -xy^2$$

$$y(0) = 5 \qquad y''(0) = 0 \qquad y'(1) = 0 \qquad y'''(1) = 2$$

```
function p8_1_17
% Shooting method for 4th-order boundary value problem
% in Problem 17, Problem Set 1.

xStart = 0; xStop = 1;    % Range of integration.
h = 0.5;                  % Step size.
freq = 1;                 % Frequency of printout.
u = [-2 6];               % Trial values of u(1)
                          % and u(2).
x = xStart;
u = newtonRaphson2(@residual,u);
[xSol,ySol] = runKut5(@dEqs,x,inCond(u),xStop,h);
printSol(xSol,ySol,freq)

    function F = dEqs(x,y)    % Differential equations.
    F = zeros(1,4);
    F(1) = y(2); F(2) = y(3); F(3) = y(4);
    F(4) = -x*y(1)^2;
    end

    function y = inCond(u)    % Initial conditions; u(1)
    y = [5 u(1) 0 u(2)];      % and u(2) are unknowns.
    end

    function r = residual(u)  % Bounday residuals.
    r = zeros(length(u),1);
    x = xStart;
    [xSol,ySol] = runKut5(@dEqs,x,inCond(u),xStop,h);
    lastRow = size(ySol,1);
    r(1) = ySol(lastRow,2);
    r(2) = ySol(lastRow,4) - 2;
    end
end

>>     x            y1            y2            y3            y4
```

```
 0.0000e+000  5.0000e+000 -3.2007e+000  0.0000e+000  7.6621e+000
 1.6059e-001  4.4913e+000 -3.1026e+000  1.2149e+000  7.3820e+000
 3.0570e-001  4.0575e+000 -2.8504e+000  2.2449e+000  6.7707e+000
 4.5121e-001  3.6699e+000 -2.4548e+000  3.1726e+000  5.9552e+000
 5.9763e-001  3.3475e+000 -1.9297e+000  3.9770e+000  5.0168e+000
 7.4552e-001  3.1082e+000 -1.2904e+000  4.6438e+000  3.9894e+000
 8.9559e-001  2.9689e+000 -5.5272e-001  5.1591e+000  2.8608e+000
 1.0000e+000  2.9398e+000 -2.1094e-015  5.4136e+000  2.0000e+000
```

Problem 18

$$y^{(4)} = -2yy''$$
$$y(0) = y'(0) = 0 \qquad y(4) = 0 \qquad y'(4) = 1$$

```
function p8_1_18
% Shooting method for 4th-order boundary value problem
% in Problem 18, Problem Set 1.

xStat = 0; xStop = 4;    % Range of integration.
h = 0.5;                  % Step size.
freq = 1;                 % Frequency of printout.
u = [-0.3 0.3];           % Trial values of u(1)
                          % and u(2).
x = xStat;
u = newtonRaphson2(@residual,u);
[xSol,ySol] = runKut5(@dEqs,x,inCond(u),xStop,h);
printSol(xSol,ySol,freq)

    function F = dEqs(x,y)    % Differential equations.
    F = zeros(1,4);
    F(1) = y(2); F(2) = y(3); F(3) = y(4);
    F(4) = -2*y(1)*y(3);
    end

    function y = inCond(u)    % Initial conditions; u(1)
    y = [0 0 u(1) u(2)];      % and u(2) are unknowns.
    end

    function r = residual(u)  % Bounday residuals.
    r = zeros(length(u),1);
    x = xStat;
    [xSol,ySol] = runKut5(@dEqs,x,inCond(u),xStop,h);
```

```
    lastRow = size(ySol,1);
    r(1) = ySol(lastRow,1);
    r(2) = ySol(lastRow,2) - 1;
    end
end

>>    x            y1           y2            y3            y4
 0.0000e+000  0.0000e+000  0.0000e+000 -3.6774e-001  2.6832e-001
 4.1892e-001 -2.8982e-002 -1.3053e-001 -2.5560e-001  2.6596e-001
 8.4027e-001 -1.0338e-001 -2.1487e-001 -1.4543e-001  2.5587e-001
 1.2915e+000 -2.1128e-001 -2.5489e-001 -3.2840e-002  2.4432e-001
 1.7771e+000 -3.3427e-001 -2.4204e-001  8.6390e-002  2.5253e-001
 2.2783e+000 -4.3921e-001 -1.6519e-001  2.2534e-001  3.1368e-001
 2.6814e+000 -4.8382e-001 -4.6300e-002  3.7198e-001  4.2481e-001
 3.0366e+000 -4.7337e-001  1.1561e-001  5.4896e-001  5.8122e-001
 3.3561e+000 -4.0500e-001  3.2370e-001  7.6328e-001  7.6553e-001
 3.6789e+000 -2.5622e-001  6.1335e-001  1.0423e+000  9.5932e-001
 3.9771e+000 -2.2517e-002  9.6891e-001  1.3471e+000  1.0612e+000
 4.0000e+000  1.4849e-014  1.0000e+000  1.3714e+000  1.0619e+000
```

Problem 19

$$\ddot{x} = -\frac{c}{m} v\,\dot{x} \qquad \ddot{y} = -\frac{c}{m} v\,\dot{y} - g \qquad v = \sqrt{\dot{x}^2 + \dot{y}^2}$$
$$x(0) = y(0) = 0 \qquad x(10\text{ s}) = 8000\,\text{m} \qquad y(10\text{ s}) = 0$$

We use the notation

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \end{bmatrix}$$

```
function p8_1_19
% Shooting method for 4th-order boundary value problem
% in Problem 19, Problem Set 1.

xStart = 0; xStop = 10;   % Range of integration.
h = 2;                   % Step size.
freq = 0;                 % Frequency of printout.
u = [1000 600];           % Trial values of u(1)
                          % and u(2).
x = xStart;
u = newtonRaphson2(@residual,u);
[xSol,ySol] = bulStoer(@dEqs,x,inCond(u),xStop,h);
```

```
printSol(xSol,ySol,freq)

    function F = dEqs(x,y)    % Differential equations.
    c = 3.2e-4; m = 20; g = 9.80665;
    v = sqrt(y(2)^2 + y(4)^2);
    F = zeros(1,4);
    F(1) = y(2); F(2) = -c/m*v*y(2);
    F(3) = y(4); F(4) = -c/m*v*y(4) - g;
    end

    function y = inCond(u)    % Initial conditions; u(1)
    y = [0 u(1) 0 u(2)];      % and u(2) are unknowns.
    end

    function r = residual(u)  % Bounday residuals.
    r = zeros(length(u),1);
    x = xStart;
    [xSol,ySol] = bulStoer(@dEqs,x,inCond(u),xStop,h);
    n = size(ySol,1);
    r(1) = ySol(n,1) - 8000;
    r(2) = ySol(n,3);
    end
end
```

```
>>    x            y1            y2            y3            y4
 0.0000e+000  0.0000e+000  8.5349e+002  0.0000e+000  5.0150e+001
 1.0000e+001  8.0000e+003  7.5089e+002  1.0248e-012 -4.8051e+001
```

$$
\begin{aligned}
v_0 &= \sqrt{\dot{x}_0^2 + \dot{y}_0^2} = \sqrt{853.49^2 + 501.50^2} = 989.9 \text{ m/s} \quad \blacktriangleleft \\
\theta &= \tan^{-1}\frac{501.50}{853.49} = 0.531\,24 \text{ rad} = 30.44^\circ \quad \blacktriangleleft
\end{aligned}
$$

Problem 20

$$
y^{(4)} = \beta y'' + 1 \qquad y(0) = y''(0) = y(1) = y''(1) = 0
$$

where $()' = dy/d\xi$.

(a)

```
function p8_1_20
% Shooting method for 4th-order boundary value problem
```

```
% in Problem 20, Problem Set 1.

xStart = 0; xStop = 1;     % Range of integration.
h = 0.25;                  % Step size.
freq = 2;                  % Frequency of printout.
u = [1 1];                 % Trial values of u(1)
                           % and u(2).
x = xStart;
u = newtonRaphson2(@residual,u);
[xSol,ySol] = bulStoer(@dEqs,x,inCond(u),xStop,h);
printSol(xSol,ySol,freq)

    function F = dEqs(x,y)    % Differential equations.
    beta = 1.65929;
    F = [y(2) y(3) y(4) beta*y(3)+1];
    end

    function y = inCond(u)    % Initial conditions; u(1)
    y = [0 u(1) 0 u(2)];      % and u(2) are unknowns.
    end

    function r = residual(u)  % Bounday residuals.
    r = zeros(length(u),1);
    x = xStart;
    [xSol,ySol] = bulStoer(@dEqs,x,inCond(u),xStop,h);
    n = size(ySol,1);
    r(1) = ySol(n,1);
    r(2) = ySol(n,3);
    end
end
```

```
>>    x            y1            y2            y3            y4
 0.0000e+000  0.0000e+000  3.5747e-002  0.0000e+000 -4.4069e-001
 5.0000e-001  1.1141e-002 -2.4393e-008 -1.0651e-001 -4.0475e-008
 1.0000e+000  4.4527e-017 -3.5747e-002  5.4083e-016  4.4069e-001
```

$$v_{\max} = \frac{w_0 L^4}{EI} y(0.5) = 0.001114 \frac{w_0 L^4}{EI} \quad \blacktriangleleft$$

(b)

Running the same program with $\beta = -1.65929$ resulted in

```
>>    x            y1            y2            y2            y4
 0.0000e+000  0.0000e+000  4.9976e-002  0.0000e+000 -5.8292e-001
```

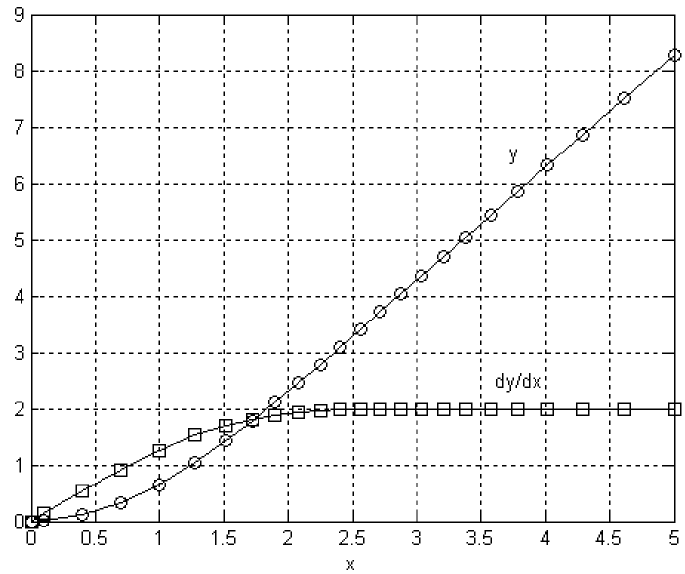
```
 5.0000e-001  1.5661e-002  4.2998e-012 -1.5099e-001  8.2968e-012
 1.0000e+000  2.3865e-012 -4.9976e-002  1.1472e-011  5.8292e-001
```

$$v_{\max} = 0.001566\frac{w_0 L^4}{EI} \quad \blacktriangleleft$$

Problem 21

$$y''' = -yy'' \qquad y(0) = y'(0) = 0 \qquad y'(\infty) = 2$$

```
function p8_1_21
% Shooting method for 3rd-order boundary value
% problem in Problem 21, Problem Set 8.1.

xStart = 0; xStop = 5;   % Range of integration.
h = 0.1;                 % Step size.
freq = 1;                % Frequency of printout.
u1 = 0; u2 = 2;          % Trial values of unknown
                         % initial conditions [u].
x = xStart;
u = brent(@residual,u1,u2);
[xSol,ySol] = runKut5(@dEqs,x,inCond(u),xStop,h);
plot(xSol,ySol(:,1),'k-o'); hold on
plot(xSol,ySol(:,2),'k-s'); grid on
xlabel('x')
gtext('y'); gtext('dy/dx')

    function F = dEqs(x,y)   % 1st-order differential eqs.
    F = [y(2) y(3) -y(1)*y(3)];
    end

    function y = inCond(u)   % Initial conditions.
    y = [0 0 u];
    end

    function r = residual(u) % Boundary residual.
    x = xStart;
    [xSol,ySol] = runKut5(@dEqs,x,inCond(u),xStop,h);
    n = size(ySol,1);
    r = ySol(n,2) - 2;
    end
end
```



Problem 22

As in Example 8.4, we introduce the dimensionless variables

$$\xi = \frac{x}{L} \qquad y = \frac{EI}{w_0 L^4} v$$

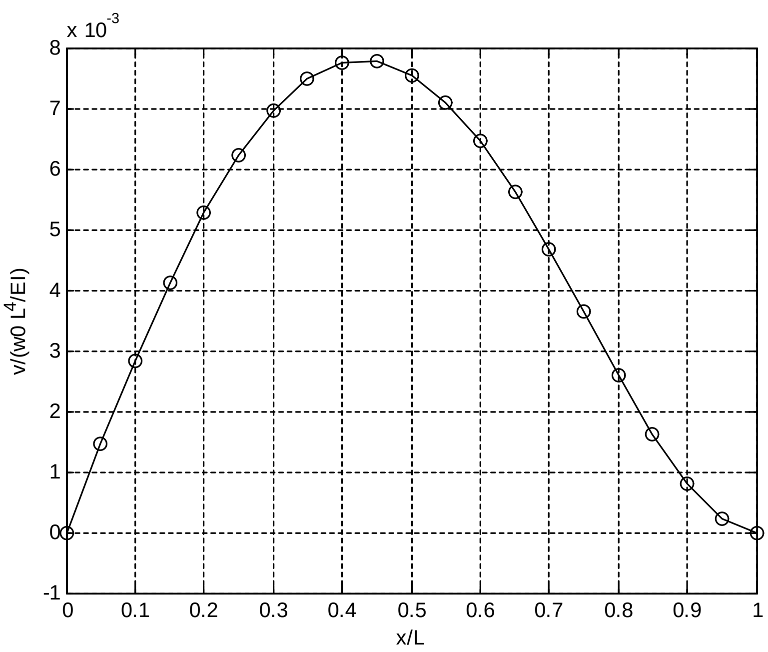
which transforms the differential equation into

$$\frac{d^4 y}{d\xi^4} = 1 + \xi$$

Here is the modified function `shoot4` that solves the problem:

```
function p8_1_22
% Shooting method for 4th-order boundary value
% problem in Prob. 8.1.22
xStart = 0; xStop = 1;          % Range of integration.
h = 0.05;                        % Step size.
freq = 1;                        % Frequency of printout.
u = [0 1];                       % Trial values of u(1)
                                 % and u(2).
x = xStart;
u = newtonRaphson2(@residual,u);
[xSol,ySol] = bulStoer(@dEqs,x,inCond(u),xStop,h);
printSol(xSol,ySol,freq)
```

```
plot(xSol,ySol(:,1),'k-o')
grid on
xlabel('x/L'); ylabel('v/(w0 L^4/EI)')

    function F = dEqs(x,y)    % Differential equations
        F = [y(2) y(3) y(4) 1 + x];
    end

    function y = inCond(u)    % Initial conditions; u(1)
        y = [0 u(1) 0 u(2)];  % and u(2) are unknowns.
    end

    function r = residual(u)  % Boundary residuals.
        r = zeros(length(u),1);
        x = xStart;
        [xSol,ySol] = bulStoer(@dEqs,x,inCond(u),xStop,h);
        lastRow = size(ySol,1);
        r(1 )= ySol(lastRow,1);
        r(2) = ySol(lastRow,2);
    end
end
```



PROBLEM SET 8.2

Problem 1

$$y'' = (2+x)y \qquad y(0) = 0 \qquad y'(1) = 5$$

With $f = (2+x)y$ Eqs. (8.11) become

$$\begin{aligned}
y_1 &= 0 \\
y_{i-1} - 2y_i + y_{i+1} - h^2(2+x_i)y_i &= 0, \quad i = 2,3,\ldots,n-1 \\
2y_{n-1} - 2y_n - h^2(2+x_n)y_n &= 0
\end{aligned}$$

$$\begin{aligned}
y_1 &= 0 \\
y_{i-1} - \left[2 + h^2(2+x_i)\right] y_i + y_{i+1} &= 0, \quad i = 2,3,\ldots,n-1 \\
2y_{n-1} - \left[2 + h^2(2+x_n)\right] y_n - y_n &= 0
\end{aligned}$$

Problem 2

$$y'' = y + x^2 \qquad y(0) = 0 \qquad y(1) = 1$$

Using $f = y + x^2$, Eqs. (8.11) become

$$\begin{aligned}
y_1 &= 0 \\
y_{i-1} - 2y_i + y_{i+1} - h^2(y_i + x_i^2) &= 0, \quad i = 2,3,\ldots,n-1 \\
y_n - 1 &= 0
\end{aligned}$$

$$\begin{aligned}
y_1 &= 0 \\
y_{i-1} - \left(2 + h^2\right) y_i + y_{i+1} &= h^2 x_i^2, \quad i = 2,3,\ldots,n-1 \\
y_n &= 1
\end{aligned}$$

Problem 3

$$y'' = e^{-x}y' \qquad y(0) = 1 \qquad y(1) = 0$$

With $f = e^{-x}y$ Eqs. (8.11) are

$$\begin{aligned} y_1 &= 1 \\ y_{i-1} - 2y_i + y_{i+1} - h^2 e^{-x_i} y_i &= 0, \quad i = 2, 3, \ldots, n-1 \\ y_n &= 0 \end{aligned}$$

$$\begin{aligned} y_1 &= 1 \\ y_{i-1} - \left(2 + h^2 e^{-x_i}\right) y_i + y_{i+1} &= 0, \quad i = 2, 3, \ldots, n-1 \\ y_n &= 0 \end{aligned}$$

Problem 4

$$y^{(4)} = y'' - y \qquad y(0) = y(1) = 0 \qquad y'(0) = 1 \qquad y'(1) = -1$$

Substituting $f = y'' - y$ into Eqs. (8.13) we get

$$\begin{aligned} y_{-1} - 4y_0 + 6y_1 - 4y_2 + y_3 - h^4 \left(\frac{y_0 - 2y_1 + y_2}{h^2} - y_1 \right) &= 0 \\ y_0 - 4y_1 + 6y_2 - 4y_3 + y_4 - h^4 \left(\frac{y_1 - 2y_2 + y_3}{h^2} - y_2 \right) &= 0 \\ y_1 - 4y_2 + 6y_3 - 4y_4 + y_5 - h^4 \left(\frac{y_2 - 2y_3 + y_4}{h^2} - y_3 \right) &= 0 \\ &\vdots \\ y_{n-3} - 4y_{n-2} + 6y_{n-1} - 4y_n + y_{n+1} - h^4 \left(\frac{y_{n-2} - 2y_{n-1} + y_n}{h^2} - y_{n-1} \right) &= 0 \\ y_{n-1} - 4y_{n-1} + 6y_n - 4y_{n+1} + y_{n+2} - h^4 \left(\frac{y_{n-1} - 2y_n + y_{n+1}}{h^2} - y_n \right) &= 0 \end{aligned}$$

From Table 8.1 equivalent the boundary conditions are

$$\begin{aligned} y_1 &= 0 \qquad y_0 = y_2 - 2h \\ y_n &= 0 \qquad y_{n+1} = y_{n-1} - 2h \end{aligned}$$

Therefore, the finite difference equations become

$$\begin{aligned} y_1 &= 0 \\ -\left(4 + h^2\right) y_1 + \left(7 + 2h^2 + h^4\right) y_2 - \left(4 + h^2\right) y_3 + y_4 &= 2h \\ y_1 - (4 + h^2) y_2 + (6 + 2h^2 + h^4) y_3 - (4 + h^2) y_4 + y_5 &= 0 \\ &\vdots \\ y_{n-3} - \left(4 + h^2\right) y_{n-2} + \left(7 + 2h^2 + h^4\right) y_{n-1} - \left(4 + h^2\right) y_n &= 2h \\ y_n &= 0 \end{aligned}$$

Problem 5

$$y^{(4)} = -9y + x \qquad y(0) = y''(0) = 0 \qquad y'(0) = y'''(0) = 0$$

With $f = -9y + x$ Eqs. (8.13) yield

$$\begin{aligned}
y_{-1} - 4y_0 + 6y_1 - 4y_2 + y_3 - h^4\left(-9y_1 + x_1\right) &= 0 \\
y_0 - 4y_1 + 6y_2 - 4y_3 + y_4 - h^4\left(-9y_2 + x_2\right) &= 0 \\
y_1 - 4y_2 + 6y_3 - 4y_4 + y_5 - h^4\left(-9y_3 + x_3\right) &= 0 \\
&\vdots \\
y_{n-3} - 4y_{n-2} + 6y_{n-1} - 4y_n + y_{n+1} - h^4\left(-9y_{n-1} + x_{n-1}\right) &= 0 \\
y_{n-2} - 4y_{n-1} + 6y_n - 4y_{n+1} + y_{n+2} - h^4\left(-9y_n + x_n\right) &= 0
\end{aligned}$$

According to Table 8.1 the boundary conditions are equivalent to

$$\begin{aligned}
y_1 &= 0 \qquad y_0 = 2y_1 - y_2 \\
y_{n+1} &= y_{n-1} \qquad y_{n+2} = 2y_{n+1} - 2y_{n-1} + y_{n-1} = y_{n-2}
\end{aligned}$$

The finite difference equations now become

$$\begin{aligned}
y_1 &= 0 \\
-2y_1 + (5 + 9h^4)y_2 - 4y_3 + y_4 &= h^4 x_2 \\
y_1 - 4y_2 + (6 + 9h^4)y_3 - 4y_4 + y_5 &= h^4 x_3 \\
&\vdots \\
y_{n-3} - 4y_{n-2} + (7 + 9h^4)y_{n-1} - 4y_n &= h^4 x_{n-1} \\
2y_{n-2} - 8y_{n-1} + (6 + 9h^4)y_n &= h^4 x_n
\end{aligned}$$

Problem 6

$$y'' = xy \qquad y(1) = 1.5 \qquad y(2) = 3$$

The finite difference equations are

$$\begin{aligned}
y_1 &= 1.5 \\
y_{i-1} - 2y_i + y_{i+1} - h^2 x_i y_i &= 0, \quad i = 2, 3, \ldots, n-1 \\
y_n &= 3
\end{aligned}$$

or

$$\begin{aligned}
y_0 &= 1.5 \\
y_{i-1} - \left(2 + h^2 x_i\right) y_i + y_{i+1} &= 0, \quad i = 2, 3, \ldots, n-1 \\
y_n &= 3
\end{aligned}$$

The following program is based on the function `fDiff6` in Example 8.6.

```
function p8_2_6
% Finite difference method for the second-order,
% linear boundary value problem in Problem 6,
% Problem Set 8.2.

xStart = 1; xStop = 2;      % Range of integration.
n = 21;                     % Number of mesh points.
freq = 2;                   % Printout frequency.

h = (xStop - xStart)/(n-1);
x = linspace(xStart,xStop,n)';
[c,d,e,b] = fDiffEqs(x,h,n);
[c,d,e] = LUdec3(c,d,e);
printSol(x,LUsol3(c,d,e,b),freq)

function [c,d,e,b] = fDiffEqs(x,h,n)
% Sets up the tridiagonal coefficient matrix and the
% constant vector of the finite difference equations.
h2 = h*h;
d = -h2.*x - 2
c = ones(n-1,1);
e = ones(n-1,1);
b = zeros(n,1);
d(1) = 1; e(1) = 0; b(1) = 1.5;
d(n) = 1; c(n-1) = 0; b(n) = 3;
```

The program prints every second point of the solution:

```
>>      x            y1
   1.0000e+000   1.5000e+000
   1.1000e+000   1.5372e+000
   1.2000e+000   1.5914e+000
   1.3000e+000   1.6647e+000
   1.4000e+000   1.7597e+000
   1.5000e+000   1.8793e+000
   1.6000e+000   2.0272e+000
   1.7000e+000   2.2075e+000
   1.8000e+000   2.4255e+000
   1.9000e+000   2.6872e+000
   2.0000e+000   3.0000e+000
```

Problem 7

$$y'' = -2y' - y \qquad y(0) = 0 \qquad y(1) = 1$$

The finite difference equations are

$$\begin{aligned} y_1 &= 0 \\ y_{i-1} - 2y_i + y_{i+1} - h^2\left(-2\frac{y_{i+1} - y_{i-1}}{2h} - y_i\right) &= 0, \quad i = 2, 3, \ldots, n-1 \\ y_n &= 1 \end{aligned}$$

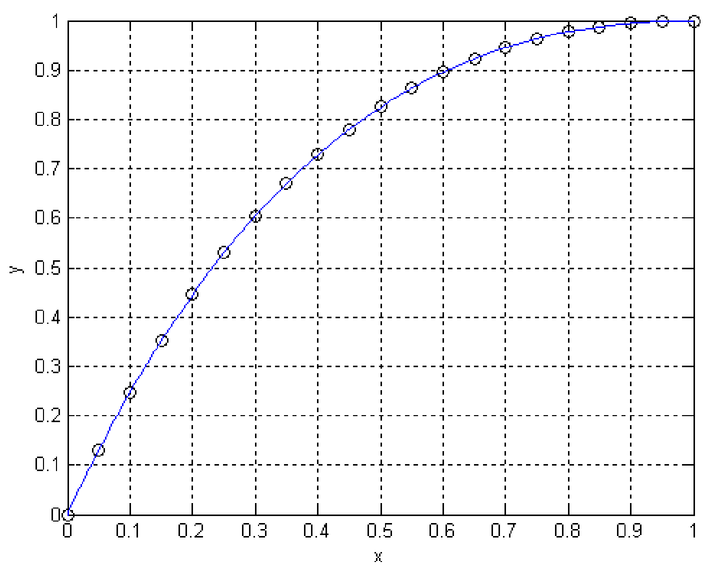
or

$$\begin{aligned} y_1 &= 0 \\ (1-h)y_{i-1} - (2-h^2)y_i + (1+h)y_{i+1} &= 0, \quad i = 2, 3, \ldots, n-1 \\ y_n &= 1 \end{aligned}$$

```
function p8_2_7
% Finite difference method for the second-order,
% linear boundary value problem in Problem 7,
% Problem Set 8.2.

xStart = 0; xStop = 1;      % Range of integration.
n = 21;                     % Number of mesh points.
freq = 2;                   % Printout frequency.

h = (xStop - xStart)/(n-1);
x = linspace(xStart,xStop,n)';
[c,d,e,b] = fDiffEqs(x,h,n);
[c,d,e] = LUdec3(c,d,e);
y = LUsol3(c,d,e,b);
printSol(x,y,freq)
plot(x,y,'ko'); hold on
fplot('x*exp(1-x)',[0,1]); grid on
xlabel('x'); ylabel('y')

function [c,d,e,b] = fDiffEqs(x,h,n)
% Sets up the tridiagonal coefficient matrix and the
% constant vector of the finite difference equations.
h2 = h*h;
d = ones(n,1)*(-2 + h^2);
c = ones(n-1,1)*(1 - h);
e = ones(n-1,1)*(1 + h);
b = zeros(n,1);
d(1) = 1; e(1) = 0;
```

```
d(n) = 1; c(n-1) = 0; b(n) = 1;
>>      x             y1
   0.0000e+000    0.0000e+000
   1.0000e-001    2.4612e-001
   2.0000e-001    4.4536e-001
   3.0000e-001    6.0442e-001
   4.0000e-001    7.2915e-001
   5.0000e-001    8.2464e-001
   6.0000e-001    8.9533e-001
   7.0000e-001    9.4509e-001
   8.0000e-001    9.7725e-001
   9.0000e-001    9.9472e-001
   1.0000e+000    1.0000e+000
```

The plot shows the numerical solution (open circles) together with the analytical solution (solid line).



Problem 8

$$y'' = -\frac{1}{x}y' - \frac{1}{x^2}y \qquad y(1) = 0 \qquad y(2) = 0.638961$$

The finite difference equations are

$$\begin{aligned} y_1 &= 0 \\ y_{i-1} - 2y_i + y_{i+1} - h^2\left(-\frac{y_{i+1} - y_{i-1}}{2hx_i} - \frac{y_i}{x_i^2}\right) &= 0, \quad i = 2, 3, \ldots, n-1 \\ y_n &= 0.638961 \end{aligned}$$

or

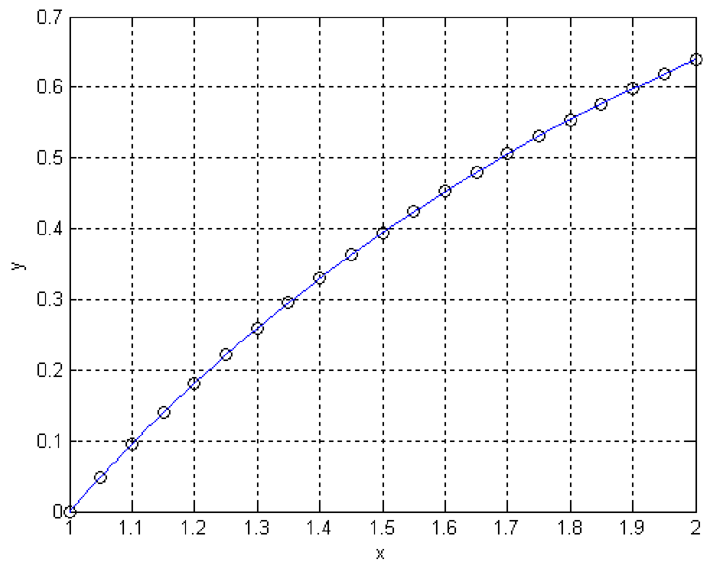
$$
\begin{aligned}
y_1 &= 0 \\
\left(1 - \frac{h}{2x_i}\right) y_{i-1} - \left(2 - \frac{h^2}{x_i^2}\right) y_i + \left(1 + \frac{h}{2x_i}\right) y_{i+1} &= 0, \quad i = 2, 3, \ldots, n-1 \\
y_n &= 0.638961
\end{aligned}
$$

Here are the finite difference equations:

```
function p8_2_8
% Finite difference method for the second-order,
% linear boundary value problem in Problem 8,
% Problem Set 8.2.

xStart = 1; xStop = 2;      % Range of integration.
n = 21;                     % Number of mesh points.
freq = 2;                   % Printout frequency.

h = (xStop - xStart)/(n-1);
x = linspace(xStart,xStop,n)';
[c,d,e,b] = fDiffEqs(x,h,n);
[c,d,e] = LUdec3(c,d,e);
y = LUsol3(c,d,e,b);
printSol(x,y,freq)
plot(x,y,'ko'); hold on
fplot('sin(log(x))',[1,2]); grid on
xlabel('x'); ylabel('y')

function [c,d,e,b] = fDiffEqs(x,h,n)
% Sets up the tridiagonal coefficient matrix and the
% constant vector of the finite difference equations.
h2 = h*h;
d = h2./x./x - 2;
c = -0.5*h./x(2:n) + 1;
e = 0.5*h./x(1:n-1) + 1;
b = zeros(n,1);
d(1) = 1; e(1) = 0;
d(n) = 1; c(n-1) = 0; b(n) = 0.638961;
>>       x             y1
   1.0000e+000   0.0000e+000
   1.1000e+000   9.5170e-002
   1.2000e+000   1.8132e-001
   1.3000e+000   2.5937e-001
   1.4000e+000   3.3016e-001
   1.5000e+000   3.9445e-001
   1.6000e+000   4.5289e-001
```

```
1.7000e+000    5.0608e-001
1.8000e+000    5.5452e-001
1.9000e+000    5.9868e-001
2.0000e+000    6.3896e-001
```

The plot shows the numerical solution (open circles) together with the analytical solution (solid line).



Problem 9

$$y'' = y^2 \sin y \qquad y'(0) = 0 \qquad y(\pi) = 1$$

The finite difference equations are

$$\begin{aligned} -2y_1 + 2y_2 - h^2 F(x_1, y_1, y_1') &= 0 \\ y_{i-1} - 2y_i + y_{i+1} - h^2 F(x_i, y_i, y_i') &= 0, \quad i = 2, 3, \ldots, n-1 \\ y_n &= 1 \end{aligned}$$

In arriving at the first equation, we utilize the equivalent boundary condition $y_0 = y_2$. The quadratic $y = (x/\pi)^2$ was chosen for the starting solution (note that its satisfies the prescribed boundary conditions). The following program is based on the function `fDiff7` in Example 8.7.

```
function p8_2_9
% Finite difference method for the second-order,
% nonlinear boundary value problem in Problem 9,
% Problem Set 8.2.
```

```
xStart = 0; xStop = pi;      % Range of integration.
n = 21;                      % Number of mesh points.
freq = 2;                    % Printout frequency.
x = linspace(xStart,xStop,n)';
y = x.*x/pi^2;               % Starting values of y.

h = (xStop - xStart)/(n-1);
y = newtonRaphson2(@residual,y,1.0e-5);
printSol(x,y,freq)

    function r = residual(y);
    % Residuals of finite difference equations: left-hand
    % sides of Eqs (8.11).
    r = zeros(n,1);
    r(1) = -2*y(1) + 2*y(2)...
           - h*h*y2Prime(x(1),y(1),0);
    r(n) = y(n) - 1;
    for i = 2:n-1
        r(i) = y(i-1) - 2*y(i) + y(i+1)...
             - h*h*y2Prime(x(i),y(i),(y(i+1) - y(i-1))/(2*h));
    end
    end

    function F = y2Prime(x,y,yPrime)
    % Second-order differential equation F = y".
    F = (y^2)*sin(y);
    end
end
>>      x           y1
   0.0000e+000   4.1338e-001
   3.1416e-001   4.1678e-001
   6.2832e-001   4.2714e-001
   9.4248e-001   4.4498e-001
   1.2566e+000   4.7127e-001
   1.5708e+000   5.0757e-001
   1.8850e+000   5.5631e-001
   2.1991e+000   6.2132e-001
   2.5133e+000   7.0875e-001
   2.8274e+000   8.2892e-001
   3.1416e+000   1.0000e+000
```

Problem 10

$$y'' = -2y(2xy' + y) \qquad y(0) = \frac{1}{2} \qquad y'(1) = -\frac{2}{9}$$

The finite difference equations are

$$\begin{aligned}
y_1 &= 0.5 \\
y_{i-1} - 2y_i + y_{i+1} - h^2 F(x_i, y_i, y_i') &= 0, \quad i = 2, 3, \ldots, n-1 \\
y_{n-1} - 2y_n + y_{n+1} - h^2 F(x_n, y_n, y_n') &= 0
\end{aligned}$$

The bounday condition $y_n' = -2/9$ is equivalent to

$$\frac{y_{n+1} - y_{n-1}}{2h} = -\frac{2}{9} \qquad y_{n+1} = y_{n-1} - \frac{4}{9}h$$

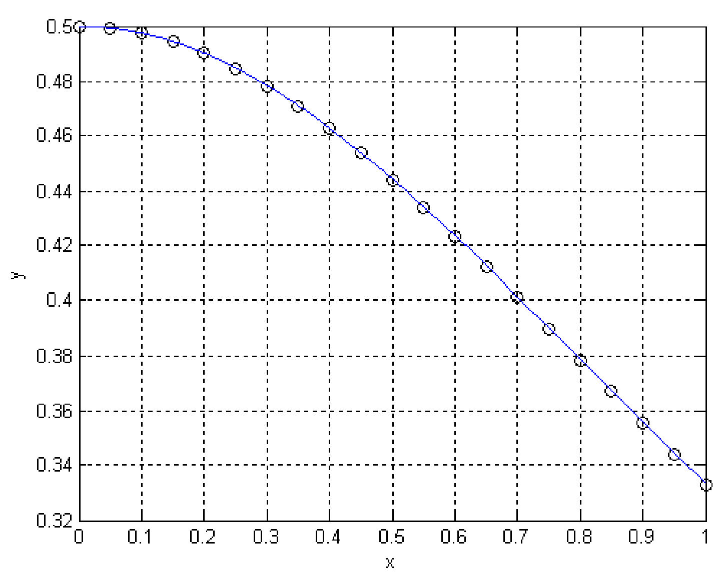
so that the last finite difference equation becomes

$$2y_{n-1} - 2y_n - \frac{4}{9}h - h^2 F\left(x_n, y_n, -\frac{2}{9}\right) = 0$$

```
function p8_2_10
% Finite difference method for the second-order,
% nonlinear boundary value problem in Problem 10,
% Problem Set 8.2.

xStart = 0; xStop = 1;        % Range of integration.
n = 21;                       % Number of mesh points.
freq = 2;                     % Printout frequency.
x = linspace(xStart,xStop,n)';
y = -2/9*x + 0.5;             % Starting values of y.
h = (xStop - xStart)/(n-1);
y = newtonRaphson2(@residual,y,1.0e-5);
printSol(x,y,freq)
plot(x,y,'ko'); hold on
fplot('1/(2+x.*x)',[0 1]); grid on
xlabel('x'); ylabel('y')

    function r = residual(y);
    % Residuals of finite difference equations: left-hand
    % sides of Eqs (8.11).
    r = zeros(n,1);
    r(1) = y(1) - 0.5;
    r(n) = 2*y(n-1) - 2*y(n) - 4/9*h...
         - h*h*y2Prime(x(n),y(n),-2/9);
    for i = 2:n-1
        r(i) = y(i-1) - 2*y(i) + y(i+1)...
```

```
            - h*h*y2Prime(x(i),y(i),(y(i+1) - y(i-1))/(2*h));
    end
    end

    function F = y2Prime(x,y,yPrime)
    % Second-order differential equation F = y".
    F = -2*y*(2*x*yPrime + y);
    end
end
>>       x              y1
   0.0000e+000   5.0000e-001
   1.0000e-001   4.9746e-001
   2.0000e-001   4.9009e-001
   3.0000e-001   4.7831e-001
   4.0000e-001   4.6275e-001
   5.0000e-001   4.4418e-001
   6.0000e-001   4.2342e-001
   7.0000e-001   4.0126e-001
   8.0000e-001   3.7842e-001
   9.0000e-001   3.5548e-001
   1.0000e+000   3.3293e-001
```

The plot shows the numerical solution (open circles) together with the analytical solution (solid line).



Problem 11

$$y'' = \begin{cases} -0.25x & 0 < x < 0.25 \\ -\dfrac{0.25}{\gamma}\left[x - 2\left(x - 0.25\right)^2\right] & 0.25 < x < 0.5 \end{cases}$$

$$y(0) = 0 \qquad y'(0.5) = 0$$

The finite difference equations are

$$\begin{aligned} y_1 &= 0 \\ y_{i-1} - 2y_i + y_{i+1} &= \begin{cases} -0.25x_i h^2 & 0 < x_i < 0.25 \\ -\dfrac{0.25}{\gamma}\left[x_i - 2\left(x_i - 0.25\right)^2\right] h^2 & 0.25 < x_i < 0.5 \end{cases} \\ y_{n-1} - 2y_n + y_{n+1} &= -\frac{0.25}{\gamma}\left[x_n - 2\left(x_n - 0.25\right)^2\right] h^2 \end{aligned}$$

The bounday condition $y_n' = 0$ is equivalent to $y_{n+1} = y_{n-1}$ so that the last finite difference equation becomes

$$2y_{n-1} - 2y_n = -\frac{0.25}{\gamma}\left[x_n - 2\left(x_n - 0.25\right)^2\right] h^2$$

```
function p8_2_11
% Finite difference method for the second-order,
% linear boundary value problem in Problem 11,
% Problem Set 8.2.

xStart = 0; xStop = 0.5;    % Range of integration.
n = 21;                     % Number of mesh points.
freq = 0;                   % Printout frequency.

h = (xStop - xStart)/(n-1);
x = linspace(xStart,xStop,n)';
[c,d,e,b] = fDiffEqs(x,h,n);
[c,d,e] = LUdec3(c,d,e);
printSol(x,LUsol3(c,d,e,b),freq)

function [c,d,e,b] = fDiffEqs(x,h,n)
% Sets up the tridiagonal coefficient matrix and the
% constant vector of the finite difference equations.
h2 = h*h;
m = ceil(n/2);
d = ones(n,1)*(-2);
```

```
c = ones(n-1,1);
e = ones(n-1,1);
b = ones(n,1);
b(1:m) = -0.25*x(1:m)*h2;
for i = m+1:n
    b(i) = -0.25/1.5*(x(i) - 2*(x(i) - 0.25)^2)*h2;
end
% If n is odd, average the b's at the midpoint node
if rem(m,2) ~= 0;
    b(m) = -0.125/1.5*(x(m) - 2*(x(m) - 0.25)^2)*h2...
           - 0.125*x(m)*h2;
end
d(1) = 1; e(1) = 0; b(1) = 0; c(n-1) = 2;
```

```
>>      x             y1
   0.0000e+000   0.0000e+000
   5.0000e-001   6.6243e-003
```

Thus the numerical solution gives

$$v_{\max} = 0.006624\frac{w_0 L^4}{EI} \quad \blacktriangleleft$$

whereas the analytical solution is

$$v_{\max} = \frac{61}{9216}\frac{w_0 L^4}{EI} = 0.006619\frac{w_0 L^4}{EI}$$

Problem 12

$$y'' = -\frac{1-x}{1+[(\delta-1)x]^4} \qquad y(0) = y(1) = 0$$

The finite difference equations are

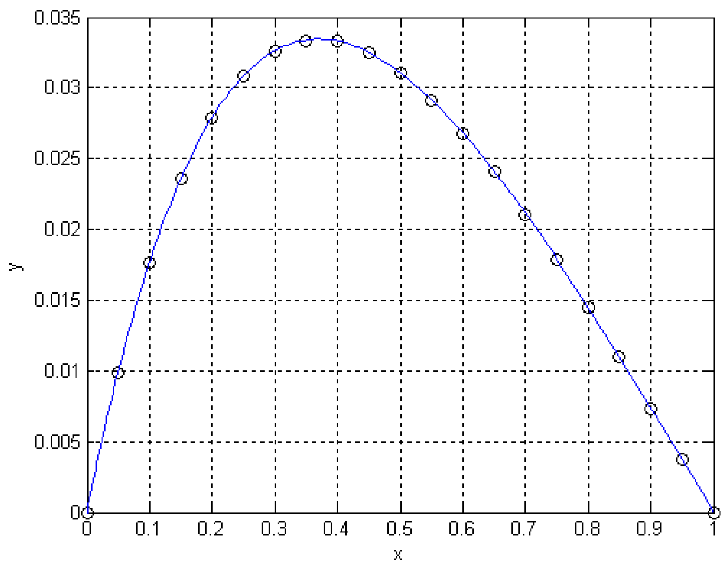
$$\begin{aligned} y_1 &= 0 \\ y_{i-1} - 2y_i + y_{i+1} &= -\frac{1-x_i}{1+[(\delta-1)x_i]^4}h^2, \quad i = 2,3,\ldots,n-1 \\ y_n &= 0 \end{aligned}$$

```
function p8_2_12
% Finite difference method for the second-order,
% linear boundary value problem in Problem 12,
% Problem Set 8.2.
```

```
xStart = 0; xStop = 1;      % Range of integration.
n = 21;                     % Number of mesh points.

h = (xStop - xStart)/(n-1);
x = linspace(xStart,xStop,n)';
[c,d,e,b] = fDiffEqs(x,h,n);
[c,d,e] = LUdec3(c,d,e);
y = LUsol3(c,d,e,b);
plot(x,y,'ko'); hold on
fplot('-(x^2)/2/(1 + 0.5*x )^2 + x/4.5',[0 1])
grid on; xlabel('x'); ylabel('y')

function [c,d,e,b] = fDiffEqs(x,h,n)
% Sets up the tridiagonal coefficient matrix and the
% constant vector of the finite difference equations.
h2 = h*h;
d = ones(n,1)*(-2);
c = ones(n-1,1);
e = ones(n-1,1);
b = zeros(n,1);
for i = 2:n-1
    b(i) = -(1 - x(i))/(1 + 0.5*x(i))^4*h2;
end
d(1) = 1; e(1) = 0; d(n) = 1; c(n-1) = 0;
```

The plot shows the numerical solution (open circles) together with the analytical solution (solid line)

$$y = -\frac{x^2}{2(1+0.5x)^2} + \frac{x}{4.5} \quad \text{with } \gamma = 1.5$$



Problem 13

$$y^{(4)} = x \qquad y(0) = y''(0) = y(1) = y''(1) = 0$$

Taking into account the boundary conditions $y_0 = -y_2$ and $y_{n+1} = -y_{n-1}$, the finite difference equations are

$$\begin{aligned} y_1 &= 0 \\ -4y_1 + 5y_2 - 4y_3 + y_4 &= h^4 x_2 \\ y_{i-2} - 4y_{i-1} + 6y_i - 4y_{i+1} + y_{i+2} &= h^4 x_i, \quad i = 3, 4, \ldots, n-2 \\ y_{n-3} - 4y_{n-2} + 5y_{n-1} - 4y_n &= h^4 x_{n-1} \\ y_n &= 0 \end{aligned}$$

```
function p8_2_13
% Finite difference method for the 4th-order,
% linear boundary value problem in Problem 13,
% Problem Set 8.2.

xStart = 0; xStop = 1;     % Range of integration.
n = 21;                    % Number of mesh points.
freq = 1;                  % Printout frequency.
h = (xStop - xStart)/(n-1);
x = linspace(xStart,xStop,n)';
[d,e,f,b] = fDiffEqs(x,h,n);
[d,e,f] = LUdec5(d,e,f);
```

```
printSol(x,LUsol5(d,e,f,b),freq)

function [d,e,f,b] = fDiffEqs(x,h,n)
% Sets up the pentadiagonal coefficient matrix and the
% constant vector of the finite difference equations.
h4 = h^4;
d = ones(n,1)*6;
e = ones(n-1,1)*(-4);
f = ones(n-2,1);
b = zeros(n,1);
b(2:n-1) = h4*x(2:n-1);
d(1) = 1; d(2) = 5; d(n-1) = 5; d(n) = 1;
e(1) = 0; f(1) = 0; e(n-1) = 0; f(n-2) = 0;
```

Only the points needed for the computation of end slopes and mid-span displacement are shown below.

```
>>      x            y1
   0.0000e+000   0.0000e+000
   5.0000e-002   9.7048e-004
   1.0000e-001   1.9202e-003

   5.0000e-001   6.5234e-003

   9.0000e-001   2.1767e-003
   9.5000e-001   1.1076e-003
   1.0000e+000   0.0000e+000
```

$$y(0.5) = 6.523 \times 10^{-3}$$

From Tables 5.3:

$$\begin{aligned} y'(0) &\approx \frac{-3y(0) + 4y(0.05) - y(0.1)}{2h} \\ &= \frac{-3(0) + 4(0.970484) - 1.92019}{0.1} \times 10^{-3} = 19.62 \times 10^{-3} \end{aligned}$$

$$\begin{aligned} y'(1) &= \frac{y(0.9) - 4y(0.95) + 3y(1.0)}{2h} \\ &= \frac{2.17669 - 4(1.10764) + 3(0)}{0.1} \times 10^{-3} = -22.54 \times 10^{-3} \end{aligned}$$

Therefore, the mid-span displacement and the end slopes are (the numbers in parenthesis are the numerical factors obtained from the analytical solution):

$$\begin{aligned}
v(0.5L) &= 6.523 \times 10^{-3}\frac{w_0 L^4}{EI} \quad \blacktriangleleft \quad (6.510)\\
\left.\frac{dv}{dx}\right|_{x=0} &= 19.62 \times 10^{-3}\frac{w_0 L^3}{EI} \quad \blacktriangleleft \quad (19.44)\\
\left.\frac{dv}{dx}\right|_{x=L} &= -22.54 \times 10^{-3}\frac{w_0 L^3}{EI} \quad \blacktriangleleft \quad (-22.22)
\end{aligned}$$

Problem 14

$$y^{(4)} = \beta y'' + 1 \qquad y(0) = y''(0) = y(1) = y''(1) = 0$$

The finite difference equations are

$$\begin{aligned}
y_1 &= 0\\
y_0 - 4y_1 + 6y_2 - 4y_3 + y_4 - h^4\left(\beta\frac{y_1 - 2y_2 + y_3}{h^2} + 1\right) &= 0\\
y_1 - 4y_2 + 6y_3 - 4y_4 + y_5 - h^4\left(\beta\frac{y_2 - 2y_3 + y_4}{h^2} + 1\right) &= 0\\
&\vdots\\
y_{n-3} - 4y_{n-2} + 6y_{n-1} - 4y_n + y_{n+1} - h^4\left(\beta\frac{y_{n-2} - 2y_{n-1} + y_n}{h^2} + 1\right) &= 0\\
y_n &= 0
\end{aligned}$$

After using the boundary conditions $y_0 = -y_2$ and $y_{n+1} = -y_{n-1}$, we get

$$\begin{aligned}
y_1 &= 0\\
-\left(4 + h^2\beta\right)y_1 + \left(5 + 2h^2\beta\right)y_2 - \left(4 + h^2\beta\right)y_3 + y_4 &= h^4\\
y_1 - \left(4 + h^2\beta\right)y_2 + \left(6 + 2h^2\beta\right)y_3 - \left(4 + h^2\beta\right)y_4 + y_5 &= h^4\\
&\vdots\\
y_{n-3} - \left(4 + h^2\beta\right)y_{n-2} + \left(5 + 2h^2\beta\right)y_{n-1} - \left(4 + h^2\beta\right)y_n &= h^4\\
y_n &= 0
\end{aligned}$$

(a)

```
function p8_2_14
% Finite difference method for the 4th-order,
% linear boundary value problem in Problem 14,
% Problem Set 8.2.
```

```
xStart = 0; xStop = 1;    % Range of integration.
n = 21;                   % Number of mesh points.
freq = 10;                % Printout frequency.
h = (xStop - xStart)/(n-1);
x = linspace(xStart,xStop,n)';
[d,e,f,b] = fDiffEqs(x,h,n);
[d,e,f] = LUdec5(d,e,f);
printSol(x,LUsol5(d,e,f,b),freq)

function [d,e,f,b] = fDiffEqs(x,h,n)
% Sets up the pentadiagonal coefficient matrix and the
% constant vector of the finite difference equations.
beta = 1.65929
h2 = h*h; h4 = h2*h2;
d = ones(n,1)*(6 + 2*h2*beta);
e = ones(n-1,1)*(-4 - h2*beta);
f = ones(n-2,1);
b = ones(n,1)*h4;
b(1) = 0; b(n) = 0;
d(1) = 1; d(n) = 1;
d(2) = 5 + 2*h2*beta; d(n-1) = 5 + 2*h2*beta;
e(1) = 0; f(1) = 0; e(n-1) = 0; f(n-2) = 0;
```

```
>> beta =
    1.6593
     x             y1
   0.0000e+000   0.0000e+000
   5.0000e-001   1.1159e-002
   1.0000e+000   0.0000e+000
```

$$v_{\max} = 0.011\,16 \frac{w_0 L^4}{EI} \blacktriangleleft$$

(b)

Running the program with negative β yields

```
>> beta =
   -1.6593
     x             y1
   0.0000e+000   0.0000e+000
   5.0000e-001   1.5699e-002
   1.0000e+000   0.0000e+000
```

$$v_{\max} = 0.015\,70 \frac{w_0 L^4}{EI} \blacktriangleleft$$

Problem 15

$$y^{(4)} = -\gamma y + 1 \qquad y(0) = y''(0) = y(1) = y''(1) = 0$$

The finite difference equations are

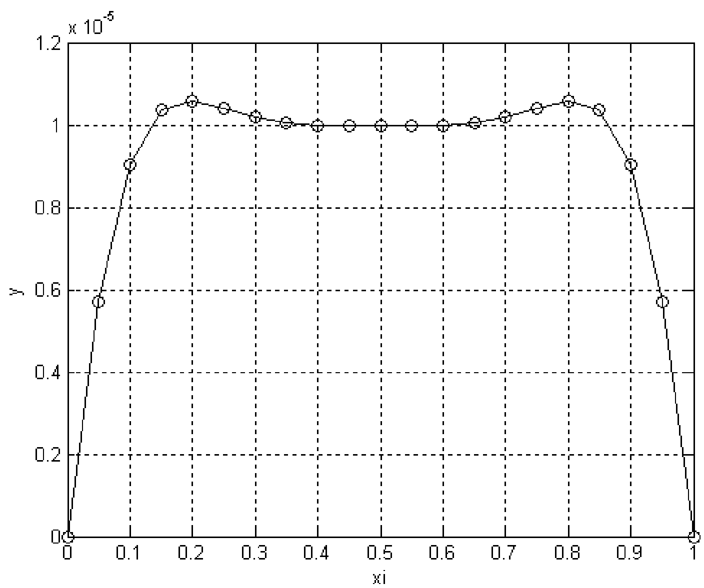
$$\begin{aligned}
y_1 &= 0 \\
y_0 - 4y_1 + 6y_2 - 4y_3 + y_4 - h^4\left(-\gamma y_2 + 1\right) &= 0 \\
y_1 - 4y_2 + 6y_3 - 4y_4 + y_5 - h^4\left(-\gamma y_3 + 1\right) &= 0 \\
&\vdots \\
y_{n-3} - 4y_{n-2} + 6y_{n-1} - 4y_n + y_{n+1} - h^4\left(-\gamma y_{n-1} + 1\right) &= 0 \\
y_n &= 0
\end{aligned}$$

With the boundary conditions $y_0 = -y_2$ and $y_{n+1} = -y_{n-1}$ these equations become

$$\begin{aligned}
y_1 &= 0 \\
-4y_1 + \left(5 + h^4\gamma\right) y_2 - 4y_4 + y_4 &= h^4 \\
y_1 - 4y_2 + \left(6 + h^4\gamma\right) y_3 - 4y_4 + y_5 &= h^4 \\
&\vdots \\
y_{n-3} - 4y_{n-2} + \left(5 + h^4\gamma\right) y_{n-1} - 4y_n &= h^4 \\
y_n &= 0
\end{aligned}$$

```
function p8_2_15
% Finite difference method for the 4th-order,
% linear boundary value problem in Problem 15,
% Problem Set 8.2.

xStart = 0; xStop = 1;    % Range of integration.
n = 21;                   % Number of mesh points.
h = (xStop - xStart)/(n-1);
x = linspace(xStart,xStop,n)';
[d,e,f,b] = fDiffEqs(x,h,n);
[d,e,f] = LUdec5(d,e,f);
y = LUsol5(d,e,f,b);
plot(x,y,'k-o'); grid on
xlabel('xi'); ylabel('y')

function [d,e,f,b] = fDiffEqs(x,h,n)
% Sets up the pentadiagonal coefficient matrix and the
% constant vector of the finite difference equations.
gamma = 1.0e5;
```

```
h2 = h*h; h4 = h2*h2;
d = ones(n,1)*(6 + h4*gamma);
e = ones(n-1,1)*(-4);
f = ones(n-2,1);
b = ones(n,1)*h4;
b(1) = 0; b(n) = 0;
d(1) = 1; d(n) = 1;
d(2) = 5 + h4*gamma; d(n-1) = 5 + h4*gamma;
e(1) = 0; f(1) = 0; e(n-1) = 0; f(n-2) = 0;
```



Problem 16

$$
\begin{aligned}
y^{(4)} &= \begin{cases} -\gamma y & \text{in } 0 < x < 0.25 \\ -\gamma y + 1 & \text{in } 0.25 < x < 0.5 \end{cases} \\
y''(0) &= y'''(0) = y'(0.5) = y'''(0.5) = 0
\end{aligned}
$$

The finite difference equations are

$$
\begin{aligned}
y_{-1} - 4y_0 + 6y_1 - 4y_2 + y_3 - h^4(-\gamma y_1) &= 0 \\
y_0 - 4y_1 + 6y_2 - 4y_3 + y_4 - h^4(-\gamma y_2) &= 0 \\
y_1 - 4y_2 + 6y_3 - 4y_4 + y_5 - h^4(-\gamma y_3) &= 0 \\
&\vdots \\
y_{n-3} - 4y_{n-2} + 6y_{n-1} - 4y_n + y_{n+1} - h^4(-\gamma y_{n-1} + 1) &= 0 \\
y_{n-2} - 4y_{n-1} + 6y_n - 4y_{n+1} + y_{n+2} - h^4(-\gamma y_n + 1) &= 0
\end{aligned}
$$

Substituting the equivalent boundary conditions in Table 8.1:

$$\begin{aligned} y_0 &= 2y_1 - y_2 \qquad y_{-1} = 2y_0 - 2y_2 + y_3 = 4y_1 - 4y_2 + y_3 \\ y_{n+1} &= y_{n-1} \qquad y_{n+2} = 2y_{n+1} - 2y_{n-1} + y_{n-2} = y_{n-2} \end{aligned}$$

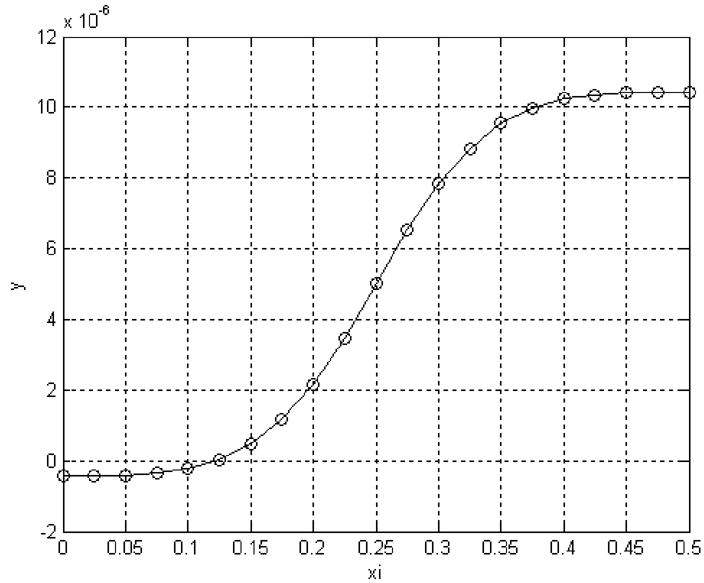
the finite difference equations become

$$\begin{aligned} \left(2 + h^4\gamma\right) y_0 - 4y_1 + 2y_2 &= 0 \\ -2y_0 + \left(5 + h^4\gamma\right) y_1 - 4y_2 + y_3 &= 0 \\ y_0 - 4y_1 + \left(6 + h^4\gamma\right) y_2 - 4y_3 + y_4 &= 0 \\ &\vdots \\ y_{m-3} - 4y_{m-2} + \left(7 + h^4\gamma\right) y_{m-1} - 4y_m &= h^4 \\ 2y_{m-2} - 8y_{m-1} + \left(6 + h^4\gamma\right) y_m &= h^4 \end{aligned}$$

To obtain a symmetric coefficient matrix, the first and last equations must be divided by 2.

```
function p8_2_16
% Finite difference method for the 4th-order,
% linear boundary value problem in Problem 16,
% Problem Set 8.2.

xStart = 0; xStop = 0.5;  % Range of integration.
n = 21;                   % Number of mesh points.
h = (xStop - xStart)/(n-1);
x = linspace(xStart,xStop,n)';
[d,e,f,b] = fDiffEqs(x,h,n);
[d,e,f] = LUdec5(d,e,f);
y = LUsol5(d,e,f,b);
plot(x,y,'k-o'); grid on
xlabel('xi'); ylabel('y')

function [d,e,f,b] = fDiffEqs(x,h,n)
% Sets up the pentadiagonal coefficient matrix and the
% constant vector of the finite difference equations.
gamma = 1.0e5;
h2 = h*h; h4 = h2*h2;
d = ones(n,1)*(6 + h4*gamma);
e = ones(n-1,1)*(-4);
f = ones(n-2,1);
b = zeros(n,1);
d(1) = 1 + 0.5*h4*gamma;
d(2) = 5 + h4*gamma;
d(n-1) = 7 + h4*gamma;
d(n) = 3 + 0.5*h4*gamma;
```

```
e(1) = -2;
m = ceil(n/2); b(m+1:n) = h4;
% If n is odd, average the b's at the midpoint node.
if rem(m,2) ~= 0; b(m) = 0.5*h4; end
b(n) = 0.5*h4;
```



Problem 17

Introducing the central finite difference approximations into the differential equation

$$y'' = r(x) + s(x)y + t(x)y'$$

we obtain

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

where we used the notation $r_i = r(x_i)$ etc. After collecting terms, the finite difference equations become

$$\left(1 + \frac{h}{2}t_i\right) y_{i-1} - \left(2 + h^2 s_i\right) y_i + \left(1 - \frac{h}{2}t_i\right) y_{i+1} = h^2 r_i$$

The first and last of these equations are modified when the boundary conditions are introduced. The boundary conditions at the left end are prescribed by `alpha = [code,value]`, where `code` specifies the variable on which the condition is imposed (`code = 0` refers to y and `code = 1` refers to y') and `value` is the prescribed value. The boundary conditions at the right end are prescibed by `beta = [code,value]` in the same manner.

```
function p8_2_17
% Problem 17, Problem Set 8.2:
% solution of linear, 2nd-order boundary value problem
% by the finite difference method.

xStart = 1;           % Value of x at left end
xStop = 2;            % Value of x at right end
n = 21;               % Number of mesh points
freq = 2;             % Prinout frequency
alpha = [0 0];        % Boundary conds. at left end
beta  = [0 0.638961]; % Boundary conds. at right end

h = (xStop - xStart)/(n - 1);
x = linspace(xStart,xStop,n);
[c,d,e,b] = eqs(@coeff,alpha,beta,x,h,n);
[c,d,e] = LUdec3(c,d,e);
y = LUsol3(c,d,e,b);
printSol(x,y,freq);

function [r,s,t] = coeff(z)
% Specify r,s,t in y'' = r(x) + s(x)y + t(x)y'
r = 0;
s = -1/z^2;
t = -1/z;

function [c,d,e,b] = eqs(coeff,alpha,beta,x,h,n)
h2 = h*h;
b = zeros(n,1); c = zeros(n-1,1);
d = zeros(n-1,1); e = zeros(n-1,1);
for i = 2:n-1
    [r,s,t] = coeff(x(i));
    c(i-1) = 1 + h*t/2;
    d(i) = -(2 + h2*s);
    e(i) = 1 - h*t/2;
    b(i) = h2*r;
end

% Apply boundary condition at left end.
code = alpha(1); value = alpha(2);
if code == 0    % y = value
    d(1) =1; e(1) = 0; b(1) = value;
else            % y' = value
    [r,s,t] = coeff(x(1));
    d(1) = -(2 + h2*s); e(1) = 2;
    b(1) = h2*(r + t*value) + 2*h*value;
```

```
end

% Apply boundary conditions at right end.
code = beta(1); value = beta(2);
if code == 0     % y = value
    d(n) = 1; c(n-1) = 0; b(n) = value;
else             % y' = value
    [r,s,t] = coeff(x(n));
    d(n) = -(2 + h2*s); c(n-1) = 0;
    b(n) = h2*(r + t*value) - 2*h*value;
end
```

```
>>       x             y1
   1.0000e+000   0.0000e+000
   1.1000e+000   9.5170e-002
   1.2000e+000   1.8132e-001
   1.3000e+000   2.5937e-001
   1.4000e+000   3.3016e-001
   1.5000e+000   3.9445e-001
   1.6000e+000   4.5289e-001
   1.7000e+000   5.0608e-001
   1.8000e+000   5.5452e-001
   1.9000e+000   5.9868e-001
   2.0000e+000   6.3896e-001
```

Problem 18

It is convenient to introduce he variable $x = r/a$. The differential equation and the boundary conditions then become

$$\frac{d^2T}{dx^2} = -\frac{1}{x}\frac{dT}{dx} \qquad T\big|_{x=0.5} = 0 \qquad T\big|_{x=1} = 200^\circ\ \mathrm{C}$$

Using 11 mesh points, the finite difference equations, Eqs. (8.11), are

$$\begin{aligned} T_1 &= 0 \\ T_{i-1} - 2T_i + T_{i+1} - h^2\left(-\frac{1}{x_i}\frac{T_{i+1} - T_{i-1}}{2h}\right) &= 0, \quad i = 2, 3, \ldots 10 \\ T_{11} &= 200 \end{aligned}$$

or

$$
\begin{aligned}
T_1 &= 0 \\
\left(1 - \frac{h}{2x_i}\right) T_{i-1} - 2T_i + \left(1 + \frac{h}{2x_i}\right) T_{i+1} &= 0, \quad i = 2, 3, \cdots 10 \\
T_{11} &= 200
\end{aligned}
$$

The following program is based on Example 8.6. It utilizes the tridiagonal structure of the equations.

```
function problem8_2_18
xStart = 0.5; xStop = 1;
n = 11;
h = (xStop - xStart)/(n-1);
x = zeros(n,1); y = zeros(n,2);
x(1) = xStart;
for i = 2:n
    x(i) = x(i-1) + h;
    y(i,2) = 200*(1 - log(x(i))/log(0.5)); % Analytical soln.
end
[c,d,e,b] = fdEqs(x,h,n);
[c,d,e] = LUdec3(c,d,e);
y(:,1) = LUsol3(c,d,e,b);                  % Numerical soln.
printSol(x,y,1)

function[c,d,e,b] = fdEqs(x,h,n)
% Sets ub finite difference (tridiagonal) equations
h2 = h*h;
d = ones(n,1)*(-2);
c = zeros(n-1,1);
e = zeros(n-1,1);
for i = 1:n-1
    c(i) = 1 - h/2/x(i+1);
    e(i) = 1 + h/2/x(i);
end
b = zeros(n,1);
e(1) = 0; d(1) = 1;
b(n) = 200; d(n) = 1; c(n-1) = 0;
```

In the printout y1 is the numerical solution and y2 is the analytical solution. The two are in good agreement.

```
     x            y1            y2
  5.0000e-001   0.0000e+000   0.0000e+000
  5.5000e-001   2.7492e+001   2.7501e+001
  6.0000e-001   5.2594e+001   5.2607e+001
```

```
6.5000e-001   7.5687e+001   7.5702e+001
7.0000e-001   9.7070e+001   9.7085e+001
7.5000e-001   1.1698e+002   1.1699e+002
8.0000e-001   1.3560e+002   1.3561e+002
8.5000e-001   1.5310e+002   1.5311e+002
9.0000e-001   1.6959e+002   1.6960e+002
9.5000e-001   1.8520e+002   1.8520e+002
1.0000e+000   2.0000e+002   2.0000e+002
```