```
function [ ] = Solver( Length, Steps, x0 )
%SOLVER Summary of this function goes here
%   Detailed explanation goes here
dx = (Length-x0)/Steps;
X = x0+dx:dx:Length;

for i = 1:Steps
    M(i) = C_exact(x0+i*dx);
end

K = TriDiag(X)
F = F_vector(X)

U = (K\F)'
M

end
```

```
Error using Solver (line 4)
Not enough input arguments.
```

Published with MATLAB® 7.14

```
function [ C ] = C_exact( x )
%C_EXACT Summary of this function goes here
%   Detailed explanation goes here

if x > pi/3
    C = 0.01*sin(3*x);
else
    C = sin(3*x);
end

end
```

Published with MATLAB® 7.14

```
function [ F ] = F_vector( X )
%F_VECTOR Summary of this function goes here
%   Detailed explanation goes here
m = length(X);
F = zeros(m,1);

for i = 1:m-1
    F(i,1) = TwoPointRule(@f_x, i, i, X, X(i), X(i+1));
end

F(m,1) = TwoPointRule(@f_x, i, i, X, X(m-1), X(m));

end
```

Published with MATLAB® 7.14

```
function [ K ] = TriDiag( X )
%TRIDIAG Summary of this function goes here
%   Detailed explanation goes here
steps = length(X);
K = zeros(steps);

for i = 1:steps-1
    K(i,i+1) = TwoPointRule(@Coeff, i, i+1, X, X(i), X(i+1));
    K(i, i) = TwoPointRule(@Coeff, i, i, X, X(i), X(i+1));
    K(i+1,i) = TwoPointRule(@Coeff, i+1, i, X, X(i), X(i+1));
end

K(steps,steps) = K(steps-1, steps-1);
```

```
end
```

Published with MATLAB® 7.14

```
function [ Coeff ] = Coeff( x, X, i, j )
%COEFF Summary of this function goes here
%   Detailed explanation goes here
if x > pi/3
    E = 100;
else
    E = 1;
end

A = Phi_i_prime(x, X, i);
B = Phi_i_prime(x, X, j);
C = Phi_i(x, X, i);
D = Phi_i(x, X, j);
Coeff = A*B*E - C*D;
end
```

```
Error using Coeff (line 4)
Not enough input arguments.
```

Published with MATLAB® 7.14

```
function [ L ] = Phi_i( x, X_points, i )
%Phi_i is the lagrange polynomial evaluated at Xi

L = 1;
m = length(X_points);

for j = 1:m
    if j ~= i
        L = L*(x-X_points(j))/(X_points(i) - X_points(j));
    end
end

end
```

Published with MATLAB® 7.14

```
function [ Sum ] = Phi_i_prime( x, X_points, i )
%PHI_I_PRIME Summary of this function goes here
%   Detailed explanation goes here

L = 1;
Sum = 0;
m = length(X_points);

for k = 1:m
    if k ~= i
       for j = 1:m
           if j ~= k
               if j ~= i
                   L =L*(x-X_points(j))/(X_points(i)-X_points(j));
               end
           end
       end
       Sum = Sum + L*(-X_points(k))/(X_points(i)-X_points(k));
       L=1;
    end
end
```

Published with MATLAB® 7.14

```
function [ TwoPtVal ] = TwoPointRule( func, i, j, X, x1, x2 )
%TWOPOINTRULE calculates an integral value using a single application of
% the Two Point Gaussian Quadrature

%INPUT
% func -> Function handle for a function with one input argument
% x1 -> Initial value of x
% x2 -> Final value of x

% Initialize step size, weight, and the two evaluation points
h = (x2-x1)/2;
w = 1/sqrt(3);
xa = (h * -w) + h;
xb = (h*w) + h;

% Calculate Value of the function
TwoPtVal = h* (func(xa,X,i,j) + func(xb,X,i,j));

end
```

```
Error using TwoPointRule (line 11)
Not enough input arguments.
```

Published with MATLAB® 7.14

```
function [ F ] = f_x( x, X, i, j )
%F_X Summary of this function goes here
%   Detailed explanation goes here
if x > (pi/3)
    F = 9.01*sin(3*x);
else
    F = 10*sin(3*x);
end

F = F*Phi_i(x,X,i);

end
```

```
Error using f_x (line 4)
Not enough input arguments.
```

Published with MATLAB® 7.14