NUCLEAR ENGINEERING, UC BERKELEY

ME 280A Finite Element Analysis HOMEWORK 2: HIGHER ORDER ELEMENTS

Xin Wang

October 16, 2014

1 INTRODUCTION

The objective of this project is to solve a 1-D differential equation using linear, quadratic and cubic elements.

$$
\begin{aligned}
\frac{d}{dx}(A_1(x)\frac{du}{dx}) &= f(x) \\
f(x) &= 256sin(\frac{3\pi kx}{4})cos(16\pi x) \\
A_1(x) &= 0.2 \\
L &= 1 \\
u(0) &= 0 \\
A1(L)\frac{du}{dx}(L) &= 1
\end{aligned} \tag{1}
$$

2 ANALYTICAL SOLUTION

This equation can be solved analytically after two integrations:

$$
\begin{aligned}
u(x) &= 5\frac{4087\pi x + 1536\sqrt{2}x + \frac{137216sin(\frac{61\pi x}{4})}{61\pi} - \frac{124928sin(\frac{67\pi x}{4}))}{67\pi}}{4087\pi} \\
\frac{du}{dx} &= 5(1 + \frac{512(67cos(\frac{61\pi x}{4}) - 61cos(\frac{67\pi x}{4}))}{4087\pi} - \frac{512(67cos(\frac{61\pi}{4}) - 61cos(\frac{67\pi}{4}))}{4087\pi}
\end{aligned} \tag{2}
$$

3 FINITE ELEMENT METHOD

The first step of FEM is to derive the weak form of the differential equation:

$$\int_\Omega \frac{dv}{dx}A_1\frac{du}{dx}dx = \int_\Omega fvdx + A_1\frac{du}{dx}v\,|_{\Gamma t} \tag{3}$$

We approximate the real solution u by

$$u(x) = \sum_{j=1}^{N} a_j\phi_j(x) \tag{4}$$

and we choose the test function v with the same approximation functions

$$v(x) = \sum_{i=1}^{N} b_i \phi_i(x) \tag{5}$$

where N is the number of degree of freedom(number of nodes).

Then the equation becomes:

$$\int_{\Omega} \frac{d}{dx}(\sum_{j=1}^{N+1} a_j \phi_j(x)) A_1 \frac{d}{dx}(\sum_{i=1}^{N+1} b_i \phi_i(x)) dx = \int_{\Omega} f(\sum_{i=1}^{N+1} b_i \phi_i(x)) dx + (\sum_{i=1}^{N+1} b_i \phi_i(x) t) \mid_{\Gamma t}, \forall b_i \tag{6}$$

We can regroup the terms into:

$$\sum_{i=1}^{N+1} b_i \int_{\Omega} (\sum_{j=1}^{N+1} a_j \frac{d}{dx} \phi_j(x) A_1 \frac{d}{dx} \phi_i(x)) dx = \sum_{i=1}^{N+1} b_i \int_{\Omega} f \phi_i(x) dx + \sum_{i=1}^{N+1} b_i (\phi_i(x) t) \mid_{\Gamma t} \tag{7}$$

As the equation should be valid for any b_i, we obtain the matrix system to solve:

$$\begin{aligned} K_{ij} &= \int_{\Omega} \frac{d}{dx} \phi_j(x) A_1 \frac{d}{dx} \phi_i(x) dx \\ R_i &= \int_{\Omega} f \phi_i(x) dx + \phi_i(x) t \mid_{\Gamma t} \\ Ka &= R \end{aligned} \tag{8}$$

In this homework, piece-wise linear, quadratic or cubic basis functions are used. The numerical computation is carried over the corresponding master elements and mapped to the global elements. Shape functions are defined for elements with different polynomial orders(figure 1).

Linear shape functions are:

$$\begin{aligned} \hat{\phi}_1 &= \frac{1-\xi}{2} \\ \hat{\phi}_2 &= \frac{1+\xi}{2} \end{aligned} \tag{9}$$

and

$$\begin{aligned}
\frac{d\hat{\phi}_1}{d\xi} &= -\frac{1}{2} \\
\frac{d\hat{\phi}_2}{d\xi} &= \frac{1}{2}
\end{aligned} \tag{10}$$

Quadratic shape functions are:

$$\begin{aligned}
\hat{\phi}_1 &= \frac{\xi(\xi-1)}{2} \\
\hat{\phi}_2 &= -(\xi-1)(\xi+1) \\
\hat{\phi}_3 &= \frac{\xi(\xi+1)}{2}
\end{aligned} \tag{11}$$

and

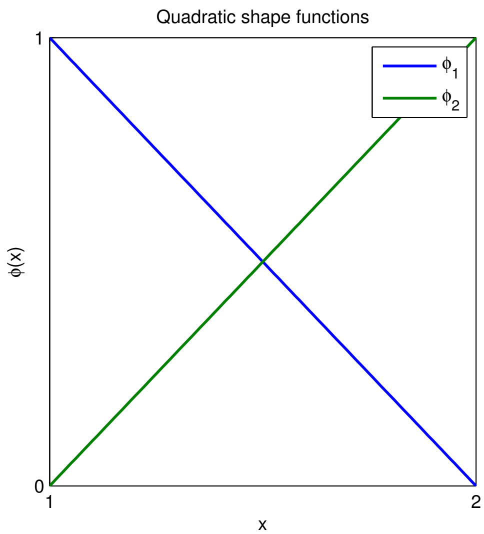
$$\begin{aligned}
\frac{d\hat{\phi}_1}{d\xi} &= \frac{2\xi-1}{2} \\
\frac{d\hat{\phi}_2}{d\xi} &= -2\xi \\
\frac{d\hat{\phi}_3}{d\xi} &= \frac{2\xi+1}{2}
\end{aligned} \tag{12}$$

The cubic shape functions are:

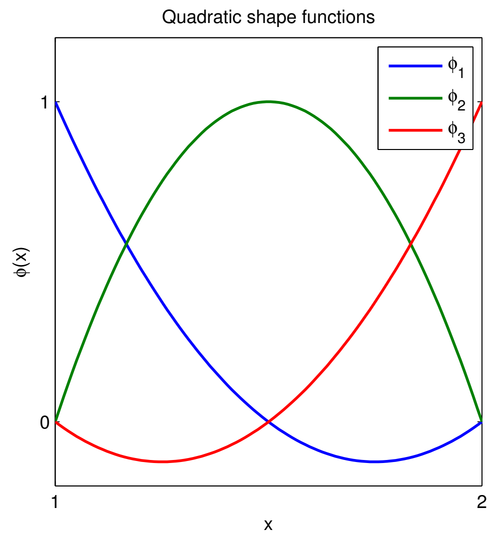
$$\begin{aligned}
\hat{\phi}_1 &= \frac{-(\xi-1)(3\xi+1)(3\xi-1)}{16} \\
\hat{\phi}_2 &= \frac{9(\xi-1)(\xi+1)(3\xi-1)}{16} \\
\hat{\phi}_3 &= \frac{-9(\xi-1)(\xi+1)(3\xi+1)}{16} \\
\hat{\phi}_4 &= \frac{(\xi+1)(3\xi+1)(3\xi-1)}{16}
\end{aligned} \tag{13}$$

and

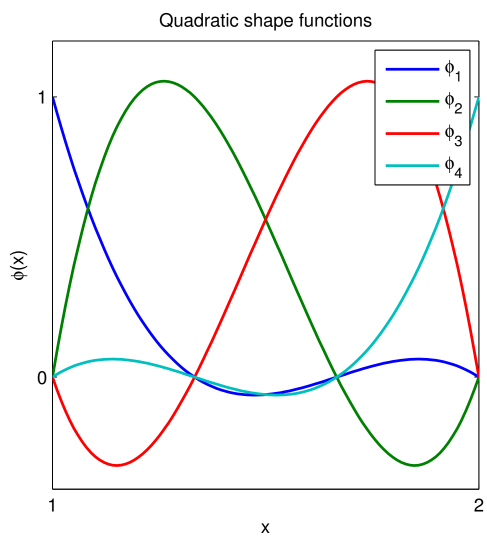
$$\begin{aligned}
\frac{d\hat{\phi}_1}{d\xi} &= -\frac{1}{16}(27\xi^2-18\xi-1) \\
\frac{d\hat{\phi}_2}{d\xi} &= \frac{9}{16}(9\xi^2-2\xi-3) \\
\frac{d\hat{\phi}_3}{d\xi} &= -\frac{9}{16}(9\xi^2+2\xi-3) \\
\frac{d\hat{\phi}_4}{d\xi} &= \frac{1}{16}(27\xi^2+18\xi-1)
\end{aligned} \tag{14}$$



(a) linear shape function



(b) quadratic shape function



(c) cubic shape function

Figure 1: nodel shape functions on an element [1, 2]

Corresponding global coordinate x can be calculate x from ξ:

$$x = \sum \chi_i \hat{\phi}_i \tag{15}$$

where χ_i are the coordinates of the nodes in the global element.

3.1 Apply the boundary conditions

After computing the matrices K and R element by element and assembling them, one needs to modify the K and R matrices in order to incorporate the boundary conditions. In this homework, we deal with two types of boundary conditions:

3.1.1 Dirichlet boundary condition

Dirichlet boundary condition gives the value of u on an end point(x=0 or x=L), i.e the value of a(0 or N). We force the test function v(0) =0. The implementation of Dirichlet boundary condition in 1D is to modify the first or the last line of K and R for boundary condition at the left or right end of the domain.

For Dirichlet BC on u(0):

K(1,1) =1 and K(1, 2:N) =0

R(1) = u(0)

For Dirichlet BC on u(L):

K(N,N) =1 and K(N, 1:N-1) =0

R(N) = u(L)

a = K/R results in for the line corresponds to Dirichlet boundary condition:

a(1) = R(1)/K[1,1] = R(1)

or

a(N) = R(N)/K[N,N] = R(N)

3.1.2 Neumann boundary condition

Neumann boundary condition gives the value of $A1(x)\frac{du}{dx}$ on an end point.

For Neumann BC on x=0:

$R(1) = R(1) + A_1\frac{du}{dx}[x = 0]$

For Neumann BC on x=L:

R(N) = u(L) $R(N) = R(N) + A_1\frac{du}{dx}[x = L]$

4 Gaussian Quadrature Integration

One can integrate a polynomial of order 2N-1 exactly using N Gaussian points and appropriate Gaussian weights. So the integration of a function f over the domain [-1 ,1] can be computed numerically as:

$$\int_{-1}^{1} f(\xi)d\xi = \sum_{i=1}^{N} w_i f(\xi_i) \tag{16}$$

5 Error Calculations

The error is defined as

$$\begin{aligned}
e^N &= \frac{\| u - u^N \|_{A_1(\Omega)}}{\| u \|_{A_1(\Omega)}} \\
\| u \|_{A_1(\Omega)} = \sqrt{\int_\Omega \frac{du}{dx} A_1 \frac{du}{dx} dx} &= \sqrt{\int_\Omega A_1 (\frac{du}{dx})^2 dx} \\
\| u - u^N \|_{A_1(\Omega)} &= \sqrt{\int_\Omega A_1 (\frac{d(u-u^N)}{dx})^2 dx}
\end{aligned} \tag{17}$$

To compute the error numerically, we calculate the two following quantities element by element, and then assemble them to obtain the overall error:

$$\begin{aligned}
\| u \|^2_{A_1(\Omega)} &= \int_\Omega A_1 (\frac{du}{dx})^2 dx \\
\| u - u^N \|^2_{A_1(\Omega)} &= \int_\Omega A_1 (\frac{d(u-u^N)}{dx})^2 dx \\
&= A_1 \int_\Omega (\frac{du}{dx} - \frac{du_N}{dx})^2 dx \\
&= A_1 \sum_{e=1}^{N_e} \int_{x_e}^{x_{e+1}} (\frac{du}{dx} - \frac{du_N}{dx})^2 dx \\
&= A_1 \sum_{e=1}^{N_e} \int_{-1}^{1} (\frac{du}{d\xi} - \frac{du_N}{d\xi})^2 J d\xi
\end{aligned} \tag{18}$$

6 Postprocessing

Following the same procedures as in homework 1 would give the matrix A for the system. However, the postprocessing is more complicated for higher order element.

In order to plot the result, we calculate the value of the numerical solution for 10 points in each elements. For any given ξ, the corresponding value for the solution is straightforward:

$$\begin{aligned} u(x) &= u(x(\xi)) = \sum_{i=1}^{P+1} a_i^e \hat{\phi}_i \\ x(\xi) &= \sum x^e \hat{\phi}(\xi) \end{aligned} \tag{19}$$

For each element, we can also calculate the derivative of u(x):

$$\frac{du}{dx} = \frac{d}{dx} \sum_{i=1}^{P+1} a_i \phi_i = \left(\frac{d}{d\xi} \sum_{i=1}^{P+1} a_i \hat{\phi}_i\right) \frac{d\xi}{dx} \tag{20}$$

7 Results

Minimum number of elements and number of nodes to achieve to error criteria for different order of elements are tabulated bellow:

P	Ne_{opt}	$N_{opt} = Ne_{Opt} * P + 1$
1	1348	1349
2	94	189
3	34	103

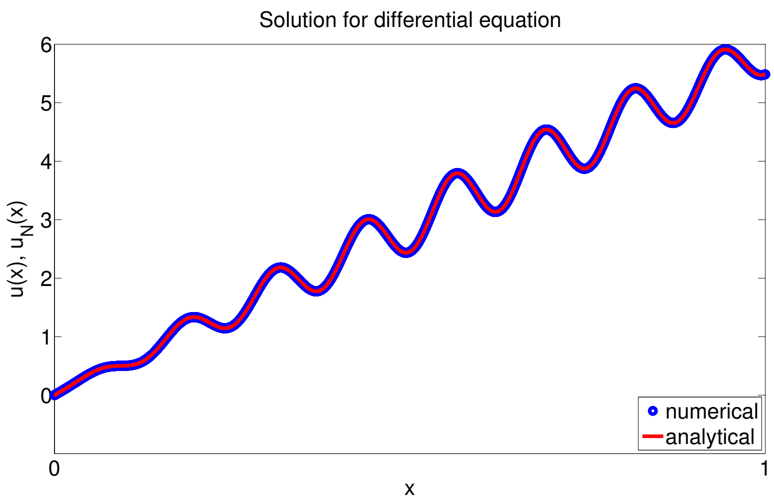
The numerical solution and the analytical solution are compared in figure 2.

7.1 Relationship between the error and the element size

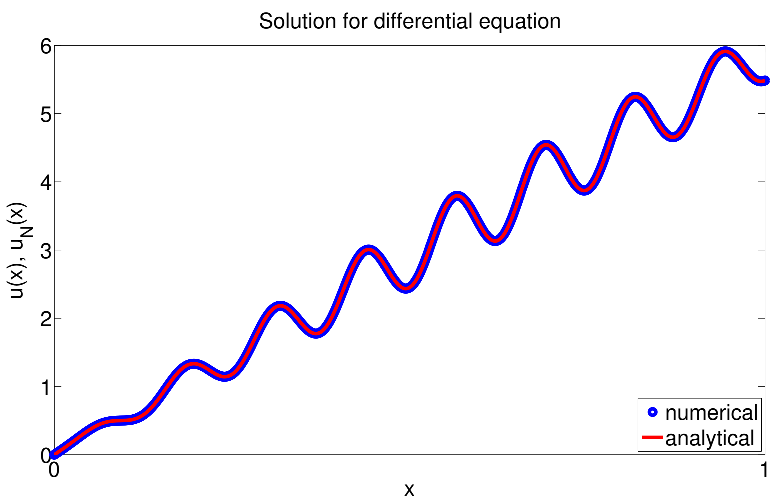
If the element size is too big, the numerical solution is far from the real solution, it's not interesting to study the relationship between error and element size in this region. So we study the correlation in the region around the

8 Conclusion

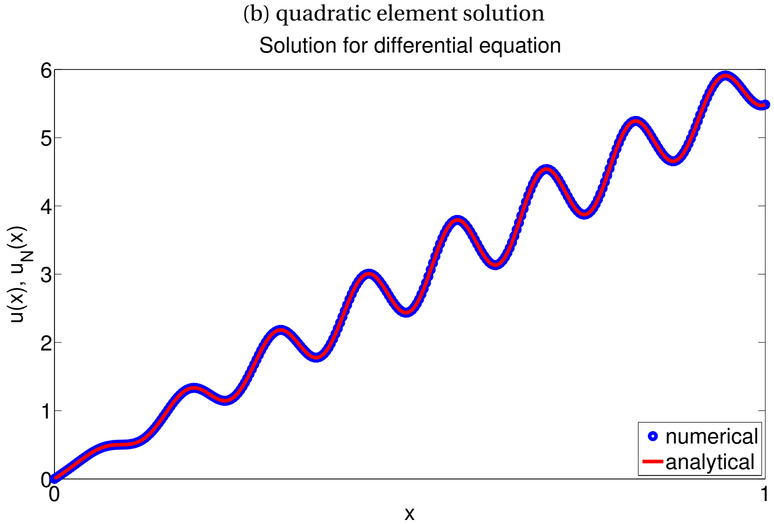
This homework extends the previous one to higher order of element. As the error of ... . Having higher order element shape functions can capture the fluctuations



(a) linear element solution

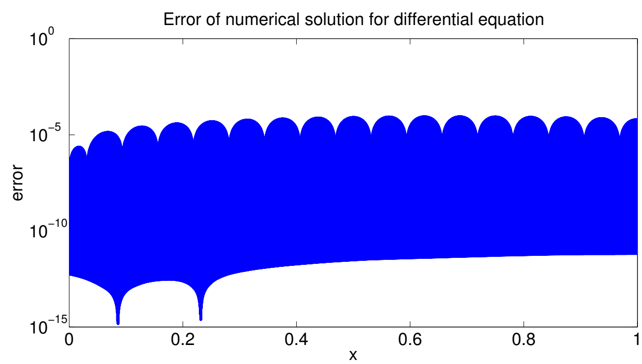


(b) quadratic element solution

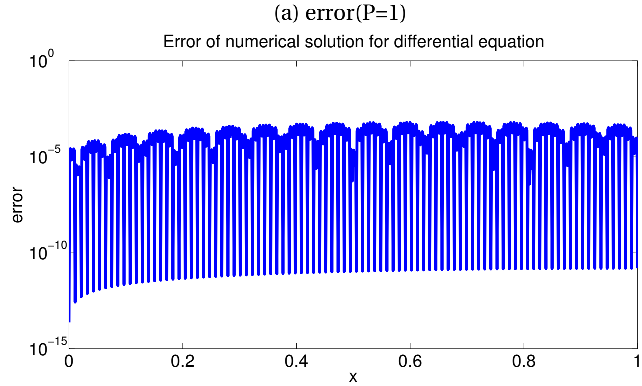


(c) cubic element solution

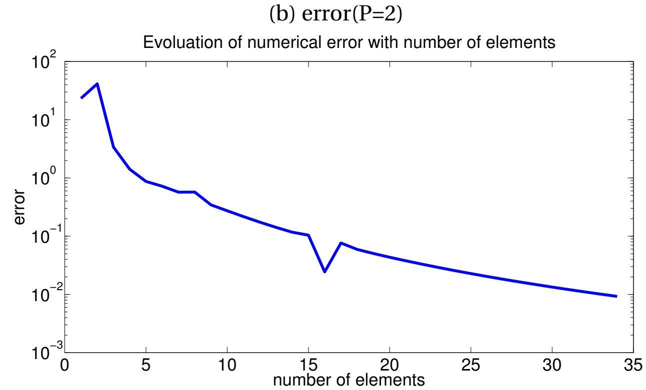
Figure 2: Comparison of analytical solution and numerical solution



(a) error(P=1)



(b) error(P=2)



(c) error(P=3)

Figure 3: Evoluation of numerical error with number of elements