Exploring the Finite Element Method to Numerically Solve Boundary Value Problems

Kelly Smith

May 10, 2007

Program in Applied Mathematics,
University of Arizona, Tucson, AZ

Abstract

The concepts of how the finite element method (FEM) is formed and utilized to numerical find a solution to boundary value problems (BVP) are explored. The results from a program that implements this method for a one-variable BVP are provided and analyzed. Following formal definitions, an example shows how this method can solve a two-variable BVP as well.

1 Introduction

Finite Element Analysis is a recently developed area in numerical analysis to solve differential and partial differential equations. The first ideas that contributed to this method arose in 1941 by Alexander Hrennikoff, while working at MIT on lattice analogies of structural membranes and modeling plate bending. Similarly, Richard Courant began his work, independently of Hrennikoff, in 1942 on torsion problems. Both engineers pictured their continuum problem as a divided space of piecewise approximations. Then in the 1950's the improvement of the computer capabilities and availability to government research centers allowed advancements in numerical analysis. In 1953-54, N.J. Turner in the Boeing Dynamics unit and John H. Argyris at the University of Stuttgart in Germany published articles on numerical models that used this unnamed method of finite elements. However, this method finally became recognized and re-invented in the 1956 paper of "Stiffness and Deflection of Complex Structures" from the Boeing group [1].

Throughout the years the Finite Element method has been in the fore front of aeronautical and structural engineering. NASA Structural Analysis Program, NASA Langley Research Center, and Wright-Patterson Airforce Base are noted for contributions. However, this method finally became mainstream when academia gained insight into this area in the 1970's. Thousands of articles and numerous books were published from academia. Thus, countless areas of science utilize this method since partial differential equations can explain many properties of materials, fluid flows, and dynamics in the real world [1].

So when a scientist must solve a boundary value problem (BVP), the Finite Element Method is a numerical tool to help build the solution to the original equation. Essentially, the domain of the problem to split into many pieces, or rather elements, on which the ODE or PDE is evaluated. Then, by taking a linear combination of all the solutions on all the elements, one can approximate the BVP on the entire domain.

The order of the paper is as follows: Section 2 describes the formulation and process for numerically solving a one variable boundary value problem with the FEM, Section 3 provides formal definitions and examples for clarification, Section 4 describes how the FEM works to numerically solve two variable BVPs.

2 Numerical Solving One Variable Boundary Value Problems

Beginning with one variable BVPs, connections are made between weak-formulations of boundary-value problems, Hilbert spaces, and then the formulation of symmetric variational problems. A Matlab program is given in order to demonstrate the simulations.

2.1 Weak Formulation of Boundary value Problems

Consider the following one dimensional problem [2]:

$$u'' = -f \tag{BVP}$$
$$u(0) = u'(1) = 0 \tag{1}$$

Thus, if u is the solution and v is a function such that $v(0)$=0, multiply both sides of the differential equation by v and integrate on $(0, 1)$. The weak formulation is defined.

$$\begin{aligned}\int_0^1 f(x)v(x)dx &= \int_0^1 -u''(x)v(x)dx \\ &= \int_0^1 u'(x)v'(x)dx \\ &= a(u, v)\end{aligned}$$

This is done by integrating the left side by parts in the first line. Also, u'(1)=v(0)=0, so the boundary terms are zero to get the second line. Let $V = \{v \in L^2(0, 1) : a(v, v) < \infty$ and $v(0) = 0\}$, then the weak formulation of BVP, where u is the solution, is defined as:

$$u \in V \text{ and u satisfies } a(u, v) = (f, v)\ \forall v \in V. \tag{2}$$

Theorem 2.1 *Suppose $f \in C^0([0, 1])$ and $u \in C^2([0, 1])$ satisfies the weak formulation 2, Then u solves the boundary value problem $u'' = -f$.*

Proof Let $f \in C^0([0, 1])$ and $u \in C^2([0, 1])$ satisfies the weak formulation 2.

$$\begin{aligned}(f, v) &= a(u, v) \\ &= \int_0^1 -u''(x)v(x)dx + u'(1)v(1)\end{aligned}$$

The second line is from integrating $a(u, v)$ by parts. Since this is true for every $v \in V$, pick $v(x) = x$. Thus in order to get $\int_0^1 f(x)v(x)dx = \int_0^1 -u''(x)v(x)$, or rather for $-u'' = f$, $u'(1)$ needs to be zero. Thus, u solves $-u'' = f$ and u also satisfies the boundary values of $u(0) = u'(1) = 0$.

2.2 Symmetric Variational Problem

Since the Weak formulation of the BVP resembles an inner product, it makes sense to express the BVP with respect to a Hilbert space H setting. However,

$a(u, v)$ is not the same as a inner product since a constant function, say v, $a(v, v) = a(0, 0) = 0$ but $v \neq 0$. So in order the preserve the coercivity property of $a(v, v)$, meaning $a(v, v) = 0$ if and only if $v = 0$, the BVP must be expressed on a subspace of Hilbert space H. It is known that the $V = \{v \in L^2(0, 1) : a(v, v) < \infty \text{ and } v(0) = 0\}$, (also used in section 2.1) is a subspace of a Hilbert space.

Proposition 2.2 *If $a(.,.)$ is bounded (or rather continuous), coercive, symmetric on $V \subset H$, then $(V, a(.,.))$ is a Hilbert Space.*

So assuming the following three properties:

1). $(H, a(.,.))$ is a Hilbert space 2). V is closed subspace of H 3). $a(.,.)$ is a bounded, symmetric, bilinear form that is coercive on V , the Symmetric Variational Problem is:

Given $F \in V'$,the dual space of V, find $u \in V$ such that $a(u, v) = F(v) \forall v \in V$.

Theorem 2.3 *Suppose conditions 1-3 hold, then there exists a unique $u \in V$ solving the symmetric variational problem [2].*

Proof We know $a(.,.)$ is an inner product on $V \subset H$. Also, we know $(V, a(.,.))$ is a Hilbert Space. The Riesz Representation Theorem states that every linear functional $L(v)$ on Hilbert Space H can be unique represented as an inner product of (u, v) for some $u \in H$. Thus $\exists!\ u \in V$ such that $a(u, v) = F(v)$. ▮

It is also noted that the symmetric variational problem of BVP $a(u, v) = \int_0^1 f(x)v(x)dx$ can be obtained by analysis on the equation of $G(t) = a(u, u) - 2F(u)$, where $u = u_0 + tv$.

$$\begin{aligned}
G(t) &= a(u_0 + tv, u_0 + tv) - 2F(u_0 + tv) \\
&= a(u_0, u_0) + 2a(u_0, tv) + a(tv, tv) - 2F(u_o) - 2f(tv) \\
&= a(u_0, u_0) + 2ta(u_0, v) + t^2 a(v, v) - 2F(u_o) - 2tf(v)
\end{aligned}$$

Note that the above holds since $a(.,.)$ is symmetric. When the derivative of G is taken with respect to t, one sees that:

$$G'(t) = 2a(u_0, v) + 2ta(v, v) - 2F(v)$$

and if $G'(t) = 0$ is set to zero,

$$\begin{aligned}
0 &= 2a(u_0, v) + 2ta(v, v) - 2F(v) \\
0 &= 2a(u_0 + tv, v) - 2F(v)
\end{aligned}$$

To get the symmetric variational problem, t must be zero.

$$\begin{aligned} 0 &= a(u_0, v) - F(v) \\ a(u_0, v) &= F(v) \end{aligned}$$

So G reaches a minimum at $t = 0$, and thus $a(u_0, v) = F(v)$ is obtained. When u_0 satisfies this equation, once again, u_0 is a solution to the BVP.

In the above subsections, the boundary conditions of u(0)=u'(1)=0 are used. However, in the computer problems to follow, the boundary conditions of u(0)=u(1)=0 are used. The same analysis is used for these boundary conditions, except that the space of functions used are $V = \{v \in L^2(0,1) : a(v,v) < \infty$, $v(0) = 0$, $v(1) = 0\}$, the difference being that the functions now must be defined to be v(1)=0 as well.

2.3 FEM method

In order to do the process of finite element method, piecewise polynomial space must be defined. Let x_n be the points in $[0,1]$, not necessarily uniform distance from each other. Let $S = \big(v | v \in C^0([0,1])$, v is a linear polynomial on $[x_{n-1}, x_n]$ $\forall n$, and $v(0) = 0\big)$ [2]. Let

$$\phi^0 = \begin{cases} 1-x & x \in [0,1] \\ 0 & otherwise \end{cases} \qquad \phi^1 = \begin{cases} x & x \in [0,1] \\ 0 & otherwise \end{cases} \tag{3}$$

Let $I = [x_{n-1}, x_n]$, and define $\phi_n^i = \phi^i\big(\frac{x - x_{n-1}}{x_n - x_{n-1}}\big)$, see Figure 1.

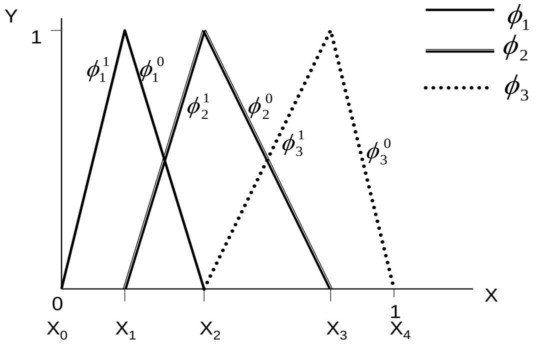


Figure 1: With $\{x_1, \ldots, x_5\}$ in interval $[0,1]$, ϕ_1, ϕ_2, and ϕ_3 would be the basis for all piecewise linear functions with respect to the given $\{x_n\}$ on $[0,1]$. So as n increases, $v = a_1\phi_1 + ...a_{n-2}\phi_{n-2}$ can converge to a nonlinear function. Note that each ϕ_i is $\phi_i(x_j) = \delta_{ij}$.

Theorem 2.4 *A basis element in S is defined as* $\phi_n = \phi_n^0 + \phi_n^1$.

Proof Show ϕ_n is linear independent. If $\sum_{k=1}^n c_k\phi_k(x_n) = 0$, we have constructed $\phi_k(x_n) = \delta_{nk}$, so the sum is equal to $c_j\phi_j(x_j) = 0$. But $\phi_j(x_j) = 1$, thus $c_j = 0$. This is true $\forall x_j$. So $\{\phi_n\}$ is linear independent. Now show that $\{\phi_n\}$ spans S. $\forall v \in S$, v is linear on the subintervals. Construct the interpolant of v, defined as $v_I = \sum_{k=1}^n v(x_k)\phi_k$. Note that the interpolant is a linear combination of $\{\phi_n\}$. $v - v_I$ is linear on the subintervals $[x_{n-1}, x_n]\forall n$ and $v(x_j) - v_I(x_j) = 0 \forall x_n$. Thus $v - v_I = 0$ on $[0,1]$. So $v = v_I$. Thus any $v \in S$ can be written as the linear combination of $\{\phi_n\}$. So $\{\phi_n\}$ spans S. ▮

Thus, $\{\phi_n\}$ are triangle like functions in $[x_{n-1}, x_n]$. $\{\phi_n\}$ form interpolants that are approximations to solutions for BVP.

Proposition 2.5 *if* $u \in V$, *then*$||u - u_I||_E < h||u''||_2$, *where h is the size of the interval [2].*

Proof Let error between u and u_I be $e = u - u_I$, $e' = u' + u_I'$, and $e'' = u''$ since u_I is linear on the subintervals. We know e is zero at the interpolation points, in particular at 0 and 1. so $\exists\ \alpha$ such that $e'(\alpha) = 0$. So $|e'(y)| = \int_\alpha^y e''(x)dx$ where $\alpha \in [0,1]$. By Cauchy-Swartz inequality,

$$|e'(y)| \leq \left(\int_\alpha^y dx\right)^{1/2}\left(\int_\alpha^y (e''(x))^2 dx\right)^{1/2} \leq |y - \alpha|^{1/2}\left(\int_0^1 (e''(x))^2 dx)\right)^{1/2}$$

Finish by squaring and integrating with respect to y. So

$$\int_0^1 (e'(y))^2 dy \leq 1 \int_0^1 (e''(x))^2 dx$$

and since $e = u - u_I$ and $e'' = u''$, we get $||u - u_I||_E < h||u''||_2$. ▮

Proposition 2.5 is important to note because it is saying that the error between the numerical solution and exact solution depends upon the mesh size h. The numerical simulations in the following subsection demonstrate that when a mesh size in decreased, the error in the numerical solution is decreased as well.

2.4 Computer Program

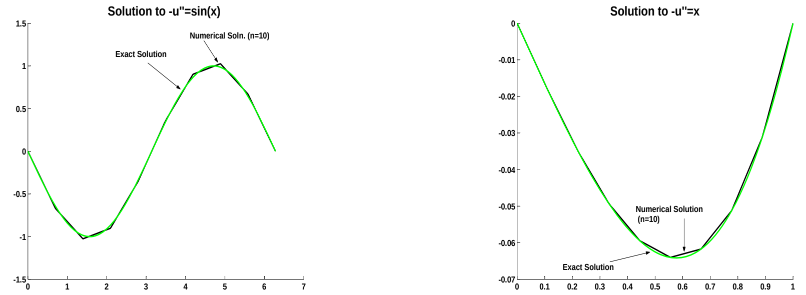
So we have the one variable BVP of $-u'' = f$, u(0)=u(1)=0, where

$$u = a_1\phi_1 + a_2\phi_2 + \cdots + a_{n-2}\phi_{n-2} \text{ and } u' = a_1\phi_1' + a_2\phi_2' + \cdots + a_{n-2}\phi_{n-2}'.$$

Here ϕ_n is a basis element as in Theorem 2.4 (also see Figure 1 as a visual aid). If there are n points in the discretation of the given interval, then there will be $n-2$ number of basis elements ϕ_i in order to compensate for the boundaries which are zero at each end of the given interval.

So to solve $u'' = -f$, multiply both sides ϕ_i and integrate the left hand side by parts to get $(u, \phi_i) = (f, \phi_i)$. This is done $\forall \phi_i i = 1, \ldots, n$. A system of linear equations is formed to solve for $\{a_n\}$. Once the $\{a_n\}$ is acclaimed, the solution u can be formed.

With the computer program I wrote (see Appendix for code), two example problems are presented. The first is $u'' = -\sin(x)$ with boundary conditions of $u(0) = u(2\pi) = 0$. The exact solution is $u = -\sin(x)$. $[0, 2\pi]$ was split into 10 uniform length intervals 2(a). However, as seen in figure 1, $\{x_n\}$ do not have to be equally spaced. The second problem is $u'' = -x$,with boundary condition u(0)=u(1)=0. The exact solution is $u = \frac{1}{6}x^3 - \frac{1}{6}x$ 2(b). Figure 2a shows the results of both problems. The plots for $n \geq 20$ are not shown because one can not distinguish the numerical solution from the exact solution.



(a) Numerical Soln ($n = 10$) $u'' = -sin(x)$

(b) Numerical Soln. ($n = 10$) to $-u'' = x$

Figure 2: The exact solution to $-u'' = sin(x)$ is $u = -sin(x)$. The exact solution to $-u'' = x$ is $u = \frac{1}{6}x^3 - \frac{1}{6}x$. Each respective interval, either $[0, 2\pi]$ or $[0, 1]$ is divided by 10 discretation points. ($n = 10$), the numerical solution is close to the real solution.

To understand the convergence rate of FEM on these two tested BVPs, the total error is plotted versus the number of discretizing points in the given interval. Error is defined as the L^2 error. Let F be the exact solution and N be the numerical solution for n discretation points.

$$L^2 Error = \left[\frac{1}{n}\sum_{k=1}^{s}(F(y_k) - N(y_k))^2\right]^{\frac{1}{2}}$$

where $\{y_k\}$ are 500 equally spaced points in the interval on which the BVP is being evaluated on.

The case of $-u'' = x$ is in Figure 3. The L^2 error for $-u'' = x$ can be represented as a power series of $Error = 0.0269n^{-1.638}$, with a R^2 error of 0.9609. The errors for the numerical solution to $-u'' = \sin(x)$ are similar, and can be represented as $Error = 0.5331n^{-1.6486}$, $R^2 = 0.9623$.

Pointwise Error is also found for each problem, see Figure 4. With F as the exact solution and N as the numerical solution, pointwise error is defined at each y_k as:

$$|F(y_k) - N(y_k)|$$

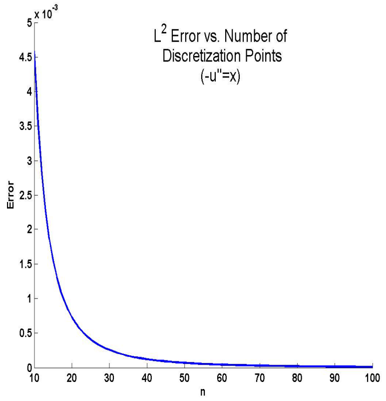
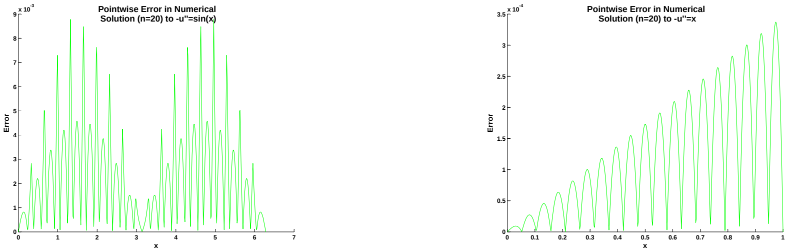


Figure 3: This plot shows the L^2 Error of the numerical solution to the exact solution of $-u'' = x$ as the number of discretizing points increase.



(a) Pointwise Error in Numerical Soln. ($n = 20$) to $-u'' = sin(x)$

(b) Pointwise Error in Numerical Soln. ($n = 20$) to $-u'' = x$

Figure 4: Pointwise Error between exact solution F and Numerical Solution N (using n=20 discretation points). Pointwise error is calculated at each $\{y_k\}$, $|F(y_k) - N(y_k)|$, where $\{y_k\}$ is a set of 500 uniform points in the respective interval on which the BVP is being evaluated on.

3 Formal Construction and Definition of Finite Elements

So the method introduced in the previous section to solve a one dimensional ordinary differential equation is called the finite element method. This method can be extend from the one-dimensional case into a general case of finite element spaces. Before exploring higher dimensional cases, a few definitions have to be introduced [2].

Definition Let

- $K \subseteq \Re^n$ be a bounded closed set with nonempty interior and piecewise smooth boundary,

- P be a finite-dimensional space of functions on K,

- $N = \{N_1, N_2, \ldots, N_k\}$ be a basis for P'.

Then (K, P, N) is called a finite element [2].

Be better understanding, K is the element domain, P is the space of shape functions, and N is the set of nodal variables.

Definition Let (K, P, N) be a finite element. The basis $\{\phi_1, \phi_2, \ldots, \phi_k\}$ of P (dual to N since $N_i(\phi_j) = \delta_{ij}$) is called the nodal basis of P.

So to relate this to the one-dimensional case, K would be the interval $[0, 1] \subseteq \Re$, P is the set of piecewise linear functions, and $N = \{N_1, N_2\}$, where $N_1(v) = v(0)$ and $N_2(v) = v(1) \forall v \in P$. The nodal basis was $\phi_0 = 1 - x$ and $\phi_1 = x$ (refer to equation 3). In the K would be the interval $[a, b] \subseteq \Re$, P_k is the set of all polynomials of degree less than or equal to k, and $N_k = \{N_1, N_2, \ldots, N_k\}$, where $N_i(v) = v(a + (b - a)i/k) \forall v \in P_k$ and $i = 0, 1, \ldots, k$. Then (K,P_k,N_k) is a finite element and resides on the following Lemma.

Definition N, the set of nodal variables, determines P, the set of shape functions, if $\psi \in P$ with $N_i(\psi) = 0 \forall i = 1, 2, \ldots, k$, then $\psi = 0$.

Lemma 3.1 *(Lemma 3.1.4): Let P be a d-dimensional vector space and let $\{N_1, N_2, \ldots, N_d\}$ be a subset of dual space P'. Then the following two statements are equivalent.*
(a) $\{N_1, N_2, \ldots, N_d\}$ is a basis for P'
(b) given $v \in P$ with $N_i(v) = 0 \forall i = 1, 2, \ldots, d$, then $v \equiv 0$, or rather N determines P.

Proof Let $\{\phi_1, \phi_2, \ldots, \phi_d\}$ be some basis for P and by assumption of (a) let $\{N_1, N_2, \ldots, N_d\}$ be a basis for P'. Then for any $L \in P'$

$$L = \alpha_1 N_1 + \alpha_2 N_2 + \cdots + \alpha_d N_d$$

which is equivalent to

$$y_i := L(\phi_i) = \alpha_1 N_1(\phi_i) + \alpha_2 N_2(\phi_i) + \cdots + \alpha_d N_d(\phi_i),\ i = 1, 2, \ldots, d$$

This can be put into matrix form. Let $B = (N_j(\phi_i))i, j = 1, 2, \ldots, d$ and the system of equations become $B\alpha = y$. B is solvable, thus invertible, since $N_j(\phi_i) = \delta_{ij}$, making B the identity matrix and each column in linear independent. So statement (a) is equivalent to saying B is invertible. However, lets now assume we are given statement (b) and $v \in P$ can be written as $v = \beta_1 phi_1 + \cdots + \beta_d \phi_d$. Then $N_i(v) = \beta_1 N_i(phi_1) + \cdots + \beta_d N_i(\phi_d) = 0 \forall i = 1, \ldots, d$. So if we look at $i = 1$:

$$\begin{aligned}
&N_1(v) = \beta_1 N_1(phi_1) + \beta_2 N_2(\phi_2) + \cdots + \beta_d N_1(\phi_d) = 0\\
&N_1(v) = \beta_1 N_1(phi_1) = 0 \text{ since } N_i(\phi_j) = \delta_{ij}\\
&N_1(v) = \beta_1 = 0\\
&\text{So } \beta_1 = 0
\end{aligned}$$

Then each β must be zero. So if $C = N_i(\phi_j) = \delta_{ij}$, $C\beta = 0$ only has solution of $\{\beta\} = 0$. So C is invertible. But $C = B^T$. So if you can form C, you can form B, and visa versa. Thus statements (a) and (b) are equivalent. ∎

So if we can prove that N determines P, where P is a space of shape functions on K, a bounded and closed set with nonempty interior and piecewise smooth boundary,then N will be a basis for P' and thus,(K, P, N) will be a finite element.

4 Numerical Solving Two Variable Boundary Value Problems

So in the two variable BVP, I chose to define a Lagrange Element as the discretation of the domain. The Lagrange Element is described in three functionals that disappear along the respective leg of a triangle. Each functional is dependent on two variables, say x and y. L_i is the functional that describes side i of the triangle and point z_j is a point on (or in the center) of the triangle, such as a vertex or midpoint along L_i. We define $N_j(L_i) = L_i(z_j) = \delta_{ij}$. See figure 5.

Thus, the basis elements of $\Re^2$ need to be defined. Let's find coefficients of $L_1 := ax + by + c$, using the right triangle connecting points (0,0), (1,0), and (0,1).

$$\begin{aligned}
N_1(L_1) = L_1(z_1) = L_1(0, 0) = 1\\
N_2(L_1) = L_1(z_2) = L_1(1, 0) = 0\\
N_3(L_1) = L_1(z_3) = L_1(0, 1) = 0
\end{aligned}$$

After evaluating L_1, the coefficients are $a = -1$, $b = -1$ and $c = 1$. Doing the same for L_2 and L_3 yields the $L_2 := x$ and $L_3 := y$.

Next, one needs to show that $N = N_1, N_2, N_3$ does determine P_1, which is the set of all polynomials in two variables of degree ≤ 1. The following lemma is needed.

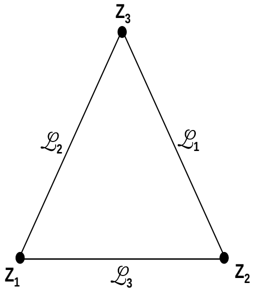


Figure 5:

Lemma 4.1 *Let V be a polynomial of degree $d \geq 1$ that vanishes on a hyperplane L. Then we can write $V = LQ$, where Q is a polynomial of degree $d - 1$.*

So suppose that $V \in P$ is a linear combination of L_1, L_2, and L_3. $V|_{L_1}$ is a linear function a one variable and vanishes at z_2 and z_3, since L_1 vanishes at z_2 and z_3. Thus $V = 0$ on L_1 since it is linear and vanishes at two points. By Lemma 4.1, V can be written as $V = cL_1$, where c=constant. Since $V = 0$, $N_1(V) = V(z_1) = cL(z_1) = 0$. We know $L(z_1) = 1$, thus $c = 0$. One can repeat same process for $V|_{L_2}$and $V|_{L_3}$. Thus $V = cL_i = 0$ for $i = 1, 2, 3$. By lemma 3.1 and for a given K (a bounded and closed set with nonempty interior and piecewise smooth boundary), (K, P, N) is be a finite element, which implies the Lagrange Element can be used in the discretation of the domain.

4.1 Method

The two variable BVP is given by

$$\Delta u = -f \tag{4}$$

$$\tag{5}$$

where u=0 on the boundary and the derivative in the direction of the normal to the boundary is zero.

The first step to solving the BVP is doing a triangulation of the domain, see Figure 6.

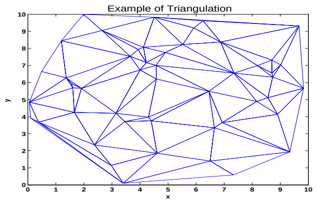
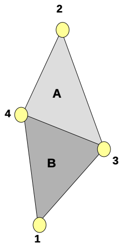


Figure 6: Example of a triangulation of the domain of a boundary value problem.

This can be easily done with the $Matlab^{®}$ program function called $Delaunay.m$. However, there must be triangle vertices, or rather called nodes, on the boundary of the given domain in order to account for the boundary values assigned in the problem. The second step would be to evaluate L_1, L_2, and L_3 on each triangle in the domain. The interpolant of u would still be $u = a_1\phi_1 + \cdots + a_n\phi_n$, where it is possible for each ϕ_i to be a sum of L functionals.

As an example, consider the following scenario in Figure 7 of two triangles as a triangulation, where the domain is the area of the triangles. Note that this would not solve a two variable BVP since there are no nodes on the inside of the domain. However, one can still write the equations to demonstrate how the interpolant u can be formed, see Figure 7(b).



(a) Example of two triangles, A and B, with nodes 1,2,3,and 4 in a domain.

$$u=\mu_1(L_{1B})+\mu_2(L_{2A})+\mu_3(L_{3A}+L_{3B})+\mu_4(L_{4A}+L_{4B})$$

$$\Delta u=\underset{\overset{\uparrow}{\phi_1}}{\mu_1(\Delta L_{1B})}+\underset{\overset{\uparrow}{\phi_2}}{\mu_2(\Delta L_{2A})}+\underset{\overset{\uparrow}{\phi_3}}{\mu_3(\Delta L_{3A}+\Delta L_{3B})}+\underset{\overset{\uparrow}{\phi_4}}{\mu_4(\Delta L_{4A}+\Delta L_{4B})}$$

(b) Equations that form the interpolant of u.

Figure 7: A simple example of how the interpolant of u can be written.

Similarly to the one variable BVP case, we can define a weak formulation function $a(u, \phi_i)$.

$$\begin{aligned} (f, \phi_i) &= \int_\Omega (-\Delta u)\phi_i d\Omega \\ &= \int_\Omega \nabla u \nabla \phi_i d\Omega \\ &= a(u, \phi_i) \end{aligned}$$

Then replace $\nabla u = \mu_1 \nabla \phi_1 + \cdots + \mu_1 \nabla \phi_1$. If this is done $\forall\{\phi_i\}$, a system of equations is formed and one can solve for $\{\mu_n\}$

5 Conclusions

Clearly, this method to solve boundary value problems (BVP) is versatile and can be extended to BVP of higher number of variables. The shape functions are made relative to the dimension of the domain. We saw that in the one variable case, the shape functions are essentially lines in the xy-plane. In the two dimensional case, the shape functions are triangles. This allowed the domain to be divided in the x and y variable. Then $\{\phi_i\}$ were formed from linear combinations of the shape functions. one can multiple both sides of the BVP by each ϕ_i separately, and then integrate over the domain. Since this is done for each ϕ_i, a system of equations is formed. The solution to the system yields the coefficients to form the interpolant of u.

Future work for this project involves producing a code that can solve a two-variable BVP. Improvement in the computational efficiency in the one-variable BVP solver can be considered as well.

6 Acknowledgments

I want to thank Shankar Venkataramani for being my advisor and guiding me through this first year graduate term paper.

References

[1] ASME International. *Iteractive Timeline.* (http://www.asme.org/Communities/History/Resources/Interactive_Timeline.cfm) (viewed April 27, 2007)

[2] Brenner S., Scott L. *The Mathematical Theory of Finite Element Methods, 2^{nd} Edition*, Springer-Verlag New York, Inc., 2002.

7 Appendix

The following is the code to numerical solve a one-variable boundary value problem.

```
clear all\\
 hold on \\
 n=10;  %NUMBER OF DISCRETATION POINTS


y=linspace(0,1,n);

%y=[0,3,6,8,11,13,15,16,19,19.5]
x=eye(n,1);

  for i=1:n
      x(i)=2*pi*y(i)      %use if eq. is -u''=sinx(x)
      %x(i)=y(i);         %use if eq. is -u''=x
end
%x=y
%---------------------------------------MAKE SYSTEM OF EQUATIONS KA=F---------

%----------------------------MAKING K MATRIX-------------------------------------------
diagonalK=zeros(n-2,1); Topdiagonal=zeros(n-3,1);
Botdiagonal=zeros(n-3,1);

k=zeros(n-2,n-2);


for i=1:n-2
 %this is slope of the phi's, slope=rise/run = 1/(x(i+1)-x(i))
diagonalK(i)= 1/(x(i+1)-x(i)) + 1/(x(i+2)-x(i+1)) ;
%remember x(1) = 0 which is really x_0  so if you want x_3-x_4 =>> x(4)-x(5)
end

for i=1:n-3 Topdiagonal(i)= 1/(x(i+1)- x(i+2)) ; end

for i=1:n-2 k(i,i)=diagonalK(i); end for i=1:n-3
  k(i,i+1)=Topdiagonal(i);
  k(i+1,i)=Topdiagonal(i);
end k;
%-----------------------------------MAKE F in Ka=F   F_i= integral (f*phi_i)--------
but the interval size which this is being evaluated is so
%small that f looks like a constant function.  so this integral can be
%simplified to F_i= integral f(x_i)phi_i dx.
%\int phi_i = 1/2 (base * height) = (1/2)(x_i+1 - x_i-1)
```

```
f=eye(n-2,1);
for i=1:n-2    % u''=-sin(x)
    f(i)=-sin(x(i+1)) * (1/2)*(x(i+2)-x(i));
     % need to start evaluting with x_2 since phi_1 peaks at x_2
    %f(i)=-x(i+1)*(1/2)*(x(i+2)-x(i));
end

%----------------------------SOLVE SYSTEM ka=f--------------------
[r,Q]=mgs(k);
%Qr=k,   Qr a = f,   ra=c,  and Qc=f so c=Q*f
c=Q'*f ;%since Q unitary
%now have y vector, solve ra=y by back substituion
a=zeros(n-2,1); d=n-2; for i=d:-1:1
    if i<d
         for g=i+1:d
         a(i)=a(i)-r(i,g).*a(g);
          end
         a(i)=(c(i)+a(i))/r(i,i);

     else
        a(i)=(1/r(i,i))*c(i);
    end
end
%----------------------------WHAT ARE PHI's-------------------------


s=500 ;     %number of points in x(1) to x(n)
r=linspace(x(1),x(n),s)  ;  % finer points between x_1 and x_n.

phi1=zeros(s,n-2)  ;   %positive slope side of phi's
%and there are 2 less phi functions than actual interpolation points
phi0=zeros(s,n-2)  ;    %negative slope side of phi's


for i=1:n-2  %last part of last phi begins at x_n-1  make phi1 postive slope phi's
  for m=1:s %s
      if r(m)>=x(i) &&  r(m)<=x(i+1)
           phi1(m,i)=(r(m)-x(i))/(x(i+1)-x(i));
            %so we scaled down phi(x)=x to the interval [x_e, x_e+1],
            %phiSCALED= phi[(x-x_e)/(x_e+1-x_e)]
       end% end making postive phi i

       if r(m)>=x(i+1) && r(m)<=x(i+2)
                phi0(m,i)=1-(r(m)-x(i+1))/(x(i+2)-x(i+1));
                 %this scale down from phi(x)=1-x
```

```
      end  %end making negative slope phi i
   end  %went through every point r in interval x_1 to x_n
end  %finished evaluting all phi's

%in order to get total triangle phi, phi= phi1(:,H)+phi0(:,H)


u=zeros(s,1);  %add all phi's together
for i=1:n-2
   u= u + a(i).*(phi1(:,i)+phi0(:,i));
end


for i=1:s
   %p(i)=(1/6)*r(i)^3-(1/6)*r(i);
   p(i)=-sin(r(i));
end

plot(r,u,'k')
hold on
%end %begin for look for n's at top
plot(r,p,'g')
hold off
```