Chapter 6

Finite Element Methods for 1D Boundary Value Problems

The finite element (FE) method was developed to solve complicated problems in engineering, notably in elasticity and structural mechanics modeling involving elliptic PDEs and complicated geometries. But nowadays the range of applications is quite extensive. We will use the following 1D and 2D model problems to introduce the finite element method

$$\text{1D:} \quad -u''(x) = f(x)\,, \quad 0 < x < 1\,, \quad u(0) = 0\,, \quad u(1) = 0\,;$$

$$\text{2D:} \quad -(u_{xx} + u_{yy}) = f(x, y)\,, \quad (x, y) \in \Omega, \quad u(x, y)\Big|_{\partial\Omega} = 0\,,$$

where Ω is a bounded domain in (x, y) plane with the boundary $\partial\Omega$.

6.1 The Galerkin FE method for the 1D model

We illustrate the finite element method for the 1D two-point BVP

$$-u''(x) = f(x)\,, \quad 0 < x < 1\,, \quad u(0) = 0\,, \quad u(1) = 0\,,$$

using the Galerkin finite element method described in the following steps.

1. Construct a variational or weak formulation, by multiplying both sides of the differential equation by a test function $v(x)$ satisfying the boundary conditions (BC) $v(0) = 0$, $v(1) = 0$ to get

   $$-u''v = fv\,,$$

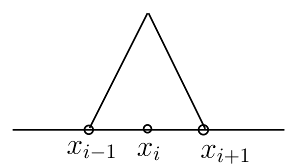
   and then integrating from 0 to 1 (using integration by parts) to have the following,

$$\begin{aligned}\int_0^1 (-u''v)\,dx &= \left. -u'v\right|_0^1 + \int_0^1 u'v'dx \\ &= \int_0^1 u'v'dx \\ \Longrightarrow \quad \int_0^1 u'v'dx &= \int_0^1 fv\,dx\,, \text{ the weak form.}\end{aligned}$$

2. Generate a mesh, *e.g.*, a uniform Cartesian mesh $x_i = i\,h$, $i = 0, 1, \cdots, n$, where $h = 1/n$, defining the intervals (x_{i-1}, x_i), $i = 1, 2, \cdots, n$.

3. Construct a set of basis functions based on the mesh, such as the piecewise linear functions $(i = 1, 2, \cdots, n-1)$

$$\phi_i(x) = \begin{cases} \dfrac{x - x_{i-1}}{h} & \text{if } x_{i-1} \le x < x_i, \\ \dfrac{x_{i+1} - x}{h} & \text{if } x_i \le x < x_{i+1}, \\ 0 & \text{otherwise}, \end{cases}$$



often called the hat functions, see the right diagram for a hat function.

4. Represent the approximate (FE) solution by a linear combination of the basis functions

$$u_h(x) = \sum_{j=1}^{n-1} c_j \phi_j(x)\,,$$

where the coefficients c_j are the unknowns to be determined. On assuming the hat basis functions, obviously $u_h(x)$ is also a piecewise linear function, although this is not usually the case for the true solution $u(x)$. Other basis functions are considered later. We then derive a linear system of equations for the coefficients by substituting the approximate solution $u_h(x)$ for the exact solution $u(x)$ in the weak form $\int_0^1 u'v'dx = \int_0^1 fv\,dx$, *i.e.*,

$$\int_0^1 u_h'v'dx = \int_0^1 fvdx\,, \text{ (noting that errors are introduced!)}$$

$$\begin{aligned}\Longrightarrow \quad \int_0^1 \sum_{j=1}^{n-1} c_j \phi_j' v'dx &= \sum_{j=1}^{n-1} c_j \int_0^1 \phi_j' v'dx \\ &= \int_0^1 fv\,dx\,.\end{aligned}$$

Next, choose the test function $v(x)$ as ϕ_1, ϕ_2, $\cdots$, ϕ_{n-1} successively, to get the system of linear equations (noting that further errors are introduced):

$$\left(\int_0^1 \phi_1'\phi_1' dx\right) c_1 + \cdots + \left(\int_0^1 \phi_1'\phi_{n-1}' dx\right) c_{n-1} = \int_0^1 f\phi_1 dx$$
$$\left(\int_0^1 \phi_2'\phi_1' dx\right) c_1 + \cdots + \left(\int_0^1 \phi_2'\phi_{n-1}' dx\right) c_{n-1} = \int_0^1 f\phi_2 dx$$
$$\cdots \quad \cdots \quad \cdots\cdots \quad \cdots \quad \cdots \quad \cdots \quad \cdots \quad \cdots$$
$$\left(\int_0^1 \phi_i'\phi_1' dx\right) c_1 + \cdots + \left(\int_0^1 \phi_i'\phi_{n-1}' dx\right) c_{n-1} = \int_0^1 f\phi_i dx$$
$$\cdots \quad \cdots \quad \cdots\cdots \quad \cdots \quad \cdots \quad \cdots \quad \cdots \quad \cdots$$
$$\left(\int_0^1 \phi_{n-1}'\phi_1' dx\right) c_1 + \cdots + \left(\int_0^1 \phi_{n-1}'\phi_{n-1}' dx\right) c_{n-1} = \int_0^1 f\phi_{n-1} dx\,,$$

or in the matrix-vector form:

$$\begin{bmatrix} a(\phi_1,\phi_1) & a(\phi_1,\phi_2) & \cdots & a(\phi_1,\phi_{n-1}) \\ a(\phi_2,\phi_1) & a(\phi_2,\phi_2) & \cdots & a(\phi_2,\phi_{n-1}) \\ \vdots & \vdots & \vdots & \vdots \\ a(\phi_{n-1},\phi_1) & a(\phi_{n-1},\phi_2) & \cdots & a(\phi_{n-1},\phi_{n-1}) \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_{n-1} \end{bmatrix} = \begin{bmatrix} (f,\phi_1) \\ (f,\phi_2) \\ \vdots \\ (f,\phi_{n-1}) \end{bmatrix},$$

where

$$a(\phi_i,\phi_j) = \int_0^1 \phi_i'\phi_j' dx\,, \qquad (f,\phi_i) = \int_0^1 f\phi_i dx\,.$$

The term $a(u,v)$ is called a bilinear form since it is linear with each variable (function), and (f,v) is called a linear form. If ϕ_i are the hat functions, then in particular we get

$$\begin{bmatrix} \frac{2}{h} & -\frac{1}{h} & & & & \\ -\frac{1}{h} & \frac{2}{h} & -\frac{1}{h} & & & \\ & -\frac{1}{h} & \frac{2}{h} & -\frac{1}{h} & & \\ & & \ddots & \ddots & \ddots & \\ & & & -\frac{1}{h} & \frac{2}{h} & -\frac{1}{h} \\ & & & & -\frac{1}{h} & \frac{2}{h} \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ \vdots \\ c_{n-2} \\ c_{n-1} \end{bmatrix} = \begin{bmatrix} \int_0^1 f\phi_1 dx \\ \int_0^1 f\phi_2 dx \\ \int_0^1 f\phi_3 dx \\ \vdots \\ \int_0^1 f\phi_{n-2} dx \\ \int_0^1 f\phi_{n-1} dx \end{bmatrix}.$$

5. <u>Solve the linear system of equations</u> for the coefficients and hence obtain the approximate solution $u_h(x) = \sum_i c_i\phi_i(x)$.

6. Carry out the error analysis (*a prior or a posteriori* error analysis).

Questions are often raised about how to appropriately

- represent ODE or PDE problems in a weak form;
- choose the basis functions ϕ, *e.g.*, in view of ODE/PDE, mesh, and the boundary conditions, *etc.*;
- implement the finite element method;
- solve the linear system of equations; and
- carry out the error analysis,

which will be addressed in subsequent chapters.

6.2 Different mathematical formulations for the 1D model

Let us consider the 1D model again,

$$\begin{aligned} &-u''(x) = f(x), \quad 0 < x < 1, \\ &u(0) = 0, \quad u(1) = 0. \end{aligned} \tag{6.1}$$

There are at least three different formulations to consider for this problem:

1. the (D)-form, the original differential equation;
2. the (V)-form, the variational form or weak form

$$\int_0^1 u'v'dx = \int_0^1 fv\,dx \tag{6.2}$$

   for any test function $v \in H_0^1(0,1)$, the Sobolev space for functions in integral forms like the C^1 space for functions (see later), and as indicated above, the corresponding finite element method is often called the Galerkin method; and

3. the (M)-form, the minimization form

$$\min_{v(x)\in H_0^1(0,1)} \left\{ \int_0^1 \left(\frac{1}{2}(v')^2 - fv \right) dx \right\}, \tag{6.3}$$

   when the corresponding finite element method is often called the Ritz method.

   As discussed in subsequent subsections, under certain assumptions these three different formulations are equivalent.

6.2.1 A physical example

From the viewpoint of mathematical modeling, both the variational (or weak) form and the minimization form are more natural than the differential formulation. For example, suppose we seek the equilibrium position of an elastic string of unit length, with two ends fixed and subject to an external force.

The equilibrium is the state that minimizes the total energy. Let $u(x)$ be the displacement of the string at a point x, and consider the deformation of an element of the string in the interval $(x, x+\Delta x)$, see Fig. 6.1 for an illustration. The potential energy of the deformed element is

$$
\begin{aligned}
&\tau \cdot \text{increase in the element length}\\
&= \tau\left(\sqrt{\left(u(x+\Delta x)-u(x)\right)^2+(\Delta x)^2}-\Delta x\right)\\
&= \tau\left(\sqrt{\left(u(x)+u_x(x)\Delta x+\frac{1}{2}u_{xx}(x)(\Delta x)^2+\cdots-u(x)\right)^2+(\Delta x)^2}-\Delta x\right)\\
&\simeq \tau\left(\sqrt{\left[1+u_x^2(x)\right](\Delta x)^2}-\Delta x\right)\\
&\simeq \frac{1}{2}\tau u_x^2(x)\Delta x\,,
\end{aligned}
$$

where τ is the coefficient of the elastic tension that we assume to be constant. If the external force is denoted by $f(x)$, the work done by this force is $-f(x)u(x)$ at every point x. Consequently, the total energy of the string (over $0 < x < 1$) is

$$
F(u) = \int_0^1 \frac{1}{2}\tau u_x^2(x)\,dx - \int_0^1 f(x)u(x)\,dx\,,
$$

from work-energy principle: the change in the kinetic energy of an object is equal to the net work done on the object. Thus to minimize the total energy, we seek the extremal u^* such that

$$
F(u^*) \le F(u)
$$

for all admissible $u(x)$, *i.e.*., the “minimizer" u^* of the functional $F(u)$ (a function of functions).

Using the principal of the virtual work, we also have

$$
\int_0^1 u'v'dx = \int_0^1 fvdx
$$

for any admissible function $v(x)$.

On the other hand, the force balance yields the relevant differential equation. The external force $f(x)$ is balanced by the tension of the elastic string given by Hooke's law, see Fig. 6.1 for an illustration, such that

$$
\begin{aligned}
&\tau\left(u_x(x+\Delta x)-u_x(x)\right) \simeq -f(x)\Delta x\\
\text{or}\qquad &\tau\frac{u_x(x+\Delta x)-u_x(x)}{\Delta x} \simeq -f(x)\,,
\end{aligned}
$$

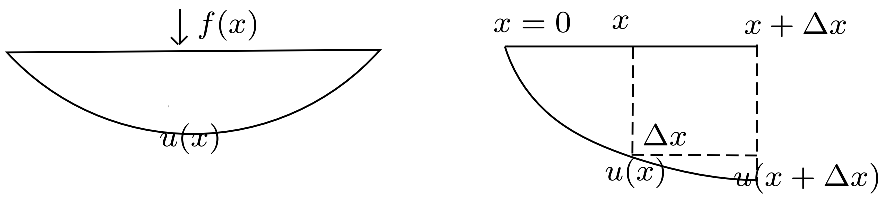


Figure 6.1. *A diagram of elastic string with two ends fixed, the displacement and force.*

thus, for $\Delta x \to 0$ we get the PDE

$$-\tau u_{xx} = f(x)\,,$$

along with the boundary condition $u(0) = 0$ and $u(1) = 0$ since the string is fixed at the two ends.

The three formulations are equivalent representations of the same problem. We show the mathematical equivalence in the next sub-section.

6.2.2 Mathematical equivalence

At the beginning of this chapter, we proved that (D) is equivalent to (V) using integration by parts. Let us now prove that under certain conditions (V) is equivalent to (D), and that (V) is equivalent to (M), and that (M) is equivalent (V).

Theorem 6.1. *(V) → (D). If u_{xx} exists and is continuous, then*

$$\int_0^1 u'v'dx = \int_0^1 fvdx, \quad \forall\, v(0) = v(1) = 0, \quad v \in H^1(0,1),$$

implies that $\;-u_{xx} = f(x)$.

Recall that $H^1(0,1)$ denotes a Sobolev space, which here we can regard as the space of all functions that have a first order derivative.

Proof: From integration by parts, we have

$$\begin{aligned} & \int_0^1 u'v'dx = u'v|_0^1 - \int_0^1 u''v\,dx, \\ \Longrightarrow \quad & -\int_0^1 u''vdx = \int_0^1 fv\,dx, \\ \text{or} \quad & \int_0^1 (u''+f)\,v\,dx = 0\,. \end{aligned}$$

Since $v(x)$ is arbitrary and continuous, and u'' and f are continuous, we must have

$$u'' + f = 0, \qquad i.e., \quad -u'' = f.$$

Theorem 6.2. *(V) → (M). Suppose $u^*(x)$ satisfies*

$$\int_0^1 u^{*\prime} v' dx = \int_0^1 v f dx$$

for any $v(x) \in H^1(0,1)$, and $v(0) = v(1) = 0$. Then

$$F(u^*) \le F(u) \quad or$$
$$\frac{1}{2}\int_0^1 (u^*)_x^2 dx - \int_0^1 f u^* dx \le \frac{1}{2}\int_0^1 u_x^2 dx - \int_0^1 f u dx.$$

Proof:

$$\begin{aligned}
F(u) &= F(u^* + u - u^*) = F(u^* + w) \ (\text{where } w = u - u^*, \quad w(0) = w(1) = 0)\,,\\
&= \int_0^1 \left(\frac{1}{2}(u^* + w)_x^2 - (u^* + w) f\right) dx\\
&= \int_0^1 \left(\frac{(u^*)_x^2 + w_x^2 + 2(u^*)_x w_x}{2} - u^* f - w f\right) dx\\
&= \int_0^1 \left(\frac{1}{2}(u^*)_x^2 - u^* f\right) dx + \int_0^1 \frac{1}{2} w_x^2 dx + \int_0^1 \left((u^*)_x w_x - f w\right) dx\\
&= \int_0^1 \left(\frac{1}{2}(u^*)_x^2 - u^* f\right) dx + \int_0^1 \frac{1}{2} w_x^2 dx + 0\\
&= F(u^*) + \int_0^1 \frac{1}{2} w_x^2 dx\\
&> F(u^*).
\end{aligned}$$

The proof is completed.

Theorem 6.3. *(M) → (V). If $u^*(x)$ is the minimizer of $F(u^*)$, then*

$$\int_0^1 (u^*)_x v_x dx = \int_0^1 f v dx$$

for any $v(0) = v(1) = 0$ and $v \in H^1(0,1)$.

Proof: Consider the auxiliary function:

$$g(\epsilon) = F(u^* + \epsilon v)\,.$$

Since $F(u^*) \le F(u^* + \epsilon v)$ for any ϵ, $g(0)$ is a global or local minimum such that

$g'(0) = 0$. To obtain the derivative of $g(\epsilon)$, we have

$$
\begin{aligned}
g(\epsilon) &= \int_0^1 \left\{ \frac{1}{2}(u^* + \epsilon v)_x^2 - (u^* + \epsilon v) f \right\} dx \\
&= \int_0^1 \left\{ \frac{1}{2}\left((u^*)_x^2 + 2(u^*)_x v_x \epsilon + v_x^2 \epsilon^2 \right) - u^* f - \epsilon v f \right\} dx \\
&= \int_0^1 \left(\frac{1}{2}(u^*)_x^2 - u^* f \right) dx + \epsilon \int_0^1 \left((u^*)_x v_x - f v \right) dx + \frac{\epsilon^2}{2} \int_0^1 v_x^2 dx \,.
\end{aligned}
$$

Thus we have

$$
g'(\epsilon) = \int_0^1 \left((u^*)_x v_x - f v \right) dx + \epsilon \int_0^1 v_x^2 dx
$$

and

$$
g'(0) = \int_0^1 \left((u^*)_x v_x - f v \right) dx = 0
$$

since $v(x)$ is arbitrary, *i.e.*, the weak form is satisfied.

However, the three different formulations may not be equivalent for some problems, depending on the regularity of the solutions. Thus although

$$
\text{(D)} \Longrightarrow \text{(M)} \Longrightarrow \text{(V)}\,,
$$

in order for (V) to imply (M) the differential equation is usually required to be self-adjoint, and for (M) or (V) to imply (D) the solution of the differential equation must have continuous second order derivatives.

6.3 Key components of the FE method for the 1D model

In this section, we discuss the model problem (6.1) using the following methods:

- Galerkin method for the variational or weak formulation;
- Ritz method for the minimization formulation.

We also discuss another important aspect of finite element methods, namely, how to assemble the stiffness matrix using the element by element approach. The first step is to choose an integral form, usually the weak form, say
$\int_0^1 u'v'\,dx = \int_0^1 f v\,dx$ for any $v(x)$ in the Sobolev space $H^1(0,1)$ with $v(0) = v(1) = 0$.

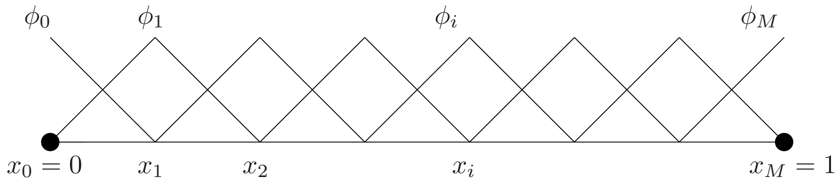


Figure 6.2. *Diagram of a mesh and hat basis functions.*

6.3.1 Mesh and basis functions

For a 1D problem, a mesh is a set of points in the interval of interest, say, $x_0 = 0$, x_1, x_2, $\cdots$, $x_M = 1$. Let $h_i = x_{i+1} - x_i$, $i = 0, 1, \cdots, M-1$.

- x_i is called a *node*, or nodal point.
- (x_i, x_{i+1}) is called an element.
- $h = \max\limits_{0 \le i \le M-1} \{ h_i \}$ is the mesh size, a measure of how fine the partition is.

Define a finite dimensional space *on* the mesh

Let the solution be in the space V, which is $H_0^1(0,1)$ in the model problem. Based on the mesh, we wish to construct a subspace

V_h (a finite dimensional space) $\subset V$ (the solution space) ,

such that the discrete problem is contained in the continuous problem.

Any such a finite element method is called conforming one. Different *finite* dimensional spaces generate different finite element solutions. Since V_h has finite dimension, we can find a set of basis functions

$$\phi_1,\ \phi_2,\ \cdots, \phi_{M-1} \subset V_h$$

that are *linearly independent*, *i.e.*, if

$$\sum_{j=1}^{M-1} \alpha_j \phi_j = 0\,,$$

then $\alpha_1 = \alpha_0 = \cdots = \alpha_{M-1} = 0$. Thus V_h is the space *spanned* by the basis functions:

$$V_h = \left\{ v_h(x),\ \ v_h(x) = \sum_{j=1}^{M-1} \alpha_j \phi_j \right\}.$$

The simplest finite dimensional space is the *piecewise continuous linear* function space defined over the mesh:

$$V_h = \Big\{\ v_h(x),\ v_h(x) \text{ is continuous piecewise linear},\ v_h(0) = v_h(1) = 0 \Big\}.$$

It is easy to show that V_h has a finite dimension, even though there are an infinite number of elements in V_h.

Find the dimension of V_h

A linear function $l(x)$ in an interval (x_i, x_{i+1}) is uniquely determined by its values at x_i and x_{i+1}:

$$l(x) = l(x_i)\frac{x - x_{i+1}}{x_i - x_{i+1}} + l(x_{i+1})\frac{x - x_i}{x_{i+1} - x_i}.$$

There are $M-1$ nodal values $l(x_i)$'s, $l(x_1)$, $l(x_2)$, $\cdots$, $l(x_{M-1})$ for a piecewise continuous linear function over the mesh, in addition to $l(x_0) = l(x_M) = 0$. Given a vector $[\,l(x_1), l(x_2), \cdots, l(x_{M-1})]^T \in R^{M-1}$, we can construct a $v_h(x) \in V_h$ by taking $v_h(x_i) = l(x_i)$, $i = 1, \cdots, M-1$. On the other hand, given $v_h(x) \in V_h$, we get a vector $[v(x_1), v(x_2), \cdots, v(x_{M-1})]^T \in R^{M-1}$. Thus there is a *one to one* relation between V_h and R^{M-1}, so V_h has the finite dimension $M-1$. Consequently, V_h is considered to be equivalent to R^{M-1}.

Find a set of basis functions

The finite dimensional space can be spanned by a set of basis functions. There are infinitely many sets of basis functions, but we should choose one that:

- is simple;
- has compact (minimum) support, *i.e.*, zero almost everywhere except for a small region; and
- meets the regularity requirement *i.e.*, continuous and differentiable, except at nodal points.

The simplest is the set of hat functions

$$\begin{aligned}
&\phi_1(x_1) = 1\,, \quad \phi_1(x_j) = 0\,, \quad j = 0, 2, 3, \cdots, M, \\
&\phi_2(x_2) = 1\,, \quad \phi_2(x_j) = 0\,, \quad j = 0, 1, 3, \cdots, M, \\
&\cdots\cdots\cdots\cdots \\
&\phi_i(x_i) = 1\,, \quad \phi_i(x_j) = 0\,, \quad j = 0, 1, \cdots i-1,\ i+1, \cdots, M, \\
&\cdots\cdots\cdots\cdots \\
&\phi_{M-1}(x_{M-1}) = 1\,, \quad \phi_{M-1}(x_j) = 0\,, \quad j = 0, 1, \cdots, M.
\end{aligned}$$

They can be represented simply as $\phi_i(x_j) = \delta_i^j$, *i.e.*,

$$\phi_i(x_j) = \begin{cases} 1\,, & \text{if } i = j\,, \\ 0\,, & \text{otherwise}\,. \end{cases} \tag{6.4}$$

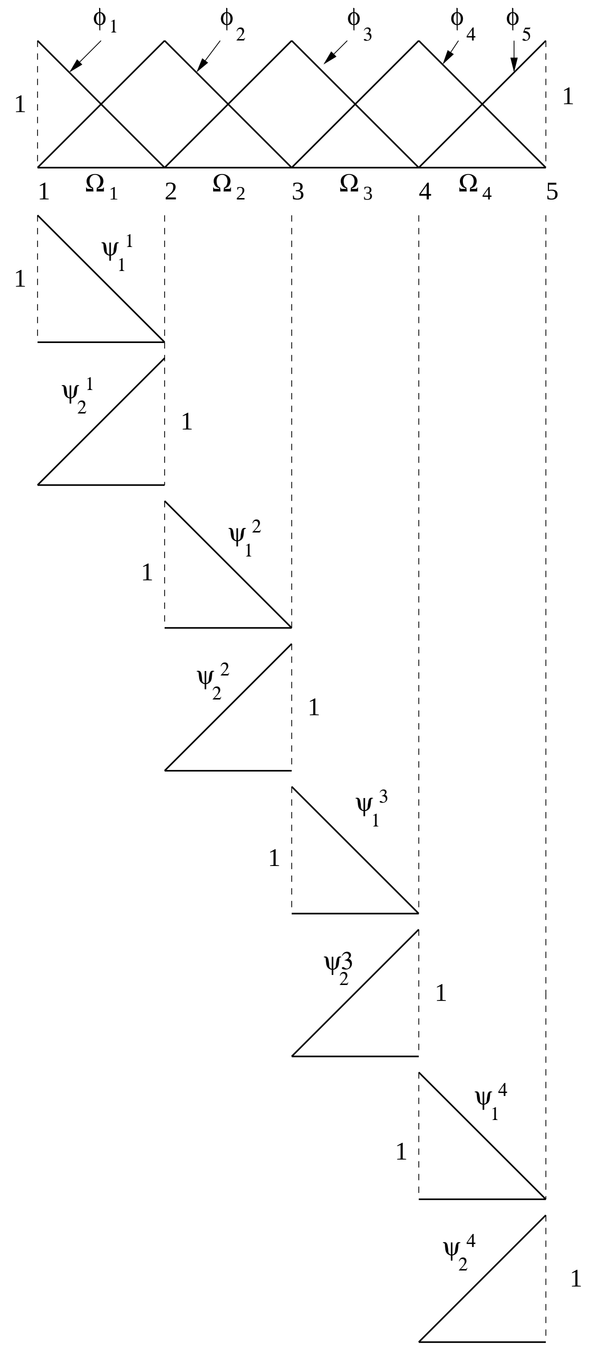


Figure 6.3. *Continuous piecewise linear basis functions ϕ_i for a 4-element mesh generated by linear shape functions ψ_1^e, ψ_2^e defined over each element. On each interior element, there are only two nonzero basis functions. The figure is adapted from Ref. [6].*

The analytic form of the hat functions for $i = 1, 2, \cdots, n-1$ is

$$\phi_i(x) = \begin{cases} 0\,, & \text{if } x < x_{i-1}\,, \\ \dfrac{x - x_{i-1}}{h_i}\,, & \text{if } x_{i-1} \le x < x_i\,, \\ \dfrac{x_{i+1} - x}{h_{i+1}}\,, & \text{if } x_i \le x < x_{i+1}\,, \\ 0\,, & \text{if } x_{i+1} \le x\,; \end{cases} \tag{6.5}$$

and the finite element solution sought is

$$u_h(x) = \sum_{j=1}^{M-1} \alpha_j\, \phi_j(x)\,, \tag{6.6}$$

and either the minimization form (M) or the variational or weak form (V) can be used to derive a linear system of equations for the coefficients α_j. On using the hat functions, we have

$$u_h(x_i) = \sum_{j=1}^{M-1} \alpha_j \phi_j(x_i) = \alpha_i \phi_i(x_i) = \alpha_i\,, \tag{6.7}$$

so α_i is an approximate solution to the exact solution at $x = x_i$.

6.3.2 The Ritz method

Although not every problem has a minimization form, the Ritz method was one of the earliest and has proven to be one of the most successful.

For the model problem (6.1), the minimization form is

$$\min_{v \in H_0^1(0,1)} F(v): \quad F(v) = \frac{1}{2}\int_0^1 (v_x)^2 dx - \int_0^1 f v\, dx\,. \tag{6.8}$$

As before, we look for an approximate solution of the form $u_h(x) = \displaystyle\sum_{j=1}^{M-1} \alpha_j \phi_j(x)$. Substituting this into the functional form gives

$$F(u_h) = \frac{1}{2}\int_0^1 \left(\sum_{j=1}^{M-1} \alpha_j \phi_j'(x) \right)^2 - \int_0^1 f \sum_{j=1}^{M-1} \alpha_j \phi_j(x) dx\,, \tag{6.9}$$

which is a multivariate function of $\alpha_1, \alpha_2, \cdots, \alpha_{M-1}$ and can be written as

$$F(v_h) = F\left(\alpha_1, \alpha_2, \cdots, \alpha_{M-1}\right).$$

The necessary condition for a global minimum (also a local minimum) is

$$\frac{\partial F}{\partial \alpha_1} = 0\,, \quad \frac{\partial F}{\partial \alpha_2} = 0\,, \quad \cdots \quad \frac{\partial F}{\partial \alpha_i} = 0\,, \quad \cdots \quad \frac{\partial F}{\partial \alpha_{M-1}} = 0\,.$$

Thus taking the partial derivatives with respect to α_j we have

$$\begin{aligned}
\frac{\partial F}{\partial \alpha_1} &= \int_0^1 \left(\sum_{j=1}^{M-1} \alpha_j \phi_j' \right) \phi_1' dx - \int_0^1 f \phi_1 dx = 0 \\
\cdots \quad & \cdots\cdots\cdots\cdots\cdots\cdots \\
\frac{\partial F}{\partial \alpha_i} &= \int_0^1 \left(\sum_{j=1}^{M-1} \alpha_j \phi_j' \right) \phi_i' dx - \int_0^1 f \phi_i dx = 0, \quad i = 1, 2, \cdots, M-1\,,
\end{aligned}$$

and on exchanging the order of integration and summation:

$$\sum_{j=1}^{M-1} \left(\int_0^1 \phi_j' \phi_i' dx \right) \alpha_j = \int_0^1 f \phi_i dx, \quad i = 1, 2, \cdots M-1.$$

This is the same system of equations that follow from the Galerkin method with the weak form, *i.e.*,

$$\begin{aligned}
\int_0^1 u' v' dx &= \int_0^1 f v\, dx \quad \text{immediately gives} \\
\int_0^1 \left(\sum_{j=1}^{M-1} \alpha_j \phi_j' \right) \phi_i' dx &= \int_0^1 f \phi_i\, dx, \quad i = 1, 2, \cdots M-1\,.
\end{aligned}$$

Comparison of the Ritz and the Galerkin FE methods

For many problems, the Ritz and Galerkin methods are theoretically equivalent.

- The Ritz method is based on the minimization form, and optimization techniques can be used to solve the problem.
- The Galerkin method usually has weaker requirements than the Ritz method. Not every problem has a minimization form, whereas almost all problems have some kind of weak form. How to choose suitable weak form and the convergence of different methods are all important issues for finite element methods.

6.3.3 Assembling the stiffness matrix element by element

Given a problem, say the model problem, after we have derived the minimization or weak form and constructed a mesh and a set of basis functions we need to form:

- the coefficient matrix $A = \{a_{ij}\} = \{\int_0^1 \phi_i' \phi_j' dx\}$, often called the *stiffness matrix for the first order derivatives*, and

- the right-hand side vector $F = \{f_i\} = \{\int_0^1 f_i \phi_i dx\}$, often called the *load vector*.

The procedure to form A and F is a crucial part in the finite element method. For the model problem, one way is by *assembling element by element*:

$$\begin{array}{ccccccc} (x_0, x_1)\,, & (x_1, x_2), & \cdots & (x_{i-1}, x_i) & \cdots & (x_{M-1}, x_M)\,, \\ \Omega_1\,, & \Omega_2\,, & \cdots & \Omega_i\,, & \cdots & \Omega_M\,. \end{array}$$

The idea is to break up the integration element by element, so that for any integrable function $g(x)$ we have

$$\int_0^1 g(x)\,dx = \sum_{k=1}^{M} \int_{x_{k-1}}^{x_k} g(x)\,dx = \sum_{k=1}^{M} \int_{\Omega_k} g(x)\,dx\,.$$

The stiffness matrix can then be written

$$A = \begin{bmatrix} \int_0^1 (\phi_1')^2 dx & \int_0^1 \phi_1'\phi_2' dx & \cdots & \int_0^1 \phi_1'\phi_{M-1}' dx \\ \int_0^1 \phi_2'\phi_1' dx & \int_0^1 (\phi_2')^2 dx & \cdots & \int_0^1 \phi_2'\phi_{M-1}' dx \\ \vdots & \vdots & \vdots & \vdots \\ \int_0^1 \phi_{M-1}'\phi_1' dx & \int_0^1 \phi_{M-1}'\phi_2' dx & \cdots & \int_0^1 (\phi_{M-1}')^2 dx \end{bmatrix}$$

$$= \begin{bmatrix} \int_{x_0}^{x_1} (\phi_1')^2 dx & \int_{x_0}^{x_1} \phi_1'\phi_2' dx & \cdots & \int_{x_0}^{x_1} \phi_1'\phi_{M-1}' dx \\ \int_{x_0}^{x_1} \phi_2'\phi_1' dx & \int_{x_0}^{x_1} (\phi_2')^2 dx & \cdots & \int_{x_0}^{x_1} \phi_2'\phi_{M-1}' dx \\ \vdots & \vdots & \vdots & \vdots \\ \int_{x_0}^{x_1} \phi_{M-1}'\phi_1' dx & \int_{x_0}^{x_1} \phi_{M-1}'\phi_2' dx & \cdots & \int_{x_0}^{x_1} (\phi_{M-1}')^2 dx \end{bmatrix}$$

$$+ \begin{bmatrix} \int_{x_1}^{x_2} (\phi_1')^2 dx & \int_{x_1}^{x_2} \phi_1'\phi_2' dx & \cdots & \int_{x_1}^{x_2} \phi_1'\phi_{M-1}' dx \\ \int_{x_1}^{x_2} \phi_2'\phi_1' dx & \int_{x_1}^{x_2} (\phi_2')^2 dx & \cdots & \int_{x_1}^{x_2} \phi_2'\phi_{M-1}' dx \\ \vdots & \vdots & \vdots & \vdots \\ \int_{x_1}^{x_2} \phi_{M-1}'\phi_1' dx & \int_{x_1}^{x_2} \phi_{M-1}'\phi_2' dx & \cdots & \int_{x_1}^{x_2} (\phi_{M-1}')^2 dx \end{bmatrix}$$

$$+ \cdots$$

$$+ \begin{bmatrix} \int_{x_{M-1}}^{x_M} (\phi_1')^2 dx & \int_{x_{M-1}}^{x_M} \phi_1'\phi_2' dx & \cdots & \int_{x_{M-1}}^{x_M} \phi_1'\phi_{M-1}' dx \\ \int_{x_{M-1}}^{x_M} \phi_2'\phi_1' dx & \int_{x_{M-1}}^{x_M} (\phi_2')^2 dx & \cdots & \int_{x_{M-1}}^{x_M} \phi_2'\phi_{M-1}' dx \\ \vdots & \vdots & \vdots & \vdots \\ \int_{x_{M-1}}^{x_M} \phi_{M-1}'\phi_1' dx & \int_{x_{M-1}}^{x_M} \phi_{M-1}'\phi_2' dx & \cdots & \int_{x_{M-1}}^{x_M} (\phi_{M-1}')^2 dx \end{bmatrix}.$$

For the hat basis functions, it is noted that each interval has only *two* nonzero basis functions, *cf.*, Fig. 6.3. This leads to

$$A = \begin{bmatrix} \int_{x_0}^{x_1} (\phi_1')^2 dx & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix} + \begin{bmatrix} \int_{x_1}^{x_2} (\phi_1')^2 dx & \int_{x_1}^{x_2} \phi_1'\phi_2' dx & \cdots & 0 \\ \int_{x_1}^{x_2} \phi_2'\phi_1' dx & \int_{x_1}^{x_2} (\phi_2')^2 dx & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}$$

$$+ \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ 0 & \int_{x_2}^{x_3} (\phi_2')^2 dx & \int_{x_2}^{x_3} \phi_2'\phi_3' dx & \cdots & 0 \\ 0 & \int_{x_2}^{x_3} \phi_3'\phi_2' dx & \int_{x_2}^{x_3} (\phi_3')^2 dx & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & \int_{x_{M-1}}^{x_M} (\phi_{M-1}')^2 dx \end{bmatrix}.$$

The nonzero contribution from a particular element is

$$K_i^e = \begin{bmatrix} \int_{x_i}^{x_{i+1}} (\phi_i')^2 dx & \int_{x_i}^{x_{i+1}} \phi_i'\phi_{i+1}' dx \\ \int_{x_i}^{x_{i+1}} \phi_{i+1}'\phi_i' dx & \int_{x_i}^{x_{i+1}} (\phi_{i+1}')^2 dx \end{bmatrix},$$

the two by two *local stiffness matrix*. Similarly, the *local load vector* is

$$F_i^e = \begin{bmatrix} \int_{x_i}^{x_{i+1}} f\phi_i dx \\ \int_{x_i}^{x_{i+1}} f\phi_{i+1} dx \end{bmatrix},$$

and the global load vector can also be assembled element by element:

$$F = \begin{bmatrix} \int_{x_0}^{x_1} f\phi_1 dx \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} \int_{x_1}^{x_2} f\phi_1 dx \\ \int_{x_1}^{x_2} f\phi_2 dx \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \int_{x_2}^{x_3} f\phi_2 dx \\ \int_{x_2}^{x_3} f\phi_3 dx \\ \vdots \\ 0 \\ 0 \end{bmatrix} + \cdots + \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ \int_{x_{M-1}}^{x_M} f\phi_{M-1} dx \end{bmatrix}.$$

Computing local stiffness matrix K_i^e and local load vector F_i^e

In the element (x_i, x_{i+1}), there are only two nonzero hat functions centered at x_i and x_{i+1} respectively:

$$\psi_i^e(x) = \frac{x_{i+1} - x}{x_{i+1} - x_i}\,, \qquad \psi_{i+1}^e(x) = \frac{x - x_i}{x_{i+1} - x_i}\,,$$
$$(\psi_i^e)' = -\frac{1}{h_i}\,, \qquad (\psi_{i+1}^e)' = \frac{1}{h_i}\,,$$

where ψ_i^e and ψ_{i+1}^e are defined only on one particular element. We can concentrate on the corresponding contribution to the stiffness matrix and load vector from the two nonzero hat functions. It is easy to verify that

$$\int_{x_i}^{x_{i+1}} (\psi_i')^2 \, dx = \int_{x_i}^{x_{i+1}} \frac{1}{h_i^2} \, dx = \frac{1}{h_i}\,, \qquad \int_{x_i}^{x_{i+1}} \psi_i' \psi_{i+1}' \, dx = \int_{x_i}^{x_{i+1}} -\frac{1}{h_i^2} \, dx = -\frac{1}{h_i}\,,$$
$$\int_{x_i}^{x_{i+1}} (\psi_{i+1}')^2 \, dx = \int_{x_i}^{x_{i+1}} \frac{1}{h_i^2} \, dx = \frac{1}{h_i}\,.$$

The local stiffness matrix K_i^e is therefore

$$K_i^e = \begin{bmatrix} \dfrac{1}{h_i} & -\dfrac{1}{h_i} \\ -\dfrac{1}{h_i} & \dfrac{1}{h_i} \end{bmatrix},$$

and the stiffness matrix A is assembled as follows:

$$A = 0^{(M-1)\times(M-1)}\,, \quad A = \begin{bmatrix} \frac{1}{h_0} & 0 & 0 & \cdots \\ 0 & 0 & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix}, \quad A = \begin{bmatrix} \frac{1}{h_0} + \frac{1}{h_1} & -\frac{1}{h_1} & 0 & \cdots \\ -\frac{1}{h_1} & \frac{1}{h_1} & 0 & \cdots \\ 0 & 0 & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix},$$

$$\cdots\cdots \quad A = \begin{bmatrix} \frac{1}{h_0} + \frac{1}{h_1} & -\frac{1}{h_1} & 0 & 0 & \cdots \\ -\frac{1}{h_1} & \frac{1}{h_1} + \frac{1}{h_2} & -\frac{1}{h_2} & 0 & \cdots \\ 0 & -\frac{1}{h_2} & \frac{1}{h_2} & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \end{bmatrix}.$$

Thus we finally assemble the tridiagonal matrix

$$A = \begin{bmatrix} \frac{1}{h_0}+\frac{1}{h_1} & -\frac{1}{h_1} & & & & \\ -\frac{1}{h_1} & \frac{1}{h_1}+\frac{1}{h_2} & -\frac{1}{h_2} & & & \\ & -\frac{1}{h_2} & \frac{1}{h_2}+\frac{1}{h_3} & -\frac{1}{h_3} & & \\ & & \ddots & \ddots & \ddots & \\ & & & -\frac{1}{h_{M-3}} & \frac{1}{h_{M-3}}+\frac{1}{h_{M-2}} & -\frac{1}{h_{M-2}} \\ & & & & -\frac{1}{h_{M-2}} & \frac{1}{h_{M-2}}+\frac{1}{h_{M-1}} \end{bmatrix}.$$

Remark 6.1. *For a uniform mesh $x_i = ih$, $h = 1/M$, $i = 0, 1, \cdots, M$ and the integral approximated by the mid-point rule*

$$\int_0^1 f(x)\phi_i(x)dx = \int_{x_{i-1}}^{x_{i+1}} f(x)\phi_i(x) \simeq \int_{x_{i-1}}^{x_{i+1}} f(x_i)\phi_i(x)dx$$
$$= f(x_i)\int_{x_{i-1}}^{x_{i+1}} \phi_i(x)dx = f(x_i)\,,$$

the resulting system of equations for the model problem from the finite element method is identical to that obtained from the FD method.

6.4 Matlab programming of the FE method for the 1D model problem

Matlab codes to solve the 1D model problem

$$-u''(x) = f(x), \quad a < x < b; \qquad u(a) = u(b) = 0 \tag{6.10}$$

using the hat basis functions are available either through the link

http://www4.ncsu.edu/~zhilin/FD_FEM_book

or by e-mail request to the authors. The Matlab codes include the following Matlab functions:

- $U = fem1d(x)$ is the main subroutine of the finite method using the hat basis functions. The input x is the vector containing the nodal points. The output U, $U(0) = U(M) = 0$ is the finite element solution at the nodal points, where $M + 1$ is the total nodal points.

- $y = hat1(x, x1, x2)$ is the local hat function in the interval $[x1, x2]$ which takes one at $x = x2$ and zero at $x = x1$.

- $y = hat2(x, x1, x2)$ is the local hat function in the interval $[x1, x2]$ which takes one at $x = x1$ and zero at $x = x2$.
- $y = int\_hata1\_f(x1, x2)$ computes the integral $\int_{x1}^{x2} f(x)hat1dx$ using the Simpson rule.
- $y = int\_hata2\_f(x1, x2)$ computes the integral $\int_{x1}^{x2} f(x)hat2dx$ using the Simpson quadrature rule.
- The main function is *drive.m* which solves the problem, plot the solution and the error.
- $y = f(x)$ is the right hand side of the differential equation.
- $y = soln(x)$ is the exact solution of differential equation.
- $y = fem\_soln(x, U, xp)$ evaluates the finite element solution at an arbitrary point xp in the solution domain.

We explain some of these Matlab functions in the following subsections.

6.4.1 Define the basis functions

In an element $[x_1, x_2]$ there are two nonzero basis functions: one is

$$\psi_1^e(x) = \frac{x - x_1}{x_2 - x_1} \tag{6.11}$$

where the Matlab code is the file hat1.m so

```
function y = hat1(x,x1,x2)
% This function evaluates the hat function
    y = (x-x1)/(x2-x1);
return
```

and the other is

$$\psi_2^e(x) = \frac{x_2 - x}{x_2 - x_1} \tag{6.12}$$

where the Matlab code is the file hat2.m so

```
function y = hat2(x,x1,x2)
% This function evaluates the hat function
    y = (x2-x)/(x2-x1);
return
```

6.4.2 Define $f(x)$

```
function y = f(x)
    y = 1;  % for example
return
```

6.4.3 The main FE routine

```
function U = fem1d(x)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                      %
%           A simple Matlab code of 1D FE method for                   %
%                                                                      %
%                   -u'' = f(x),   a <= x <= b, u(a)=u(b)=0            %
%    Input: x, Nodal points                                            %
%    Output: U, FE solution at nodal points                            %
%                                                                      %
%    Function needed: f(x).                                            %
%                                                                      %
%    Matlab functions used:                                            %
%                                                                      %
%     hat1(x,x1,x2), hat function in [x1,x2] that is 1 at x2; and      %
%     0 at x1.                                                         %
%                                                                      %
%     hat2(x,x1,x2), hat function in [x1,x2] that is 0 at x1; and      %
%     1 at x1.                                                         %
%                                                                      %
%     int_hat1_f(x1,x2): Contribution to the load vector from hat1     %
%     int_hat2_f(x1,x2): Contribution to the load vector from hat2     %
%                                                                      %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

M = length(x);
for i=1:M-1,
  h(i) = x(i+1)-x(i);
end

A = sparse(M,M); F=zeros(M,1);          % Initialization
A(1,1) = 1; F(1)=0;
A(M,M) = 1; F(M)=0;
A(2,2) = 1/h(1); F(2) = int_hat1_f(x(1),x(2));

for i=2:M-2,                       % Assembling element by element
  A(i,i)     = A(i,i) + 1/h(i);
  A(i,i+1)   = A(i,i+1) - 1/h(i);
  A(i+1,i)   = A(i+1,i) - 1/h(i);
  A(i+1,i+1) = A(i+1,i+1) + 1/h(i);
  F(i)       = F(i) + int_hat2_f(x(i),x(i+1));
  F(i+1)     = F(i+1) + int_hat1_f(x(i),x(i+1));
end
```

```
  A(M-1,M-1) = A(M-1,M-1) + 1/h(M-1);
  F(M-1)     = F(M-1) + int_hat2_f(x(M-1),x(M));

  U = A\F;                  % Solve the linear system of equations

  return
```

6.4.4 A test example

Let us consider the test example

$$f(x) = 1\,, \quad a = 0\,, \quad b = 1\,.$$

The exact solution is

$$u(x) = \frac{x(1-x)}{2}\,. \tag{6.13}$$

A sample Matlab drive code is listed below:

```
clear all; close all;     % Clear every thing so it won't mess up with other
                          % existing variables.

%%%%%% Generate a mesh.

x(1)=0; x(2)=0.1; x(3)=0.3; x(4)=0.333; x(5)=0.5; x(6)=0.75;x(7)=1;

U = fem1d(x);

%%%%%% Compare errors:

x2 = 0:0.05:1; k2 = length(x2);
for i=1:k2,
  u_exact(i) = soln(x2(i));
  u_fem(i) = fem_soln(x,U,x2(i));  % Compute FE solution at x2(i)
end

error = norm(u_fem-u_exact,inf)  % Compute the infinity error

plot(x2,u_fem,':', x2,u_exact)   % Solid: the exact, %dotted: FE solution
hold; plot(x,U,'o')              % Mark the solution at nodal points
xlabel('x'); ylabel('u(x) & u_{fem}(x)');
title('Solid line: Exact solution, Dotted line: FE solution')

figure(2); plot(x2,u_fem-u_exact); title('Error plot')
xlabel('x'); ylabel('u-u_{fem}');  title('Error Plot')
```

Fig. 6.4 shows the plots produced by running the code. Fig. 6.4 (a) shows both the true solution (the solid line) and the finite element solution (the dashed line). The little 'o's are the finite element solution values at the nodal points. Fig. 6.4 (a) shows the error between the true and the finite element solutions at a few selected points (zero at the nodal points in this example, although may not be not so in general).

(a) (b)

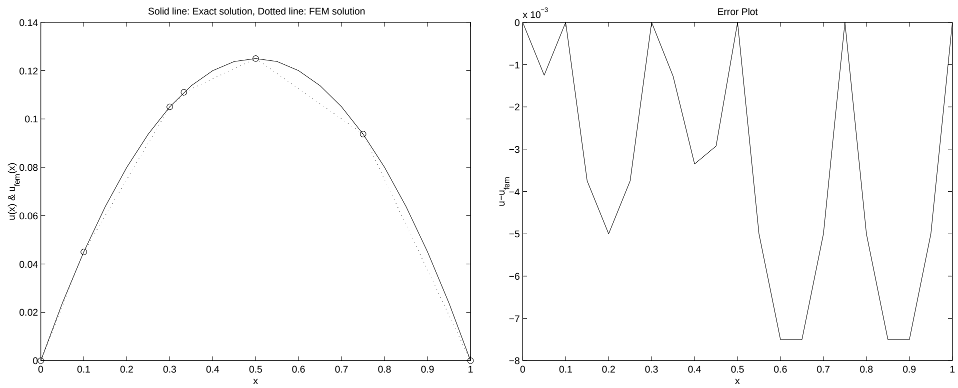


Figure 6.4. *(a) Plot of the true solution (solid line) and the finite element solution (the dashed line). (b): The error plot at some selected points.*

6.5 Exercises

1. Consider the following BVP:

$$-u''(x) + u(x) = f(x), \qquad 0 < x < 1, \quad u(0) = u(1) = 0.$$

   (a) Show that the weak form (variational form) is

$$(u', v') + (u, v) = (f, v), \quad \forall\, v(x) \in H_0^1(0,1),$$

   where

$$(u,v) = \int_0^1 u(x)v(x)dx,$$
$$H_0^1(0,1) = \left\{ v(x), \quad v(0) = v(1) = 0, \quad \int_0^1 v^2 dx < \infty \right\}.$$

   (b) Derive the linear system of the equations for the finite element approximation

$$u_h = \sum_{j=1}^{3} \alpha_j \phi_j(x)\,,$$

   with the following information:

- $f(x) = 1$;
- the nodal points and the elements are indexed as

$$x_0 = 0, \quad x_2 = \frac{1}{4}, \quad x_3 = \frac{1}{2}, \quad x_1 = \frac{3}{4}, \quad x_4 = 1.$$
$$\Omega_1 = [x_3, x_1], \quad \Omega_2 = [x_1, x_4], \quad \Omega_3 = [x_2, x_3], \quad \Omega_4 = [x_0, x_2] \, ;$$

- the basis functions are the hat functions

$$\phi_i(x_j) = \begin{cases} 1\,, & \text{if } i = j\,, \\ 0\,, & \text{otherwise}\,, \end{cases}$$

  and not re-order the nodal points and elements; and
- assemble the stiffness matrix and the load vector element by element.

2. (This problem involves modifying *drive.m*, *f.m* and *soln.m*.) Use the Matlab codes to solve

$$-u''(x) = f(x), \qquad 0 < x < 1, \quad u(0) = u(1) = 0.$$

Try two different meshes: (a) the one given in *drive.m*; (b) the uniform mesh $x_i = i\,h$, $h = 1/M$, $i = 0, 1, \cdots, M$. Take $M = 10$, done in Matlab using the command: $x = 0 : 0.1 : 1$.

Use the two meshes to solve the problem for the following $f(x)$ or exact $u(x)$:

(a) given $u(x) = \sin(\pi x)$, what is $f(x)$?

(b) given $f(x) = x^3$, what is $u(x)$?

(c) (extra credit) given $f(x) = \delta(x - 1/2)$, where $\delta(x)$ is the Dirac delta function, what is $u(x)$? **Hint:** The Dirac delta function is defined as a distribution satisfying $\int_a^b f(x)\delta(x)dx = f(0)$ for any function $f(x) \in C(a, b)$ if $x = 0$ is in the interior of the integration.

Ensure that the errors are reasonably small.

3. (This problem involves modifying *fem1d.m*, *drive.m*, *f.m* and *soln.m*.) Assume that

$$\int_{x_i}^{x_{i+1}} \phi_i(x)\phi_{i+1}(x)dx = \frac{h}{6},$$

where $h = x_{i+1} - x_i$), and ϕ_i and ϕ_{i+1} are the hat functions centered at x_i and x_{i+1} respectively. Use the Matlab codes to solve

$$-u''(x) + u(x) = f(x), \qquad 0 < x < 1, \quad u(0) = u(1) = 0\;.$$

Try to use the uniform grid $x = 0 : 0.1 : 1$ in Matlab, for the following exact $u(x)$:

(a) $u(x) = \sin(\pi x)$, what is $f(x)$?

(b) $u(x) = x(1 - x)/2$, what is $f(x)$?