PE281 Finite Element Method Course Notes

summarized by Tara LaForce

Stanford, CA 23rd May 2006

1 Derivation of the Method

In order to derive the fundamental concepts of FEM we will start by looking at an extremely simple ODE and approximate it using FEM.

1.1 The Model Problem

The model problem is:

$$\begin{array}{cc} -u'' + u = x & 0 < x < 1 \\ u\left(0\right) = 0 & u\left(1\right) = 0 \end{array} \tag{1}$$

and this problem can be solved analytically: $u\left(x\right) = x - sinhx/sinh1$. The purpose of starting with this problem is to demonstrate the fundamental concepts and pitfalls in FEM in a situation where we know the correct answer, so that we will know where our approximation is good and where it is poor. In cases of practical interest we will look at ODEs and PDEs that are too complex to be solved analytically.

FEM doesn't actually approximate the original equation, but rather the *weak form* of the original equation. The purpose of the weak form is to satisfy the equation in the "average sense," so that we can approximate solutions that are discontinuous or otherwise poorly behaved. If a function $u(x)$ is a solution to the original form of the ODE, then it also satisfies the weak form of the ODE. The weak form of Eq. 1 is

$$\int_0^1 \left(-u'' + u\right) v dx = \int_0^1 x v dx \tag{2}$$

The function $v(x)$ is called the weight function or *test function.* $v(x)$ can be any function of x that is sufficiently well behaved for the integrals to exist. The set of all functions v that also have $v(0) = 0$, $v(1) = 0$ are denoted by H. (We will put many more constraints on v shortly.)

The new problem is to find u so that

$$\begin{array}{cc} \int_0^1 \left(-u'' + u - x\right) v dx = 0 & forallv \in H \\ u\left(0\right) = 0 & u\left(1\right) = 0 \end{array} \tag{3}$$

Once the problem is written in this way we can say that the solution u belongs to the class of *trial functions* which are denoted $\tilde{H}$. When the problem is written in this way the classes of test functions H and trial functions $\tilde{H}$ are not the same. For example, u must be twice differentiable and have the property that $\int_0^1 u'' v dx < \infty$, while v doesn't even have to be continuous as long as the integral in Eq. 3 exists and is finite. It is possible to approximate u in this way, but having to work with two different classes of functions unnecessarily complicates the problem. In order to make sure that H and $\tilde{H}$ are the same we can observe that if v is sufficiently smooth then

$$\int_0^1 -u'' v dx = \int_0^1 u' v' dx - u' v|_0^1 \tag{4}$$

This formulation must be valid since u must be twice differentiable and v was arbitrary. This puts another constraint on v that it must be differentiable and that those derivatives must be well-enough behaved to ensure that the integral $\int_0^1 u' v' dx$ exists. Moreover, since we decided from the outset that $v(0) = 0$ and $v(1) = 0$, the second term in Eq. 4 is zero regardless of the behavior of u' at these points. The new problem is

$$\int_0^1 \left(u' v' + u v - x v\right) dx = 0. \tag{5}$$

Notice that by performing the integration by parts we restricted the class of test functions H by introducing v' into the equation. We have simultaneously expanded the class of trial functions $\tilde{H}$, since u is no longer required to have a second derivative in Eq. 5. The weak formulation defined in Eq. 5 is called a variational boundary-value problem.

In Eq. 5 u and v have exactly the same constraints on them:

1. u and v must be square integrable, that is: $\int_0^1 uvdx \approx \int_0^1 u^2dx < \infty$
2. The first derivatives of u and v must be square integrable, that is: $\int_0^1 u'v'dx \approx \int_0^1 (u')^2dx < \infty$ (this actually guarantees the first property)
3. We had already assumed that $v(0) = 0$ and $v(1) = 0$ and we know from the original statement of the problem that $u(0) = 0$ and $u(1) = 0$.

Now we have that $\tilde{H} = H = H_0^1$. Any function w is a member of H_0^1 if $\int_0^1 (u')^2dx < \infty$ and $w(0) = w(1) = 0$. H_0^1 is the space of admissible functions for the variational boundary-value problem (ie. all admissible test *and* trial functions are in H_0^1)

We will consider the variational form Eq. 5 to be the equation that we would like to approximate, rather than the original statement in Eq. 1. Once we have found a solution to Eq. 5 in this way we can ask the question whether this formulation is also a solution to Eq. 1: That is, whether this solution is a function satisfying Eq. 1 at every x in $0 < x < 1$, or whether we have found a solution that satisfies only the weak form of the equation. In the case that we can only find a solution to the weak form, no "classical" solution exists.

1.2 Galerkin Approximations

We now have the problem re-stated so that we are looking for $u \in H_0^1$ such that

$$\int_0^1 (u'v' + uv)\,dx = \int_0^1 xvdx \tag{6}$$

for all $v \in H_0^1$. In order to narrow down the number of functions we will consider in our approximate solutions we will make two more assumptions about H_0^1. First, we will assume that H_0^1 is a linear space of functions (that is if $v_1, v_2 \in H_0^1$ and a, b are constants then $av_1 + bv_2 \in H_0^1$.)

The second assumption is that H_0^1 is infinite dimensional. For example if we have the sine series $\psi_n(x) = \sqrt{2}sin(n\pi x)$ for $n = 1, 2, 3, ...$ and $v \in H_0^1$ then

v can be represented by $v(x) = \sum_{n=1}^{\infty} a_n \psi_n(x)$. The scalar coefficients a_n are given by $a_n = \int_0^1 v(x)\,\psi_n(x)\,dx$, just like usual. Hence infinititely many coefficients a_n must be found to define v exactly. As in Fourier analysis, many of these coefficients will be zero. We will also truncate the series in order to have managable length series, just like in discrete Fourier analysis.

Unlike in Fourier analysis, though the basis functions do not have to be sines and cosines, much less smooth functions can be used. In fact our set of basis functions do not even have to be smooth and can contain discontinuities in the derivatives, but they must be continuous. We will assume that the infinite series converges so that we can consider only the first N basis functions and get a good approximation v_N of the original test (or trial) function:

$$v \cong v_N = \sum_{i=1}^{N} \beta_i \phi_i(x) \tag{7}$$

where ϕ_i are as-yet unspecified basis functions. This subspace of functions is denoted $H_0^{(N)}$ and is a *subspace* of H_0^1. Galerkin's method consists of finding an approximate solution to Eq. 6 in a finite-dimensional subspace $H_0^{(N)}$ of $H_0^{(1}$ of admissible functions rather than in the whole space H_0^1. Now we are looking for $u_N = \sum_{i=1}^{N} \alpha_i \phi_i(x)$. The new approximate problem we have is to find $u_N \in H_0^{(N)}$ such that

$$\int_0^1 \left(u_N' v_N' + u_N v_N\right) dx = \int_0^1 x v_N dx \tag{8}$$

for all $v_N \in H_0^{(N)}$. Since the ϕ_i are known (in principle) u_N will be completely determined once the coefficients α_i have been found.

In order to find that α_n we put $\sum_{i=1}^{N} \alpha_i \phi_i(x)$ and $\sum_{i=1}^{N} \beta_i \phi_i(x)$ into Eq. 8.

$$\int_0^1 \left\{ \begin{array}{l} \frac{d}{dx}\left[\sum_{i=1}^{N} \beta_i \phi_i(x)\right] \frac{d}{dx}\left[\sum_{j=1}^{N} \alpha_j \phi_i(x)\right] + \\ \left[\sum_{i=1}^{N} \beta_i \phi_i(x)\right] \left[\sum_{j=1}^{N} \alpha_j \phi_i(x)\right] - \\ x \sum_{i=1}^{N} \beta_i \phi_i(x) \end{array} \right\} dx = 0 \tag{9}$$

for all N independent sets of β_i.

This can be expanded and factored to give

$$\sum\nolimits_{i=1}^{N}\beta_i\left(\sum\nolimits_{j=1}^{N}\left\{\int_0^1\left[\phi_j'(x)\,\phi_i'(x)+\phi_j(x)\,\phi_i(x)\right]dx\right\}\alpha_j-\int_0^1 x\phi_i(x)\,dx\right)=0 \tag{10}$$

for all N independent sets of β_i. The structure of Eq. 10 is easier to see if it is re-written as

$$\sum\nolimits_{i=1}^{N}\beta_i\left(\sum\nolimits_{j=1}^{N}K_{ij}\alpha_j-F_i\right)=0 \tag{11}$$

for all β_i. Where

$$K_{ij}=\int_0^1\left[\phi_j'(x)\,\phi_i'(x)+\phi_j(x)\,\phi_i(x)\right]dx \quad F=\int_0^1 x\phi_i(x)\,dx \tag{12}$$

and where $i,\ j=1,...,N$. The $N\times N$ matrix of K_{ij} is called the stiffness matrix and the vector F is the load vector. Since the β_i are known K_{ij} and F can be calculated directly. But the β_i were arbitrary so we can choose each element β_i for each equation. For the first equation choose $\beta_1=1$ and $\beta_n=0$ for $n\neq 1$. Now $\sum_{j=1}^{N}K_{1j}\alpha_j=F_1$. Similarly for the second equation choose $\beta_2=1$ and $\beta_n=0$ for $n\neq 2$ so that $\sum_{j=1}^{N}K_{2j}\alpha_j=F_2$. In this way we have chosen N independent equations that can be used to find the N unknowns α_i. Moreover the N coefficients α_i can be found from $\alpha_j=\sum_{j=1}^{N}\left(K^{-1}\right)_{ji}F_i$ where $\left(K^{-1}\right)_{ji}$ are the elements of the inverse of K.

The stiffness matrix K is symmetric for this simple problem, which makes the computation of the matrix faster since we don't have to compute all of the elements, symmetric matricies are also much faster to invert.

1.3 Finite Elements Basis Functions

Now we have done a great deal of work, but it may not seem like we are much closer to finding a solution to the original ODE since we still know nothing about ϕ_i. The purpose of using such a general formulation is that any set of linearly independent functions will work to solve the ODE. Now we are finally going to talk about what kind of functions we will want to use as basis functions. The finite element method is a general and systematic technique for constructing basis functions for Galerkin approximations. In

FEM the basis functions ϕ_i are defined piecewise over subregions. Over any subdomain the ϕ_i will be chosen to be polynomials of low degree, though other possibilities do exist.

- *finite elements* are the subregions of the domain over which each basis function is defined. Hence each basis function has compact support over an element. Each element has length h. The lengths of the elements do NOT need to be the same (but generally we will assume that they are.)
- *nodes* or *nodal points* are defined within each element. In Figure 1 the five nodes are the endpoints of each element (numbered 0 to 4).
- the *finite element mesh* is the collection of elements and nodal points that make up the domain and is shown in Figure 1. An element i is denoted by Ω_i.

Now we need to construct the actual basis functions using the three criteria defined before: 1) The basis functions are simple functions defined piecewise over the finite element mesh, 2) the basis functions must be in the class of test functions H_0^1, and 3) The basis functions are chosen so that the parameters α_i are the values of $u_N(x)$ at the nodal points.

The simplest set of basis functions are the "hat functions" on elements $i = 1, 2, 3$.

$$\phi_i(x) = \left\{ \begin{array}{ccc} \frac{x-x_{i-1}}{h_i} & \text{for} & x_{i-1} \le x \le x_i \\ \frac{x_{i+1}-x}{h_{i+1}} & \text{for} & x_i \le x \le x_{i+1} \\ 0 & \text{for} & x < x_{i-1}, x > x_{i+1} \end{array} \right\} \tag{13}$$

where $h_i = x_i - x_{i-1}$ is the length of element i. The derivatives are

$$\phi_i'(x) = \left\{ \begin{array}{ccc} \frac{1}{h_i} & \text{for} & x_{i-1} \le x \le x_i \\ \frac{-1}{h_{i+1}} & \text{for} & x_i \le x \le x_{i+1} \\ 0 & \text{for} & x < x_{i-1}, x > x_{i+1} \end{array} \right\} \tag{14}$$

The equations for elements 0 and 4 have been left out since we decided that $u(0) = u(1) = 1$, so no basis functions are required. In general the basis functions for the first and last elements are half of the functions since there is no $i-1$ or $i+1$ node, respectively. The hat functions are shown in Figure 2. The mathematical term for hat functions is *piece-wise linear basis functions*

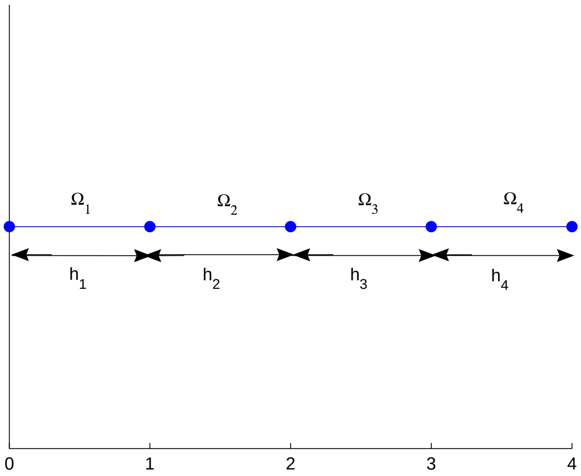

Figure 1: Four finite elements on the interval [0 1].

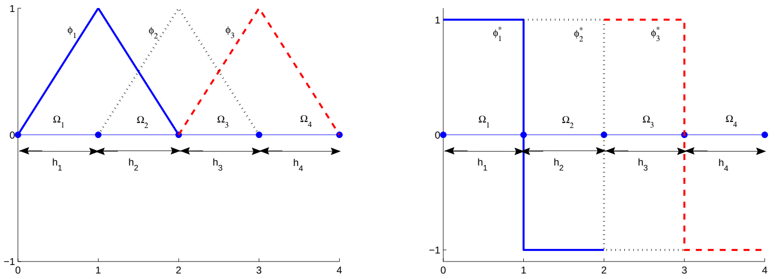

Figure 2: Four hat functions (top) and their derivatives (bottom) on the interval [0 1].

Looking at the three criteria above, clearly the functions in Eq. 13 are simple and defined element-wise. It is easy to show that they are in H_0^1, since they have square-integrable first derivatives. They also satisfy the third criteria since $\phi_i(x_j) = 1$ if $i = j$ and 0 otherwise. Hence each function contributes to the value of u_N at exactly one node and $\alpha_i = u_N(x_i)$.

It is less clear that the hat functions will give a continuous representation of v_N and u_N. Let v be the sine function with period 2 shown in Figure

3. At the nodes (0, 1, 2, 3, 4) sine has the values (0,0.7071,1,0.7071,0). The representation v_N on the finite element mesh is $v_N = 0.7071\phi_1(x) + \phi_2(x) + 0.7071\phi_3(x)$. When the elements are summed up the sine wave is approximated by piecewise linear functions between each of the nodes, and is exactly represented at each node. When more nodes are used the approximation improves and in the limit of $N \to \infty$ the sine wave would be exactly represented. In FEM we will never proceed all the way to the limit, so the interval size h will always have finite size h. This is why the term *finite elements* is used.

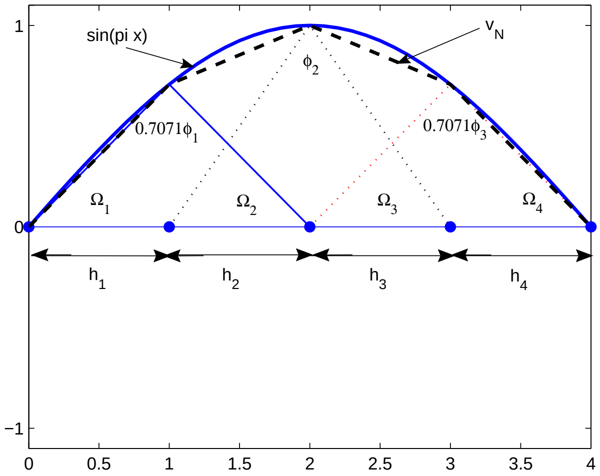

Figure 3: The finite element approximation of $sin(\pi x)$ using five nodes on the interval [0 1].

1.4 The Stiffness Matrix K and the Load Vector F for Hat Functions

Recall from Eq. 12 that each element of the stiffness matrix K is given by

$$\begin{aligned} K_{ij} &= \int_0^1 \left(\phi_i'(x)\,\phi_j'(x) + \phi_i(x)\,\phi_j(x)\right)dx \\ &= \sum_{e=1}^4 \int_{\Omega_e} \left(\phi_i'(x)\,\phi_j'(x) + \phi_i(x)\,\phi_j(x)\right)dx \\ &= \sum_{e=1}^4 K_{ij}^e \end{aligned} \tag{15}$$

similarly

$$F_i = \int_0^1 x\phi_i\,(x)dx = \sum\nolimits_{e=1}^{4} \int_{\Omega_e} x\phi_i\,(x)dx = \sum\nolimits_{e=1}^{4} F_i^e \tag{16}$$

where we have used the property that $\phi(x)$ are defined piecewise on each element 1 through 4. In order to compute an approximation of the solution to the model ODE it is necessary to compute nine elements for K_{ij} from $i, j = 1, 2, 3$ and three elements for F. But since each of the functions $\phi(x)$ are defined in the same way it is possible to compute K^e and F^e for a generic element and then to construct the matrix using the sums above. Consider a generic interior element Ω_e on the interval x_A to x_B. We will use a change of variables and rewrite this in terms of ξ, a dummy variable for x. We will have $\xi = (0, h)$. On this element exactly two of the hat functions are nonzero: $\psi_A(\xi) = 1-\frac{\xi}{h}$ and $\psi_B(\xi) = \frac{\xi}{h}$. Convince yourself that this definition is equivalent to the previous definition of the hat function, but with the origin shifted to the start of one of the interior elements. The two hat functions have derivatives $\psi_A'(\xi) = -\frac{1}{h}$ and $\psi_B'(\xi) = \frac{1}{h}$.

It is also important to notice that for the hat functions $\phi_i(x) \neq 0$ on only the elements Ω_i and Ω_{i+1}. This results in a tridiagonal sparse matrix K for any number of elements in the mesh as will be shown below. Using Eq. 15 you can see that there are three integrals that contribute to K_{ij}:

$$\begin{aligned}
&k_{AA} = \textstyle\int_0^h \left([\psi_A^e{}'\,(\xi)]^2 + [\psi_A^e\,(\xi)]^2\right)d\xi \\
&= \textstyle\int_0^h \left([1/h]^2 + [1-\xi/h]^2\right)d\xi = 1/h + h/3 \\
&k_{AB} = \textstyle\int_0^h \left(\psi_A^e{}'\,(\xi)\,\psi_B^e{}'\,(\xi) + \psi_A^e\,(\xi)\,\psi_B^e\,(\xi)\right)d\xi \\
&= \textstyle\int_0^h \left((-1/h)\,(1/h) + (1-\xi/h)\,(\xi/h)\right)d\xi = -1/h + h/6 \\
&k_{BB} = \textstyle\int_0^h \left([\psi_B^e{}'\,(\xi)]^2 + [\psi_B^e\,(\xi)]^2\right)d\xi \\
&= \textstyle\int_0^h \left([-1/h]^2 + [\xi/h]^2\right)d\xi = 1/h + h/3
\end{aligned} \tag{17}$$

Similarly the components that contribute to the load vector are:

$$\begin{aligned}
F_A^e &= \textstyle\int_0^h (x_A + \xi)\,(1 - \xi/h)\,d\xi = \frac{h}{6}\,(2x_A + x_B) \\
F_B^e &= \textstyle\int_0^h (x_A + \xi)\,(\xi/h)\,d\xi = \frac{h}{6}\,(x_A + 2x_B)
\end{aligned} \tag{18}$$

where the x_A and x_B terms come from evaluating the forcing function $f(x) = x$ at the endpoints of the generic element.

Thus each generic interior element contributes to the stiffness matrix a 2 × 2 submatrix

$$k^e = \begin{bmatrix} 1/h + h/3 & -1/h + h/6 \\ -1/h + h/6 & 1/h + h/3 \end{bmatrix} \tag{19}$$

and two entries to the load vector

$$f^e = h/6 \begin{bmatrix} 2x_A + x_B \\ x_A + 2x_B \end{bmatrix} \tag{20}$$

For the 4 element mesh we have derived the contributions to the overall stiffness matrix K from each node is given by:

$$\begin{aligned} K^1 &= \begin{bmatrix} 1/h + h/3 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} K^2 = \begin{bmatrix} 1/h + h/3 & -1/h + h/6 & 0 \\ -1/h + h/6 & 1/h + h/3 & 0 \\ 0 & 0 & 0 \end{bmatrix} \\ K^3 &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1/h + h/3 & -1/h + h/6 \\ 0 & -1/h + h/6 & 1/h + h/3 \end{bmatrix} K^4 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1/h + h/3 \end{bmatrix} \end{aligned} \tag{21}$$

where the contributions from elements 1 and 4 have only one entry because only half of the hat function exists on these elements. Similarly the contributions to the load vector are

$$F^1 = h/6 \begin{bmatrix} 2h \\ 0 \\ 0 \end{bmatrix} F^2 = h/6 \begin{bmatrix} 2h + 2h \\ h + 4h \\ 0 \end{bmatrix} \tag{22}$$

$$F^3 = h/6 \begin{bmatrix} 0 \\ 4h + 3h \\ 2h + 6h \end{bmatrix} F^4 = h/6 \begin{bmatrix} 0 \\ 0 \\ 6h + 4h \end{bmatrix} \tag{23}$$

where $h = 0.35$ for the model problem. Now $K = K^1 + K^2 + K^3 + K^4$ and $F = F^1 + F^2 + F^3 + F^4$. The final system of equations has symmetric and diagonally dominant stiffness matrix K, which is very nice to work with mathematically. The values of u_N at each node is given by $\tilde{\alpha} = K^{-1}F$ and $u_N = \sum_{i=1}^{3} \alpha_i \phi_i (x)$.

Using this we get that the approximation to the model problem is $u = 0.0353\phi_1(x) + 0.0569\phi_2(x) + 0.0505\phi_3(x)$. This is not a very accurate answer, since only four elements were used. A more accurate approximation can be obtained by using more elements, but at the cost of building and inverting a larger stiffness matrix K. The usual way of estimating the error of an FEM approximation using linear basis functions (the hat functions we derived) using the L_2 or mean-square norm is that $||e||_0 < C_2 h^2$. This is an a-priori error estimate and in general a worst-case scenario, the actual error may be substantially smaller.

2 General One Dimensional Problems

At this point we will extend the derivation above to include general linear second-order elliptic ODEs of the form

$$a_o(x)\frac{d^2u(x)}{dx^2} + a_1(x)\frac{du}{dx} + a_2(x)u(x) = f(x). \tag{24}$$

Recall that this equation is elliptic if a_o never changes sign or vanishes, ie $|a_o(x)| > \gamma > 0$. We will also focus on *two-point boundary-value problems* which are problems where half of the boundary conditions are specified at each end point.

2.1 Flow Through Porous Media

One example in which elliptic boundary value problems arise in PE is as one-dimensional flow through porous media. Start by defining the flux σ of the fluid as $\sigma(x) = -k(x)\frac{du}{dx}$. σ can be a general flux function of any type, but for porous media flow, u is hydraulic head or pressure, σ is the flow rate, k is the absolute permeability, $f(x)$ is the fluid source/sink and may represent wells or a boundary condition with flow across it, such as a constant flux boundary with an aquifer. In porous media flow we are also implicitly assuming the additional equation for Darcy's law holds, that is $\sigma = -ku'$. The permeability does not have to be constant in this formulation, but may vary with x.

2.2 One-Dimensional Heat-Loss

One example in which elliptic boundary value problems arise in PE is heat loss from a wellbore. For the moment we can consider only 1D heat loss, but radial heat loss can be approximated using FEM as well. In heat loss u is the temperature, σ is the heat flux given by Fourier's law, $f(x)$ is the heat source (in this example, the wellbore), and k is the thermal conductivity of the porous medium.

2.3 Boundary Conditions

The most general boundary conditions that we will consider are

$$\begin{array}{c} \alpha_0 \frac{du(0)}{dx} + \beta_0 u\left(0\right) = \gamma_0 \\ \alpha_l \frac{du(l)}{dx} + \beta_l u\left(l\right) = \gamma_l \end{array} \tag{25}$$

Only the value of u is specified in a *Dirichlet Boundary Condition.* Only the value of $\frac{du(0)}{dx}$ is specified in a *Neumann Boundary Condition.* The flux $-k(0)(-\frac{du(0)}{dx}) = \sigma_0$ or a linear combination of the flux and u are specified in a *natural boundary condition.*

Now we have a general ODE of the form

$$-\frac{d}{dx}\left(k\left(x\right)\frac{du\left(x\right)}{dx}\right) + c\left(x\right)\frac{du}{dx} + b\left(x\right)u\left(x\right) = f\left(x\right) \tag{26}$$

where the second derivative of u has been replaced by the definition of the flux. This ODE doesn't necessarily have a second derivative at every single point in the domain since $k(x)$ may not be continuous. u must have a second derivative over every smooth sub-domain Ω_i, but there may discontinuites at the boundary between two subdomains. We didn't worry about the existence of the second derivative in the model problem in the last section because the real physical problems we want to solve have the form of Eq. 26, with the boundary conditions defined in Eq. 25, not the form of Eq. 1.

As before, the ODE is multiplied by the test function v and integrated to give the variational form of the two-point boundary-value problem on any smooth domain Ω

$$-ku'v|_{x_{i-1}}^{x_i} + \int_{\Omega} (ku'v' + cu'v + buv)\, dx - \int_{\Omega_i} fvdx = 0 \tag{27}$$

If our domain is not smooth we can solve this problem over a series of subdomains where the ODE is smooth and sum them. There are three types of discontinuity that are possible at the edges of the Ω:

- $k(x)$ is discontinuous, $f(x)$ is continuous at x_1. This gives a jump condition across the boundary of the element, but since k is finite on both sides of the jump the flux is continuous and $[\sigma(x)] = 0$ at the boundary.
- $f(x)$ is discontinuous, $k(x)$ is continuous at x_2. This gives a jump condition across the boundary of the element, but as long as f is finite on both sides of the jump the flux is continuous and $[\sigma(x)] = 0$ at the boundary.
- $f(x)$ is discontinuous at x_3 and has a concentrated forcing term given by the $-\hat{f}\delta(x - x_i)$ which is not finite. This gives a jump condition across the boundary of the element, and the flux is not continuous and $[\sigma(x)] = \hat{f}$ at the boundary.

As a consequence when we sum over all of the elements the variational boundary value problem becomes

$$\begin{array}{l} k(0)\, u'(0)\, v(0) - k(l)\, u'(l)\, v(l) + \int_0^l (ku'v' + cu'v + buv)\, dx \\ + [\sigma(x_1)]\, v(x_1) + [\sigma(x_2)]\, v(x_2) + [\sigma(x_3)]\, v(x_3) = \int_0^l fvdx \end{array} \tag{28}$$

At the points x_1 and x_2 the jump condition is zero so those terms drop out, but the jump at x_3 is not zero, so we would have to deal with the $\hat{f}$ term in the FEM. For the rest of the derivation we will assume that we have no discontinuites of this type, but it is important to know that even very messy domains can easily be handled using FEM.

Rewriting Eq. 28 so that the homogeneous ODE is on the left and the forcing and boundary terms are on the right gives

$$\begin{array}{l} \int_0^l (ku'v' + cu'v + buv)\, dx = -v(0)\, k(0)\, [\gamma_0 - \beta_0 u(0)]/\alpha_0 \\ +v(l)\, k(l)\, [\gamma_l - \beta_l u(l)]/\alpha_l + \int_0^l fvdx + \hat{f}v(x_3) \end{array} \tag{29}$$

2.4 Galerkin Approximation

Exactly as in the first section we are looking for a finite set of basis functions $\{\phi_1, \phi_2, ..., \phi_N\}$ to approximate $u_N \in H^{(N)}$. In this case we are no longer requiring that our basis functions have the property $v(0) = v(l) = 0$ because the ODE is not necessarily zero at the endpoints, and we want to have the same test and trial functions. The stiffness matrix for Eq. 29 is given by

$$K_{ij} = \int_0^l \left(k\phi_i'\phi_j' + c\phi_j'\phi_i + b\phi_j\phi_i\right) dx \tag{30}$$

where $j = i$ or $j = i + 1$ are the only nonzero entries. The load vector is

$$F_i = \int_0^l f\phi_i\,(x)dx + -\phi_i\,(0)\,k\,(0)\,\gamma_0/\alpha_0 + \phi_i\,(l)\,k\,(l)\,\gamma_l/\alpha_l \tag{31}$$

where $\phi_i(0) = 0$ for every element except the first and $\phi_i(l) = 0$ for every element except the last. Once again we will be looking only at linear shape functions, but since we are now using v_N that do not have $v(0) = v(l) = 0$ we are going to re-define the shape functions slightly so that the origin of the shifted coordinate system is at the center of the element and $\xi = -1$ at the left endpoint and $\xi = 1$ at the right endpoint. Now the two functions defined on a generic element Ω_e are given by $\hat{\psi}_1\,(\xi) = 0.5\,(1 - \xi)$ and $\hat{\psi}_2\,(\xi) = 0.5\,(1 + \xi)$. This is exactly the same functions employed in the first section, except that now on the first and last elements the basis functions are no longer zero. The real purpose of redefining the hat functions in this way is that this formulation allows for the easy definition of higher-order approximations.

Each element makes a contribution to the stiffness matrix of the form

$$k_{ij}^e = \int_{s_1^e}^{s_2^e} \left(k\psi_i^{e\prime}\,(\xi)\,\psi_j^{e\prime}\,(\xi) + c\psi_j^{e\prime}\,(\xi)\,\psi_i^e\,(\xi) + c\psi_i^e\,(\xi)\,\psi_j^e\,(\xi)\right) d\xi \tag{32}$$

for $j = i \pm 1$ and contributes to the load vector

$$f_i^e = \int_{s_1^e}^{s_2^e} \left(\hat{f}\psi_i^e\,(\xi)\right) d\xi. \tag{33}$$

Typically the integrals are computed numerically. We are ignoring the boundary conditions and any discontinuities in the data at this point.

For linear basis functions each element has two nodes that contribute, so the first element has two equations of the form

$$\begin{aligned} k_{11}^1 u_1 + k_{12}^1 u_2 = f_1^1 \\ k_{21}^1 u_1 + k_{22}^1 u_2 = f_1^2 \end{aligned} \tag{34}$$

The *ith* interior node has two equations of the form

$$\begin{aligned} k_{11}^i u_{i-1} + k_{12}^i u_i = f_1^i \\ k_{21}^i u_{i-1} + k_{22}^i u_i = f_2^i \end{aligned} \tag{35}$$

This gives a tridiagonal matrix K and a load vector F such that

$$K = \begin{bmatrix} k_{11}^1 & k_{12}^1 & 0 & 0 \\ k_{21}^1 & k_{22}^1 + k_{11}^2 & k_{12}^2 & 0 \\ 0 & k_{21}^2 & k_{22}^2 + k_{11}^3 & k_{12}^{N-1} \\ 0 & 0 & k_{21}^{N-1} & k_{22}^{N-1} \end{bmatrix} F = \begin{bmatrix} f_1^1 \\ f_2^1 + f_1^2 \\ f_2^2 + f_1^3 \\ f_2^{N-1} \end{bmatrix} \tag{36}$$

Provided that there are no discontinuities in the initial data. (See the book for how to handle discontinuities.) This formulation isn't complete because doesn't account for any boundary conditions.

2.5 Natural Boundary Conditions

For general natural boundary conditions of the form

$$\begin{aligned} \alpha_0 \tfrac{du(0)}{dx} + \beta_0 u\left(0\right) = \gamma_0 \\ \alpha_l \tfrac{du(l)}{dx} + \beta_l u\left(l\right) = \gamma_l \end{aligned} \tag{37}$$

the matrix-vector equation including boundary conditions is

$$\begin{bmatrix} k_{11}^1 - \frac{k(0)\beta_0}{\alpha_o} & k_{12}^1 & 0 & 0 \\ k_{21}^1 & k_{22}^1 + k_{11}^2 & k_{12}^2 & 0 \\ 0 & k_{21}^2 & k_{22}^2 + k_{11}^3 & k_{12}^{N-1} \\ 0 & 0 & k_{21}^{N-1} & k_{22}^{N-1} + \frac{k(l)\beta_l}{\alpha_l} \end{bmatrix} \begin{bmatrix} \mathrm{u}_1 \\ \mathrm{u}_2 \\ \mathrm{u}_3 \\ \mathrm{u}_{\mathrm{N}-1} \end{bmatrix} = \begin{bmatrix} f_1^1 - \frac{k(0)\gamma_0}{\alpha_o} \\ f_2^1 + f_1^2 \\ f_2^2 + f_1^3 \\ f_2^{N-1} + \frac{k(l)\gamma_l}{\alpha_l} \end{bmatrix} \tag{38}$$

2.6 Neumann Boundary Conditions

For Neumann Boundary Conditions the equation is identical to Eq.38 with $\beta_o = \beta_l = 0$.

2.7 Dirichlet Boundary Conditions

For Dirichlet Boundary conditions the matrix problem reduces to a smaller problem. Since $u(0)$ and $u(l)$ are both specified we don't have to solve for them. The first and last rows do not need to be included in the equation, but the second and $N-2$ row of the load vector must be adjusted so that the new matrix-vector problem is

$$\begin{bmatrix} k_{22}^1 + k_{11}^2 & k_{12}^2 & 0 & 0 \\ k_{21}^2 & k_{22}^2 + k_{11}^3 & k_{12}^3 & 0 \\ 0 & k_{21}^3 & k_{22}^3 + k_{11}^4 & k_{12}^{N-2} \\ 0 & 0 & k_{21}^{N-2} & k_{22}^{N-2} + k_{11}^{N-1} \end{bmatrix} \begin{bmatrix} \mathrm{u_2} \\ \mathrm{u_3} \\ \mathrm{u_4} \\ \mathrm{u_{N-2}} \end{bmatrix} = \begin{bmatrix} f_2^1 + f_1^2 - \frac{k_{21}^1 \gamma_o}{\beta_o} \\ f_2^2 + f_1^3 \\ f_2^3 + f_1^4 \\ f_2^{N-2} + f_1^{N-1} - \frac{k_{12}^{N-1} \gamma_l}{\beta_l} \end{bmatrix} \tag{39}$$

Any combination of boundary conditions is also possible, and each boundary can be set up as described above independently of the other.

3 Higher-Order Approximations

In general it is possible to use any polynomial function to approximate the function on each element. In practice it is rarely desirable to use much higher than quadratic basis functions because higher-order functions have too much oscillation. In order to define basis functions of order n each element must have $n+1$ nodes. The *ith* shape functions for an *nth* order approximation for the basis functions is

$$\hat{\psi}_i(\xi) = \frac{(\xi - \xi_1)(\xi - \xi_2)\dots(\xi - \xi_{i-1})(\xi - \xi_{i+1})\dots(\xi - \xi_{n+1})}{(\xi_i - \xi_1)(\xi_i - \xi_2)\dots(\xi_i - \xi_{i-1})(\xi_i - \xi_{i+1})\dots(\xi_i - \xi_{n+1})} \tag{40}$$

Each of these functions is one at the node ξ_i and zero at ξ_j for $i \neq j$, which implies that they are all linearly independent on the element. There are $n+1$

linearly independent shape functions on each element. Using this definition with $n = 1$ gives the linear basis functions discussed in section 2.4. For $n = 2$ we have to define three nodes per element, two are at the ends of the element and one is in the center. Now the three shape functions are $\hat{\psi}_1(\xi) = \frac{1}{2}\xi(\xi-1)$, $\hat{\psi}_2(\xi) = 1 - \xi^2$, and $\hat{\psi}_3(\xi) = \frac{1}{2}\xi(\xi + 1)$.

For higher-order approximations the matrix K is defined in the same way as for linear elements except that now we have

$$u_h^e(x) = \sum_{j=1}^{N_e} u_j^e \psi_j^e(x) \tag{41}$$

on each element. Where N_e is the number of nodes per element. This means that now instead of having to find u_h at two points on every element we need to find u_h at $n + 1$ points, which makes our matrix-vector problem considerably larger. Frequently higher order methods are worth the cost because a higher-order approximation has error of order h^{n+1}, and is still computationally cheaper than dividing the grid into enough elements to get the same error from a linear approxiamtion.

Consider, for example if the interval $[0, 1]$ is divided up into four elements. A linear approximation would contain 8 basis functions and have an error on the order of $h^2 = 0.25^2 = 0.0625$. A quadratic approximation would contain 12 basis functions and have an error on the order of $h^3 = 0.25^3 = 0.015625$. In order to get an error this small using linear approximations we would have to divide the domain into eight elements so that $h^2 = 0.125^2 = 0.015625$, and we would have to compute 16 basis functions. Though the matrix-vector problem is the same size for linear elements with $h = 0.125$ and quadratic elements with $h = 0.25$, the quadratic formulation requires only 3/4 as many computations in order to assemble the matrix.

4 Two-Dimensional Problems

At long last it is time to solve some PDEs using FEM! Fortunately all of the concepts from the first two sections apply (almost) directly to two-dimensional problems. We will consider problems of the form

$$-\nabla \cdot k\left(\nabla u(x, y)\right) + b(x, y)\, u(x, y) = f(x, y) \tag{42}$$

Where the negative sign ensures that we are solving an elliptic equation. Once again we will define a flux term $\sigma(x, y) = k (\nabla u (x, y))$. Where k is the material modulus. In porous media flow k would be the absolute permeability, and in heat loss k would be thermal conductivity. As before, these do not need to be constants, or even continuous. However, we will restrict our derivation to cases where k is either continuous, or at least nicely enough behaved to not cause discontinuities in the flux. This is exactly what we did in solving ODEs.

We will formulate the PDE as a variational boundary-value problem by multiplying by a test function v and integrating over the domain (this is now an integral in two-dimensions!)

$$\int_{\Omega} \left[-\nabla \cdot k \left(\nabla u \left(x, y\right)\right) + b \left(x, y\right) u \left(x, y\right) - f \left(x, y\right)\right] v dx dy = 0 \tag{43}$$

In order to get everything in terms of first derivatives like we did in ODEs we use the product rule for differentiation to show that

$$\begin{aligned} \nabla \cdot (vk\nabla u) &= k\nabla u \cdot \nabla v + v\nabla \cdot (k\nabla u) \\ v\nabla \cdot (k\nabla u) &= \nabla \cdot (vk\nabla u) - k\nabla u \cdot \nabla v \end{aligned} \tag{44}$$

which can be inserted into Eq. 43 to give

$$\int_{\Omega} \left[k\nabla u \cdot \nabla v + buv - fv\right] dx dy - \int_{\Omega} \left[\nabla \cdot (vk\nabla u)\right] dx dy = 0 \tag{45}$$

Using the divergence theorem we obtain that

$$-\int_{\Omega} \left[\nabla \cdot (vk\nabla u)\right] dx dy = -\int_{\partial\Omega} k\nabla u \cdot n v ds \tag{46}$$

where $\partial\Omega$ is the boundary of Ω integrated counterclockwise. Now we have the final variational boundary-value problem

$$\int_{\Omega} \left[k\nabla u \cdot \nabla v + buv - fv\right] dx dy - \int_{\partial\Omega} k \frac{\partial u \left(s\right)}{\partial n} v ds = 0 \tag{47}$$

with boundary conditions

$$-k\left(s\right)\frac{\partial u\left(s\right)}{\partial n}=p\left(s\right)\left[u\left(s\right)\right)] \qquad (48)$$

where $s \in \partial\Omega$, the boundary of Ω.

4.1 Approximation Functions

The idea here is to represent the approximate solution $u_h(x, y)$ and test functions $v_h(x, y)$ by polynomials defined piecewise over geometrically simple subdomains of Ω. In one-dimension this consisted of dividing the line between $[0, l]$ up into parts. In two-dimensions there are many posssible choices of simple shapes that we could choose to divide up the domain into. We will only consider trianglar and rectangular elements.

4.1.1 Two-Dimensional Problems on Triangular Mesh

The simplest possible choice of shape function in two dimensions is a line $v_h(x, y) = a_1 + a_2x + a_3y$. Three constants need to be found, which means every element must have three nodes. A triangle with nodes at the corners would be the simplest and most logical way to satisfy this constraint. Moreover, if adjacent triangular elements are forced to share two nodes then this will define a continuous function across the element boundary.

Similarly if we wanted to use a quadratic shape function $v_h(x, y) = a_1+a_2x+a_3y + a_4x^2 + a_5xy + a_6y^2$ we have six parameters and a triangular mesh with nodes at each corner and at the midpoints of each side of the tringle would be a good choice.

4.1.2 Two-Dimensional Problems on Rectangular Mesh

Suppose instead that we wanted to use bilinear shape functions $v_h(x, y) = a_1 + a_2x + a_3y + a_4xy$. In this case we need to specify four nodes per element in order to find the four constants a_1 to a_4. The logical choice here would be to use rectangular elements with nodes defined at each corner. If adjacent elements are forced to share two nodes then this will define a continuous function across the element boundary.

There are infinitely many possible combinations of shape functions and elements. Sometimes it is desirable to use more complicated shapes in the mesh instead of triangular or rectangular. We will restrict our analysis to linear functions on triangular mesh, since that is the simplest choice.

4.1.3 Shape Functions

In two dimensions there are three basic requirements for the shape functions (and they are almost the same as in the ODE):

- The approximation to u must be continuous across element boundaries.
- The shape functions ψ_i^e must each be one at exactly one node and zero at all others.
- The basis functions must be square-integrable and have square-integrable first partial derivatives.

The linear function $v_h(x, y) = a_1 + a_2 x + a_3 y$ defines a plane in space. As a consequence the approximation of u will be made up of triangular shaped segments of planes that are continuous.

Suppose that the corners of a triangular elements are given by (x_1, y_1) (x_2, y_2) and (x_3, y_3). The shape function ψ_1 that one at (x_1, y_1) and zero at nodes (x_2, y_2) and (x_3, y_3) is found from the plane equation evaluated at each node and is

$$\psi_1^e(\xi) = \frac{1}{2A_e}\left[(x_2y_3 - x_3y_2) + (y_2 - y_3)\, x + (x_3 - x_2)\, y\right] \tag{49}$$

where $A_e = x_2y_3 + x_1y_2 + x_3y_1 - x_2y_1 - x_3y_2 - x_1y_3$ is the area of the element.

Similarly, the shape functions that are one at the node (x_2, y_2) and (x_3, y_3) are ψ_2 and ψ_3 respectively.

$$\begin{aligned} \psi_2^e(\xi) &= \tfrac{1}{2A_e}\left[(x_3y_1 - x_1y_3) + (y_3 - y_1)\, x + (x_1 - x_3)\, y\right] \\ \psi_3^e(\xi) &= \tfrac{1}{2A_e}\left[(x_1y_2 - x_2y_1) + (y_1 - y_2)\, x + (x_2 - x_1)\, y\right] \end{aligned} \tag{50}$$

4.2 Finite Element Approximations

In general we are trying to find $u_h\left(x,y\right)=\sum_{j=1}^{N}u_j\phi_j\left(x,y\right)$ such that $u_j=\hat{u}_j$ at the nodes on $\partial\Omega_h$ and

$$\begin{array}{l}\int_{\Omega_h}\left[k\left(\frac{\partial u_h}{\partial x}\frac{\partial v_h}{\partial x}+\frac{\partial u_h}{\partial y}\frac{\partial v_h}{\partial y}\right)+bu_hv_h\right]dxdy+\int_{\partial\Omega}pu_hv_hds\\=\int_{\Omega_h}fv_hdxdy+\int_{\partial\Omega_h}\gamma v_hds\end{array}\tag{51}$$

where $\gamma=p\hat{u}$. The general boundary condition has been put into the integral equation.

As before, the stiffness matrix K is given by

$$K_{ij}=\int_{\Omega_h}\left[k\left(\frac{\partial u_h}{\partial x}\frac{\partial v_h}{\partial x}+\frac{\partial u_h}{\partial y}\frac{\partial v_h}{\partial y}\right)+bu_hv_h\right]dxdy+\int_{\partial\Omega}pu_hv_hds\tag{52}$$

and the load vector F is

$$F_i=\int_{\Omega_h}fv_hdxdy+\int_{\partial\Omega_h}\gamma v_hds\tag{53}$$

Ignoring the boundary conditions for the moment, we can look at the structure of the linear basis functions and see that each function v_i will contribute to exactly three of the columns of K (ie it effects three of the u_j) in row i since there are three nodes per element. Hence the element matrix k^e are 3×3 matricies and the element load vector f^e is a 3×1 column vector.

Consider a generic element e. The contribution to the stiffness matrix from the first basis function $v=\psi_1$ is given by

$$\begin{array}{l}\int_{\Omega_h}\left[\begin{array}{l}k\left[\begin{array}{l}\left(\psi_{x1}\alpha_1+\psi_{x2}\alpha_2+\psi_{x2}\alpha_3\right)\psi_{x1}\\+\left(\psi_{y1}\alpha_1+\psi_{y2}\alpha_2+\psi_{y2}\alpha_3\right)\psi_{y1}\end{array}\right]\\+b\left(\psi_1\alpha_1+\psi_2\alpha_2+\psi_2\alpha_3\right)\psi_1\end{array}\right]dxdy=k_{11}^e\alpha_1+k_{12}^e\alpha_2+k_{13}^e\alpha_3\\k_{11}^e=\int_{\Omega_h}\left[k\left[\psi_{x1}^2+\psi_{y1}^2\right]+b\psi_1^2\right]dxdy\\k_{12}^e=\int_{\Omega_h}\left[k\left[\psi_{x2}\psi_{x1}+\psi_{y2}\psi_{y1}\right]+b\psi_1\psi_2\right]dxdy\\k_{13}^e=\int_{\Omega_h}\left[k\left[\psi_{x3}\psi_{x1}+\psi_{y3}\psi_{y1}\right]+b\psi_1\psi_3\right]dxdy\end{array}\tag{54}$$

where ψ_{yi} is the derivative of ψ_i with respect to y etc. and $u^e(x, y) = \sum_{j=1}^{3} \alpha_j \psi_j$. Similarly, the contribution to the stiffness matrix from the first basis function $v = \psi_2$ is given by

$$\begin{array}{l} k_{21}^e = k_{12}^e \\ k_{22}^e = \int_{\Omega_h} \left[k \left[\psi_{x2}^2 + \psi_{y2}^2 \right] + b\psi_2^2 \right] dxdy \\ k_{23}^e = \int_{\Omega_h} \left[k \left[\psi_{x3}\psi_{x2} + \psi_{y3}\psi_{y2} \right] + b\psi_2\psi_3 \right] dxdy \end{array} \tag{55}$$

and the contribution from ψ_3 is

$$\begin{array}{l} k_{31}^e = k_{13}^e \\ k_{32}^e = k_{23}^e \\ k_{33}^e = \int_{\Omega_h} \left[k \left[\psi_{x3}^2 + \psi_{y3}^2 \right] + b\psi_3^2 \right] dxdy \end{array} \tag{56}$$

the contribution to the load vector from each basis function on the element e are

$$f_1^e = \int_{\Omega_h} [f\psi_1]\, dxdy f_2^e = \int_{\Omega_h} [f\psi_2]\, dxdy f_3^e = \int_{\Omega_h} [f\psi_3]\, dxdy \tag{57}$$

The contribution of the first element to the total stiffness matrix and load vector are given by

$$K^1 = \begin{bmatrix} k_{11}^1 & k_{12}^1 & k_{13}^1 & 0 & 0 & 0 & 0 \\ k_{21}^1 & k_{22}^1 & k_{23}^1 & 0 & 0 & 0 & 0 \\ k_{31}^1 & k_{32}^1 & k_{33}^1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} F^1 = \begin{bmatrix} f_1^1 \\ f_2^1 \\ f_3^1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{58}$$

In the case of two-dimensional problems we can't just add each element matrix to the diagonal like we did in one dimensional problems because of the locations of the nodes. Because the mesh is triangular elements and nodes don't necessarily have a nice correspondence. For example element 6 may have as its nodes 3, 5, and 6 as in the picture. In that case, the shape functions on Ω_1 will only effect the value of u at nodes 3, 5, and 6. This means that the contribution of the sixth element to the stiffness matrix and load vector are given by

$$K^6 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & k^6_{11} & 0 & k^6_{12} & k^6_{13} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & k^6_{21} & 0 & k^6_{22} & k^6_{23} & 0 \\ 0 & 0 & k^6_{31} & 0 & k^6_{32} & k^6_{33} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} F^6 = \begin{bmatrix} 0 \\ 0 \\ f^6_1 \\ 0 \\ f^6_2 \\ f^6_3 \\ 0 \end{bmatrix} \qquad (59)$$

For two-dimensional problems the stiffness matrix is no longer tridiagional. If the nodes and elements are carefully ordered it can usually be written so that it is a sparse matrix with large blocks of zeros.

4.3 Boundary Conditions

The stiffness matrix and load vectors we have assembled do not include any boundary conditions. Implementation of boundary conditions is conceptually the same as in one-dimensional problems: Neumann boundary conditions are implemented by subtracting the flux term $\frac{k(s)\partial u(s)}{\partial n}$ from the load vector. Natural boundary conditions are implemented by subtracting the flux term $\frac{k(s)\partial u(s)}{\partial n}$ from the load vector and subtracting the value $p(s)u(s)$ from the components of the matrix that are on the boundary. Dirichlet boundary conditions are implemented by getting rid of the row and column for which u is known and adding the known value to the neighboring load vectors. In practice this can get quite complicated because the boundary is specified on every element that has one side on the edge of Ω. Moreover, the boundary conditions are usually different for different parts of the domain.

4.4 Example

In order to clarify some of these concepts we will look at an example problem. The domain, nodes and elements are shown in Figure 4. The domain of this problem is a reservoir with a constant pressure (Dirichlet) boundary condition along the $x = 0$ axis (Γ_{74}) to model influx from an aquifer, while no-flow (Neumann) boundaries exist everywhere else. The forcing term is a single production well in element 6 with constant pressure. This can also be thought of as a boundary condition. The PDE we are going to solve is

$$-\nabla \cdot k\left(\nabla u\left(x,y\right)\right) = f\left(x,y\right) \tag{60}$$

If we further assume that $k = 1$ is constant then this simplifies to the problem

$$\begin{array}{l} -\nabla^2 u\left(x,y\right) = f\left(x,y\right) \\ f\left(x,y\right) = \left\{ \begin{array}{cc} u_1 & \Omega_6 \\ 0 & \text{elsewhere} \end{array} \right\} \\ u\left(x,y\right) = 0 \;\; \text{on} \;\; \Gamma_{41} \\ \frac{\partial u(x,y)}{\partial n} = 0 \;\; \text{on} \;\; \Gamma_{74}, \Gamma_{12}, \Gamma_{25}, \Gamma_{56}, \Gamma_{67} \end{array} \tag{61}$$

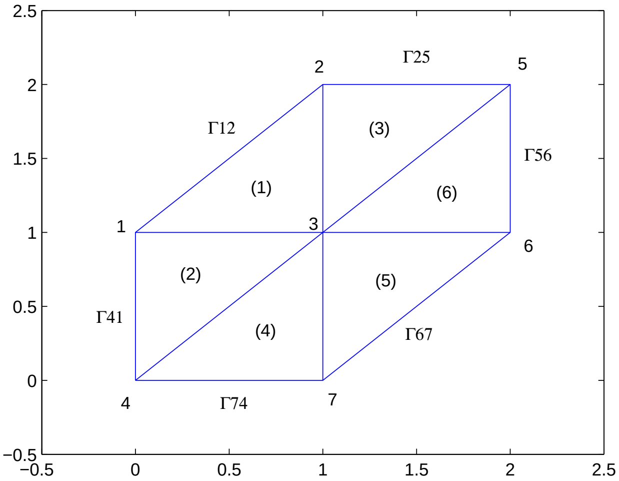

Figure 4: A six element domain with seven nodes.

The stiffness matrix and load vector for this example are given by:

$$K = \begin{bmatrix} K_{11} & K_{12} & K_{13} & K_{14} & 0 & 0 & 0 \\ K_{21} & K_{22} & K_{23} & 0 & K_{25} & 0 & 0 \\ K_{31} & K_{32} & K_{33} & K_{34} & K_{35} & K_{36} & K_{37} \\ K_{41} & 0 & K_{43} & K_{44} & 0 & 0 & K_{47} \\ 0 & K_{52} & K_{53} & 0 & K_{55} & K_{56} & 0 \\ 0 & 0 & K_{63} & 0 & K_{65} & K_{66} & K_{67} \\ 0 & 0 & K_{73} & K_{74} & 0 & K_{76} & K_{77} \end{bmatrix} F = \begin{bmatrix} 0 \\ 0 \\ f_1^6 \\ 0 \\ f_2^6 \\ f_3^6 \\ 0 \end{bmatrix} \tag{62}$$

when the boundary conditions are neglected. Each of the entries K_{ij} is the sum of the contributions of hat function to u at node i. Since node 3 is a member of every element, the row $K_{3,j}$ and the column $K_{i,3}$ are both filled.

Imposing the boundary condition $u_1 = u_4 = 0$ on Γ_{41} shrinks this to a 5×5 matrix problem

$$\begin{bmatrix} K_{22} & K_{23} & K_{25} & 0 & 0 \\ K_{32} & K_{33} & K_{35} & K_{36} & K_{37} \\ K_{52} & K_{53} & K_{55} & K_{56} & 0 \\ 0 & K_{63} & K_{63} & K_{66} & K_{67} \\ 0 & K_{73} & 0 & K_{76} & K_{77} \end{bmatrix} \begin{bmatrix} u_2 \\ u_3 \\ u_5 \\ u_6 \\ u_7 \end{bmatrix} = \begin{bmatrix} F_2 \\ F_3 \\ F_5 \\ F_6 \\ F_7 \end{bmatrix} \tag{63}$$

Since the remaining boundary conditions are no-flux they don't make any contribution to the load vector. If the boundaries were constant flux then elements 1, 2, 4, 5, 6, and 7 of the load vector would have to have the known flux subtracted off of them. This is analogous to the one-dimensional problem with Neumann boundary conditions.

The above matrix problem can be solved by finding $u = K^{-1}F$. In two dimensional problems this can be much harder to do than in 1D because the matrix has filled in and is no longer tri-diagonal. In order to get a sufficiently accurate solution, the size of the matrix K for 2D problems is also generally much larger than for a 1D problem.

References

[1] Becker,, E. B. G. F. Carey, and J. T. Oden, *Finite Elements an Introduction*, Texas Institute for Computational Mechanics, UT Austin, 1981.