Finite Element Method For the Impatient - A Quick Application

Todd K. Dupont

August 7, 2009

Introduction

The point of these notes is provide an example of using the finite element method (FEM) to move from a differential model problem to a linear-algebraic one, as computers are much better at solving the latter. There are actually several numerical methods that also perform a similar transformation (e.g., finite differences). A distinct feature of FEM is that the arrays within the global matrix problem, say $\mathbf{A} \cdot \vec{u} = \vec{b}$, are assembled from a *finite* set of N_{el} *elemental* subarrays: $\mathbf{A} = \sum_{e=1} N_{el} \mathbf{A}_e$ and $\vec{b} = \sum_{e=1}^{N_{el}} \vec{b}_e$. These elemental subarrays are found through operations on a finite set of area elements, or subdomains Ω_e, which in total constitute the model problem's total domain Ω. This breaking up of the domain generates of finite element mesh, and these meshes can be created for quite complicated domains, which is often considered a key advantage of finite element approaches or finite difference treatments.

The steps outlined in the following sections are effectively the same steps that any finite element application applies.

Model Problem

Find $u = u(x,y)$ on the domain Ω such that

$$\begin{aligned} \nabla^2 u - f &= 0, \quad \{x,y\} \in \Omega & (1) \\ u &= 0, \quad \{x,y\} \in \partial\Omega & (2) \end{aligned}$$

where $\nabla^2 \equiv \partial_x^2 + \partial_y^2$ is the Laplacian operator, $\partial\Omega$ is the boundary of the domain, and $f = f(x,y)$ is some sink term.

The model domain for this problem is the Ross Ice Shelf, which is shown, in the form of a 2-d FEM mesh, in figure 1.

Weighted-Residual, Variational, or Weak Form

Let $r(x,y) = \nabla^2 u - f$ be the residual of Poisson's equation. We would, of course, like for $r(x,y) = 0$. In FEM we move this problem to a weaker form, one where we insist that the residual be zero in a weighted-average sense, for some arbitrary weighting function $w = w(x,y)$. More specifically, we would like to find $u = u(x,y)$ such that $u(\partial\Omega) = 0$ and

$$\int_\Omega wrdA = \int_\Omega \left\{ w\nabla^2 u - wf \right\} dA \;\; = \;\; 0 \tag{3}$$

for all "permissible" w; effectively, w needs to be smooth enough to allow an integrable product with r. This problem is less stringent (weak), because the constraint is integrated, or smoothed, relative to the original problem. The variable w is often also called a test function.

Note that

$$w\nabla^2 u = w\nabla \cdot \left(\nabla^T\right) u = \nabla \cdot \left(w\nabla^T u\right) - (\nabla w) \cdot \left(\nabla^T u\right)$$

and

$$\int_\Omega \nabla \cdot \left(w\nabla^T u\right) dA = \int_{\partial\Omega} w\vec{n} \cdot \nabla^T u ds$$

where $\vec{n}$ is the outward normal on the boundary.

Thus we can rewrite the weak form of the problem as

$$\int_{\partial\Omega} w\vec{n} \cdot \nabla^T u ds - \int_\Omega \left\{ (\nabla w) \cdot \left(\nabla^T u\right) + wf \right\} dA = 0 \tag{4}$$

Because we prescribe u on the boundary, we don't want w to be arbitrary on the boundary, i.e., we are not "testing" or applying any constrain derived from Poisson's equation on the boundary. Thus $w(\partial\Omega) = 0$. This simplifies our weak form further to finding $u(x,y)$ on Ω such that $u(\partial\Omega) = 0$ and

$$\int_\Omega \left\{ (\nabla w) \cdot \left(\nabla^T u\right) + wf \right\} dA = \int_\Omega \left\{ \partial_x w \partial_x u + \partial_y w \partial_y u + wf \right\} = 0 \tag{5}$$

for arbitrary (but well-behaved) w in Ω, and $w(\partial\Omega) = 0$.

Descretization - the Galerkin Approximation

To this point u, w and f are continuous functions over Ω. To make the problem tractable we want to move from trying to find u everywhere within the domain to finding u at a specific set of points within this domain. This will turn the problem into into an linear-algebraic problem, something computers are good at solving.

$$\{x,y\} \in \Omega \quad \rightarrow \quad \{x_i,y_i\}, \quad i = 1,2,\ldots N$$

where $\{x_i,y_i\}$ is the ith point, and N is the number of points, or nodes.

We can denote the value of u (or w, or f) at the ith node as u_i (or w_i, or f_i). However, how do these functions vary between nodes? We can approximate this variation as a weighted sum over some finite set of basis (or interpolation) functions, such that

$$u(x,y) \approx \sum_{i=1}^{N} \phi_i(x,y)u_i = \vec{\phi} \cdot \vec{u} \tag{6}$$

$$w(x,y) \approx \sum_{i=1}^{N} \phi_i(x,y)w_i = \vec{\phi} \cdot \vec{w} \tag{7}$$

$$f(x,y) \approx \sum_{i=1}^{N} \phi_i(x,y)f_i = \vec{\phi} \cdot \vec{f} \tag{8}$$

where $\vec{\phi} = \begin{bmatrix} \phi_1(x,y) & \phi_2(x,y) & \ldots & \phi_N(x,y) \end{bmatrix}$ is the vector of N linearly-independent basis functions, and $\vec{u}$, $\vec{w}$ and $\vec{f}$ contain the N nodal values of u, w, and f, respectively. This approximate form, referred to as a Galerkin approximation, treats the variable of interest as a piece-wise continuous function, one whose variation between nodal points is represented as an interpolation given by the N functions ϕ_i. In this case we would like to insist that

$$\phi_i(x_j,y_j) = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases} \tag{9}$$

which will enforce $u(x_i,y_i) = u_i$, i.e., the interpolation returns the nodal values at the nodal points.

We can plug our interpolated approximations for u, w, and f into the integral found in the weak form of the model problem, yielding a new form of the problem:

Find $\vec{u}$ such that $\vec{u}(\partial_x\Omega) = 0$ and

$$\vec{w}^T \cdot \left\{ \int_\Omega \left(\partial_x \vec{\phi}^T \cdot \partial_x \vec{\phi} + \partial_y \vec{\phi}^T \cdot \partial_y \vec{\phi} \right) dA \cdot \vec{u} + \int_\Omega \vec{\phi}^T \cdot \vec{\phi} dA \cdot \vec{f} \right\} \tag{10}$$

where $\vec{w}$ is arbitrary, except that $\vec{w}(\partial\Omega) = 0$.

We can condense this problem by defining two arrays,

$$\mathbf{A} \equiv \int_\Omega \left(\partial_x \vec{\phi}^T \cdot \partial_x \vec{\phi} + \partial_y \vec{\phi}^T \cdot \partial_y \vec{\phi} \right) dA \tag{11}$$

$$\vec{b} \equiv -\int_\Omega \vec{\phi}^T \cdot \vec{\phi} dA \cdot \vec{f} \tag{12}$$

yielding

$$\vec{w}^T \left\{ \mathbf{A} \cdot \vec{u} - \vec{b} \right\} = 0 \tag{13}$$

Because the values of w_i are arbitrary (away from the boundary), and because we insist the basis functions are linearly independent, we are forced to satisfy equation (13) for each of the N cases of

$$\vec{w} = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \end{bmatrix} \tag{14}$$

$$\vec{w} = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \end{bmatrix} \tag{15}$$

$$\vec{w} = \begin{bmatrix} 0 & 0 & 1 & \dots & 0 \end{bmatrix} \tag{16}$$

$$\vec{w} = \begin{bmatrix} 0 & 0 & 0 & \dots & 1 \end{bmatrix} \tag{17}$$

Which means we are really looking for $\vec{u}$ such that $\vec{u}(\partial\Omega) = 0$ and

$$\mathbf{A} \cdot \vec{u} = \vec{b} \tag{18}$$

So we've recast the differential model problem into a linear algebra problem.

Finding the Arrays A and $\vec{b}$

Say the nodes are organized into a *mesh* of N_{el} triangular elements, where a particular set of three nodes serve as the vertices of a particular triangular area within the domain. [1]

[1] This is one of the simplest types of meshes used in FEM; meshes can be made of many elements with various shapes.

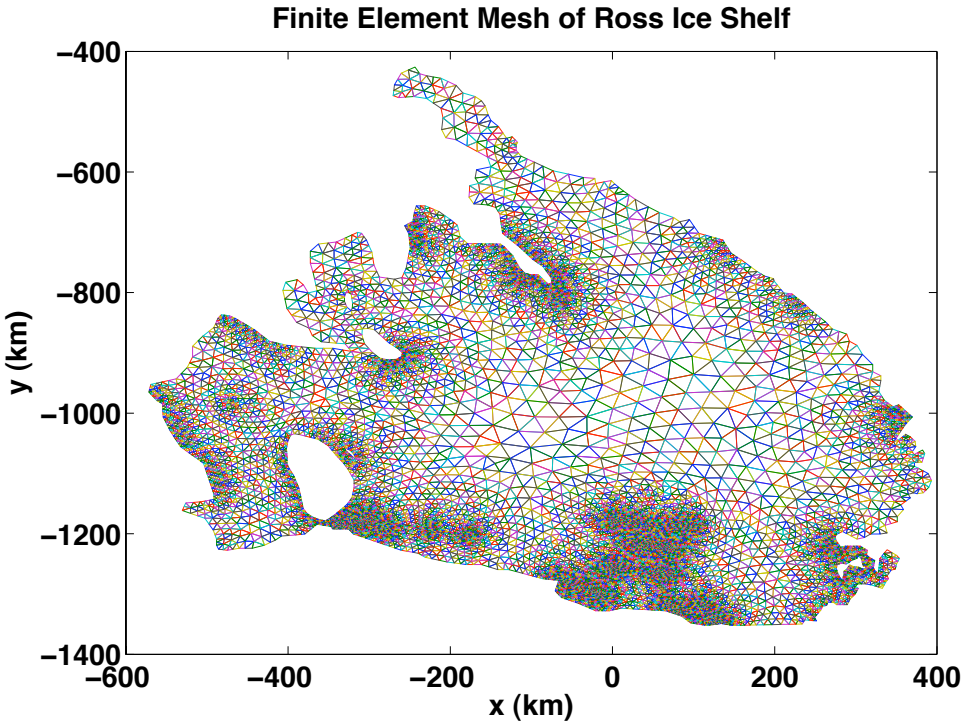

Figure 1: Ross Ice Shelf Finite Element Mesh

Given our domain Ω is now broken into N_{el} triangular subdomains we can break the integrals in the expressions for $\mathbf{A}$ and $\vec{b}$, such that

$$\mathbf{A} \equiv \sum_{e=1}^{N_{el}} \int_{\Omega_e} \left(\partial_x \vec{\phi}^T \cdot \partial_x \vec{\phi} + \partial_y \vec{\phi}^T \cdot \partial_y \vec{\phi} \right) dA \tag{19}$$

$$\vec{b} \equiv -\sum_{e=1}^{N_{el}} \int_{\Omega_e} \vec{\phi}^T \cdot \vec{\phi} dA \cdot \vec{f} \tag{20}$$

Only the basis functions interpolating between the points at the vertices of a particular element Ω_e are going to be nonzero within that element. Thus the integrals in the above expressions actually only act of that subset of three basis functions . This subset of basis functions that perform the interpolation within a given element are often called the shape functions of that element.

$$\vec{\phi}(\Omega_e) \rightarrow \vec{\psi}_e \tag{21}$$

which means the interpolation of (say) u in Ω_e can be written as

$$\vec{\psi}_e \cdot \vec{u}_e \tag{22}$$

where $\vec{u}$ are the values of u at the nodes defining Ω_e.

With this notation in hand we can re-write the expressions for $\mathbf{A}$ and $\vec{b}$ as

$$\mathbf{A} = \sum_e \int_{\Omega_e} \left\{ \partial_x \vec{\psi}_e^T \cdot \partial_x \vec{\psi}_e + \partial_y \vec{\psi}_e^T \cdot \partial_y \vec{\psi}_e \right\} dA = \sum_e \mathbf{A}_e \tag{23}$$

$$\vec{b} = -\sum_e \int_{\Omega_e} \vec{\psi}_e^T \cdot \vec{\psi}_e dA \cdot \vec{f}_e = \sum_e \vec{b}_e \tag{24}$$

where

$$\mathbf{A}_e \equiv \int_{\Omega_e} \left\{ \partial_x \vec{\psi}_e^T \cdot \partial_x \vec{\psi}_e + \partial_y \vec{\psi}_e^T \cdot \partial_y \vec{\psi}_e \right\} dA \tag{25}$$

$$\vec{b}_e \equiv -\int_{\Omega_e} \vec{\psi}_e^T \cdot \vec{\psi}_e dA \cdot \vec{f}_e \tag{26}$$

are the arrays that *sum together* to form the global arrays $\mathbf{A}$ and $\vec{b}$

To actually evaluate the integral expressions above requires that we specify the functional form of these $\vec{\psi}_e$.

The simplest shape functions on a triangular element are linear, such that

$$\vec{\psi}_e = \vec{\alpha}_e x + \vec{\beta}_e y + \vec{\gamma}_e \tag{27}$$

For the interpolation $\vec{\psi}_e \cdot \vec{u}_e = \sum_{i=1}^{3} \psi_{i,e} u_{i,e}$ to recover u_i at the i^{th} nodal position $\{x_{i,e}, y_{i,e}\}$ the following condition must be satisfied

$$\begin{bmatrix} x_{1,e} & y_{1,e} & 1 \\ x_{2,e} & y_{2,e} & 1 \\ x_{3,e} & y_{3,e} & 1 \end{bmatrix} \cdot \begin{bmatrix} \alpha_{1,e} & \alpha_{2,e} & \alpha_{3,e} \\ \beta_{1,e} & \beta_{2,e} & \beta_{3,e} \\ \gamma_{1,e} & \gamma_{2,e} & \gamma_{3,e} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{28}$$

or

$$\begin{bmatrix} \vec{x}_e & \vec{y}_e & 1 \end{bmatrix} \cdot \begin{bmatrix} \vec{\alpha}_e \\ \vec{\beta}_e \\ \vec{\gamma}_e \end{bmatrix} = \mathbf{I} \tag{29}$$

where $\mathbf{I}$ is the identity matrix.

Note that given equation (27) the derivatives in equaiton (25) are constants, i.e.,

$$\mathbf{A}_e = \int_{\Omega_e} \left\{ \vec{\alpha}_e^T \cdot \vec{\alpha}_e + \vec{\beta}_e^T \cdot \vec{\beta}_e \right\} dA = a_e \left\{ \vec{\alpha}_e^T \cdot \vec{\alpha}_e + \vec{\beta}_e^T \cdot \vec{\beta}_e \right\} \tag{30}$$

where a_e is the area of the eth element.

Similarly, knowing the coefficients in equation (27) allows us to integrate quadratic integrands that result from the product of two linear functions in equation (26).

Given the values that fill the subarrays $\mathbf{A}_e$ and $\vec{b}_e$, it is a simple matter to assembling the contributions of these into the global arrays $\mathbf{A}$ and $\vec{b}$, at which point we can solve the global matrix equation (18).