LECTURE 20

Finite Element Method : Origins

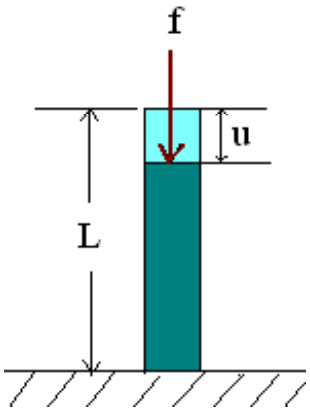
1. Step 0: Finite Element Analysis in Mechanical Engineering

The Finite Element Method has its origins in a certain numerical method for computing the deformations, strains and stresses of rigid bodies. In this setting, continuum methods, where the quantities to be computed are functions of position and time, and governed by PDEs and boundary conditions, are replaced by a reformulation of the problem by thinking of a rigid body as being composed of a finite number of simple elements and reformulating the PDE/BVP as a set of algebraic equations that can be solved by a computer.

Here is a prototypical example of this sort of analysis. Consider a load bearing column, say of length L and cross-sectional area A. Hooke's Law states that if a load f_s is placed on the column, then column will deform (shrink or expand) by

$$u = \frac{L}{AE} f$$

where E is Young's modulus of elasticity for the material the column is made of.



Let

$$k = \frac{L}{AE}$$

so that we can write

$$u = kf$$

Next consider the column rotated 90° and used a truss with loads are applied to both ends.

Here f_1 (resp. f_2) is the force applied to the left (resp. right) end of the beam and u_1 (resp. u_2) is the displacement of the left (resp. right) end of the beam. Interpreting $u = u_2 - u_1$ as the total deformation of the beam and applyin Newton's Third Law, we obtain

$$\begin{aligned} f_1 &= \frac{AE}{L}(u_1 - u_2) \\ f_2 &= \frac{AE}{L}(u_2 - u_1) \end{aligned}$$

Putting these equations in matrix form we have

$$\begin{bmatrix} f_1 \\ f_2 \end{bmatrix} = \frac{AE}{L}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}\begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

This matrix equation is called the truss *element stiffness equation*, and the symmetric matrix

$$\mathbf{K} = \frac{AE}{L}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$$

is called the *element stiffness matrix.*

Alas, the finite element method we shall develop for solving PDEs will look very little like the method used above for analyzing PDEs. Yet, this example remains at the origins of the method and so is deeply embodied in the nomenclature of the finite element method.

2. Step 2: Dynamical Systems

Next we'll consider a simple dynamical system governed by Newton's Second Law

$$\frac{d^2y}{dt^2} + ky = -g \quad , \quad 0 \le t \le T \tag{1}$$

$$y(0) = a \tag{20.1}$$

$$y(T) = b \tag{20.2}$$

(You can think of this example as a vibrating string in the presence of a gravitational field.) In this example, we reformulate Newton's Method in a manner that mimics the analysis of §1

The first step is to approximate the continuum problem by a discrete model. This we do by subdividining the time interval $[0, T]$ into N subintervals, separated by *nodes* t_i where

$$\begin{aligned} t_0 &= 0 \\ t_1 &= \frac{T}{N} \\ t_2 &= 2\frac{T}{N} \\ &\vdots \\ t_N &= N\frac{T}{N} = T \end{aligned}$$

and ascribing to each of these nodes a quantity y_i that is to represent the value of the solution at $t = t_i$.

Next, we set $h = \frac{T}{N}$ and use the approximate derivatives by considering the corresponding slopes between neighboring nodes

$$\begin{aligned} \frac{dy}{dt}(t_i) &\approx \frac{y_{i+1} - y_i}{h} := y_i' \\ \frac{d^2y}{dt^2}(t_i) &\approx \frac{y_i' - y_{i-1}'}{h} = \frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} \end{aligned} \tag{2}$$

Evaluating the continuum equation (1) that each node t_i and using the discrete approximations (2) we arrive at the following linear system for the quantities y_1

$$\begin{array}{rcll} y_0 & = & a & \\ \dfrac{y_{i+1}-2y_i+y_{i-1}}{h^2}+ky_i & = & -g & , \qquad i=1,\ldots,N-1 \\ y_N & = & b & \end{array}$$

or, as a matrix equation,

$$\begin{bmatrix} 1 & 0 & 0 & & & \\ 1 & -2+kh^2 & 1 & \ddots & & \\ 0 & 1 & -2+kh^2 & 1 & \ddots & \\ & \ddots & \ddots & \ddots & \ddots & 0 \\ & & 0 & 1 & -2+kh^2 & 1 \\ & & & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_{N-1} \\ y_N \end{bmatrix} = \begin{bmatrix} a \\ -gh^2 \\ -gh^2 \\ \vdots \\ -gh^2 \\ b \end{bmatrix} \tag{3}$$

This linear system is called the **finite difference** model of the original dynamical system. The matrix in (3) is called the *system matrix.*

3. Step 3: Energy Formulation and the Variational Finite Element Method

An alternative to Newton's Laws is the Principal of Least Action. This principle (due to Hamilton) is the following statement:

> A physical system always evolves from one configuration to another in such a way that the integral of the difference between the kinetic energy of the system and the potential energy of the system is minimized.

To see an explicit formulation of this principle, consider a single particle system where the kinetic energy is given by

$$K=\frac{1}{2}m\left(\frac{dx}{dt}\right)^2$$

and the potential energy is some prescribed function of the position x. Then if the system moves from point x_i to a point x_f over the time interval $[t_i,t_f]$, the principle of least action requires that the *action functional*

$$A\left(x\left(t\right)\right):=\int_{t_i}^{t_f}\left[\frac{1}{2}m\left(\frac{dx}{dt}\right)^2-V\left(x\left(t\right)\right)\right]dt \tag{4}$$

is minimized. Here we think the action functional as a map that takes an arbitrary C^1 function $x\left(t\right)$ on $[t_1,t_2]$ and produces some number (via the calculation of the integral on the right hand side of (4)).

Let me take a minute to demonstrate the equivalence of this principle with Newton's Second Law.

First off, consider a function of several variables $f\left(x_1,\ldots,x_n\right)$. If f has a local minimum at $\mathbf{x}=\mathbf{x}_0$, one must have $\boldsymbol{\nabla} f\left(\mathbf{x}_0\right)=\mathbf{0}$. Put another way, if f has a local minimum at $\mathbf{x}_0$ then all the first order variations of $f\left(\mathbf{x}_0+\boldsymbol{\delta}\right)$ about $\mathbf{x}_0$ must vanish. By this we mean, that if we Taylor expand f about $\mathbf{x}_0$ the terms linear in each component $\boldsymbol{\delta}$ must separately vanish:

$$\begin{array}{rcl} f\left(\mathbf{x}_0+\boldsymbol{\delta}\right) & = & f\left(\mathbf{x}_0\right)+\boldsymbol{\nabla} f\left(\mathbf{x}_0\right)\cdot\boldsymbol{\delta} \quad + \quad \text{terms quadratic and higher in the components of } \boldsymbol{\delta} \\ & = & f\left(\mathbf{x}_0\right)+\mathbf{0} \quad + \quad \text{terms quadratic and higher in the components of } \boldsymbol{\delta} \end{array}$$

We'll employ an analogous criterion for finding the extreme values of the functional A. Let $x(t)$ be an arbitary function on $[t_i, t_f]$ satisfying

$$x(t_i) = x_i \quad , \quad x(t_f) = x_f \tag{5}$$

We want to find a choice of $x(t)$ that minimizes the action functional A. We do so by setting up a small variation of the given path $x(t)$

$$x(t) \longrightarrow x_\delta(t) := x(t) + \delta(t)$$

with $\delta(t)$ satisfying

$$\delta(t_i) = 0 = \delta(t_f)$$

so that any variations $x_\delta(t)$ of continues to satisfy (5). Next, we consider how such a variation of $x(t)$ affects the action integral

$$\begin{aligned} A(x_\delta) &= \int_{t_i}^{t_f} \frac{1}{2} m \left(\frac{dx}{dt} + \frac{d\delta}{dt} \right)^2 - V(x(t) + \delta(t))\, dt \\ &= A(x) + \int_{t_i}^{t_f} \left(m \frac{dx}{dt} \frac{d\delta}{dt} - \frac{\partial V}{\partial x} \delta \right) dt + \text{terms of order } \delta^2 \end{aligned}$$

The first order variation of A is thus

$$\begin{aligned} A(x_\delta) - A(x) &\approx \int_{t_i}^{t_f} \left(m \frac{dx}{dt} \frac{d\delta}{dt} - \frac{\partial V}{\partial x} \delta \right) dt \\ &= m \frac{dx}{dt} \delta(t) \Big|_{t_i}^{t_f} - \int_{t_i}^{t_f} \left(m \frac{d^2x}{dt^2} \delta(t) + \frac{\partial V}{\partial x} \delta(x) \right) dt \\ &= -\int_{t_i}^{t_f} \left(m \frac{d^2x}{dt^2} + \frac{\partial V}{\partial x} \right) \delta(x)\, dt \end{aligned}$$

When $x(t)$ is a function that minimizes A, the corresponding first order variations should all vanish. This requires

$$m \frac{d^2x}{dt^2} = -\frac{\partial V}{\partial x}$$

which is just Newton's Second Law (once one identifies the $F(x) = -\frac{\partial V}{\partial x}$).

It turns out that the Principle of Least Action covers not only the dynamics of single particle systems, but continuum systems (such as electromagnetic fields, fluids, etc). Because of this, one typically has two equivalent formulations of a physical problem, a differential formulation in terms of PDEs and boundary conditions, and a variational formulation where some integral of the physical solution is be minimal. The physical equivalence of the two formulations then provides one with an alternative method for discovering the solution of a PDE/BVP.

The Rayleigh-Ritz Method, which is the most direct precursor of the modern finite element method, is a simple algorithm for finding such a minimizing function. Suppose the "solution" to a PDE/BVP is a function that also minimizes a particular *energy functional* $H(u)$. In the Raleigh-Ritz method, one tries to represent the actual solution as a linear combination of some sufficiently complete set of functions $\phi_i(x)$. Thus one sets

$$u(x) = \sum_{i=1}^{N} a_i \phi_i(x) \tag{6}$$

and defines

$$h(a_1, \ldots, a_n) = H\left(\sum_{i=1}^{N} a_i \phi_i \right)$$

and then tries to minimize the function $h(a_1, \ldots, a_n)$ as a function of n variables. This leads to a set of n conditions

$$\frac{\partial h}{\partial a_i} = 0$$

which can then be solved algebraically to yield an expression for minimizing function $u(x)$.

A couple of remarks should be made here. We've used general ansatzs like (6) before to solve PDEs; c.f. our use of Fourier series and Sturm-Liouville functions to expand solutions. In these situations we generally choose the functions ϕ_i because they had something to do with PDE we were trying to solve (e.g. the ϕ_i used were connected with solutions obtained by Separation of Variables) In the Raleigh-Ritz Method, the choice of the "basis functions" is usually dictated more by the geometry of the domain than the PDE itself. For in this formulation, the energy functional H is always a particular integral of functions over the entire solution domain. Because of this, one of the most important properties of a "good" choice of test functions ϕ_i is that the energy integral the defines H is readily evaluated over the solution domain when one inserts a linear combination of the test functions.

For this reason, it is generally **not** desirable to use *test functions* $\phi_i(x)$ that look anything like an actual solution. What one wants ultimately is a relatively small set of functions (to keep the number of equations to solve relatively small) that are nevertheless sufficient to get provide a good approximation to the values of the actual solution In particular, even if one expects in the end a smooth analytic solution to a PDE/BVP, one very well might use non-smooth functions $\phi_i(x)$ to approximate the solution.

4. Weak Formulation and Galerkin Finite Element Method

Suppose you want to solve a PDE of the form

$$L[u(x)] = f(x) \quad , \quad \forall\, x \in D \tag{7}$$

where L is some linear differential operator. The Raleigh-Ritz Method described above does not solve this PDE directly, but rather finds approximate solutions to an equivalent variational problem.

The (Galerkin) Finite Element Method is based on yet another equivalent formulation of the problem.

Suppose (7) is true. Then if we multiply both sides of (7) by a function $\psi(x)$ and integrate over the domain D we get

$$\int_D L[v(x)]\,\psi(x)\,dx = \int_D f(x)\,\psi(x)\,dx$$

This should be true for any function $\psi(x)$.

Suppose now we have a family of functions $\{\phi_i\}_{i=1,\ldots n}$ such that

(i) integrals of the form
$$\int_D \phi_i(x)\,\psi_j(x)\,dx$$
are readily computed.

(ii) linear combinations of the form
$$\sum a_i \phi_i(x)$$
are capable of approximating an arbitrary function to arbitrarily high degree of accuracy.

In this situation we can then construct an approximate solution of (7) by expressing it as

$$u(x) = \sum a_i \phi_i(x)$$

and trying to solve the equations

$$\int_D L\left[\sum_{i=1}^{n} a_i \phi_i(x)\right] \phi_j(x)\,dx = \int_D f(x)\,\phi_j(x)\,dx \qquad , \qquad j = 1, \ldots, n \tag{8}$$

for the coefficients a_i. Note that if L is a linear differential operator then the equations for the coefficients a_i will just be form an $n \times n$ linear system.

Now $n \times n$ linear systems are readily solved, although in general the difficulty in computing solutions by goes like $n!$. So another criterion for the functions ϕ_j that comes in play once one considers an efficient implementation of this idea is that the corresponding linear system be amenable to rapid solution. One cheap way to do this, which we shall see is also helpful in satisfying (i) and (ii) is require

(iii) the support of any particular ϕ_i is limited to just a few adjacent *finite elements* of the domain D.

By *finite element* we simply mean a particular element of a decomposition of the domain D into small, regularly shaped subdomains (like line segments, triangles, tetrahedra, etc).

Limiting the support of the functions ϕ_i to just a few finite elements, will, on the one hand, if the finite elements are small enough, allow us to satisfy (ii) by using simply linear or polynomial approximations on each finite element D_j. With the support of the ϕ_i limited to small regularly shaped finite element domains D_j, the integrals of both sides of (8) can be readily computed; and moreover, since only a few of the functions ϕ_i will be supported on any particular domain, for any particular j, the only coefficients that will apppear in the j^{th} equation of (8) will be limited to those coefficients whose corresponding function ϕ_i have supports with non-empty intersection with the support of ϕ_j.

5. Implementing the Finite Element Method

In both the Raleigh-Ritz Method and the Finite Element Method the key step is to fix a decomposition of the domain D of the problem into a set of finite elements D_i and to attach to these domain elements certain *shape functions* ϕ_i.

For a planar domain this can be done as follows.

- Choose a regular grid of points $\{\mathbf{v}_i = (x_i, y_i)\}$ in the domain D (including points on the boundary ∂D).
- Draw lines between points and their nearest neighbors. This will produce a decomposition of D intoa finite set of trianglular subdomains D_α. This decomposition is called a *triangularization* of D and the corresponding triagular subdomains will be the *finite elements* of our procedure. The set of vertices from which this triangularization was produced will be referred as the *nodes* of our implementation.
- To each node $\mathbf{v}_i$, we shall define a corresponding shape function. This will a piecewise linear function (in this implementation) satisfying

$$\begin{aligned} \phi_i(x,y) &= a_{\alpha,i}x + b_{\alpha,i}y + c_{\alpha,i} \qquad \text{if } (x,y) \in D_\alpha \text{ and } \mathbf{v}_i \text{ a vertex of } D_\alpha \\ \phi_i(x,y) &= 0 \qquad \text{if } (x,y) \in D_\beta \text{ and } \mathbf{v}_i \text{ is not a vertex of } D_\beta \\ \phi_i(\mathbf{v}_j) &= \delta_{ij} \end{aligned}$$

Note that since each of the triangular domains D_α will contain three vertices the third condition will give us three equations to determine the coefficients $a_{\alpha,i}$, $b_{\alpha,i}$, $c_{\alpha,i}$ needed to specify the values of ϕ_i on D_α.

Note also that the functions ϕ_i are non-zero only on the triangular domains the touch the vertex $\mathbf{v}_i$.

Finally, we note that if the we construct a general linear combination of the ϕ_i

$$u(x,y) = \sum_{i=1}^{n} a_i \phi_i(x,y) \tag{10}$$

Then on any particular domain D_α, $u(x,y)$ will restrict to a linear function of x, y; in fact,

$$u(x,y)|_{D_\alpha} = \sum_{\mathbf{v}_i \in D_\alpha} a_i \phi_i(x,y)$$

that is to say $u|_{D_\alpha}$ will only involve the shape functions corresponding to vertices of D_α. It is easy to see that these vertex functions will be linearly independent. Since the space of linear functions in two variables is 3-dimensional, is is clear that by adjusting the coefficients a_i we can reproduce any linear functional on D_α. Thus, the expansion (10) represents a general approximating function that is a piecewise linear function throughout D that is precisely linear on each triangular domain D_α.