Lecture 22

- The finite element method

The finite element method:

The finite element method for elliptic PDEs is the generalization to higher dimensions of the Rayleigh-Ritz type of methods we discussed for two-point boundary value problems.

The details however, are significantly different (and important).

Finite elements is a whole industry that has a strong constituency in engineering circles. The development of the method and its great strengths were responsible for people being able to compute things in the 50's and 60's (like structures of airplanes) using computers with less computing power than that in your current wristwatch.

As such, it is a highly developed field. You can find whole bookcases with books on the subject in the library. As a consequence, in this course we can only present a small glimpse into this vast universe.

The discussion of finite elements can be done at various levels, some quite mathematically formal. To the converted, everything in the world is a finite element (including the finite difference method). We will take a more pedestrian approach as a first introduction.

At the center of the finite element method is the use of non-uniform, highly irregular, grids. This has enormous advantages when one has boundary problems in which the boundary conditions are specified on irregular or complicated boundaries (for instance the body of an airplane with boundary conditions on the windows, doors, wing and tail attachments).

Handling such a problem with finite differences and square grids is tantamount to “squaring the circle”. Suppose you are trying to give Neumann data on an oval and treating the problem with a square grid. You will have to write special conditions for all the derivatives that appear in all possible situations. Imagine the problem in more than two dimensions!

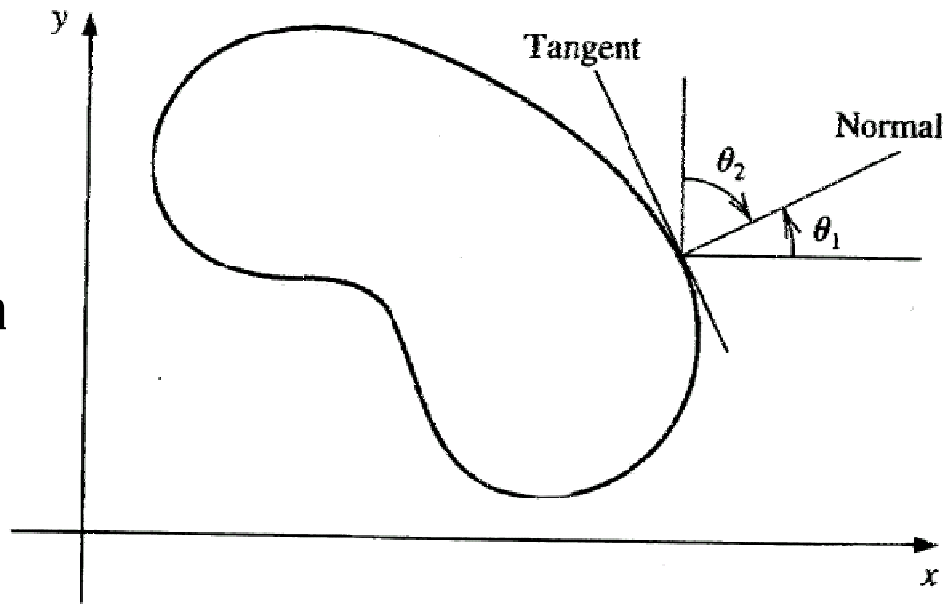
To make things concrete let us start by considering a specific equation,

$$\frac{\partial}{\partial x}\left(p(x,y)\frac{\partial u}{\partial x}\right)+\frac{\partial}{\partial y}\left(q(x,y)\frac{\partial u}{\partial y}\right)+r(x,y)u(x,y)=f(x,y),$$

In a domain with a boundary which is the union of two curves. On one of them, we specify Dirichlet data, u(x,y)=g(x,y). On the other curve, we give a more complicated boundary condition,

$$p(x,y)\frac{\partial u}{\partial x}(x,y)\cos\theta_1+q(x,y)\frac{\partial u}{\partial y}(x,y)\cos\theta_2+g_1(x,y)u(x,y)=g_2(x,y),$$

Where θ are the direction angles of the outward normal to the boundary. This condition is the kind of condition one encounters in Neumann specified problems in more than one dimension.

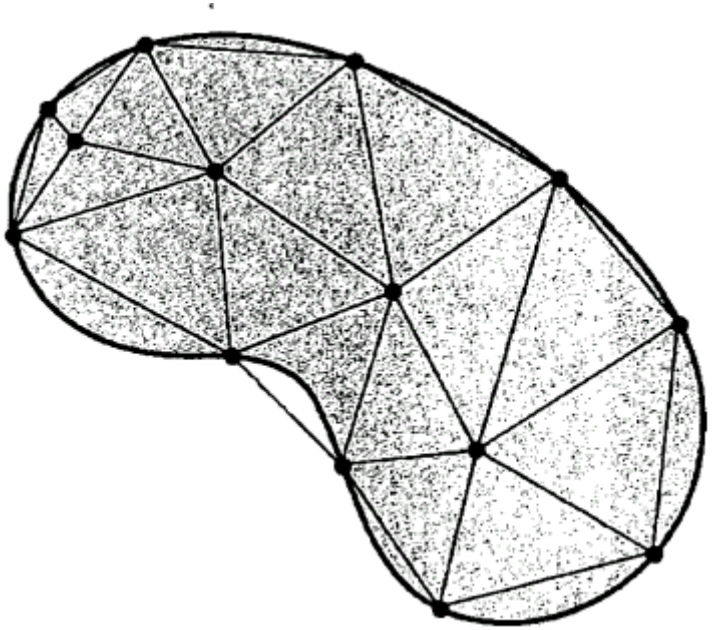


As we did in the case of Rayleigh-Ritz, if we assume that the functions p,q,r are continuous and if p,q have continuous first partial derivatives and p(x,y), q(x,y) are positive and r(x,y) is negative or zero and g_1 is positive, then a solution to the equation minimizes the functional,

$$I[w] = \int\int_{\mathcal{D}} \left\{ \frac{1}{2}\left[p(x,y)\left(\frac{\partial w}{\partial x}\right)^2 + q(x,y)\left(\frac{\partial w}{\partial y}\right)^2 - r(x,y)w^2 \right] + f(x,y)w \right\} dx\,dy$$
$$+ \int_{S_2} \{-g_2(x,y)w + \frac{1}{2}g_1(x,y)w^2\}\,dS$$

The strategy will be to minimize the functional in a finite dimensional subspace of functions, called just like we did in the Rayleigh-Ritz method.

To construct this basis of functions we will assume we subdivide the region of interest into a finite number of “elements”, for instance squares or triangles.



A common choice for the functionals is to take them to be linear,

$$\phi(x, y) = a + bx + cy,$$

We seek an approximation of the form,

$$\phi(x, y) = \sum_{i=1}^{m} \gamma_i \phi_i(x, y),$$

Where the $\phi_i(x,y)$'s are linearly independent piecewise-linear polynomials. The γ's are constants. Part of them we use to ensure the Dirichlet part of the boundary conditions are satisfied and we will use the rest to minimize the functional.

$$I[\phi] = I\left[\sum_{i=1}^{m} \gamma_i \phi_i\right] = \iint_{\mathcal{D}} \left(\frac{1}{2} \left\{ p(x, y) \left[\sum_{i-1}^{m} \gamma_i \frac{\partial \phi_i}{\partial x}(x, y) \right]^2 + q(x, y) \left[\sum_{i=1}^{m} \gamma_i \frac{\partial \phi_i}{\partial y}(x, y) \right]^2 \right.\right.$$

$$\left.\left. - r(x, y) \left[\sum_{i=1}^{m} \gamma_i \phi_i(x, y) \right]^2 \right\} + f(x, y) \sum_{i=1}^{m} \gamma_i \phi_i(x, y) \right) dy\, dx$$

$$+ \int_{S_2} \left\{ - g_2(x, y) \sum_{i=1}^{m} \gamma_i \phi_i(x, y) + \frac{1}{2} g_1(x, y) \left[\sum_{i=1}^{m} \gamma_i \phi_i(x, y) \right]^2 \right\} dS.$$

We assume n+1….m go for the boundary conditions and are therefore left with the following conditions for minimization,

$$\frac{\partial I}{\partial \gamma_j} = 0, \quad \text{for each } j = 1, 2, \ldots, n.$$

Or, expanding the derivatives out explicitly,

$$0 = \sum_{i=1}^{m} \left[\iint_{\mathcal{D}} \left\{ p(x, y) \frac{\partial \phi_i}{\partial x}(x, y) \frac{\partial \phi_j}{\partial x}(x, y) + q(x, y) \frac{\partial \phi_i}{\partial y}(x, y) \frac{\partial \phi_j}{\partial y}(x, y) \right.\right.$$

$$\left.\left. - r(x, y) \phi_i(x, y) \phi_j(x, y) \right\} dx\, dy + \int_{S_2} g_1(x, y) \phi_i(x, y) \phi_j(x, y)\, dS \right] \gamma_i$$

$$+ \iint_{\mathcal{D}} f(x, y) \phi_j(x, y)\, dx\, dy - \int_{S_2} g_2(x, y) \phi_j(x, y)\, dS,$$

Which we can rewrite as a linear system of equations Ax=b with,

$$\alpha_{ij} = \iint_{\mathcal{D}} \left[p(x,y) \frac{\partial \phi_i}{\partial x}(x,y) \frac{\partial \phi_j}{\partial x}(x,y) + q(x,y) \frac{\partial \phi_i}{\partial y}(x,y) \frac{\partial \phi_j}{\partial y}(x,y) - r(x,y) \phi_i(x,y) \phi_j(x,y) \right] dx\, dy + \int_{S_2} g_1(x,y) \phi_i(x,y) \phi_j(x,y)\, dS,$$

$$\beta_i = -\iint_{\mathcal{D}} f(x,y) \phi_i(x,y)\, dx\, dy + \int_{S_2} g_2(x,y) \phi_i(x,y)\, dS - \sum_{k=n+1}^{m} \alpha_{ik} \gamma_k,$$

We now need to get down to details, since the particular choice of basis of functions will determine the fate of the system of equations we will end up with. We assume the domain D is triangulated with triangles with three vertices labeled,

$$V_j^{(i)} = \left(x_j^{(i)}, y_j^{(i)} \right), \quad \text{for } j = 1, 2, 3.$$

With each vertex we associate a linear polynomial,

$$N_j^{(i)} \equiv N_j = a_j + b_j x + c_j y, \quad \text{where} \quad N_j^{(i)}(x_k, y_k) = \begin{cases} 1, & \text{if } j = k, \\ 0, & \text{if } j \neq k. \end{cases}$$

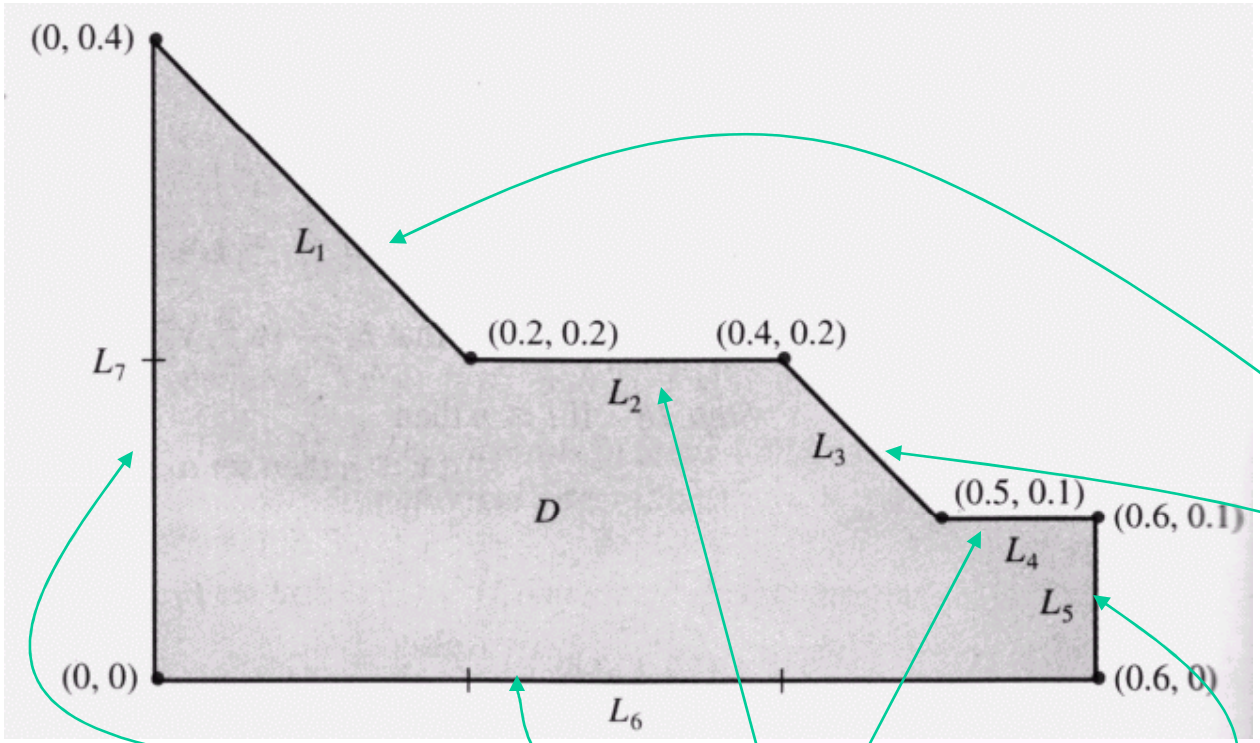
That is, the polynomial is one at the vertex in question and zero at all others in the triangle. That determines the coefficients,

$$\begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{bmatrix} \begin{bmatrix} a_j \\ b_j \\ c_j \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix},$$

If we label the nodes from 1 to n, then ϕ_i will coincide with $N^{(i)}$ when evaluated on the triangle i.

We will therefore have a ϕ_i associated with each node. The solution to the problem will be given, within each triangle by a linear combination of the ϕ_i's associated with that triangle. The coefficients will be the solution of the linear system of equations we discussed before.

To see how the method works, let us consider the solution of the Laplace equation on the domain,



This could correspond to a metal plate with fixed temperatures and heat exchange rates.

With boundary conditions,

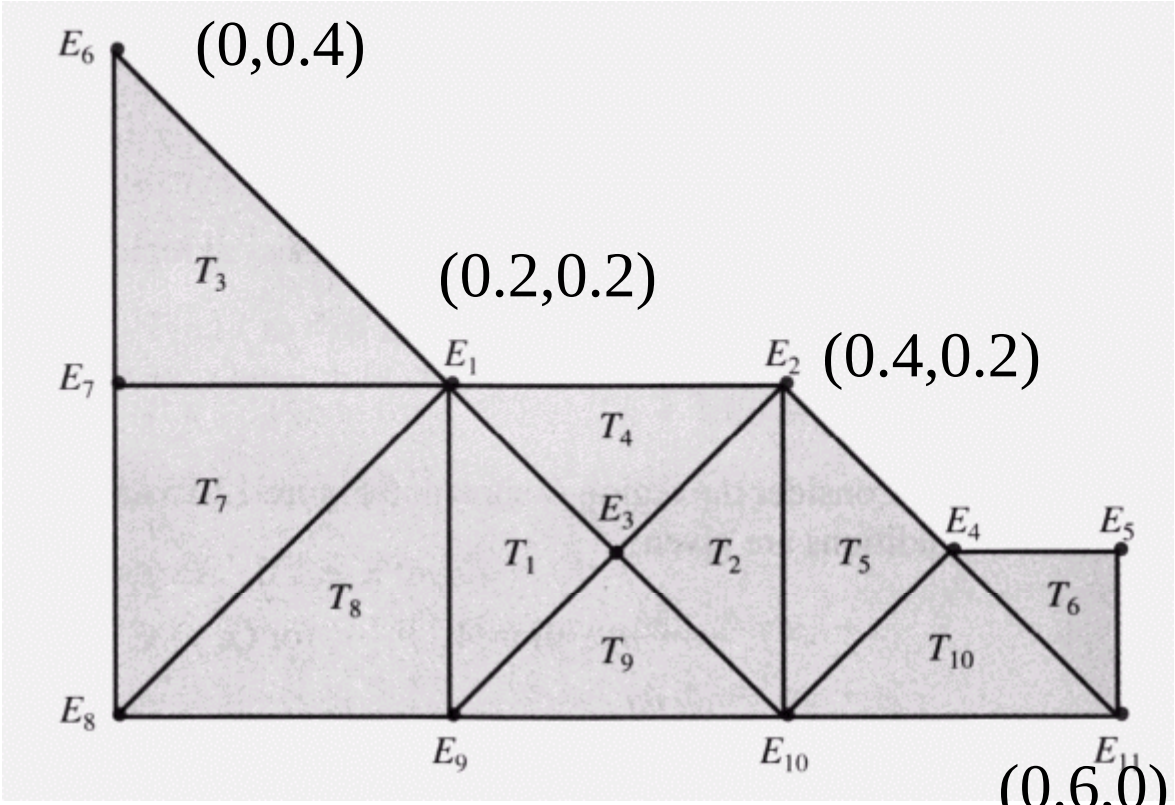
$$u(x, y) = 4, \quad \text{for } (x, y) \in L_6 \text{ and } (x, y) \in L_7,$$

$$\frac{\partial u}{\partial n}(x, y) = x, \quad \text{for } (x, y) \in L_2 \text{ and } (x, y) \in L_4,$$

$$\frac{\partial u}{\partial n}(x, y) = y, \quad \text{for } (x, y) \in L_5,$$

$$\frac{\partial u}{\partial n}(x, y) = \frac{x + y}{\sqrt{2}}, \quad \text{for } (x, y) \in L_1 \text{ and } (x, y) \in L_3,$$

We start by triangulating the region and labeling,



Next we compute the basis functions. Let us look at some examples,

On T_4,
$\phi_1 = (1-5x+5y)$,
$\phi_2 = (-2+5x+5x)$,
$\phi_3 = (2-10y)$,
On T_1,
$\phi_1 = (1-5x+5y)$,
$\phi_3 = (-2+10x)$

Boundary condition u(x,y)=4 on bottom and left sides implies,

$\gamma_k = 4$, on k = 6,7,8,9,10,11

We now need to construct the matrices a_{ij} i,j=1…5. When computing the required integrals, bear in mind that the phi's that are defined on more than one triangle have to be integrated appropriately.

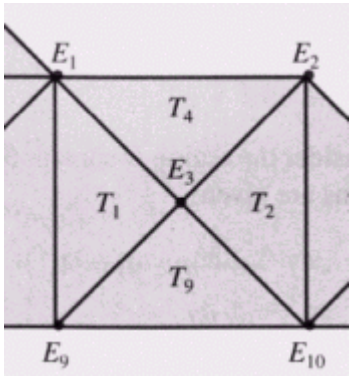
The integrals yield,

$$A = \begin{bmatrix} 2.5 & 0 & -1 & 0 & 0 \\ 0 & 1.5 & -1 & -0.5 & 0 \\ -1 & -1 & 4 & 0 & 0 \\ 0 & -0.5 & 0 & 2.5 & -0.5 \\ 0 & 0 & 0 & -0.5 & 1 \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} 6.066\bar{6} \\ 0.063\bar{3} \\ 8.0000 \\ 6.056\bar{6} \\ 2.031\bar{6} \end{bmatrix}$$

For instance, A_{13},

$$\alpha_{ij} = \iint_{\mathcal{D}} \left[p(x,y)\frac{\partial \phi_i}{\partial x}(x,y)\frac{\partial \phi_j}{\partial x}(x,y) + q(x,y)\frac{\partial \phi_i}{\partial y}(x,y)\frac{\partial \phi_j}{\partial y}(x,y) - r(x,y)\phi_i(x,y)\phi_j(x,y) \right] dx\,dy + \int_{S_2} g_1(x,y)\phi_i(x,y)\phi_j(x,y)\,dS,$$

On T_4,
ϕ_1= (1-5x+5y),
ϕ_2= (-2+5x+5x),
ϕ_3= (2-10y),
On T_1,
ϕ_1= (1-5x+5y),
ϕ_3= (-2+10x)



On T_4,
$\phi_{1,y}$= 5,
$\phi_{3,y}$= -10,
On T_1,
$\phi_{1,x}$= -5,
$\phi_{3,x}$= 10,

Each triangle has area 0.01, so the integral yields (-50)(0.01)+(-50)(0.01)=-1

Solving Ac=b, one gets,

$$\mathbf{c} = \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \gamma_3 \\ \gamma_4 \\ \gamma_5 \end{bmatrix} = \begin{bmatrix} 4.0383 \\ 4.0782 \\ 4.0291 \\ 4.0496 \\ 4.0565 \end{bmatrix}$$

The solution is obtained in each element by superposing the appropriate ϕ_i's belonging to that element with the corresponding c_i's,

So the final solution reads,

$$\begin{aligned}
T_1 &: \quad \phi(x, y) = 4.0383(1 - 5x + 5y) + 4.0291(-2 + 10x) + 4(2 - 5x - 5y), \\
T_2 &: \quad \phi(x, y) = 4.0782(-2 + 5x + 5y) + 4.0291(4 - 10x) + 4(-1 + 5x - 5y), \\
T_3 &: \quad \phi(x, y) = 4(-1 + 5y) + 4(2 - 5x - 5y) + 4.0383(5x), \\
T_4 &: \quad \phi(x, y) = 4.0383(1 - 5x + 5y) + 4.0782(-2 + 5x + 5y) + 4.0291(2 - 10y), \\
T_5 &: \quad \phi(x, y) = 4.0782(2 - 5x + 5y) + 4.0496(-4 + 10x) + 4(3 - 5x - 5y), \\
T_6 &: \quad \phi(x, y) = 4.0496(6 - 10x) + 4.0565(-6 + 10x + 10y) + 4(1 - 10y), \\
T_7 &: \quad \phi(x, y) = 4(-5x + 5y) + 4.0383(5x) + 4(1 - 5y), \\
T_8 &: \quad \phi(x, y) = 4.0383(5y) + 4(1 - 5x) + 4(5x - 5y), \\
T_9 &: \quad \phi(x, y) = 4.0291(10y) + 4(2 - 5x - 5y) + 4(-1 + 5x - 5y), \\
T_{10} &: \quad \phi(x, y) = 4.0496(10y) + 4(3 - 5x - 5y) + 4(-2 + 5x - 5y).
\end{aligned}$$

(the exact solution of the problem is xy+4).

Error estimates are hard to prove rigorously for these methods, since the choice of the elements can conspire with properties of the function. For instance, triangular or square elements with piecewise linear functions yield errors $O(h^2)$ where h is the maximum dimension of the element, but more complex elements can be constructed to yield $O(h^4)$ on functions of certain properties.

As you saw, the solution is obtained within each triangle by interpolating the functions chosen for the basis. Most finite element routines make this process transparent to the user, prompting for the function at any given x,y.

Some free finite element packages:

http://felt.sourceforge.net/ http:/ / netlib.org/ pltmg/

Welcome to the WWW home page for FElt. FElt is an open source system for finite element analysis; this document provides an overview of the features in that system.

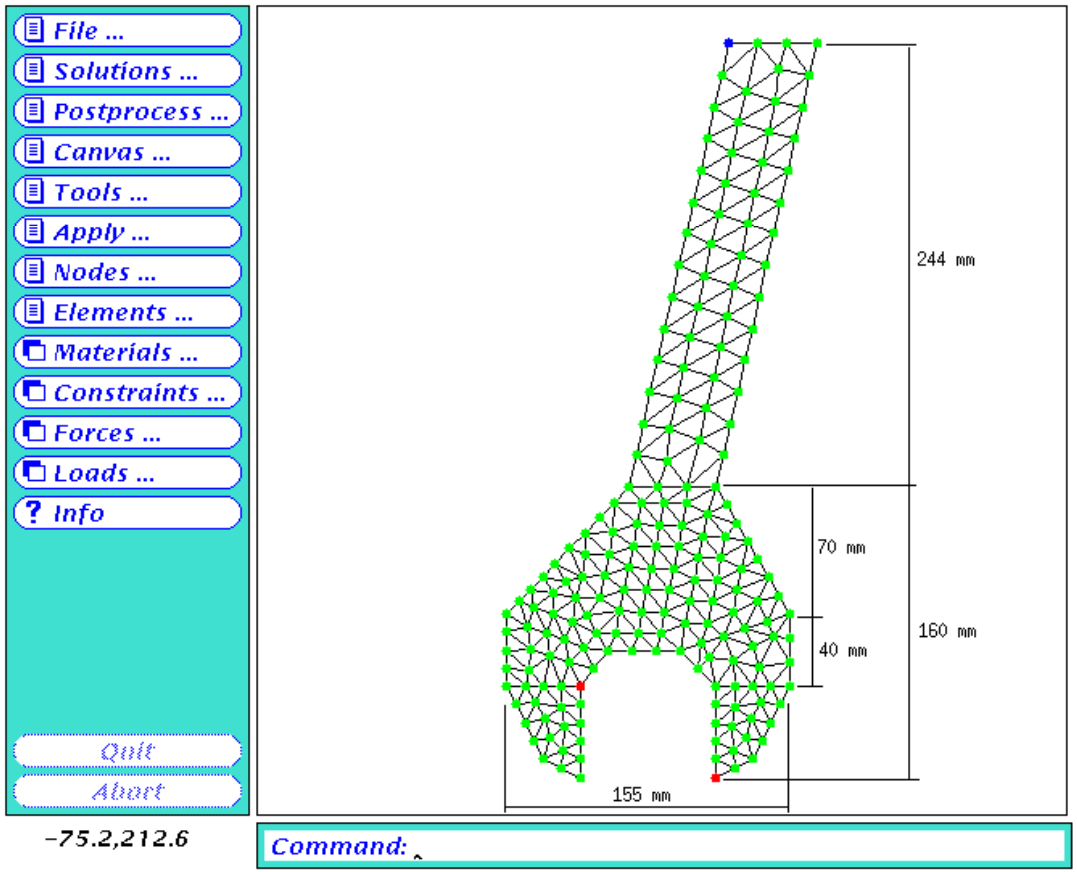
- General overview of the FElt system components.
- Example problems that demonstrate what kinds of problems FElt can solve, how FElt can help you set them up, and what kinds of output and post-processing options are available.
- Traditional documentation for FElt, including a FAQ, README's, man pages, and our comprehensive User's Guide.
- The SourceForge project page gives you access to the project administration (mailing lists, file releases, etc.).
- File releases on SourceForce is usually the fastest way to download the latest versions.
- Or use FTP if you want to download older versions or binaries that haven't been posted as a file release.

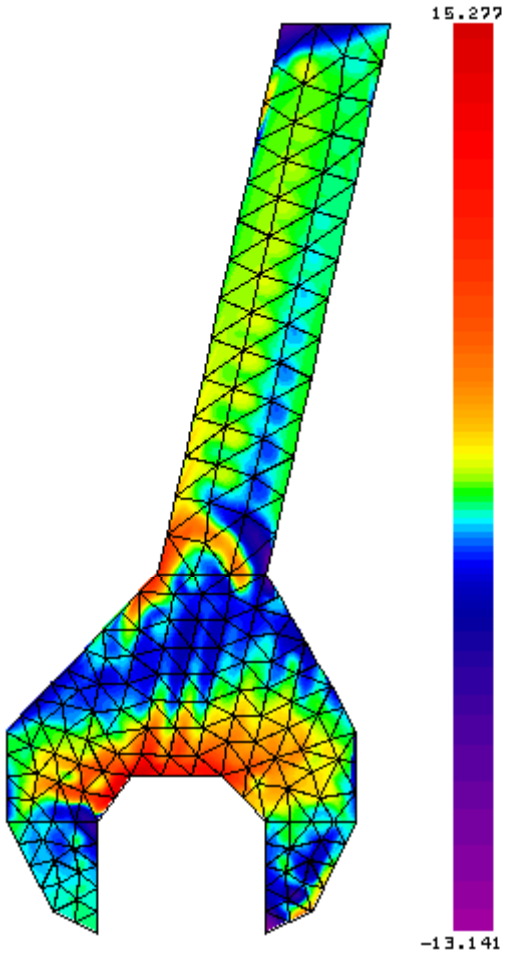
General Overview

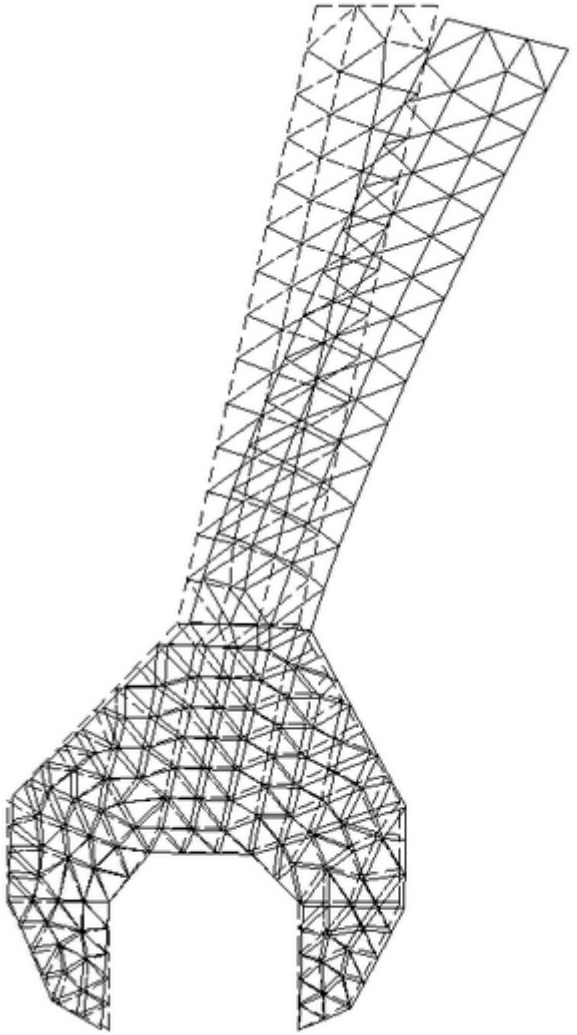
The current version of FElt knows how to solve linear static and dynamic structural and thermal analysis problems; it can also do modal and spectral analysis for dynamic problems. FElt's element library currently contains fourteen elements. The FAQ contains some additional information about what what kinds of problems FElt can solve as well as some information about expandability if you think you'd like to hack on FElt for your own purposes. A user can access FElt's capabilities through several different interfaces. The three most important are felt, the basic command-line application for solving FE problems given a standard FElt input file, burlap, our powerful, interactive, scripting environment that combines the flexibility of Matlab-like programs with FElt's finite element know-how and velvet, the full-featured CAD like pre- and post-processor that uses the X Window System for a graphical environment. All three applications use an intuitive, ASCII based syntax for problem definition. This powerful syntax allows you to substitute analytic functions in place of numeric values (sin(60) instead of 0.866025) and even more importantly allows for time-dependent forcing and boundary conditions to be specified as analytic functions of time or in the more traditional fashion as a series of discrete time, magnitude pairs. This feature makes it quite easy to specify a wide range of functions.

Example: Wrench

```
start-node    = 1        /* unnecessary as this is the default */
start-element = 1

triangular mesh
element-type = CSTPlaneStress
angtol = 20    dmin = 0.5     min = 100          /* except for min and max */
angspc = 30    kappa = 0.25   max = 300          /* these are defaults     */

boundary = [
    (0,50)
    (20,10)
    (40,0)
    (40,50)
    (55,70)
    (95,70)
    (115,50)
    (115,0)
    (135,10)
    (155,50)
    (155,90)
    (115,160)
    (170,404)
    (122,404)
    (67,160)
    (0,90)
]

end
```




15.277
-13.141



Summary

- Finite elements are the multidimensional generalization of Rayleigh-Ritz.
- The details are many…
- But the field is highly developed with many elaborate packages that handle the details.