Programming Finite Element Method

Earth 409: Earth System Modeling Course Project

By: **Prithvi Thakur**

Date: December 14th, 2018

1 Introduction

Most partial differential equations with more than one variable usually can not be solved analytically. We learnt finite difference methods in the course where we approximate the derivatives of the solution as a Taylor series expansion of space and time discretizations. This method transforms the partial differential equation into a system of algebraic equations which can be solved easily.

While the finite difference methods are easier to implement, they do not work well for the cases with complex geometry. Some examples would include systems with complicated boundary conditions which can not be represented analytically, but can be represented as a piece-wise continuous function. Another example where finite element methods would outperform finite differences is when we have variable material properties (rigidity, bulk modulus, density, etc) in space.

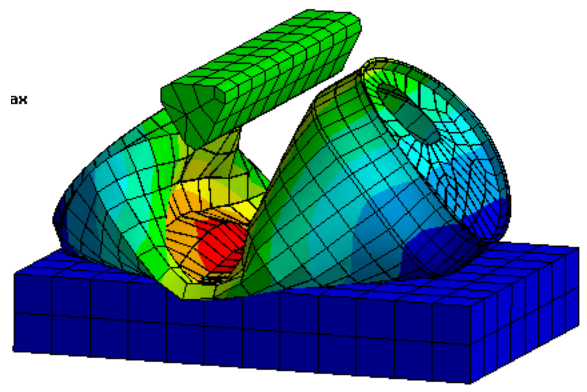

Figure 1: Stress contours of a deformed can[1]. Such a complex geometry is difficult to implement in finite difference or pseudospctral methods.

In this project, I explain the concept and implementation behind the finite element methods in one dimension. We will work with a simple one dimensional Navier-Stokes equation, which is useful in the applications of linear elasticity, e.g., stresses on a rod. I will then show an example using python's finite element solver, fenics, for a two dimensional static linear elasticity problem.

2 Linear Elasticity: Navier-Stokes in one dimension

Consider a rod of finite length made of linearly elastic material. We consider the rod to be in steady state, i.e., a load was applied slowly enough so that there is no acceleration anywhere on the rod, and it was applied sufficiently long ago so that all the vibrations have died out. The fundamental equations for static response of the elastic rod are [2]:

- The divergence of stress (σ) is balanced by the body forces(b): equilibrium equation
- The stress (σ) is related to the strain(ε) in a linear manner: material constitutive equation
- The strain(ε) is given by the displacement(u) gradient: strain definition

These equations can be written as:

$$\frac{d\sigma}{dx} + b = 0 \tag{1}$$

$$\sigma = \varepsilon E \tag{2}$$

$$\varepsilon = \frac{du}{dx} \tag{3}$$

We substitude stress and strain in the equilibrium equation to get an expression relating displacement to elastic properties. We get:

$$\frac{d^2u}{dx^2} + \frac{b}{E} = 0 \tag{4}$$

For simplicity, we consider the forcing term to be one, therefore the equation becomes

$$-\frac{d^2u}{dx^2} = 1 \tag{5}$$

Since this equation is second order in x, we need two boundary conditions to compute the solution. I will use a fixed (zero) displacement boundary on the two sides of the finite rod. Let the rod be a unit length placed in a cartesian coordinate system with its left end placed at $x = 0$ and parallel to the x axis. We can write the differential equation with its boundary conditions and the domain as:

$$-\frac{d^2u}{dx^2} = 1;\ x_\epsilon[0,1];\ u(0) = u(1) = 0 \tag{6}$$

$$-u^{''}(x) = 1;\ x_\epsilon[0,1];\ u(0) = u(1) = 0 \tag{7}$$

3 Methodology

The idea behind finite element methods is to break up the domain into multiple elements, and calculate the solution for each element at a local level. In order to do this efficiently, we can represent our solution $u(x)$ as a set of piecewise functions with the property that these functions are zero outside the element in consideration. These are known as basis functions. Using this expansion and some variational calculus, we can transform our equation into a set of ordinary differential equations which can be solved using linear algebra.

As with all other numerical methods, we start with a mesh discretization of the space. We use linearly spaced elements inside the domain $[-1, 1]$. Let the number of elements be M.

Next, we need to define the basis functions. For simplicity, we use a simple hat function as our basis function. More complicated higher order basis functions can be used if the geometry is curved or non-planar. Fig. 2. shows a typical example of the hat basis function[2]. One of the biggest advantages of finite element method is that the basis functions are piecewise, therefore elements with non-uniform size will be just as easy to work with as elements with uniform size. This is because we are essentially solving for each element within the domain of the basis functions, e.g. [0,1] for hat functions.

The next step is to derive the weak or the variational form of the differential equation. This is a two-step process: first, we multiply our differential

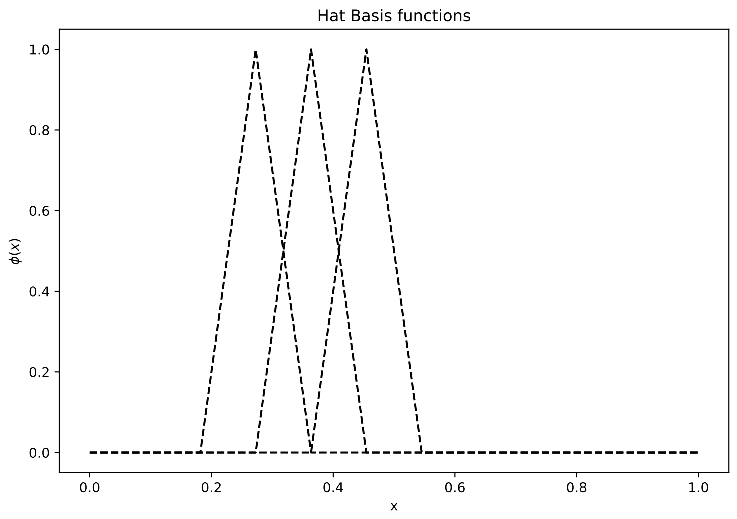

Figure 2: Hat basis functions for three consecutive elements. The values are non zero only for the element in consideration, and zero everywhere else. In this example, we consider simplest basis function. But we may have the elements further discretized into several nodes and the basis function can be higher order polynomials interpolated over the nodes.

equation with a test function, and second, we integrate the equation by parts to get rid of the derivatives of the solution. There are multiple choices for the test function, and different choices lead us to very different methods using the same formulation. If our test function is a dirac delta function, we end up with pseudospectral methods for solving partial differential equations. For finite element methods, we will let the test function be same as the basis functions we chose to approxiamte the solution. This is called the Galerkin method, and is proven to be much better than other choices [3]. The most obvious advantage of using test function the same as the basis function is that they are orthogonal, therefore unless the element is same, the product of two such functions is zero. This results in sparse matrices for inverting the solutions.

Multiplying our solution with a test function v, integrating by parts within the domain,and applying Green's theorem gives us:

$$-u^{''}v = fv;\ where\ f = 1. \tag{8}$$

$$-\int_0^1 u^{''}vdx = \int_0^1 fvdx \tag{9}$$

$$-\int_0^1 u^{'}vdx + \int_0^1 u^{'}v^{'}dx = \int_0^1 fvdx \tag{10}$$

$$\int_0^1 u^{'}v^{'}dx = \int_0^1 fvdx \tag{11}$$

The reason for using this formulation is mostly historic and a matter of convention. Finite element methods were developed by structual engineers and fluid dynamicists and the conservation of certain physical field quantities are easily represented by the variational formulation.

For a set of basis functions $\phi(x)$, our solution $u(x)$ can be written as $u(x) = \sum_{j=1}^{M} c_j\phi_j(x)$. Here c_j are the unknown coefficients to be determined. Substituting this in equation 11, we get

$$\int_0^1 \sum_{j=1}^{M} c_j\phi_j^{'}(x)v^{'}dx = \sum_{j=1}^{M}\int_0^1 c_j\phi_j^{'}(x)v^{'}dx = \int_0^1 fvdx \tag{12}$$

Next, we choose the test function $v(x) = \phi_1, \ldots, \phi_M - 1$ successively so as to get $M-2$ independent equations. These equations, together with the boundary conditions will be sufficient to determine the coefficients c_j for the solution.

$$\left(\int_0^1 \phi_1^{'}\phi_1^{'}dx\right)c_1 + \ldots + \left(\int_0^1 \phi_1^{'}\phi_{M-1}^{'}dx\right)c_1 = \int_0^1 f\phi_1 dx \tag{13}$$

$$\left(\int_0^1 \phi_2^{'}\phi_1^{'}dx\right)c_1 + \ldots + \left(\int_0^1 \phi_2^{'}\phi_{M-1}^{'}dx\right)c_1 = \int_0^1 f\phi_2 dx \tag{14}$$

$$\vdots \tag{15}$$

$$\left(\int_0^1 \phi_{N-1}^{'}\phi_1^{'}dx\right)c_1 + \ldots + \left(\int_0^1 \phi_{N-1}^{'}\phi_{M-1}^{'}dx\right)c_1 = \int_0^1 f\phi_{N-1} dx \tag{16}$$

We can write the above set of equations in matrix form as follows:

$$\begin{bmatrix} a(\phi_1,\phi_1) & a(\phi_1,\phi_2) & \ldots \\ \vdots & \ddots & \vdots \\ a(\phi_{M-1},\phi 1) & \ldots & a(\phi_{M-1},\phi_{M-1}) \end{bmatrix} \begin{bmatrix} c_1 \\ \vdots \\ c_{M-1} \end{bmatrix} = \begin{bmatrix} < f,\phi_1 > \\ \vdots \\ < f,\phi_{M-1} > \end{bmatrix} \tag{17}$$

where,

$$a(\phi_i, \phi_j) = \int_0^1 \phi_i^{'} \phi_j^{'} dx, \tag{18}$$

$$< f, \phi_i >= \int_0^1 f \phi_i dx \tag{19}$$

If the basis functions arehat functions, we get:

$$\begin{bmatrix} \frac{2}{h} & \frac{-1}{h} & \cdots \\ \frac{-1}{h} & \frac{2}{h} & \frac{-1}{h} \\ \cdots & \frac{-1}{h} & \frac{2}{h} \end{bmatrix} \begin{bmatrix} c_1 \\ \vdots \\ c_n \end{bmatrix} = \begin{bmatrix} \int_0^1 f \phi_1 dx \\ \vdots \\ \int_0^1 f \phi_{M-1} dx \end{bmatrix} \tag{20}$$

Thus, assembling the left hand matrix is straightforward. To compute the right hand side of the matrix, we need tointegrate the given function over the basis function. We can use any integration algorithm like quadrature integration or simpson's rule. I have imported simpson's method from python'sscipy library to compute the right hand side.

4 Results

I have implemented the above method in python. The right hand side of equation (20) is computed using simpson's rule for integration. The function f in our case is 1. The basis functions can be programmed using the following formulation for the i^{th} basis function:

$$\phi_i(x) = \begin{cases} 0 & x < x_{i-1} \\ \frac{x - x_{i-1}}{\Delta x} & x_{i-1} \leq x < x_i \\ 1 - \frac{x - x_i}{\Delta x} & x_i \leq x < x_{i+1} \\ 0 & x > x_{i+1} \end{cases} \tag{21}$$

We solve the system of linear equation using numpy's linear solver `numpy.linalg.solve()`. We compare the finite element solution of a one-dimensional Navier-Stokes equation with it's analytically computed solution in Fig. 3. The errors are shown in Fig. 4.

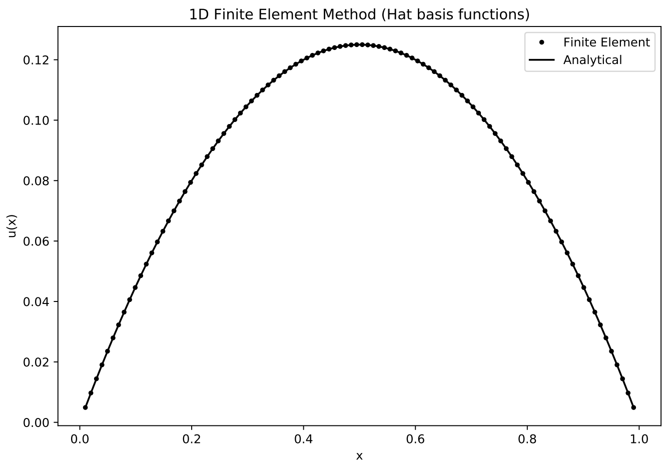

Figure 3: Analytically computed solution vs. the finite element solution. We can see that there is a pretty good match even at the boundaries.

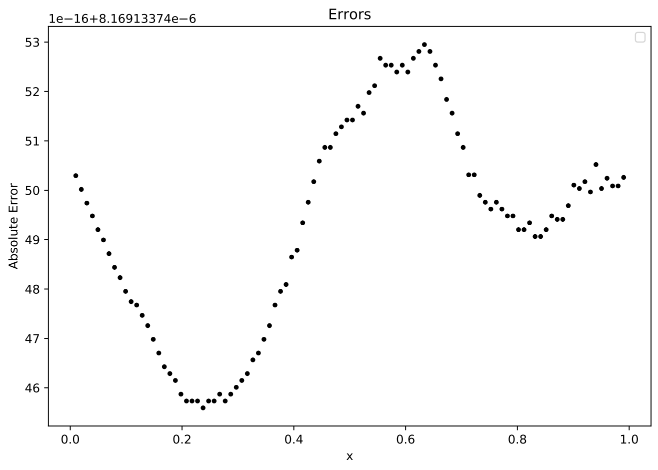

Figure 4: The errors in the finite element solution. These are in the order of 10^{-6}, which is greater than the machine precision error. The errors would be significantly reduced if we used higher order basis functions.

Next, we try a more complicated solution to linear elasticity using fenics. Fenics is a finite element solver written in C++ but wrapped around python, so that we only need to specify the weak form of the differential equation with the appropriate boundary conditions, and the solver will assemble the matrices and compute the quadrature integration for us. We start with the same equations as (1,2,3) but in two dimensions. The entire problem with boundary conditions can be written as:

$$-\nabla.\sigma = f in \Omega, \tag{22}$$

$$\sigma = \lambda tr(\varepsilon) I + 2\mu\varepsilon, \tag{23}$$

$$\varepsilon = \frac{1}{2}(\nabla u + (\nabla u)^T) \tag{24}$$

where Ω is the physical domain, λ, μ are lame's parameters for elasticity, and I is the identitiy matrix. We assume a two dimensional sheet fixed at one end and traction free at all other ends hanging under the influence of it's own gravity. We wish to compute the displacement and stresses at the steady state. The boundary conditions for this problem can be described as $u = u_D = (0, 0)$ at $x = 0$, and $f = (0, -\rho g)$. For the rest of the traction free domain, $T = 0$. We write this set of equations in the weak form similar to what is desribed above, and solve it in fenics to obtain the steady state stresses and displacements of the medium (Fig. 4, 5).

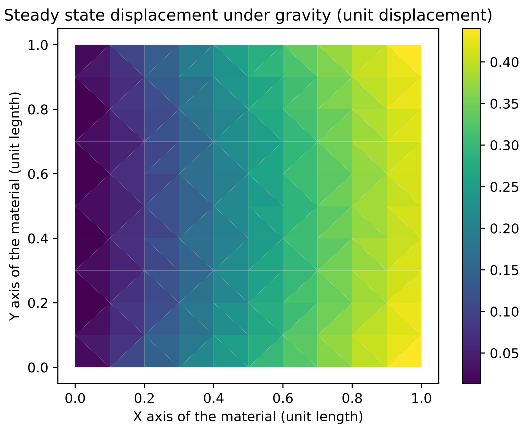

Figure 5: The displacement at steady state. We use unit dimensions for all quantities. It is fixed at the left end, therefore we see zero displacement on the left end.

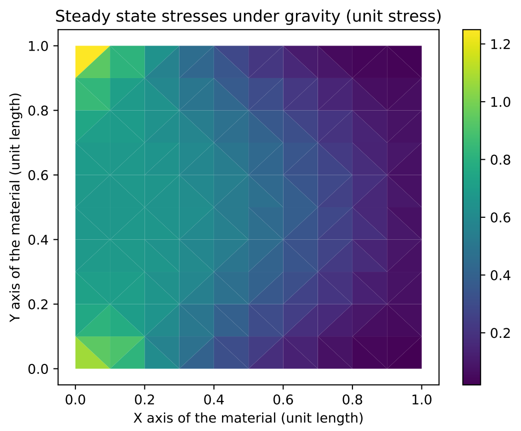

Figure 6: The stresses at steady state. Since the left end is fixed, therefore is is subjected to most stresses. The right freely hanging end has near zero stresses.

5 Conclusions

We have computed the solution to a one dimensional navier stokes equation and derived the finite element formulation for it. We further extended this to two dimensions using the python package fenics. Although the finite element methods are difficult to program, they have several advantages over finite differences and pseudospectral methods in terms of geometrical complexity and physical hetereogeneity. Therefore, finite element methods are an ideal candidate for solving real world problems that involve these complexities.

My future work would be to derive a model for a long term fault-slip to simulate multiple earthquakes and aseismic slips on a single fault. This would require solving the elastostatic and elastodynamic equations of motion with waves propagating in the medium, and switching between the two using some criteria. Finite element methods are an attractive candidate for these problems since they are multiscale in space. We want to simulate near fault large heterogeneities as well as the micro scale frictional heterogeneities. Although finite element methods in itself are not well suited for hyperbolic partial differential equations, using a discontinuous galerkin or spectral element methods is shown to work well [4].

6 References

[1] http://faculty.washington.edu/nsniadec/ME478/S13/

[2] Achenbach, J. (2012). Wave propagation in elastic solids (Vol. 16). Elsevier.

[3] https://www.wikiwand.com/en/Galerkin_method.

[4] Komatitsch, D., Tromp, J. (1999). Introduction to the spectral element method for three-dimensional seismic wave propagation. Geophysical journal international, 139(3), 806-822. [5] https://fenicsproject.org