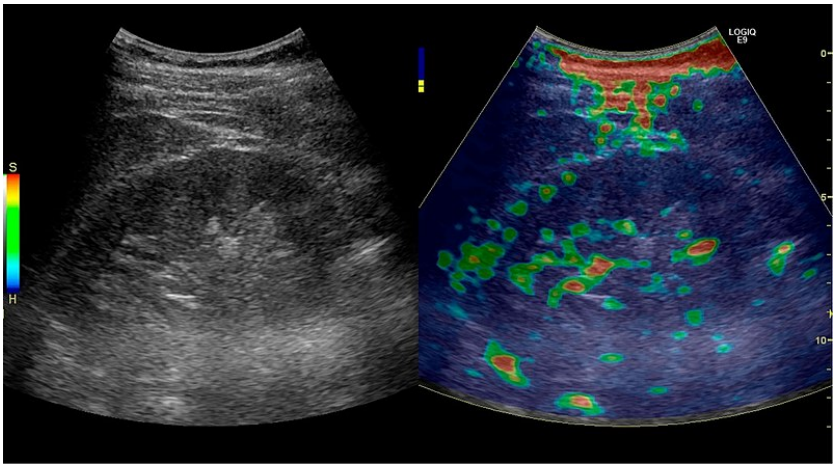
# BN5205 Finite Element Assignment Due 16 Nov 2018




Strain elastography of a normal kidney. Red depicts soft areas, and blue depicts hard areas. Hansen et al. Diagnostics 2016, 6(1), 2

Elastography is a non-invasive method for determining the elastic properties of soft tissues. The basic idea is to observe the mechanical response of the tissue in the presence of a mechanical oscillation (e.g. ultrasound). The tissue displacements can then be used to quantify the mechanical properties of the tissue. If we assume the tissue is isotropic, homogeneous, and incompressible, the Helmholtz equation (shown below) can be used to model the system:

$$\nabla^2 u + \kappa u = 0$$

where $u$ is the displacement of the tissue, and $\kappa$ is a constant which is given by:

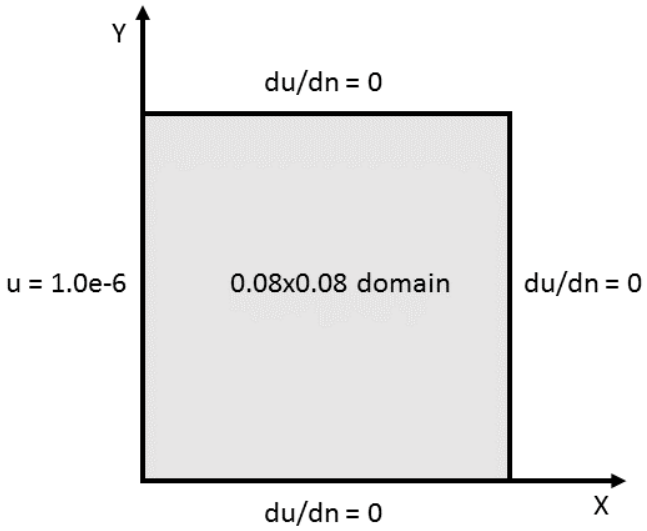
$$\kappa = \frac{\rho \omega^2}{\mu}$$

Here, $\rho$ is the tissue density, $\omega$ is the frequency of the mechanical oscillations, and $\mu$ is one of Lame's constants (shear modulus) and represents the material properties of the tissue.

In elastography, the problem is to solve for the material properties given the displacements. Here though, we will look at the simpler problem of determining the displacements given the material properties over a domain in 2D space.

**Task**

Write a program (in Python) to solve the 2D Helmholtz equation using the Galerkin Finite Element Method, and therefore calculate the displacements that arise for a given excitation frequency. The chosen domain is a square mesh with overall dimension of 0.08x0.08. Please note that this does not change with the mesh resolution. You should use bilinear basis functions, which can be integrated with 2x2 Gaussian quadrature points. The boundary conditions should be applied according to the scheme shown below:




To help you out, the following Python code is provided:

**"main.py"** is the (incomplete) main finite element code. This provides a skeleton for you to use along with some necessary initializations (including Gaussian quadrature points and weights). It will generate the meshes for you. The mesh resolution can be changed through the "elemsPerSide" (elements per side) variable.
**"psi.py"** contains the bilinear basis functions and derivatives *(no modifications needed)*.
**"dxidxJ.py"** Contains the (incomplete) x<->xi scaling function.

*Note that you should **not** assume your elements will always be square. I will test your code on a non-square (but still bilinear) domain.*

**Hand in:**

1. Your workings/derivation to show how you started with the governing equation and got to the equation that was implemented in your code
2. Your (well commented) code, including instructions on how to run it if necessary.
3. Given $\mu = 1000Pa, \rho = \frac{1060kg}{m^3}, and\ \omega = 100Hz$, run your code at 2x2, 4x4, 8x8, 16x16, 32x32 & 64x64 elements per side resolutions. From this, determine the most appropriate resolution to use to solve this problem. Please explain how you reached your conclusion.
4. Clinically, the oscillation frequency used is typically 10-1000Hz. Can your implementation get an accurate answer over this range? If yes, justify your conclusion. If not, what you recommend should be done such that it can?

***Hint:*** With $\kappa = 0$ the governing equation reduces to Laplace's equation for which there are analytic solutions you can use to test your code, e.g. $u = x^2 - y^2$.
***Hint:*** With $\kappa = 0$ and eight elements per side, each element is a unit square solving Laplace's equation. The element stiffness matrix for this case is explicitly given in the notes.