Bake, shake or break - and other applications for the FEM

Programming project in TMA4220 - part 2

by Kjetil André Johannessen and Trond Kvamsdal
TMA4220 - Numerical solution of partial differential equations using the finite element method

5: Do real-life experimentation using your FEM code

In the second part of the problem set you are going to make use of your finite element library which you have now built. We are going to apply this to one of several real-life applications. The three first tasks introduce the main equations you can choose from

- $\frac{\partial^2 u}{\partial t^2} = \alpha \nabla^2 u$ - the heat equation
- $\nabla \boldsymbol{\sigma}(\boldsymbol{u}) = -\boldsymbol{f}$ - the linear elasticity equation
- $\rho \frac{\partial^2 u}{\partial t^2} = \nabla \boldsymbol{\sigma}(\boldsymbol{u})$ - the free vibration equation

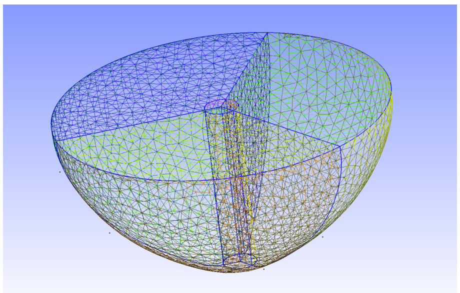
and include some very rough ”getting-started” theory on these. After this, it follows other applications of the same equations, but ultimately you are free to choose and solve whatever problem you want at this stage.

5.1: Making a princess cake (bake)

In this task you will take a deeper look into how to make a princess cake. The general idea is to bake the skirt in cake dough, turn this upside down, decorate the skirt and put a doll into the center such that it looks like she is wearing the skirt. You will be asked to model the cake dough during cooking and predict the temperature distribution in this.

Figure 1: The target princess cake

(a) The physical cake mold form



(b) The computational finite element mesh

Figure 2: The geometry which you are going to solve the heat equation on

a) The heat equation

The heat equation reads

$$\begin{aligned} \frac{\partial u}{\partial t} &= \alpha \nabla^2 u \\ u(t,x,y,z)|_{\partial\Omega} &= u^D \\ u(t,x,y,z)|_{t=0} &= u_0(x,y,z) \end{aligned} \tag{1}$$

where α is an positive constant defined by

$$\alpha = \frac{\kappa}{c_p \rho}$$

with κ^{**} being the thermal conductivity, ρ^{**} the mass density and c_p^{**} the specific heat capacity of the material.

We are going to *semidiscretize* the system by projecting the spatial variables to a finite element subspace X_h. Multiply (1) by a test function v and integrate over the domain Ω to get

$$\iiint\limits_{\Omega} \frac{\partial u}{\partial t} v \, dV = -\iiint\limits_{\Omega} \alpha \nabla u \nabla v \, dV$$

Note that we have only semidiscretized the system, and as such our unknown u is given as a linear combination of the *spatial* basis functions, and continuous in time, i.e.

$$u_h(x,y,z,t) = \sum_{i=1}^{n} u_h^i(t) \varphi_i(x,y,z).$$

The variational form of the problem then reads: Find $u_h \in X_h^D$ such that

$$\begin{aligned} \iiint\limits_{\Omega} \tfrac{\partial u}{\partial t} v \, dV &= -\iiint\limits_{\Omega} \alpha \nabla u \nabla v \, dV, \quad \forall v \in X_h \\ \Rightarrow \quad \sum_i \iiint\limits_{\Omega} \varphi_i \varphi_j dV \tfrac{\partial u_h^i}{\partial t} &= -\sum_i \iiint\limits_{\Omega} \alpha \nabla \varphi_i \nabla \varphi_j dV \, u_h^i \quad \forall j \end{aligned}$$

which in turn can be written as the linear system

$$\boldsymbol{M} \frac{\partial \boldsymbol{u}}{\partial t}(t) = -\boldsymbol{A}\boldsymbol{u}(t) \tag{2}$$

which is an ordinary differential equation (ODE) with the matrices defined as

$$\begin{aligned} \boldsymbol{A} = [A_{ij}] &= \iiint\limits_{\Omega} \alpha \nabla \varphi_i \nabla \varphi_j \, dV \\ \boldsymbol{M} = [M_{ij}] &= \iiint\limits_{\Omega} \varphi_i \varphi_j \, dV. \end{aligned}$$

Construct the matrix $\boldsymbol{A}$ and $\boldsymbol{M}$ as defined above.

b) Time integration

The system (2) is an ODE, which should be familiar from previous courses. Very briefly an ODE is an equation on the form

$$\frac{\partial y}{\partial t} = f(t, y)$$

where y may be a vector. The simplest ODE solver available is Eulers method

$$y_{n+1} = y_n + hf(t_n, y_n).$$

More sophisticated include the improved eulers methods

$$y_{n+1} = y_n + \frac{h}{2}\left(f(t_n, y_n) + f(t_{n+1}, y_n + hf(t_n, y_n))\right)$$

or the implicit trapezoid rule

$$y_{n+1} = y_n + \frac{h}{2}\left(f(t_n, y_n) + f(t_{n+1}, y_{n+1})\right)$$

and the famous Runge Kutta methods.

Choose an ODE scheme (based on your previous experience and expertize) and implement your time integration. Why did you choose the solver you did?

c) Experimentation

The boundary conditions are the physical variables which we have control over. The initial condition $u(t, x, y, z)|_{t=0}$ is the cake dough as it is prior to any cooking. A proper choice here would be room temperature, say 20°C.

During the cooking in the oven, you may apply different boundary conditions as you see fit. One option would be non-homogeneous Dirichlet boundary conditions of, say 225°C. This would correspond to the oven temperature. Another option is to enforce a heat *flux* into your domain which would be formulated as Neumann boundary conditions.

One of the key goals in this task is to see the effect that the center metallic rod has on the solution. It's purpose is to make sure that the cake is more or less evenly cooked at the end, so you don't have any raw dough in the middle of your domain after taking it out of the oven; see figure 3.

For a computational realization of the internal rod, you should apply different material properties to all elements within this domain. In the geometry files which are available for downloading from the course webpage, all elements in the rod have been tagged with 1001, while all dough elements are tagged with 1000.

(**) Physical proprties of cake dough and aluminium

Sorry, but you'll have to figure out this by yourself.

Figure 3: Raw cake dough in the middle. And yes, I actually made this cake.

5.2 Structural analysis (break)

We are in this problem going to consider the linear elasticity equation. The equations describe deformation and motion in a continuum. While the entire theory of continuum mechanics is an entire course by itself, it will here be sufficient to only study a small part of this: the linear elasticity. This is governed by three main variables $\boldsymbol{u}, \boldsymbol{\varepsilon}$ and $\boldsymbol{\sigma}$ (see table 1). We will herein describe all equations and theory in terms of two spatial variables (x, y), but the extension into 3D space should be straightforward.

$\boldsymbol{u} = \begin{bmatrix} u_x \\ u_y \end{bmatrix}$ -	the *displacement* vector measures how much each spatial point has moved in (x, y)-direction
$\boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_{xx} & \varepsilon_{xy} \\ \varepsilon_{xy} & \varepsilon_{yy} \end{bmatrix}$ -	the *strain* tensor measures how much each spatial point has deformed or stretched
$\boldsymbol{\sigma} = \begin{bmatrix} \sigma_{xx} & \sigma_{xy} \\ \sigma_{xy} & \sigma_{yy} \end{bmatrix}$ -	the *stress* tensor measures how much forces per area are acting on a particular spatial point

Table 1: Linear elasticity variables in two dimensions

Note that the subscript denotes vector component and *not* derivative, i.e. $u_x \neq \frac{\partial u}{\partial x}$.

These three variables can be expressed in terms of each other in the following way:

$$\begin{aligned} \boldsymbol{u} &= \boldsymbol{u}(\boldsymbol{x}) && (3) \\ \boldsymbol{\varepsilon} &= \boldsymbol{\varepsilon}(\boldsymbol{u}) && (4) \\ \boldsymbol{\sigma} &= \boldsymbol{\sigma}(\boldsymbol{\varepsilon}) && (5) \end{aligned}$$

The primary unknown $\boldsymbol{u}$ (the displacement) is the one we are going to find in our finite element implementation. From (3) we will have two displacement values for each finite element "node", one in each of the spatial directions.

The relation (4) is a purely geometric one. Consider an infinitesimal small square of size dx and dy, and its deformed geometry as depicted in figure 4. The strain is defined as the stretching of the element, i.e. $\varepsilon_{xx} = \frac{length(ab) - length(AB)}{length(AB)}$. The complete derivations of these quantities is described well in the Wikipedia article on strain, and the result is the following relations

$$\begin{aligned} \varepsilon_{xx}(\boldsymbol{u}) &= \frac{\partial u_x}{\partial x} \\ \varepsilon_{yy}(\boldsymbol{u}) &= \frac{\partial u_y}{\partial y} \\ \varepsilon_{xy}(\boldsymbol{u}) &= \frac{\partial u_x}{\partial y} + \frac{\partial u_y}{\partial x}. \end{aligned} \qquad (6)$$

Note that these relations are the *linearized* quantities, which will only be true for small deformations.

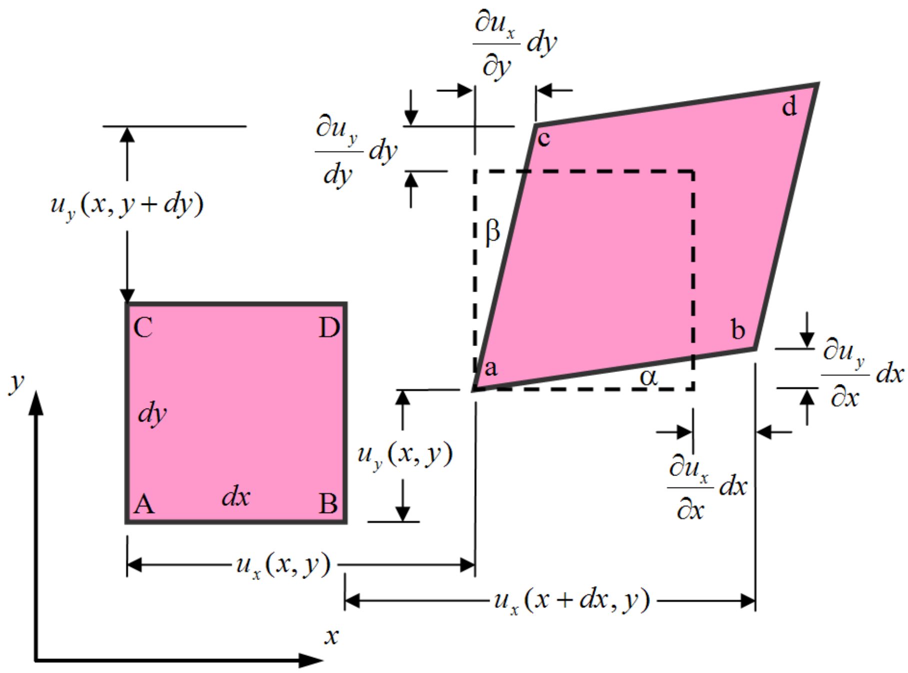


Figure 4: An infinitesimal small deformed rectangle

For the final relation, which connects the deformation to the forces acting upon it, we turn to the material properties. Again, there is a rich literature on the subject, and different relations or physical laws to describe different materials. In our case, we will study small deformations on solid materials like metal, wood or concrete. It is observed that such materials behave elastically when under stress of a certain limit, i.e. a deformed geometry will return to its initial state if all external forces are removed. Experiment has shown that the Generalized Hooks Law is proving remarkable accurate under such conditions. It states the following. Consider a body being dragged to each side by some stress σ_{xx} as depicted in figure 5. Hooks law states that the forces on a spring is linearly dependant on the amount of stretching multiplied by some stiffness constant, i.e. $\sigma_{xx} = E\varepsilon_{xx}$. The constant E is called Young's modulus. Generalizing upon this law, we see that materials typically contract in the y-direction, while being dragged in the x-direction. The ratio of compression vs expansion is called Poisson's ratio ν and is expressed as $\varepsilon_{yy} = -\nu\varepsilon_{xx}$. This gives the following relations

$$\begin{aligned}\varepsilon_{xx} &= \frac{1}{E}\sigma_{xx} \\ \varepsilon_{yy} &= -\frac{\nu}{E}\sigma_{xx}\end{aligned}$$

Due to symmetry conditions, we clearly see that when applying a stress σ_{yy} in addition to σ_{xx} we get

$$\begin{aligned}\varepsilon_{xx} &= \frac{1}{E}\sigma_{xx} - \frac{\nu}{E}\sigma_{yy} \\ \varepsilon_{yy} &= \frac{1}{E}\sigma_{yy} - \frac{\nu}{E}\sigma_{xx}\end{aligned}$$

Finally, it can be shown (but we will not) that the relation between the shear strain and shear stress

Figure 5: Deformed geometry under axial stresses

is $\varepsilon_{xy} = 2\frac{1+\nu}{E}\sigma_{xy}$. Collecting the components of $\boldsymbol{\varepsilon}$ and $\boldsymbol{\sigma}$ in a vector, gives us the compact notation

$$\begin{aligned} \bar{\varepsilon} &= C^{-1}\bar{\sigma} \\ \begin{bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{bmatrix} &= \begin{bmatrix} \frac{1}{E} & -\frac{\nu}{E} & 0 \\ -\frac{\nu}{E} & \frac{1}{E} & 0 \\ 0 & 0 & 2\frac{1+\nu}{E} \end{bmatrix} \begin{bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{xy} \end{bmatrix} \end{aligned}$$

or conversely

$$\begin{aligned} \bar{\sigma} &= C\bar{\varepsilon} \\ \begin{bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{xy} \end{bmatrix} &= \frac{E}{1-\nu^2} \begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \frac{1-\nu}{2} \end{bmatrix} \begin{bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{bmatrix} \end{aligned} \tag{7}$$

For a body at static equilibrium, we have the governing equations

$$\begin{aligned} \nabla\boldsymbol{\sigma}(\boldsymbol{u}) &= -\boldsymbol{f} \\ \left[\frac{\partial}{\partial x}, \frac{\partial}{\partial y}\right] \begin{bmatrix} \sigma_{xx} & \sigma_{xy} \\ \sigma_{xy} & \sigma_{yy} \end{bmatrix} &= -[f_x, f_y] \end{aligned} \tag{8}$$

and some appropriate boundary conditions

$$\boldsymbol{u} = \boldsymbol{g}, \quad on \quad \partial\Omega_D \tag{9}$$

$$\boldsymbol{\sigma}\cdot\hat{\boldsymbol{n}} = \boldsymbol{h}, \quad on \quad \partial\Omega_N \tag{10}$$

a) Weak form

Show that (8) can be written as the scalar equation

$$\sum_{i=1}^{2}\sum_{j=i}^{2}\int_{\Omega}\varepsilon_{ij}(\boldsymbol{v})\sigma_{ij}(\boldsymbol{u})\,dA = \sum_{i=1}^{2}\int_{\Omega}v_i f_i\,dA + \sum_{i=1}^{2}\sum_{j=1}^{2}\int_{\partial\Omega}v_i\sigma_{ij}\hat{\boldsymbol{n}}\,d\boldsymbol{S}$$

(where we have exchanged the subscripts (x, y) with $(1, 2)$) by multiplying with a test function $\boldsymbol{v} = \begin{bmatrix} v_1(x,y) \\ v_2(x,y) \end{bmatrix}$ and integrating over the domain Ω Moreover, show that this can be written in compact vector form as

$$\begin{aligned}\int_{\Omega}\bar{\varepsilon}(\boldsymbol{v})^T C\bar{\varepsilon}(\boldsymbol{u})\,dA &= \int_{\Omega}\boldsymbol{v}^T\boldsymbol{f}\,dA + \int_{\partial\Omega}\boldsymbol{v}^T\boldsymbol{\sigma}\hat{\boldsymbol{n}}\,d\boldsymbol{S} \\ &= \int_{\Omega}\boldsymbol{v}^T\boldsymbol{f}\,dA + \int_{\partial\Omega}\boldsymbol{v}^T\boldsymbol{h}\,d\boldsymbol{S}\end{aligned}$$

b) Galerkin projection

As in 2b) let $\boldsymbol{v}$ be a test function in the space X_h of piecewise linear functions on some triangulation T. Note that unlike before, we now have *vector* test functions. This means that for each node $\hat{i}$, we will have two test functions

$$\begin{aligned}\boldsymbol{\varphi}_{\hat{i},1}(\boldsymbol{x}) &= \begin{bmatrix} \varphi_{\hat{i}}(\boldsymbol{x}) \\ 0 \end{bmatrix} \\ \boldsymbol{\varphi}_{\hat{i},2}(\boldsymbol{x}) &= \begin{bmatrix} 0 \\ \varphi_{\hat{i}}(\boldsymbol{x}) \end{bmatrix}\end{aligned}$$

Let these functions be numbered by a single running index $i = 2\hat{i} + d$, where i is the node number in the triangulation and d is the vector component of the function.

Show that by inserting $\boldsymbol{v} = \boldsymbol{\varphi}_j$ and $\boldsymbol{u} = \sum_i \boldsymbol{\varphi}_i u_i$ into (11) you get the system of linear equations

$$A\boldsymbol{u} = \boldsymbol{b}$$

where

$$\begin{aligned}A = [A_{ij}] &= \int_{\Omega}\bar{\varepsilon}(\boldsymbol{\varphi}_i)^T C\bar{\varepsilon}(\boldsymbol{\varphi}_j),\ dA \\ b = [b_i] &= \int_{\Omega}\boldsymbol{\varphi}_i^T\boldsymbol{f}\,dA + \int_{\partial\Omega}\boldsymbol{\varphi}_i^T\boldsymbol{h}\,d\boldsymbol{S}\end{aligned}$$

(**Hint:** $\bar{\varepsilon}(\cdot)$ is a linear operator)

c) Test case

Show that

$$\boldsymbol{u} = \begin{bmatrix} (x^2-1)(y^2-1) \\ (x^2-1)(y^2-1) \end{bmatrix}$$

is a solution to the problem

$$\begin{aligned} \nabla \boldsymbol{\sigma}(\boldsymbol{u}) &= -\boldsymbol{f} \; in \; \Omega \\ \boldsymbol{u} &= \mathbf{0} \; on \; \partial\Omega \end{aligned} \tag{11}$$

where

$$\begin{aligned} f_x &= \frac{E}{1-\nu^2}\left(-2y^2 - x^2 + \nu x^2 - 2\nu xy - 2xy + 3 - \nu\right) \\ f_y &= \frac{E}{1-\nu^2}\left(-2x^2 - y^2 + \nu y^2 - 2\nu xy - 2xy + 3 - \nu\right) \end{aligned}$$

and $\Omega = \{(x,y) \; : \; \max(|x|,|y|) \leq 1\}$ is the refereance square $(-1,1)^2$.

d) Implementation

Modify your Poisson solver to solve the problem (11). Verify that you are getting the correct result by comparing with the exact solution. The mesh may be obtained through the Grid function `getPlate( )`.

e) Extension into 3d

Modify your 3d Poisson solver to assemble the stiffness matrix from linear elasticity in three dimensions.

f) Experimentation

Import a 3d mesh from Minecraft or create one using your choice of meshgenerator. Apply gravity loads as the bodyforces acting on your domain, this will be the right hand side function f in (8). In order to get a non-singular stiffness matrix you will need to pose some Dirichlet boundary conditions. Typically you should introduce zero displacements (homogeneous Dirichlet conditions) where your structure is attached to the ground. This would yield a stationary solution.

g) Stress analysis

Solving (8) with a finite element method gives you the primary unknown: the displacement u. If you are interested in derived quantities such as the stresses, these can be calculated from (7). Note that σ is in essence the derivative of u which means that since u is C^0 across element boundaries, then σ will be discontinuous. To get stresses at the nodal values, we propose to average the stresses over all neighbouring elements.

Loop over all elements and evaluate (the constant) stresses on that element. For each node, assign the stresses to be the average stress over all neighbouring elements. This method is called ”Stress Recovery”.

Figure 6: Block-structured mesh from the computer game Minecraft

5.3 Vibration analysis (shake)

Do problem 5.2a) - 5.2d) and read the theory on linear elasticity.

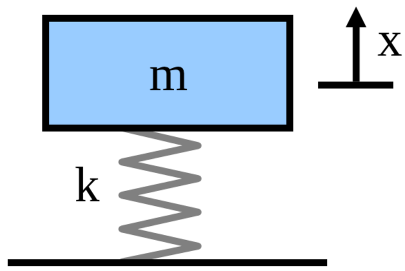


Figure 7: Mass-spring-model

The forces acting on a point mass m by a spring is given by the well known Hooks law:

$$m\ddot{x} = -kx$$

This can be extended to multiple springs and multiple bodies as in figure 8

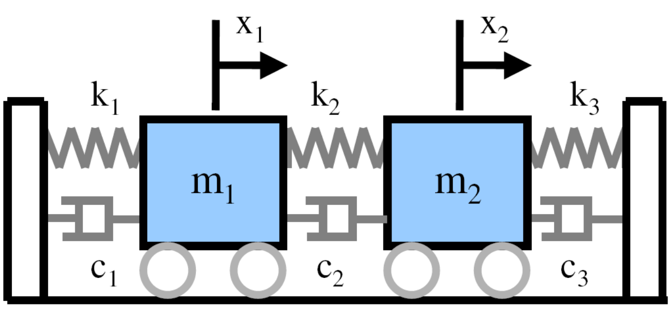


Figure 8: 2 degree-of-freedom mass spring model

The physical laws will now become a system of equations instead of the scalar one above. The forces acting on m_1 is the spring k_1 dragging in negative direction and k_2 dragging in the positive direction.

$$m_1\ddot{x_1} = -k_1x_1 + k_2(x_2 - x_1)$$

This is symmetric, and we have an analogue expression for m_2. The system can be written in matrix form as

$$\begin{aligned} \begin{bmatrix} m_1 & 0 \\ 0 & m_2 \end{bmatrix} \begin{bmatrix} \ddot{x}_1 \\ x_2 \end{bmatrix} &= \begin{bmatrix} -k_1 - k_2 & k_2 \\ k_2 & -k_2 - k_3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \\ M\ddot{\boldsymbol{x}} &= A\boldsymbol{x} \end{aligned}$$

When doing continuum mechanics, it is the exact same idea, but the actual equations differ some. Instead of discrete equations, we have continuous functions in space and the governing equations are

$$\rho\ddot{\boldsymbol{u}} = \nabla\boldsymbol{\sigma}(\boldsymbol{u})$$

semi-discretization yields the following system of equations

$$M\ddot{\boldsymbol{u}} = -A\boldsymbol{u} \tag{12}$$

with the usual stiffness and mass matrix

$$\begin{aligned} \boldsymbol{A} = [A_{ij}] &= \iiint_\Omega \bar{\varepsilon}(\boldsymbol{\varphi}_i)^T C\bar{\varepsilon}(\boldsymbol{\varphi}_j)\, dV \\ \boldsymbol{M} = [M_{ij}] &= \iiint_\Omega \rho\boldsymbol{\varphi}_i^T\boldsymbol{\varphi}_j\, dV. \end{aligned}$$

e)

Build the 3d mass matrix as given above.

We are now going to search for solutions of the type:

$$\boldsymbol{u} = \boldsymbol{u}e^{\omega it} \tag{13}$$

which inserted into (12) yields

$$\omega^2 M\boldsymbol{u} = A\boldsymbol{u} \tag{14}$$

f)

Equation (14) is called a generalized eigenvalue problem (the traditional being with $M = I$). Find the 20 first eigenvalues ω_i and eigenvectors $\boldsymbol{u}_i$ corresponding to this problem.

g)

Let x_0 be your initial geometric description (the nodal values). Plot an animation of the eigenmodes by

$$\boldsymbol{x} = \boldsymbol{x}_0 + \alpha\boldsymbol{u}_i\sin(t)$$

You may want to scale the vibration amplitude by some visually pleasing scalar α, and choose the time steps appropriately. Note that for visualization purposes, you will not use the eigenfrequency ω_i since you are interested in viewing (say) 1-5 complete periods of the vibration, but for engineering purposes this is a very important quantity.

h)

Recreate the experiment as presented in the youtube video from figure 10

In its simplest form, one should be able to construct this setup using a bluetooth speaker, your smartphone and a sound-wave app. Note however that the frequencies will depend on the material you choose. Does the thickness of the plate make any difference? Does the choice of material influence the patterns? How well were you able to recreate both the patterns and the frequencies.

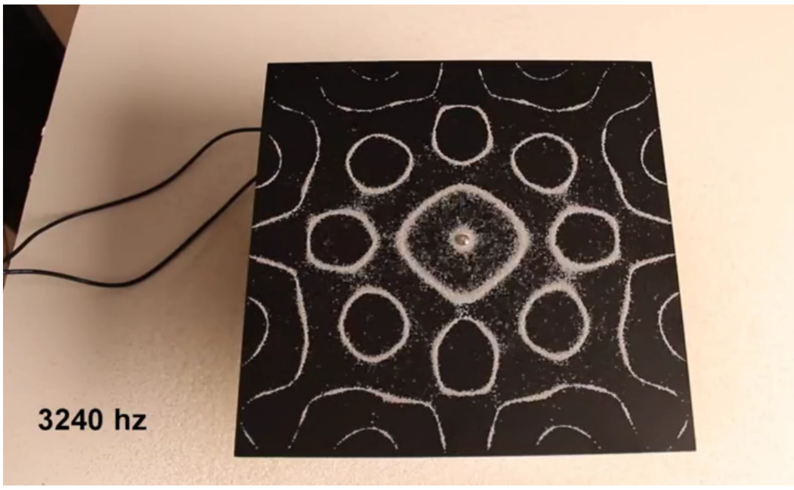


Figure 9: A vibrating plate with table salt on it

Figure 10: http://youtu.be/wvJAgrUBF4

5.4 Harmonic sounds

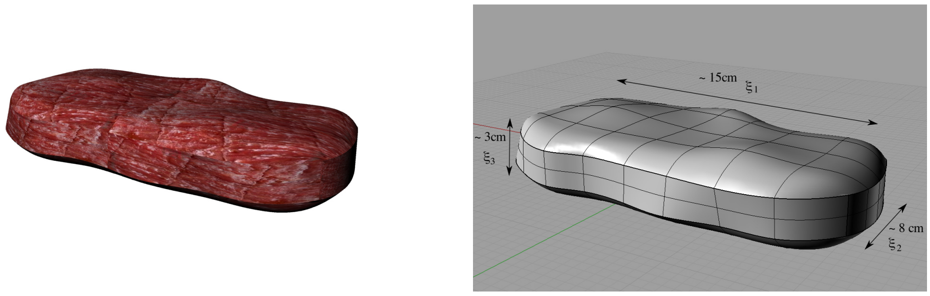
Do problem 5.3, but swap the experimentation with the following:

Another interesting question is with regards to ”harmonic” frequencies. The 1st nonzero frequency is often called the fundamental frequency, and the rest is called overtones. If the overtones are multiplies of the natural frequency (i.e. $f_i = nf_0$, where n is an integer and f_0 is the fundamental frequency) the sound is said to be harmonic. This is an important part in all musical instruments. More information can be found in the Wikipedia article on pitch.

5.5 Cooking beef

Do problem 5.1, but swap the experimentation with the following:

We are going to cook a beef in the best possible way. By alternating the boundary conditions, we are able to simulate either cooking this in a frying pan or in the oven. Simulate the cooking process and find the optimal way of prepearing your meat.



(a) The physical beef geometry

(b) The computational finite element mesh

Figure 11: The geometry which you are going to solve the heat equation on

The boundary conditions are the physical variables which we have control over. The initial condition $u(t, x, y, z)|_{t=0}$ is the beef as it is prior to any cooking. A proper choice here would be room temperature, say 20°C.

The actual cooking will be a product of the dirichlet boundary conditions. Frying the beef on a pan will result in a high (how high?) temperature on the bottom and room temperature on the other sides of the beef. What should be done to turn the beef and fry the other side? When should we turn it? Cooking it in an oven would result in a uniform boundary conditions on *all* sides of say 225°C. How long will it have to stay in? Is it a good idea to keep it in room temperature after cooking (and how does this change the boundary conditions)? More exotic cooking techniques include wrapping it in plastic and putting it in a water bath (not boiling) for some time, and only frying it on a pan for seconds prior to serving. This is called Sous-Vide.

Experiment around by cooking it in a number of ways using different boundary conditions. The optimality criterion is left up to the student. How well is your optimal beef cooked?

(**) Physical proprties of meat

It is hard to generalize too much on the physical properties of the beef as they are dependant on a number of variables outside the scope of this task. Not only are they dependant on the meat composition (i.e. what primal cut it is derived from), but it is also dependant on the temperature. Try and find good approximations for these numbers. A start may be the work of Pan and Singh (”Physical and Thermal Properties of Ground Beef During Cooking”) which suggests that the density ρ is in the range 1.006 to 1.033 g/cm^3 and the thermal conductivity κ in the range 0.35 to 0.41 W/m·K. The specific heat capacity is not mentioned in the abstract, but may be commented on in the actual article for those that get their hands on the entire document.

Unconfirmed sources list the specific heat capacity c_p of meat as 3 973 J/kg·K. You may use these values, or better yet: find more reliable, documented values.

5.6 What if?

What if a huge mountain - Denali, say - had the bottom inch of its base disappear? What would happen from the impact of the mountain falling 1 inch? What about 1 foot? What if the mountain's base were raised to the present height of the summit, and then the whole thing were allowed to drop to the earth?

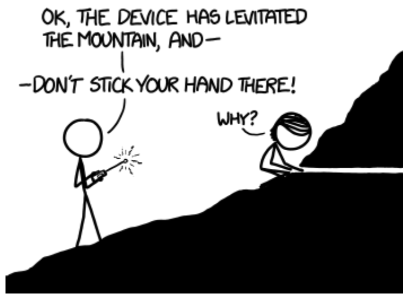


Figure 12: Don't stick your hand in there

Randall Munroe, the author of the webcomic XKCD, has a blog titled "what if". In short these are absurd questions answered in a scientific and accurate sense. It is a goldmine of inspiration and you might find interesting questions or cases to study from this blog. A link to series is included below

Figure 13: http://what-if.xkcd.com/57/

5.7 Custom game

a) Equation

Choose any equation of the above, or perhaps your own (preferably linear) equation and discretize this in a finite element framework. Add appropriate boundary conditions.

b) Geometry

Create a custom geometry using the matlab function `delauney`, the free software gmsh or any other method you would like.

c) Solve problem

Assemble all matrices, and solve all system of equations.

d) Conclusions

Plot the results in GLview, Paraview or Matlab. Experiment around with different boundary conditions, geometry or material parameters to do an investigation of your choice.