Delft University of Technology

Finite Elements Method

AP3001-FE

Lab Assignment

Authors:
Hemant Sharma (5379113)
Smit Chaudhary (5373900)

November 13, 2020

Contents

1 Introduction and Problem Statement

We will model a tsunami using Finite Elements Method. We do so under 2 different constraints. At first, we model our world in 2 dimensions where the end of region denote "end of the world" and we impose absorbing Robin boundary condition. In the second part, to be a little more realistic, we still remain in a 2 dimensional flat world but we enforce periodic boundary conditions on it. We can have absorption or no absorption even with a periodic boundary condition.

The way we go about this is we derive the weak formulation of the wave equation and get the Galerkin equations. We then use this to calculate the local element matrices and element vectors and then assemble all those local elements into a global finite element matrix. This gives us a system of linear equations which can easily be solved to get a finite element approximation of the phenomenon.

Mathematically, what we want to solve can be stated as follows:

Let Ω' denote the surface of earth. One can divide that into the surface covered by land and the surface covered by the oceans. Note that for our simulation, the surface covered by land is not relevant. Thus let Ω be the surface that is covered by the oceans.

The boundary $\partial\Omega$ can be written as $\partial\Omega = \Gamma_1 \cap \Gamma_2$. Here, Γ_1 denotes the boundary that is between the oceans and the land masses. Similarly Γ_2 denotes the boundary between the oceans and the end of our modeling region.

The pressure u associated with the wave follows the Helmholtz equation:

$$-\nabla^2 u - k^2 u = f \text{ on } \Omega \tag{1}$$

k denotes the wave number of the wave. The function f is the source of the wave and here, we take that to be a quickly decaying bell function centered at the epicenter of the tsunami.

We consider that the waves die at the coastal lines since you can't have a tsunami on land. That translates to :

$$u = 0 \text{ on } \Gamma_1 \tag{2}$$

To enforce absorbing boundary condition at the end of our simulation region, we need to have

$$\frac{\partial u}{\partial n} + iku = 0 \text{ on } \Gamma_2 \tag{3}$$

Now, if we want to enforce periodic boundary condition for a 2-dimensional system, we would have to put the following conditions on Γ_2:

$$u(\overline{x}(\text{right end}) = u(\overline{x}(\text{left end})) \tag{4}$$

$$u(\overline{x}(\text{north end}) = u(\overline{x}(\text{south end})) \tag{5}$$

We use gmsh to create an approximated outline of the land on the surface of earth.

2 Mathematical Model

2.1 Weak Formulation

Our tsunami is given by a wave which follows Helmholtz equation on Ω. Furthermore, we are given boundary conditions on the coasts and "end of the world". We use finite element methods to find an approximate solution of the equation with the given constraints.

First, we find the weak formulation associated with the given differential equation and its boundary condition. To do that, we multiply the equation with a variable ϕ and integrate the resulting equation on Ω. After integration by parts we get the following equation.

Find u that is sufficiently smooth, $u \,\epsilon\, \Sigma$, such that:

$$-\int_\Gamma \phi \frac{\partial u}{\partial n} + \int_\Omega \nabla\phi\nabla u - \int_\Omega k^2\phi u = \int_\Omega \phi f \quad \forall\phi \,\epsilon\, \Sigma \tag{6}$$

where, Σ is a set given by:

$$\Sigma = \{u : \Omega \to \mathbb{R}, \text{u is suff. smooth, u} = 0 \text{ on } \Gamma_1\}$$

Adding the boundary conditions to this equation, we get that the first term is zero on the first boundary but has a different on the second boundary, so the equation is changed to:

$$\int_{\Gamma_2} \phi iku + \int_\Omega \nabla\phi\nabla u - \int_\Omega k^2\phi u = \int_\Omega \phi f \quad \forall\phi \,\epsilon\, \Sigma \tag{7}$$

So far, we have not put any constraints on ϕ. It is clear to see that we do not need to put any restrictions on ϕ, it just has to be sufficiently smooth on the surface and boundary.

2.2 Galerkin Equations

To find the Galerkin equations, we use the following substitution:

$$\Sigma \to \Sigma^n$$

$$\phi \text{ is restricted to } \phi_1,\ \phi_2,\ ...,\ \phi_n$$

$$u \to u^n = \sum_{j=1}^{n} c_j\phi_j \,\epsilon\, \Sigma^n$$

After these substitutions, our equation is transformed into a sum over all c_js

$$\sum_{j=1}^{n} c_j(\int_{\Gamma_2} ik\phi_i\phi_j + \int_\Omega \nabla\phi_i\nabla\phi_j - \int_\Omega k^2\phi_i\phi_j) = \int_\Omega \phi_i f \quad \forall \text{ i} = 1,\ 2,\ ...,\ \text{n} \tag{8}$$

This can be written more succinctly as

$$\sum_{j=1}^{n} (S_{ij} + T_{ij})\, c_j = F_j \ \forall \text{ i} = 1,\ 2,\ ...,\ \text{n} \tag{9}$$

Here, S_{ij} denotes the integral over the domain Ω and T_{ij} denotes the integral over the domain Γ_2

3 Numerical Model

We have already argued above that we are looking for $u^n = \sum_{j=1}^{n} c_j\phi_j \ \epsilon$

Now, to numerically solve these, we have to choose the basis functions, namely ϕ_i so that the analysis becomes convenient. Firstly, note that we divide our region in small triangular elements. And we approximate the function u as u^n where u^n is a piece-wise linear function.

Thus, our function u^n is such that it is linear in each element.

To do this, we choose the basis functions such that they are non-zero only for one element at a time.

Having established the nature of the basis functions, let us utilise their simplicity to calculate the matrices S and T and the vector F

Note that $\int_\Omega \nabla\phi_i\nabla\phi_j - \int_\Omega k^2\phi_i\phi_j)$ is zero most of the times since either ϕ_i or ϕ_j or both are zero in most of the domain.

This prompts us to define an element matrix. Thus, we define a small local matrix where we consider only the relevant indices of the nodes such that it is not identically zero.

For a triangular mesh, we would have 3 such points. Thus, one would get

$$S_{ij}^{el} = \sum_{i,j=1,2,3} \left(\int_{el} \nabla\phi_i\nabla\phi_j - \int_{el} k^2\phi_i\phi_j \right) \tag{10}$$

Thus, this small 3×3 matrix is a small element matrix as is denoted by the 'el' in the superscript.

In the same manner, one can calculate T_{ij}^{el} and F_i^{el} as

$$T_{ij}^{el} = \sum_{i,j=1,2,3} \left(\int_{el} ik\phi_i\phi_j \right) \tag{11}$$

$$F_i^{el} = \sum_{i,j=1,2,3} \int_{el} \phi_i f \tag{12}$$

These local dense element matrices and vector contribute to the global and sparse matrices and vectors. We need to assemble them by adding the contributions of all the local element matrices.

One last aspect is calculating these local contributions themselves. For this, we again leverage the fact that these basis functions are piece-wise linear.

$$\phi_i = ax + by + c \tag{13}$$

To calculate the integral where we get ϕ, we use Newton-Cotes method to calculate the integral. Additionally, the terms where we get $\nabla\phi$, one gets

$$\nabla\phi = [a, b]^T \tag{14}$$

The exact implementation and the ways to calculate certain parameters (for eg. the area of the element), see the implementation of code.

4 Implementation

4.1 Geometry generation

We used paint to get the coordinates for different points on the atlas. Paint works in the fourth quadrant, similarly our map is in fourth quadrant. Our x varies from 0 to 1000 and y varies from -570 to 0. Using these value we constructed the .geo file, from this we generated the .msh file. We got the following mesh:

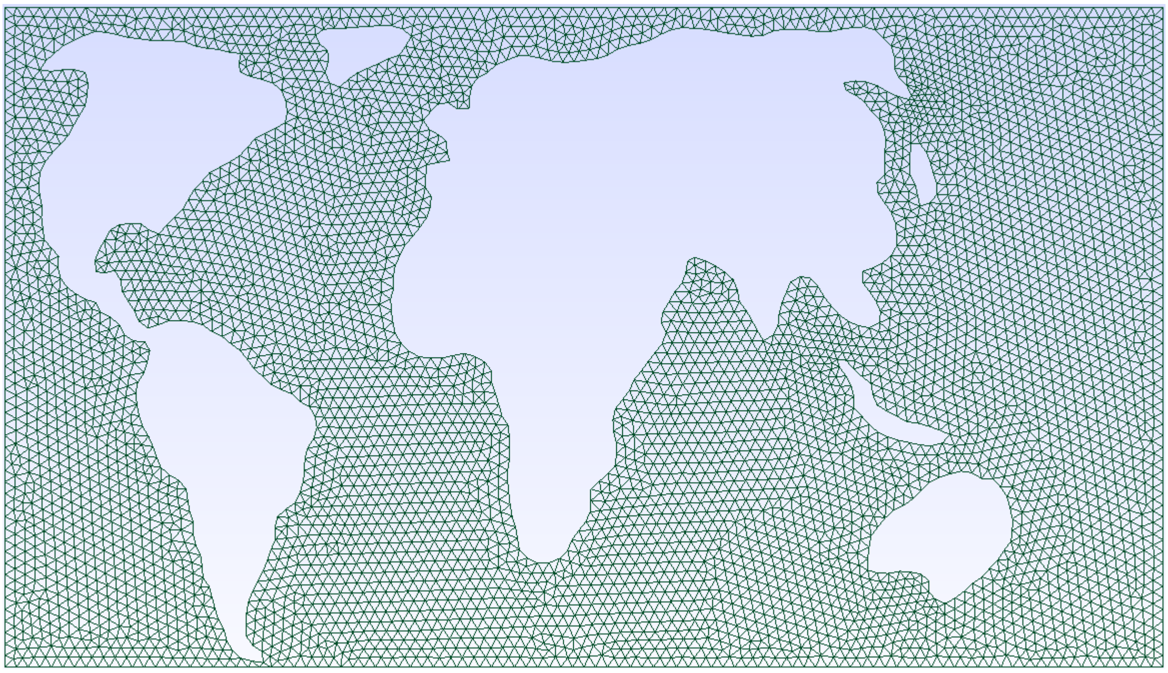

Figure 1: World after Meshing

We generated similar meshes with different size of elements, but for finer meshes, we got an error because of limited memory.

4.2 Implementation of code

4.2.1 Without periodic boundary conditions

Firstly, we read the mesh to take out border, sea and the coast. Next, since we have $u = 0$ on Γ_1, we define `interior_vertices` which has a bit misleading name since actually, these are vertices not only in the interior but also on the border of our simulation region.

Further, we create a function that maps local indices to global indices which will be necessary to assemble the global matrix from the local matrices.

Next, we calculate the element matrix. As promised in the numerical analysis part, here we will see how we actually calculate the integral that forms the elementary matrix. Note that there are 2 terms in the elementary matrix. One that includes the dot product of two gradients and another that consists the product of two functions. We can directly calculate the gradients and their dot products. From the coordinates of the vertices of the element, we also calculate the area of the element that is to be used in Newton-Cotes formula.

Then we add the contributions of all the terms to the global matrix using the function `assemble_matrix`

Following the exact same procedure but this time for the calculation of the vector, we perform the calculation and assemble a full global vector.

On the same lines as the functions `elementmatrix` and `assemblematrix`, we define the functions `elementmatrix_boundary` and `assemblematrix_boundary` which are needed to calculate the matrix T. Here, the key is to only have 2 vertices since only 2 of an element can be on boundary. Except that, the procedure is same as before.

Then we define the function $f(\overline{x})$. We model this as a quickly decaying Gaussian in 2-dimensions. Note that this is the same function as in

$$-\nabla^2 u - k^2 u = f \text{ on } \Omega \tag{15}$$

Next, we calculate all these matrices and vectors, assemble them into a global system and then calculate u^n from

$$u^n = (S + T)^{-1} F \tag{16}$$

4.2.2 With periodic boundary conditions and absorbing boundary conditions

We perform the task of implementing periodic boundary conditions in 2 different ways. Firstly, here we do not get rid of the absorbing boundary condition. Once we establish how to enforce periodic boundary condition, getting rid of the absorption term, that is the term with $\imath$k will be fairly straightforward.

Adding to the previous work, we create a function that locates the points which are on the boundary. We simply check if the x and/or y coordinates of the point are on the boundary of our region. And then we arrange them into arrays such that the i^{th} point of array for the left end and the i^{th} point of the array for the right end correspond to one another.

Then, we define a function called `make_periodic` which replaces the points on one end with the corresponding point on the other end to enforce periodicity.

But this poses a problem for us. If two sets of points are practically identical, this would also mean that the matrix S would have two columns which are identical. This translates into saying that the matrix S is not invertible and thus it is not possible to calculate

$$u^n = (S + T)^{-1} F$$

Thus, we have to be careful and remove that set of points from our consideration. Thus, we get rid of the points on the east and the south side.

Then here again, same as before we perform $u^n = (S + T)^{-1} F$

4.2.3 With periodic boundary conditions and without absorbing boundary conditions

Here, we just make the slight modification that we do not need to calculate the term with $\imath$k

The rest of the procedure as same as before.

5 Results

5.1 Tsunami without periodic boundary conditions

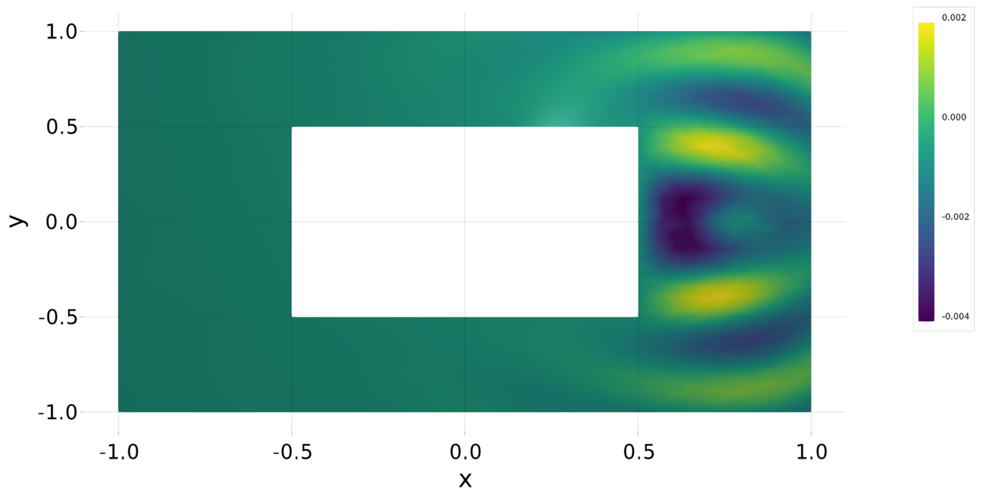

Figure 2: Tsunami with a basic mesh

In the above case, we chose (0.75, 0) as the epicenter of the earthquake. It is clear to see that there is nothing on the left side of the mesh, because we are not using periodic boundary conditions.

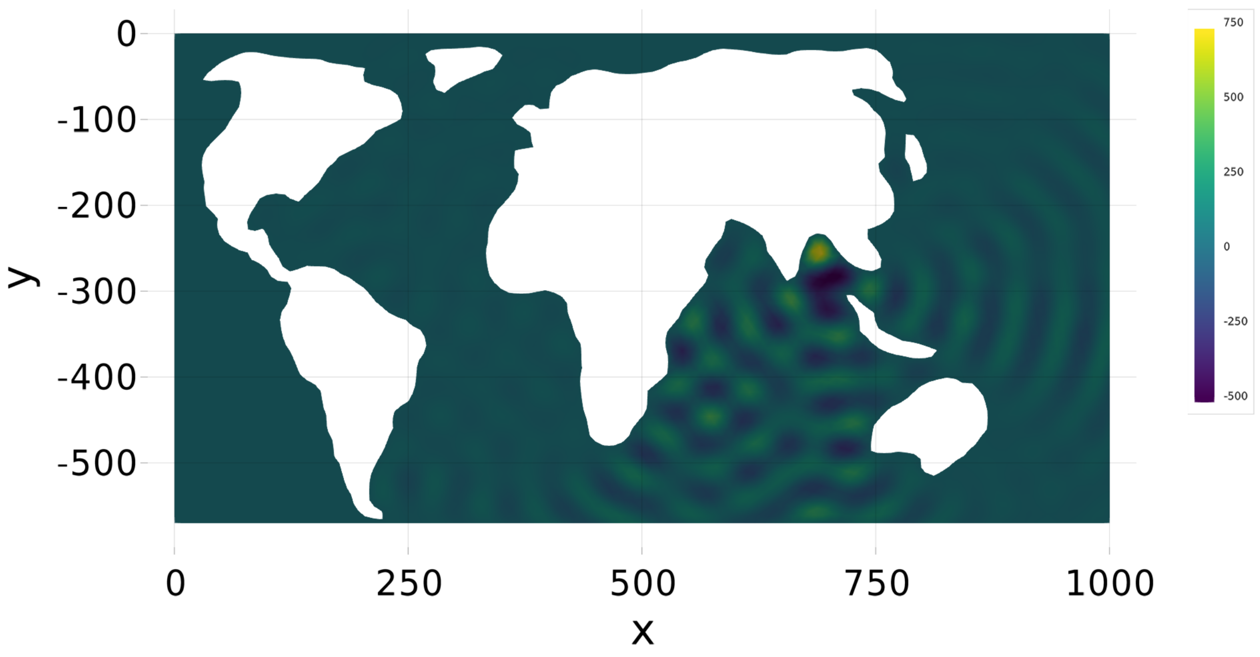

Figure 3: Tsunami in map of the world

Here, we generated the tsunami near Malaysia and India. The wavelength was close to 2000 Km (considering circumference at equator is 40075 Km, and we go round there in 1000 steps). Because of non periodic nature of boundaries, we don't see waves going away from Indonesia reaching Chile or California.

5.2 Periodic Boundary Conditions

Here we have kept the wavelength of the wave close to 4000 Km. And the epicenter of the earthquake was in South China Sea. We can see that the wave goes across the pacific ocean to Chile, Peru and California.

5.2.1 With absorbing boundaries

We did not remove the absorbing boundary condition on Γ_2, on our first try. We can see here, that the wave is travelling across the edge of the world to the other side but the wave on the "other side" is damped.

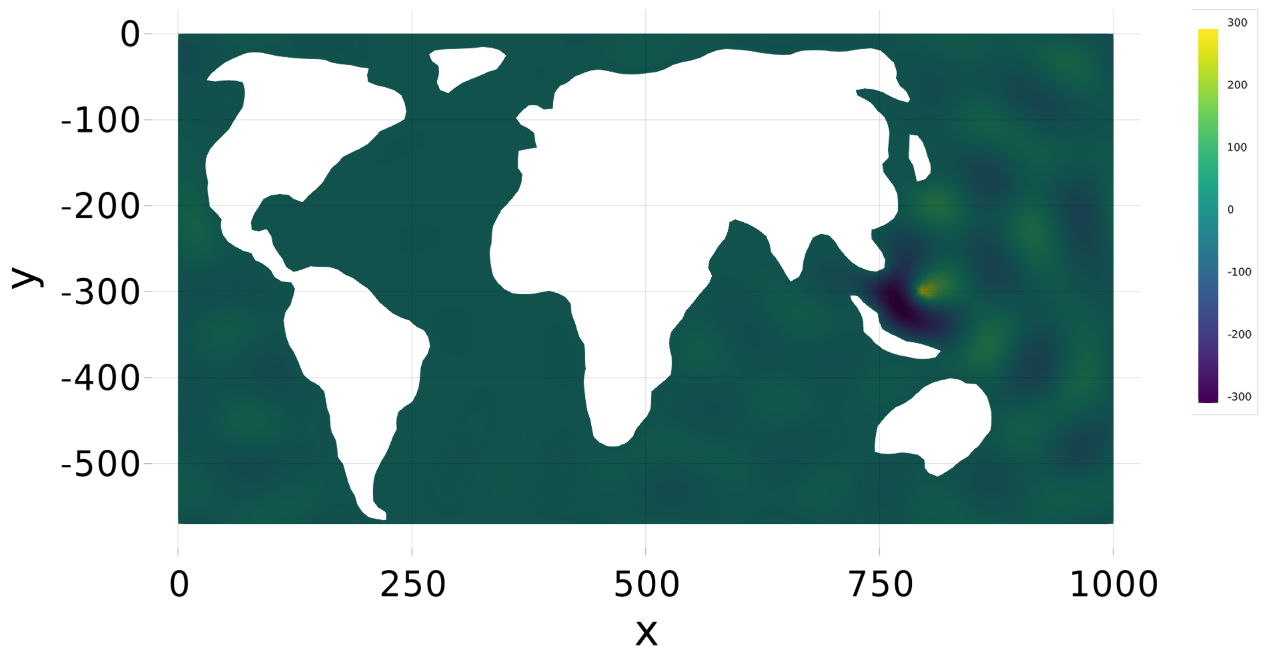

Figure 4: Tsunami with periodic boundary condition and absorbing Boundary conditions

5.2.2 Without absorbing boundaries

Now, we move to a more realistic case where the "end of the world" boundaries do not absorb the wave,.

We can observe that the wave travels much better(as compared to the case where we still had contribution from boundary conditions) from Indonesia to California and South America, and from South of the map to north of the map.

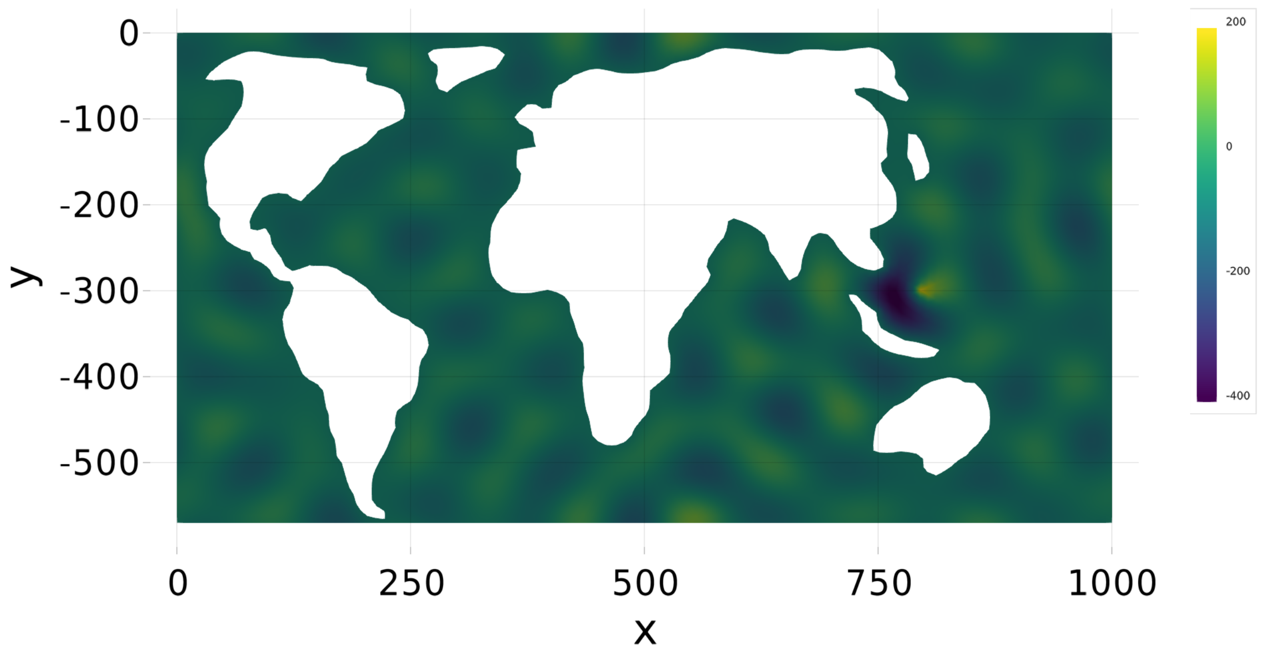

Figure 5: Tsunami with periodic boundary condition and without absorbing Boundary conditions

6 Future Work

In this work we have used Mercator projection of the earth. Thus we have completely ignored the curvature of the earth. For a future work, we could use a different projection of the globe, which approximates the earth more accurately. Or we could use a simple 3-D finite element method to simulate the wave on a globe directly.