



Norwegian University of
Science and Technology

TDT24

Short Introduction to Finite Element Method

Gagandeep Singh

Contents

1	Introduction	2
2	What Is a Differential Equation?	2
2.1	Boundary conditions	3
2.2	Solution Methods	3
2.3	Finite Difference	5
2.3.1	Differential Formula	5
2.4	Finite Element	6
2.4.1	Strong Formulation: Poission's Equation	7
2.4.2	Motivation	8
2.4.3	Weak Formulation: Poission's Equation	8
2.4.4	Advantages of Weak form Compared to Strong Form	9
3	Basic Steps in FE Approach	9
3.0.5	Approximating the Unknown u	9
3.0.6	What is Approximation in FE?	10
3.0.7	Basis Functions	11
3.0.8	Basis Functions 2D	13
4	A Practical Example	16
4.1	The Poisson Equation	16
4.2	Minimization formulation	16
4.3	Weak formulation	17
5	Geometric representation	17
5.1	Element Integral	17
5.2	Load Vector	19
6	Did we understand the difference between FE and FD?	20
7	Finite Element on GPU	21

1 Introduction

In this short report, the aim of which is to introduce the finite element method, a very rigorous method for solving partial differential equations (PDEs), I will take you through the following:

- What is a PDE?
- How to solve a PDE?
- Introduce finite difference method
- Introduce finite element method.
- A practical example using Laplace Equation.
- Fundamental difference from finite difference.

2 What Is a Differential Equation?

Popular science could describe a partial differential equation as a kind of mathematical divination or fortune-telling. In a PDE, we always have a mathematical expression describing the relation between dependent and independent variables through multiplications and partials. Along with this expression we have given initial conditions and boundary requirements. Solution of the PDE can then use this little information, and give us all the later states of this information. This is just like a fortuneteller who asks you about some information and tells you what's going on in your life. Well, I do not need to tell the difference between a PDE and fortuneteller. The relation must be something like "true" and the negation of it.

It is not easy to master the theory of partial differential equations. Unlike the theory of ordinary differential equations, which relies on the "fundamental existence and uniqueness theorem", there is no single theorem which is central to the subject. Instead, there are separate theories used for each of the major types of partial differential equations that commonly arise.

Mathematically, a partial differential equation is an equation which contains partial derivatives, such as the (wave) equation

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} \tag{1}$$

In Eq (1), variables x and t are independent of each other, and u is regarded as a function of these. It describes a relation involving an unknown function u of several independent variables (here x and t) and its partial derivatives with respect to those variables. The independent variables x and t can also depend on some other variables, in which case we must carry out the partial derivatives of x and t .

Another simple elliptic model equation is the Poisson Equation, given by

$$\Delta u = \nabla^2 u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f, \quad \text{defined on } \Omega \in \mathbb{R}^2 \tag{2}$$

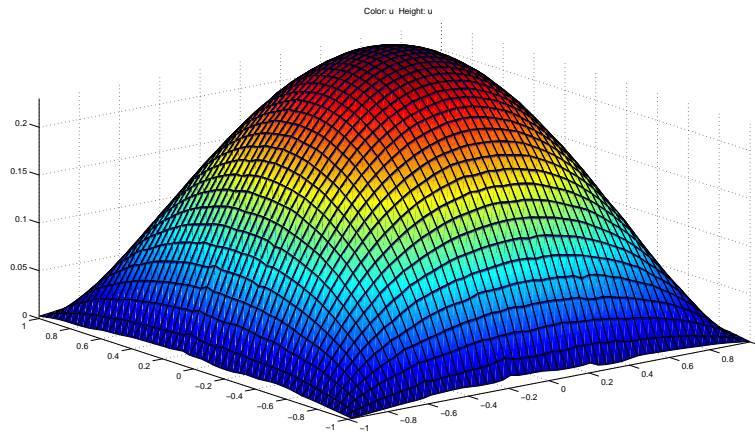


Figure 1: Solution of Eq (2) with $f = 1$

We will use this equation in the practical example and when introducing the finite element method. Notice that u depends symmetrically on x and y . So, if we choose a symmetric Ω , the solution will be symmetric. If $\Omega \in (-1, 1)^2$, the solution u will have the form shown in Fig 1. Here we have used $f = 1$.

2.1 Boundary conditions

In Fig 1 I have used $u = 0$ on the entire boundary, that is, along the lines $(x, \pm 1)$ and $(\pm 1, y)$. Using a constant value along the entire boundary is called Homogeneous Dirichlet. When we use $u = g(x, y)$ along the entire boundary, it is called Inhomogeneous Dirichlet. It is also possible to specify a constant solution $u = const$ (or $u = g(x, y)$) along a (continuous) part of the boundary. On the rest of the boundary we can specify flux variation using so-called Neumann boundary conditions. This is called Mixed boundary conditions. Neumann boundary conditions specify the *directional* derivative of u along a normal vector. We denote it as $\frac{du}{dn} = \vec{n} \cdot \nabla u = g$. Function $g = g(x, y)$ is given and in the end we have known values of u at some (continuous) part of the boundary and the directional derivative on another. Fig 3 shows another solution of Eq (2) using Inhomogeneous boundary conditions on $\Omega \in (-1, 1)^2$. Boundary conditions are shown in Fig 2. There are also some other boundary conditions, but these are the most common ones occurring in physical models and simulations. When we introduce finite element method, we will show how these conditions are fundamental and reflects in the final formulation.

2.2 Solution Methods

Depending on the nature of the physical problem and the corresponding mathematical formulation, we can use between a range of solution methods. The most known solution method is *finite difference method*, which essentially replaces every occurrence of partials with a discrete approximation using grid information. The finite difference method may be a good method to use when the geometry is sim-

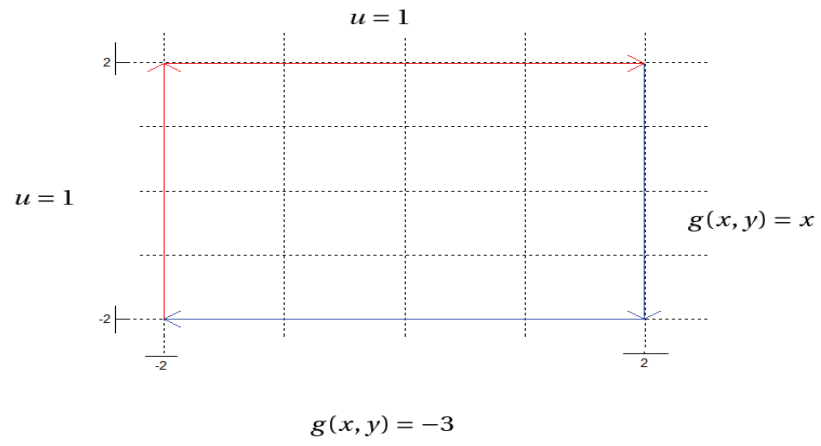


Figure 2: Boundary conditions on $\Omega \in (-2, 2)^2$

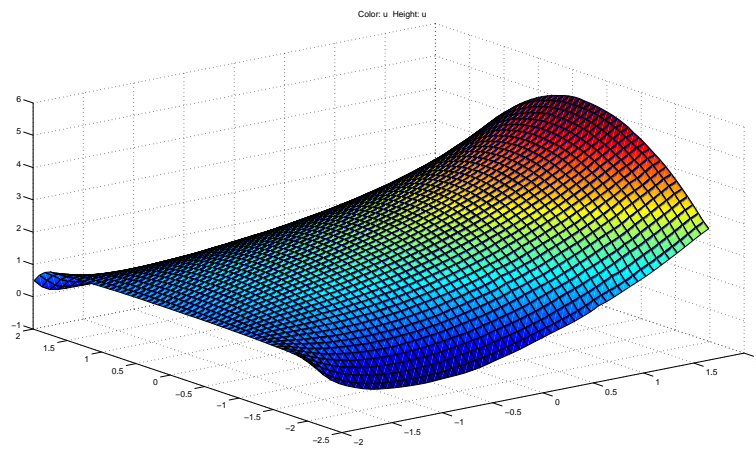


Figure 3: Solution of Eq (2) using boundary conditions shown in Fig 2

ple and not very high accuracy is required. But, when the geometry becomes more complex, finite difference becomes unreasonably difficult to implement.

The finite element method is conceptually quite different from finite difference when it comes to the discretization of the PDE, but there are some similarities. It is the most commonly used method to solve all kinds of PDEs. Matlab has *Partial Differential Equation Toolbox* which implements finite element with pre-specified known PDEs. COMSOL is also based on finite element method.

In the next section, I will mention some few points on finite difference, then go straight to finite element method. We will then be in a position to discuss some of the differences between finite difference and finite element.

2.3 Finite Difference

In Eq (2), we have an operator working on u . Let us denote this operator by L . We can then write

$$L = \nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \quad (3)$$

Then the differential equation can be written like $Lu = f$. If for example $L = \nabla^2 - 2\nabla + 2$, the PDE becomes $\nabla^2 u - 2\nabla u + 2u = f$. Finite difference methods are based on the concept of approximating this operator by a discrete operator.

2.3.1 Differential Formula

Let me remind you of Taylor's formula with the residual part of the functions of two variables:

$$u(x+h, y+k) = \sum_{m=0}^n \frac{1}{m!} \left(h \frac{\partial}{\partial x} + k \frac{\partial}{\partial y} \right)^m u(x, y) + r_n \quad (4)$$

where

$$r_n = \frac{1}{(n+1)!} \left(h \frac{\partial}{\partial x} + k \frac{\partial}{\partial y} \right)^{n+1} u(x+\theta h, y+\theta k), \quad 0 < \theta < 1 \quad (5)$$

In (4), I have used step lengths h and k in x and y direction, respectively. Let us use $h = k$ in the following for simplicity. Let us now develop this series of $u(x+h, y)$. I denote it by $u^{m+1, n}$. The rest follows from it.

$$u^{m+1, n} = u^{m, n} + h u_x^{m, n} + \frac{h^2}{2!} u_{xx}^{m, n} + \frac{h^3}{3!} u_{xxx}^{m, n} + O(h^4) \quad (6)$$

The subscript in u_x refers to the partial derivative of u with respect to x . Now, let us develop the series of $u^{m-1, n}$, $u^{m, n+1}$, and $u^{m, n-1}$.

$$\begin{aligned} u^{m-1, n} &= u^{m, n} - h u_x^{m, n} + \frac{h^2}{2!} u_{xx}^{m, n} - \frac{h^3}{3!} u_{xxx}^{m, n} + O(h^4) \\ u^{m, n+1} &= u^{m, n} + h u_y^{m, n} + \frac{h^2}{2!} u_{yy}^{m, n} + \frac{h^3}{3!} u_{yyy}^{m, n} + O(h^4) \\ u^{m, n-1} &= u^{m, n} - h u_y^{m, n} + \frac{h^2}{2!} u_{yy}^{m, n} - \frac{h^3}{3!} u_{yyy}^{m, n} + O(h^4) \end{aligned} \quad (7)$$

If we suppose $u_{xx}^{m,n}$ is small compared to $u_x^{m,n}$ and $h \rightarrow 0$, we see from Eq (6) and the first equation in (7) that

$$\begin{aligned} u_x^{m,n} &= \frac{1}{h} [u^{m+1,n} - u^{m,n}] + O(h) && \text{Forward difference} \\ u_x^{m,n} &= \frac{1}{h} [u^{m-1,n} - u^{m,n}] + O(h) && \text{Backward difference} \end{aligned} \quad (8)$$

In the same way, we get

$$\begin{aligned} u_y^{m,n} &= \frac{1}{h} [u^{m,n+1} - u^{m,n}] + O(h) && \text{Forward difference} \\ u_y^{m,n} &= \frac{1}{h} [u^{m,n-1} - u^{m,n}] + O(h) && \text{Backward difference} \end{aligned} \quad (9)$$

In the same way, if we suppose $u_{xxx}^{m,n}$ is small compared to $u_{xx}^{m,n}$ and $h \rightarrow 0$, we see from Eq (6) and the first equation in (7) that

$$u_{xx}^{m,n} = \frac{1}{h^2} [u^{m+1,n} - 2u^{m,n} + u^{m-1,n}] + O(h^2) \quad \text{Central difference} \quad (10)$$

We also get a second order central difference in y from second and third equation in Eq (7)

$$u_{yy}^{m,n} = \frac{1}{h^2} [u^{m,n+1} - 2u^{m,n} + u^{m,n-1}] + O(h^2) \quad \text{Central difference} \quad (11)$$

Let us look at Poisson Eq (2). It contains two separate second derivatives, and we can use Eq (10) and (11) to approximate these with $O(h^2)$.

$$\nabla^2 u = u_{xx} + u_{yy} = u_{xx}^{m,n} + u_{yy}^{m,n} + O(h^2) \quad (12)$$

This expression gives us the well-known five point formula. One thing this expression tells us, is that in order to use finite difference methods like this one, we need a relatively structured grid. Otherwise, we will have difficulties defining the direction of step length.

In the next section, I will introduce finite element method. I will start from what is called "point of departure" for the element method.

2.4 Finite Element

One remark that can be made already, is that the finite element method is mathematically very rigorous. It has a much stronger mathematical foundation than many other methods (It has a more elaborate mathematical foundation than many other methods), and particularly finite difference. It uses results from real and functional analysis. I will not touch any of this in this short introduction. We will take the path of engineers; forget the mathematical foundation and use the final result.

In the practical example in the end, I have included what we call *the minimization principle*. It looks like the one used when deriving the Conjugate gradient method by minimizing a quadratic function. The answer to this minimization function is our solution. But there is another result, called *Weak formulation*, which, when true,

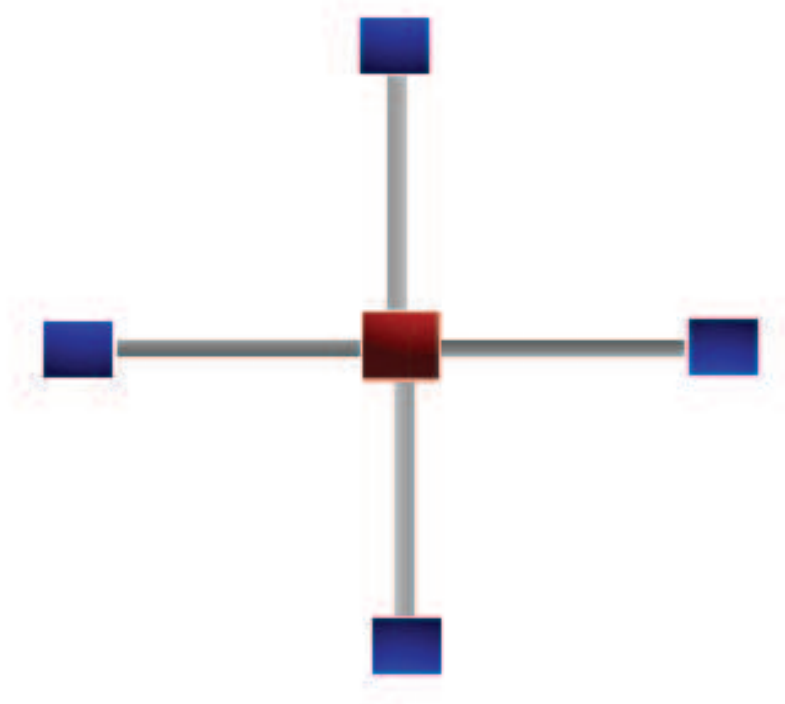


Figure 4: The five point stencil for the Poisson Equation

also makes the minimization formulation true. So, we will start at the weak formulation and discuss the results we arrive at. I will leave the minimization formulation in the practical example for those of you who may like minimization principles.

Finite Element (FE) is a numerical method to solve arbitrary PDEs, and to achieve this objective, it is a characteristic feature of the FE approach that the PDE in question is first reformulated into an equivalent form, and this form has the weak form.

2.4.1 Strong Formulation: Poisson Equation

Let us rewrite the Poisson Equation stated in the beginning and choose $\Omega = (0, 1)^2$. I intend to use only Homogeneous Dirichlet boundary conditions. This gives

$$\Delta u = \nabla^2 u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f, \quad \text{defined on } \Omega = (0, 1)^2 \quad (13)$$

$$u(\partial\Omega) = 0 \quad \partial\Omega = \{(x, y) | x = 0, 1 \text{ or } y = 0, 1\}$$

This is called the *strong formulation* in finite element, and says no more than the original PDE formulation. Still, there is a reason why we call it *strong*. We will discuss everything in terms of strong and weak forms.

2.4.2 Motivation

- The Poisson problem has a strong formulation; a minimization formulation; and a weak formulation.
- The minimization/weak formulations are more general than the strong formulation in terms of regularity and admissible data.
- The minimization/weak formulations are defined by; a space X , a bilinear form a and linear form l .
- The minimization/weak formulations identify:
 - ESSENTIAL boundary conditions - Dirichlet - reflected in X
 - NATURAL boundary conditions - Neumann - reflected in a and l

2.4.3 Weak Formulation: Poisson's Equation

As I wrote, the weak formulation is a re-formulation of the original PDE (Strong form), and it is from this form that the final FE approach is established. To establish the weak form of PDE Eq (13), multiply it with an arbitrary function, so-called *weight-function*, $v(x, y)$ to obtain

$$v\nabla^2 u = f v \quad (14)$$

We may even integrate this expression over Ω .

$$\int_{\Omega} v\nabla^2 u = \int_{\Omega} f v \quad (15)$$

I previously claimed that $v(x, y)$ is arbitrary. This is not completely true, since the manipulations that $v(x, y)$ undergoes should be meaningful. Let us define a space of functions and call it H^1

$$H^1(\Omega) = \left\{ v : \Omega \rightarrow \mathbb{R} : \int_{\Omega} v^2, \int_{\Omega} v_x^2, \int_{\Omega} v_y^2 < +\infty \right\} \quad (16)$$

H^1 is a function space where all the functions are bounded [quadratic integrable]. We want our functions to be well-behaved, so that we can define operations on them within the rules of (say) integration. Let us now define a sub-space of H^1 where we can find our solution u . We call this X and

$$X = \{ v \in H^1(\Omega) : v|_{\partial\Omega} = 0 \} \quad (17)$$

Before I explain to you why we have chosen $H^1(\Omega)$ as our function space, let us carry out the integration in Eq (15) a little further. Let $u, v \in X$. We know from calculus that $\nabla(v\nabla u) = \nabla v \cdot \nabla u + v\nabla^2 u$. We can write

$$\int_{\Omega} v\nabla^2 u = \int_{\Omega} \nabla(v\nabla u) - \int_{\Omega} \nabla v \cdot \nabla u \quad (18)$$

Using Gauss's theorem on $\nabla(v\nabla u)$ we get

$$\int_{\Omega} \nabla(v\nabla u) = \int_{\partial\Omega} \underbrace{v}_{v|_{\partial\Omega}=0} \nabla u \cdot \hat{n} dS = 0 \quad (19)$$

In Eq (19), we have transformed a surface integral to a line integral and dS refers to an infinitesimal line segment. I have not put dA on surface integrals, so be aware of that. Eq (18) now reduces to

$$\int_{\Omega} v \nabla^2 u = - \int_{\Omega} \nabla v \cdot \nabla u \quad (20)$$

and so, we get

$$- \int_{\Omega} \nabla v \cdot \nabla u dA = \int_{\Omega} f v dA \quad (21)$$

2.4.4 Advantages of Weak form Compared to Strong Form

This equation ((21)) is the final weak formulation. It is equivalent to the strong form posed above. You can reverse all the steps, and get back to the original equation. Let me explain the space H^1 . Firstly, if you look at the strong form, we have two separate partial derivatives of u , so the strong form requires that u be continuously differentiable until at least second partial derivative. Our new formulations has lowered this requirement to only first partial derivatives by transforming one of the partial derivatives onto the weight-function $v(x, y)$. This is the first big advantage of a weak formulation (now you can guess naming conventions). The subspace X is not difficult to understand; it is a subspace of H^1 because our weak form requires that the functions are in H^1 ; our strong form requires that u be 0 along the boundary, so X is the subspace of all function which are zero on the boundary. Now remember from the Motivation section above that I wrote "Dirichlet are essential and reflected in X ". This is automated through using weak form. There are several technical facts about Dirichlet conditions, but we will let them be.

3 Basic Steps in FE Approach

We are already half-way towards the final formulation [which will be a linear system $Au = b$]. These are the steps we follow

- Establish the strong formulation [finished]
- Obtain the weak formulation [finished]
- Choose approximations for the unknown functions, in this case u [next - 1]
- Choose the weight functions v [next - 2]
- Solve the system

3.0.5 Approximating the Unknown u

Let me state our route straight away; the region is divided into smaller parts and the approximation is carried out over these smaller parts and then, based on these results, it is established for the entire region. If we split the whole domain into smaller parts, i.e. finite elements, for which we are able to give an approximation of u , then we are able to approximate u for the whole geometry of any form.

So, we say that the "approximation is carried out elementwise", hence the name.

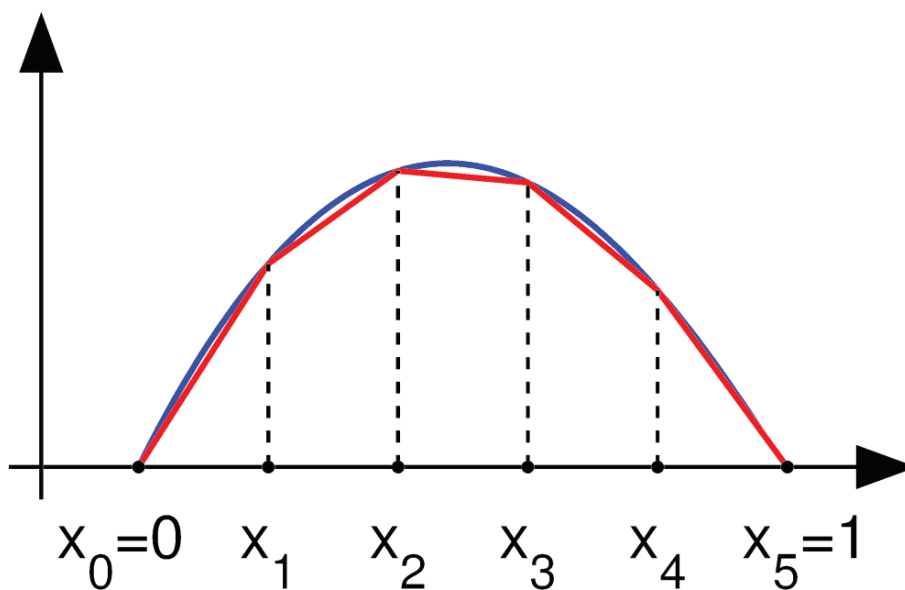


Figure 5: A continuous function (blue) approximated by a piecewise linear polynomial (red)

3.0.6 What is Approximation in FE?

To give you an idea of what we mean by approximation in this particular FE context, consider the function (blue line) in Fig 5. This function, call it g , is continuous as it looks. Say we have values of this function at equally distant points with step length h (equally distant for simplicity). Then, for example, we can approximate this continuous function by an approximate function (red line), call it \tilde{g} , such that they have equal values at these points. Between the points, we have a linear approximation of g . We see that \tilde{g} restricted to one element, denote it by $\tilde{g}|_T$, is a linear polynomial, so it has the form

$$\tilde{g}|_T = ax + b \quad a \text{ and } b \text{ constant} \quad (22)$$

The constants a and b vary from element to element, but they are constant. It is also possible to use quadratic and cubic polynomials restricted to one element, but we will use linear polynomials.

The problem we are working on is $2D$, so how would a two-dimensional approximation of a two dimensional surface look like if it were to be composed of linear approximations when restricted to elements? Well, first of all we need to know what kind of element shape we are using. In introductory courses, we see the triangular shape. Fig 6 shows a *mesh* of our domain Ω partitioned into triangles. The domain is first discretized into nodes, and then a well-defined triangulation is performed between the nodes. Elements with four and even more nodes are possible. The good thing with triangular elements is that they can approximate a huge range of geometries. Quadratic elements have difficulties at corners.

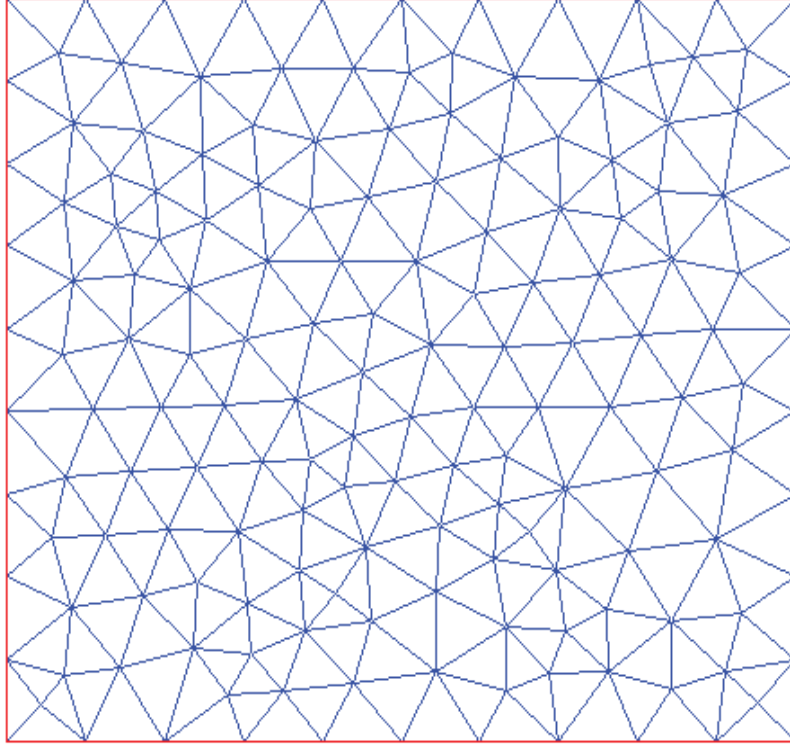


Figure 6: Domain Ω partitioned into finite triangular elements.

3.0.7 Basis Functions

Let us step back to 1D and look at some figures showing what in FE is called basis functions. In figures 5, we see the approximation (red line) of an underlying function (we called it g). Here we assumed that we knew this function, so that we could just make an interpolation between the given points (at equal distance) and make a piecewise continuous linear polynomial. Well, in the case of our differential equation, we do not know the function u .

Before I give any explanation, let us consider the same 1D function $g(x)$ function as shown in Fig 5. Let us define 6 equi-distant points $\{0 = x_0, x_1, x_2, x_3, x_4, x_5 = 1\}$. At the end-points $x_0 = 0$ and $x_5 = 1$, we have $g(x) = 0$. At the internal *nodes* $g(x) \neq 0$. Let us define four functions $\phi_i, i = 1, \dots, 4$, for the internal nodes. One characteristic for all these functions is the following.

$$\phi_i(x_j) = \delta_{ij} \quad (23)$$

Every $\phi_i(x)$ is equal to 1 only at the i_{th} node and zero at all the other ones. Another characteristic is that every $\phi_i(x)$ is only non-zero at element number i and $i + 1$, that

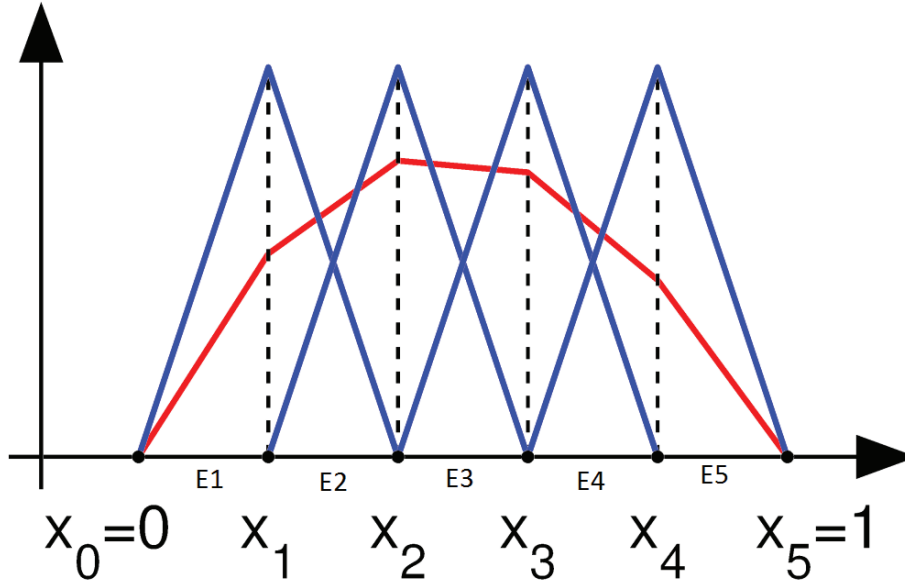


Figure 7: Basis functions for the six-node discretization of domain $(0, 1)$.

is, elements that share node i . Otherwise, all the functions look like Fig 7. Now, given these facts about these functions, suppose we have an underlying function $g(x)$ which we need to approximate using piecewise continuous linear polynomials at each of these elements. This is where these functions come and play a role. If you look closely at one element of Fig 7, you will see that each element has two functions, crossing each other at the middle and with opposite slopes. More specifically, for element i (written as Ei in the Figure), these are $\phi_i|_{Ei}$ (ϕ_i restricted to element Ei) and $\phi_{i-1}|_{Ei}$ (ϕ_{i-1} restricted to element Ei). These two restrictions (functions) can approximate an arbitrary linear polynomial on element Ei . Let us write our approximation \tilde{g} (red line in Fig 7) as a linear combination of these $\phi_i(x)$'s.

$$\tilde{g}(x) = \sum_{i=0}^5 g_i \phi_i(x) \quad (24)$$

Using this expression, we get

$$\tilde{g}(x_j) = \sum_{i=0}^5 g_i \phi_i(x_j) = \sum_{i=0}^5 g_i \delta_{ij} = g_j \quad (25)$$

Well, this tells us that the value of \tilde{g} at the nodes $x_i, i = 0, \dots, 5$, is equal to the coefficient g_i ($g_0 = 0$ and $g_5 = 0$). If we can find these coefficients, we can know the approximate values of $g(x)$ at the nodes. This is why we call these basis functions the *nodal basis*.

Finally and before we jump ahead to 2D, let me explain why I use the word *basis*

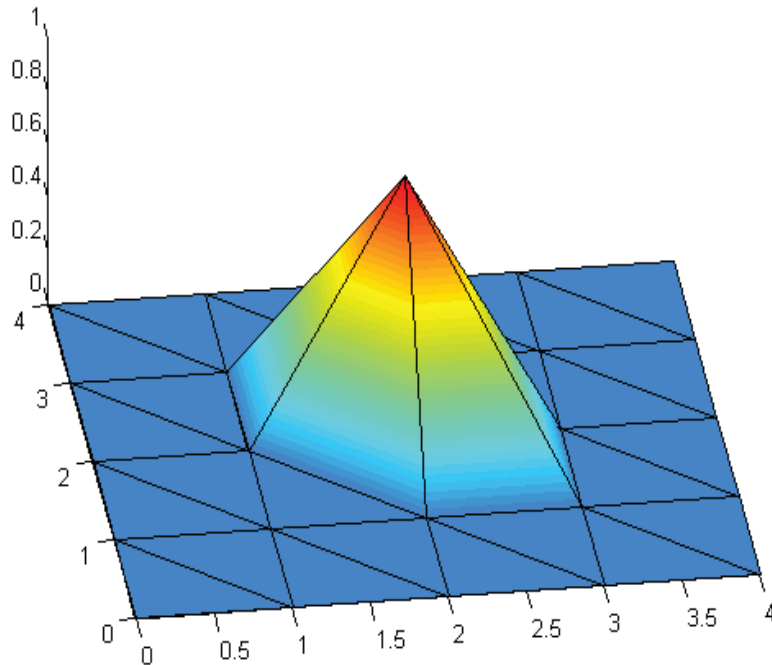


Figure 8: Basis function ϕ_i . It is only non-zero at elements that share this node.

functions. If we want to approximate our continuous function g with a piecewise continuous linear function \tilde{g} , these four functions (four for Fig 7) are what we need, not one less or one more. These functions are linearly independent of each other; it is not possible to make one out of a combination of others. For example, only one of these functions, ϕ_i , is non-zero (equal to 1) at node i .

3.0.8 Basis Functions 2D

Take a look at Fig 6. This is our *discretized* 2D-domain. It is made up of triangles and each triangle has three nodes. One node can be shared by several triangles. Each triangle is an element. Our final formulation of the FE approach will give us a linear system of equations with dimension $N \times N$, where N is the number of nodes. This means that after solving this system, we will find values of u at the nodes. This is directly comparable to the 1D problem of approximation we looked at in the previous section.

We need to define *basis functions* for our 2D-domain. As in 1D, we will have as many basis functions as we have nodes. Each node has one basis function. We denote these functions $\phi_i(x, y)$ for node number i . As in 1D $\phi_i(x_j, y_j) = \delta_{ij}$. Each $\phi_i(x, y)$ is non-zero only in elements which share node i . Fig 8 and 9 shows two such functions, one completely internal, and one which has interface to boundary. Principally, there is no difference between the two. Think only of the internal one if it confuses you.

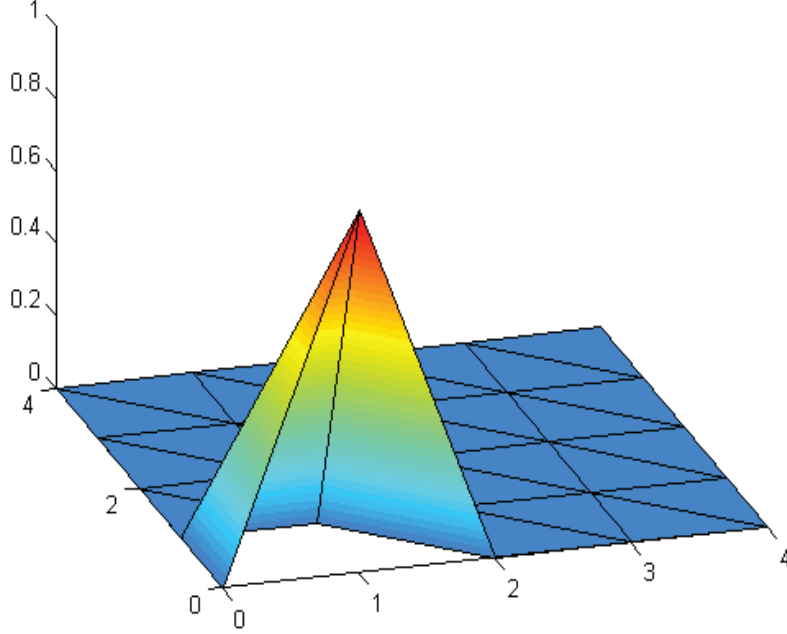


Figure 9: Basis function ϕ_i . It is only non-zero at elements that share this node.

Now let us go back to our weak form (21). Here we look for a function $u(x, y)$ which is continuously differentiable. Note that all of our ϕ_i functions are continuously differentiable. In our specific problem, we are in search of an approximate function $\hat{u}(x, y)$ which is piecewise linear on each element. As in 1D, we know that we can write this function as a linear combination of our basis functions.

$$\hat{u}(x, y) = \sum_{i=1}^N \hat{u}_i \phi_i(x, y) \quad (26)$$

And again,

$$\hat{u}(x_j, y_j) = \sum_{i=1}^N \hat{u}_i \phi_i(x_j, y_j) = \hat{u}_j \quad \text{Nodal basis} \quad (27)$$

To make the problem more convenient and easy to read, let us define two forms a and l as

$$a(u, v) \equiv \int_{\Omega} \nabla v \cdot \nabla u \quad \text{and} \quad l(v) \equiv \int_{\Omega} f v \quad (28)$$

Note that $a(u, v) = a(v, u)$. Using the integral rules, we also find that a and l are bilinear and linear forms, respectively. Now we can rewrite our weak form as

$$a(u, v) = l(v) \quad u, v \in X \quad (29)$$

Going towards our final formulation, let us express our approximation \hat{u} and our weight function (which is arbitrary) and in X v as

$$\begin{aligned}\hat{u} &= \sum_{i=1}^N \hat{u}_i \phi_i \\ v &= \sum_{j=1}^N v_j \phi_j\end{aligned}\tag{30}$$

The number N is the total number of nodes, including boundary nodes. Note that we are using v in terms of our basis functions. This is called the Galerkin method. We put these two expressions in our weak form (29) and complete the expression

$$\begin{aligned}a\left(\sum_{j=1}^N v_j \phi_j, \sum_{i=1}^N \hat{u}_i \phi_i\right) &= l\left(\sum_{j=1}^N v_j \phi_j\right) \\ &= a\left(v_1 \phi_1 + \sum_{j=2}^N v_j \phi_j, \sum_{i=1}^N \hat{u}_i \phi_i\right) \\ &= a\left(v_1 \phi_1, \sum_{i=1}^N \hat{u}_i \phi_i\right) + a\left(\sum_{j=2}^N v_j \phi_j, \sum_{i=1}^N \hat{u}_i \phi_i\right) \\ &= \sum_{j=1}^N a\left(v_j \phi_j, \sum_{i=1}^N \hat{u}_i \phi_i\right) \\ &= \sum_{j=1}^N \sum_{i=1}^N a(v_j \phi_j, \hat{u}_i \phi_i) \\ &= \sum_{j=1}^N v_j \sum_{i=1}^N a(\phi_j, \phi_i) \hat{u}_i = \sum_{j=1}^N v_j l(\phi_j)\end{aligned}\tag{31}$$

During and right before this long expression, particularly regarding the Galerkin method, I have skipped a number of details which may be important for the overall understanding of FE, but the intention here is the final formulation without deepening every step to the bottom. The last expression can be written in compact form as

$$v^T A \hat{u} = v^T F \implies A \hat{u} = F\tag{32}$$

Here,

$$\begin{aligned}v &= [v_1 \ v_2 \ \dots \ v_N]^T \\ \hat{u} &= [\hat{u}_1 \ \hat{u}_2 \ \dots \ \hat{u}_N]^T\end{aligned}\tag{33}$$

A and F are given as follows

$$A = \begin{bmatrix} a(\phi_1, \phi_1) & a(\phi_1, \phi_2) & \dots & a(\phi_1, \phi_N) \\ a(\phi_2, \phi_1) & a(\phi_2, \phi_2) & \dots & a(\phi_2, \phi_N) \\ \vdots & \vdots & \vdots & \vdots \\ a(\phi_N, \phi_1) & a(\phi_N, \phi_2) & \dots & a(\phi_N, \phi_N) \end{bmatrix}$$

$$F = [l(\phi_1) \ l(\phi_2) \ \dots \ l(\phi_N)]^T$$

One thing we could have shown at this point is that the system $A\hat{u} = F$ is not SPD (symmetric positive definite), a property that is a prerequisite for the Conjugate gradient method. The reason for this is that we have not yet specified the boundary conditions. We have just set up the system for all nodes. We need to change this system so that $u = 0$ along the boundary. Remember that we had $u = 0$ in our strong form. Remember also that we have $f = 1$, so that there is no need to use quadrature or any other numerical technique to calculate the $l(\phi_i)$ integral, since this integral is easy to calculate.

If our node numbering is given as $1, 2, 3, \dots, N$, we easily change our linear system to an SPD by setting $\hat{u}_i = 0$ for all nodes i which are boundary nodes. If for example the fifth node is a boundary node we remove the fifth column and fifth row from the matrix A . After removing all columns and row corresponding to boundary nodes, we will have a final SPD linear system, which can be solved for example by the CG-method.

The rest of the report contains a relatively detailed example which explains how to set up a linear system for the Poisson Equation. As I said, I have included both minimization and weak formulations, but do skip the minimization principle if it doesn't interest you.

Towards the practical implementation, it is also convenient to define a reference triangular element and define a 1-1 mapping between a general physical element in the real/physical domain. All the integrals which make up A and F , are performed in this reference element. This is nothing more than a convenient substitution.

Finally, there are some other details that I will leave out as regards to assembling/-making the matrix from what is called element matrices.

4 A Practical Example

4.1 The Poisson Equation

We have given the Poisson Equation $\Delta u = \nabla^2 u = f = 1$, defined on Ω shown in Fig 10. The boundary is denoted by Γ .

4.2 Minimization formulation

Find $u \in V^D$ such that

$$u = \arg \min_{w \in V^D} J(w) \quad (34)$$

$$\begin{aligned} J(w) &= \frac{1}{2} \int_{\Omega} \nabla w \cdot \nabla w \, dA - \int_{\Omega} f w \, dA \\ &= \frac{1}{2} \int_{\Omega} \nabla w \cdot \nabla w \, dA \end{aligned} \quad (35)$$

$$V^D = \{v \in H^1(\Omega) \mid v|_{\Gamma} = 0\} \quad (36)$$

$$H^1(\Omega) = \left\{ v : \mathbb{R} \rightarrow \mathbb{R} \mid \int_{\Omega} v^2, \int_{\Omega} v_x^2, \int_{\Omega} v_y^2 < +\infty \right\} \quad (37)$$

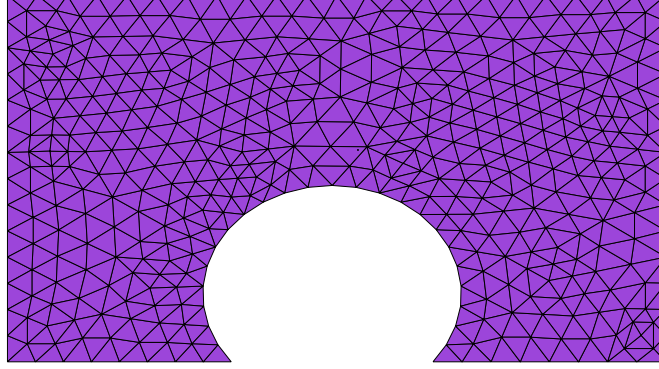


Figure 10: Finite element mesh used for the practical example. Along the boundary, $u = 0$.

4.3 Weak formulation

Let $w = u + v \in V^D, u \in V^D, v \in V^D$.

$$\begin{aligned}
 J(w) &= J(u + v) \\
 &= \frac{1}{2} \int_{\Omega} \nabla(u + v) \cdot \nabla(u + v) dA \\
 &= \frac{1}{2} \int_{\Omega} \nabla u \cdot \nabla u dA + \underbrace{\int_{\Omega} \nabla u \cdot \nabla v dA}_{\delta J_v(u)} + \underbrace{\frac{1}{2} \int_{\Omega} \nabla v \cdot \nabla v dA}_{>0 \text{ for } v \neq 0}
 \end{aligned} \tag{38}$$

$$\begin{aligned}
 \delta J_v(u) &= \int_{\Omega} \nabla u \cdot \nabla v dA = \int_{\Omega} \nabla(v \nabla u) dA - \int_{\Omega} v \nabla^2 u dA \\
 &= \int_{\Gamma} \underbrace{v}_{=0} \nabla u \cdot \vec{n} dS - \int_{\Omega} v \underbrace{\nabla^2 u}_{=0} dA = 0
 \end{aligned} \tag{39}$$

Equations (38) og (39) show that

$$J(w) \geq J(u) \quad \forall w \in V^D, u \in V^D \quad \textbf{Weak formulation} \tag{40}$$

5 Geometric representation

5.1 Element Integral

Let $\phi_i|_T = N_i$. This is the function ϕ_i restricted to one element. T stands for element. We want to compute the integral over the element T

$$a(N_i, N_j) = \int_T \frac{\partial N_i}{\partial x} \frac{\partial N_j}{\partial x} + \frac{\partial N_i}{\partial y} \frac{\partial N_j}{\partial y} dA \tag{41}$$

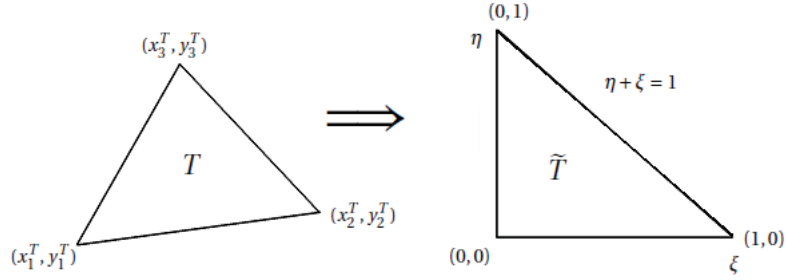


Figure 11: Mapping between physical and reference element

In terms of programming, it is easier to calculate the element integral ?? by the use of substitution. We define a linear (and affine) mapping between T and a reference element \tilde{T} as shown in Figure 11. Basis functions for \tilde{T} corresponding to $N_1(x, y)$, $N_2(x, y)$ and $N_3(x, y)$ for T are given by

$$\begin{aligned}\hat{N}_1(\xi, \eta) &= 1 - \xi - \eta \\ \hat{N}_2(\xi, \eta) &= \xi \\ \hat{N}_3(\xi, \eta) &= \eta\end{aligned}\quad (42)$$

Relation between the coordinates (x, y) og (ξ, η) is given by

$$\begin{aligned}x &= x_1^T \hat{N}_1 + x_2^T \hat{N}_2 + x_3^T \hat{N}_3 \\ y &= y_1^T \hat{N}_1 + y_2^T \hat{N}_2 + y_3^T \hat{N}_3\end{aligned}\quad (43)$$

We have

$$\frac{\partial N_i}{\partial x} = \frac{\partial \hat{N}_i}{\partial \xi} \frac{\partial \xi}{\partial x} + \frac{\partial \hat{N}_i}{\partial \eta} \frac{\partial \eta}{\partial x}; \quad \frac{\partial N_i}{\partial y} = \frac{\partial \hat{N}_i}{\partial \xi} \frac{\partial \xi}{\partial y} + \frac{\partial \hat{N}_i}{\partial \eta} \frac{\partial \eta}{\partial y}\quad (44)$$

Vi need to find $\frac{\partial \xi}{\partial x}$, $\frac{\partial \xi}{\partial y}$, $\frac{\partial \eta}{\partial x}$ og $\frac{\partial \eta}{\partial y}$. Since $x = x(\xi, \eta)$ og $y = y(\xi, \eta)$, we get

$$\begin{bmatrix} dx \\ dy \end{bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} \end{bmatrix} \begin{bmatrix} d\xi \\ d\eta \end{bmatrix}\quad (45)$$

The Jacobian of matrix (45) is given by

$$|J| = \begin{vmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} \end{vmatrix} = \begin{vmatrix} x_2^T - x_1^T & x_3^T - x_1^T \\ y_2^T - y_1^T & y_3^T - y_1^T \end{vmatrix}\quad (46)$$

The inverse of (45) is given by

$$\begin{bmatrix} d\xi \\ d\eta \end{bmatrix} = \begin{bmatrix} \frac{\partial \xi}{\partial x} & \frac{\partial \xi}{\partial y} \\ \frac{\partial \eta}{\partial x} & \frac{\partial \eta}{\partial y} \end{bmatrix} \begin{bmatrix} dx \\ dy \end{bmatrix} = \frac{1}{|J|} \begin{bmatrix} \frac{\partial y}{\partial \eta} & -\frac{\partial x}{\partial \eta} \\ -\frac{\partial y}{\partial \xi} & \frac{\partial x}{\partial \xi} \end{bmatrix} \begin{bmatrix} dx \\ dy \end{bmatrix}\quad (47)$$

This gives us

$$\begin{aligned}\frac{\partial \xi}{\partial x} &= \frac{1}{|J|} \frac{\partial y}{\partial \eta} & \frac{\partial \xi}{\partial y} &= -\frac{1}{|J|} \frac{\partial x}{\partial \eta} \\ \frac{\partial \eta}{\partial x} &= -\frac{1}{|J|} \frac{\partial y}{\partial \xi} & \frac{\partial \eta}{\partial y} &= \frac{1}{|J|} \frac{\partial x}{\partial \xi}\end{aligned}\quad (48)$$

It follows that

$$\begin{aligned}
\frac{\partial N_1}{\partial x} &= \frac{1}{|J|}(y_2^T - y_3^T) & \frac{\partial N_1}{\partial y} &= \frac{1}{|J|}(x_3^T - x_2^T) \\
\frac{\partial N_2}{\partial x} &= \frac{1}{|J|}(y_3^T - y_1^T) & \frac{\partial N_2}{\partial y} &= -\frac{1}{|J|}(x_3^T - x_1^T) \\
\frac{\partial N_3}{\partial x} &= -\frac{1}{|J|}(y_2^T - y_1^T) & \frac{\partial N_3}{\partial y} &= \frac{1}{|J|}(x_2^T - x_1^T)
\end{aligned} \tag{49}$$

All the terms in (49) are constant, and the area of T is denoted by A_T . The element stiffness matrix for T is then given by

$$\begin{aligned}
K^T &= A_T \begin{bmatrix} \partial_x N_1^2 + \partial_y N_1^2 & \partial_x N_1 \partial_x N_2 + \partial_y N_1 \partial_y N_2 & \partial_x N_1 \partial_x N_3 + \partial_y N_1 \partial_y N_3 \\ \partial_x N_2 \partial_x N_1 + \partial_y N_2 \partial_y N_1 & \partial_x N_2^2 + \partial_y N_2^2 & \partial_x N_2 \partial_x N_3 + \partial_y N_2 \partial_y N_3 \\ \partial_x N_3 \partial_x N_1 + \partial_y N_3 \partial_y N_1 & \partial_x N_3 \partial_x N_2 + \partial_y N_3 \partial_y N_2 & \partial_x N_3^2 + \partial_y N_3^2 \end{bmatrix} \tag{50} \\
&= \frac{A_T}{|J|^2} \begin{bmatrix} (y_2 - y_3)^2 + (x_3 - x_2)^2 & (y_2 - y_3)(y_3 - y_1) - (x_3 - x_2)(x_3 - x_1) & (x_3 - x_2)(x_2 - x_1) - (y_2 - y_3)(y_2 - y_1) \\ (y_2 - y_3)(y_3 - y_1) - (x_3 - x_2)(x_3 - x_1) & (y_3 - y_1)^2 + (x_3 - x_1)^2 & -(y_3 - y_1)(y_2 - y_1) - (x_3 - x_1)(x_2 - x_1) \\ (x_3 - x_2)(x_2 - x_1) - (y_2 - y_3)(y_2 - y_1) & -(y_3 - y_1)(y_2 - y_1) - (x_3 - x_1)(x_2 - x_1) & (y_2 - y_1)^2 + (x_2 - x_1)^2 \end{bmatrix}
\end{aligned}$$

5.2 Load Vector

In the Laplace equation stated above we have chosen to use constant load function $f = 1$. This is only to avoid unnecessary complications and numerical quadrature in the final formulation.

We know that the entry F_i in right-hand side vector F is given by

$$F_i = l(\phi_i) = \int f \phi_i(x, y) dA = \int \phi_i(x, y) dA \quad f = 1 \tag{51}$$

As for element matrix, ϕ_i is non-zero at several elements (the ones that share node i), so we could also write this expression as follows

$$F_{hi} = \sum_{k=1}^S \int N_i(x, y) dA \tag{52}$$

where S is the number of elements in Ω over which the support of N_i is unequal zero. Using the framework of mapping between T and \tilde{T} , we can define \hat{N}_i in terms of reference coordinates (ξ, η) . Using $dA = |det J| d\xi d\eta$ we get

$$F_{hi} = \sum_{k=1}^S \int \hat{N}_i(\xi, \eta) |det J| d\xi d\eta = \sum_{k=1}^S |det J| \int \hat{N}_i(\xi, \eta) d\xi d\eta \tag{53}$$

The Jacobian J is dependent on the element S . The integral in \hat{N}_i is simply

$$\int \hat{N}_i(\xi, \eta) d\xi d\eta = \frac{1}{6} \quad \forall i, i = 1, 2, 3 \tag{54}$$

So,

$$F_{hi} = \sum_{k=1}^S \int \hat{N}_i(\xi, \eta) |det J| d\xi d\eta = \frac{1}{6} \sum_{k=1}^S |det J| \tag{55}$$

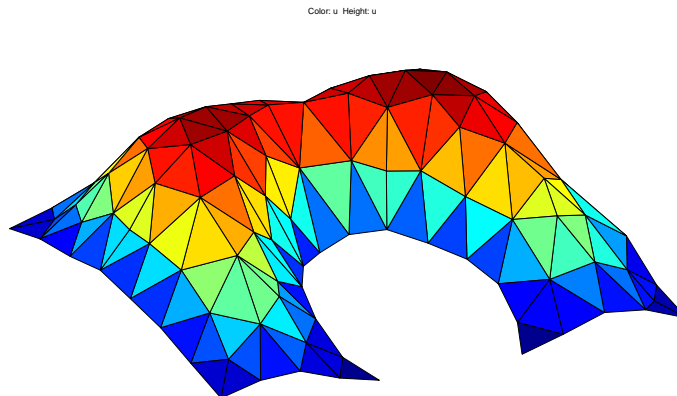


Figure 12: Solution of Poisson Equation on a coarse grid.

I will not go into further details regarding setting up the matrix A and vector F . The element matrix that we derived above is only for one element, so we have as many 3×3 matrices as we have elements. To make the total matrix, we need to put these matrices at proper places. The same is done for F . After putting the system together, we need to remove the columns and rows corresponding to boundary nodes, because there we have given function values, in our case $u = 0$ along the boundary. Then it is just a matter of using the desired method to solve the system and plotting it. Fig 12 show the solution of our Poisson Equation using a coarse grid, and Fig 13 shows the same solution using a fine grid. I have included Fig 12 to remark how the solution is made up of triangular planes which are piecewise continuous across the boundary.

6 Did we understand the difference between FE and FD?

FD here refers to finite difference. So did we understand the main difference between finite element and finite difference? Firstly, as I wrote in the beginning, we associate a differential operator (call it L) to every PDE. In the case of Poisson, it is just $L = \nabla^2$. The finite difference method discretizes this operator by replacing every differential with an approximate expression using differential formulas. It does it by making a grid of nodes, and uses the step length, which in general may vary, between nodes to find expressions for these formulas. Taylor series is all we need to begin with.

What is common between FE and FD, is the idea of discretization of the domain on which our PDE is defined. We have nodes both in FE and FD. The main difference lies in the concept of approximation. FD discretizes the operator, while FE discretizes the underlying space X . If you remember, we derived a weak formulation in FE, and based on it, we defined a space of functions in which we supposed ours was to be found. This was as follows

$$X = \{v \in H^1(\Omega) : v|_{\partial\Omega} = 0\} \quad (56)$$

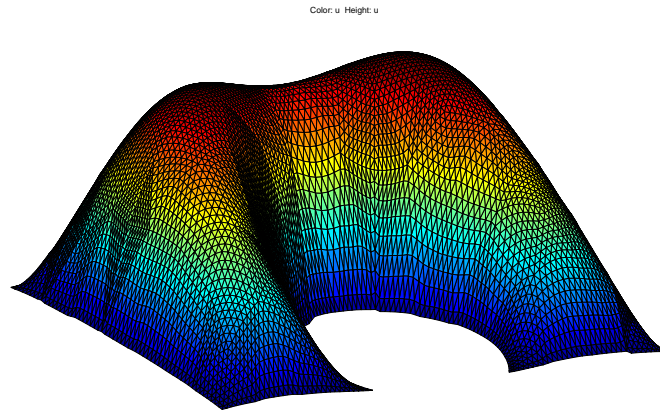


Figure 13: Solution of Poisson's Equation on a fine grid.

and

$$H^1(\Omega) = \left\{ v : \Omega \rightarrow \mathbb{R} : \int_{\Omega} v^2, \int_{\Omega} v_x^2, \int_{\Omega} v_y^2 < +\infty \right\} \quad (57)$$

The space X is of infinite dimension; there is an infinite number of functions which are zero on the boundary and quadratic integrable until first partial derivatives. In our FE approximation, we now know that we have, in this specific case, used functions which are linear polynomials per element. As the figures above shows, these are continuous across the element boundaries, but look like straight planes in one element. This means that we have used functions from a subspace of X , call it X_h which looks like

$$X_h = \{v \in X : v|_T = \mathbb{P}_1\} \quad (58)$$

That is, X_h is the space of all functions in X , which, when restricted to one element, are polynomials of degree one. FE discretizes the function space; FD discretizes the differential operator.

7 Finite Element on GPU

I have no plan to go into details here, but the following figure may give you an idea as to how this can be done. Each color corresponds to one block.

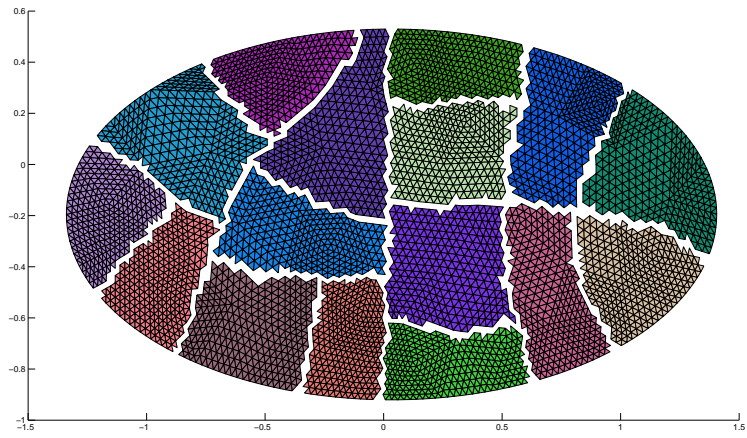


Figure 14: Partitioning of finite element grid.

References

- [1] Niels Ottosen & Hans Pettersen : *Introduction to The Finite Element Method*. Prentice , 1992, 1st Edition, 1992.
- [2] Dietrich Braess : *Finite elements*. Cambridge, 3rd Edition, 2007.
- [3] Notes in TMA4220 : <http://www.math.ntnu.no/emner/TMA4220/2007h/notat.imf>.