Scientific Computing
Lecture 5

Discretization

Nikolay Koshev, Nikolay Yavich

October 14, 2023

Skoltech
Skolkovo Institute of Science and Technology

Structure of the lecture

- Introduction
- Finite Differences Method (FDM)
- Finite Elements Method (FEM)
- Galerkin approach
- **EXAMPLE:** 3D Cauchy problem for elliptic equation

Scientific Computing
Lecture 5

Part 1: Introduction

October 14, 2023

Skoltech

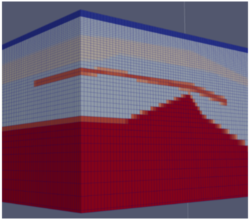
Skolkovo Institute of Science and Technology

Why do we need to descretize the problem?

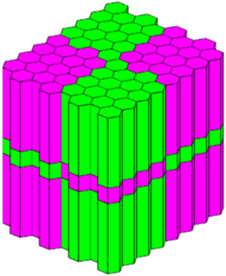
- Generally, the solution of complex problem could be rarely found analytically; and, if if could, you don't need a computer...
- The continuous function $f(x)$ is defined for the infinite number of points
- From the point of view of computer, it seems an infinite-dimensional vector
- Having only two states of the bit, computers do not understand infinity.

Three main approaches for macroscopic problems:

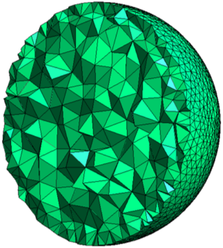
Finite Difference Method (FDM)



Finite Volume Method (FVM)



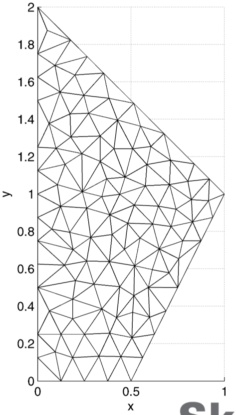
Finite Element Method (FEM)



The computational domain: $\Omega \subset \mathbb{R}^n$, $\quad n = 2$ or $n = 3$.
The boundary: $\partial\Omega$: the boundary of computational domain;

$$-\Delta u(\mathbf{x}) = f(\mathbf{x}), \quad \mathbf{x} \in \Omega, \quad , u|_{\partial\Omega} = \varphi(\mathbf{x}).$$

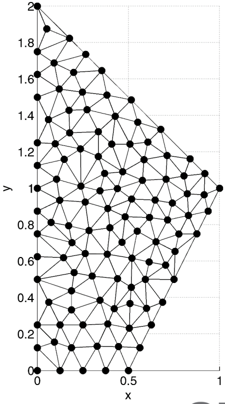
- Cover the computational domain with some mesh.



The computational domain: $\Omega \subset \mathbb{R}^n$, $\quad n = 2$ or $n = 3$.
The boundary: $\partial\Omega$: the boundary of computational domain;

$$-\Delta u(\mathbf{x}) = f(\mathbf{x}), \quad \mathbf{x} \in \Omega, \quad , u|_{\partial\Omega} = \varphi(\mathbf{x}).$$

- Cover the computational domain with some mesh.
- Define the locations of known and unknown variables: nodes (vertices), cellular centers of mass, centers of links, sides etc.

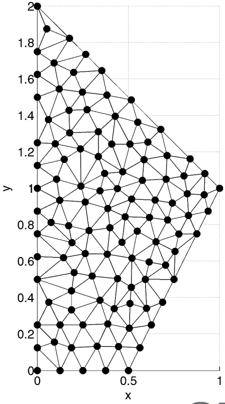


The computational domain: $\Omega \subset \mathbb{R}^n, \quad n = 2$ or $n = 3$.
The boundary: $\partial\Omega$: the boundary of computational domain;

$$-\Delta u(\mathbf{x}) = f(\mathbf{x}), \quad \mathbf{x} \in \Omega, \quad , u|_{\partial\Omega} = \varphi(\mathbf{x}).$$

- Cover the computational domain with some mesh.
- Define the locations of known and unknown variables.
- Construct the system of equations to solve numerically:

$$Au_h = g.$$



Skoltech
Skolkovo Institute of Science and Technology

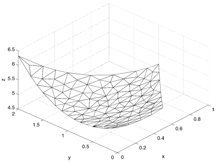
Poisson equation 2D example: solution steps

The computational domain: $\Omega \subset \mathbb{R}^n, \quad n = 2$ or $n = 3$.
The boundary: $\partial\Omega$: the boundary of computational domain;

$$-\Delta u(\mathbf{x}) = f(\mathbf{x}), \quad \mathbf{x} \in \Omega, \quad , u|_{\partial\Omega} = \varphi(\mathbf{x}).$$

- Cover the computational domain with some mesh.
- Define the locations of known and unknown variables.
- Construct the system of equations to solve numerically.
- Solve the SLAE:

$$u_h = A^{-1} g$$

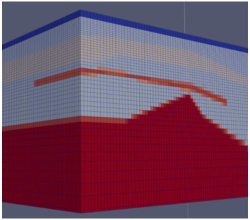


Skoltech
Skolkovo Institute of Science and Technology

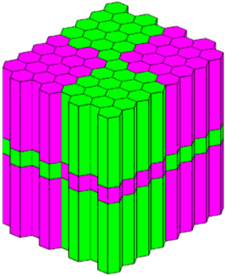
Macroscopic approaches

Three main approaches for macroscopic problems:

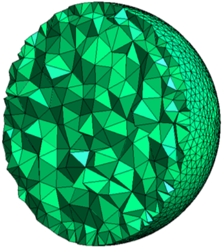
Finite Difference Method (FDM)



Finite Volume Method (FVM)



Finite Element Method (FEM)



Scientific Computing
Lecture 3

Part 4: Discretization. Finite differences.

Nikolay Koshev

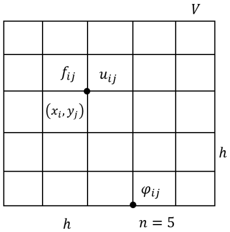
October 14, 2023

Skoltech
Skolkovo Institute of Science and Technology

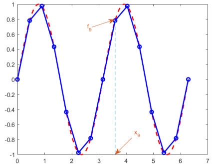
FDM: The mesh

- Let the computational domain be $\Omega \subset \mathbb{R}^r$
- All $r-$dimensional points to be processed are therefore belong to this domain: $\mathbf{x} \in \Omega$
- **Nodes:** A finite set of points $\{x_n\}$ being considered in real computations. Here n is a multiindex.
- $x_n = (x^1_{n_1}, x^2_{n_2}, ..., x^r_{n_r})$; $n_i = 0, ..., N_i - 1$, where N_i is a number of nodes at i-th dimension.
- For example, for $r = 3$, the nodes will be denoted as: $x_{ijk}, i = 0, ..., N_x - 1, j = 0, ..., N_y - 1, k = 0, ..., N_k - 1$.
- The falue of some function depending on x will be denoted as: $f_n = f(x_n)$. For $r = 3$ it then will be f_{ijk}.

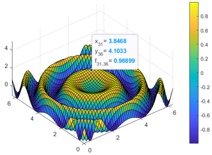
Uniform mesh: $x_n = (n_1 h_1, n_2 h_2, ..., n_r h_r)$.



The structure of the mesh



The function $u(x) = sin(2x)$, approximated with a mesh $x_i = i/N$



The function $u(x, y) = sin((x - \pi)^2 + (y - \pi)^2)$, approximated with a mesh $x_i = i/N, y_j = j/N, N = 50$.

Approximation of the first derivatives. For short notations, consider the 2d case:

$$x_i = ih_x, \quad y_j = jh_y, \quad f_{ij} = u(x_i, y_j).$$

- Backward difference:

$$\frac{\partial u}{\partial x}(x_i, y_j) \equiv \left(\frac{\partial u}{\partial x}\right)_{ij} \approx \frac{u_{ij} - u_{i-1,j}}{h}.$$

- Forward difference:

$$\frac{\partial u}{\partial x}(x_i, y_j) \equiv \left(\frac{\partial u}{\partial x}\right)_{ij} \approx \frac{u_{i+1,j} - u_{ij}}{h}.$$

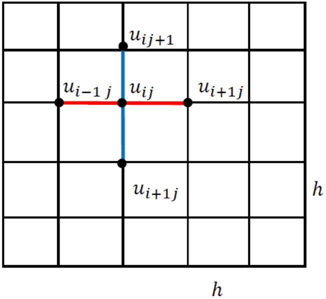
- Symmetric difference:

$$\frac{\partial u}{\partial x}(x_i, y_j) \equiv \left(\frac{\partial u}{\partial x}\right)_{ij} \approx \frac{u_{i+1,j} - u_{i-1,j}}{2h}.$$

- The same is for other variables;

$$u_{xx}(x_i, y_j) \approx \frac{u_{i+1,j} - 2u_{ij} + u_{i-1,j}}{h_x^2}.$$

$$u_{yy}(x_i, y_j) \approx \frac{u_{i,j+1} - 2u_{ij} + u_{i,j-1}}{h_y^2}.$$



The Poisson equation in this approximation takes the form

$$-\left(\frac{u_{i+1,j} - 2u_{ij} + u_{i-1,j}}{h_x^2} + \frac{u_{i,j+1} - 2u_{ij} + u_{i,j-1}}{h_y^2}\right) = f_{ij}.$$

Approximation of 2D Neumann problem for Poisson eq.

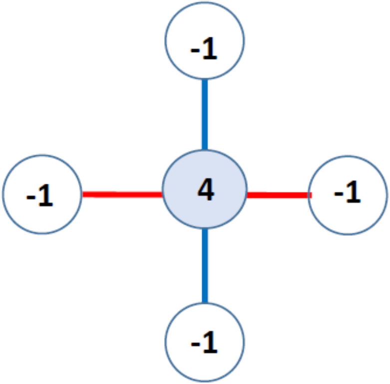
- **The Poisson equation in this approximation takes the form**

$$-\left(\frac{u_{i+1,j} - 2u_{ij} + u_{i-1,j}}{h_x^2} + \frac{u_{i,j+1} - 2u_{ij} + u_{i,j-1}}{h_y^2}\right) = f_{ij}.$$

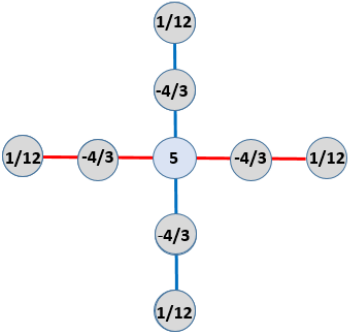
- The boundary conditions mean we can exclude variables u_{ij} for indices i, j belonging to the boundary, changing them with correspondent values φ_{ij}.
- The scheme has a square matrix: easy to see that indices vary such that $i = 1, ..., N_i - 2$, $j = 1, ..., N_j - 2$, which means $(N_i - 2) * (N_j - 2)$ equations. Since the boundary values are excluded, so we have the same number of unknown variables.

The Cross template consists of five nodes. If the steps $h_x = h_y = h$ (the most often case):

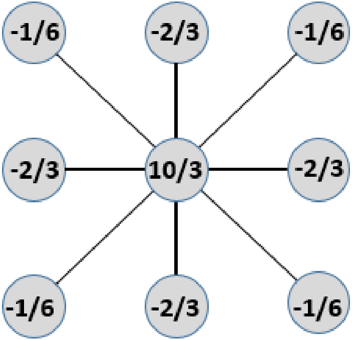
$$-u_{i-1,j} - u_{i+1,j} + 4u_{ij} - u_{i,j-1} - u_{i,j+1} = h^2 f_{ij}$$



Fourth order scheme



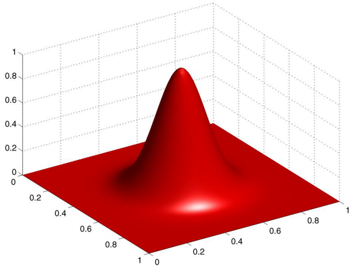
Fourth order compact scheme (Numerov's scheme)



Non-uniform meshes: example

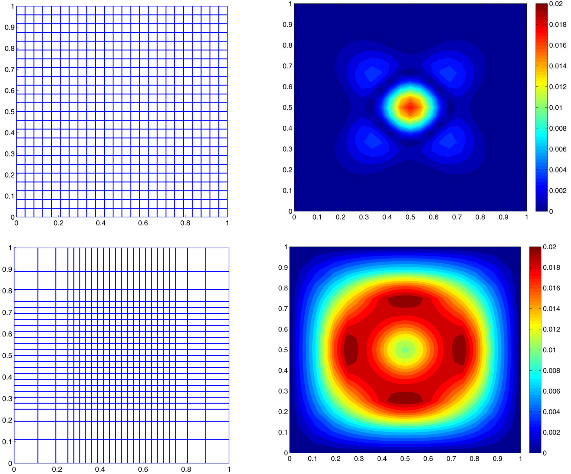
$$-\Delta u = f, \quad \text{in}\Omega,$$
$$u = \varphi, \quad \text{in}\Gamma.$$

$$u = exp\left(- 40\left(x - \frac{1}{2}\right)^2 - 40\left(y - \frac{1}{2}\right)^2 \right)$$



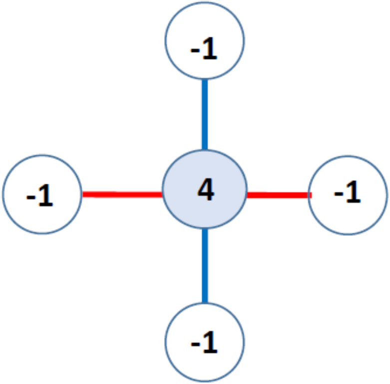
Non-uniform meshes: example

The mesh $\qquad\qquad$ $|u - u_h|$



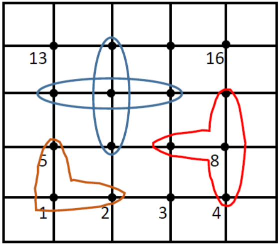
The Cross template consists of five nodes. If the steps $h_x = h_y = h$ (the most often case):

$$-u_{i-1,j} - u_{i+1,j} + 4u_{ij} - u_{i,j-1} - u_{i,j+1} = h^2 f_{ij}$$



The structure of SLAE

$$A\,u_h = g$$



$$\left[\begin{array}{cccc|cccc|cccc|cccc}
4 & -1 & & & -1 & & & & & & & & & & & \\
-1 & 4 & -1 & & & -1 & & & & & & & & & & \\
 & -1 & 4 & -1 & & & -1 & & & & & & & & & \\
 & & -1 & 4 & & & & -1 & & & & & & & & \\
\hline
-1 & & & & 4 & -1 & & & -1 & & & & & & & \\
 & -1 & & & -1 & 4 & -1 & & & -1 & & & & & & \\
 & & -1 & & & -1 & 4 & -1 & & & -1 & & & & & \\
 & & & -1 & & & -1 & 4 & & & & -1 & & & & \\
\hline
 & & & & -1 & & & & 4 & -1 & & & -1 & & & \\
 & & & & & -1 & & & -1 & 4 & -1 & & & -1 & & \\
 & & & & & & -1 & & & -1 & 4 & -1 & & & -1 & \\
 & & & & & & & -1 & & & -1 & 4 & & & & -1 \\
\hline
 & & & & & & & & -1 & & & & 4 & -1 & & \\
 & & & & & & & & & -1 & & & -1 & 4 & -1 & \\
 & & & & & & & & & & -1 & & & -1 & 4 & -1 \\
 & & & & & & & & & & & -1 & & & -1 & 4
\end{array}\right]$$

Skolkovo Institute of Science and Technology

The structure of SLAE

$$
\mathbf{A} = \left[\begin{array}{c|c|c|c}
\mathbf{C} & -\mathbf{E} & & \\ \hline
-\mathbf{E} & \mathbf{C} & -\mathbf{E} & \\ \hline
 & -\mathbf{E} & \mathbf{C} & -\mathbf{E} \\ \hline
 & & -\mathbf{E} & \mathbf{C}
\end{array}\right]
\qquad
\mathbf{C} = \left[\begin{array}{c|c|c|c}
\mathbf{4} & \mathbf{-1} & & \\ \hline
\mathbf{-1} & \mathbf{4} & \mathbf{-1} & \\ \hline
 & \mathbf{-1} & \mathbf{4} & \mathbf{-1} \\ \hline
 & & \mathbf{-1} & \mathbf{4}
\end{array}\right]
$$

$$
\left[\begin{array}{cccc|cccc|cccc|cccc}
4 & -1 & & & -1 & & & & & & & & & & & \\
-1 & 4 & -1 & & & -1 & & & & & & & & & & \\
 & -1 & 4 & -1 & & & -1 & & & & & & & & & \\
 & & -1 & 4 & & & & -1 & & & & & & & & \\ \hline
-1 & & & & 4 & -1 & & & -1 & & & & & & & \\
 & -1 & & & -1 & 4 & -1 & & & -1 & & & & & & \\
 & & -1 & & & -1 & 4 & -1 & & & -1 & & & & & \\
 & & & -1 & & & -1 & 4 & & & & -1 & & & & \\ \hline
 & & & & -1 & & & & 4 & -1 & & & -1 & & & \\
 & & & & & -1 & & & -1 & 4 & -1 & & & -1 & & \\
 & & & & & & -1 & & & -1 & 4 & -1 & & & -1 & \\
 & & & & & & & -1 & & & -1 & 4 & & & & -1 \\ \hline
 & & & & & & & & -1 & & & & 4 & -1 & & \\
 & & & & & & & & & -1 & & & -1 & 4 & -1 & \\
 & & & & & & & & & & -1 & & & -1 & 4 & -1 \\
 & & & & & & & & & & & -1 & & & -1 & 4
\end{array}\right]
$$

SLAE: Matrix spectral properties

Finite-difference SLAE properties:

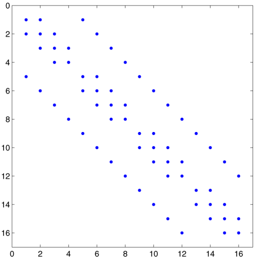
- The system contains a big number of unknown variables and equations
- The matrix is square, sparse, invertible, symmetric, positive definite
- Big condition number (often ill-conditioned):

$$\kappa(A) = O\left(\frac{1}{h^2}\right)$$

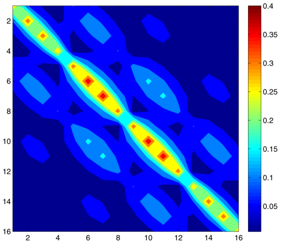
- $H = A^{-1}$ - approximates fundamental (Green's) function.
- H_{ij} - solution in the node i with the point source, located in the node j.
- The matrix A^{-1} is dense while the matrix A is sparse.

It's better to avoid usage of inverse matrices in real computations.

A



A^{-1}



Scientific Computing
Lecture 3

Part 5: FEM discretization

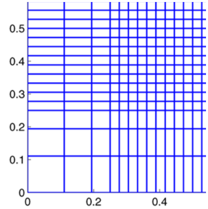
Nikolay Koshev

October 14, 2023

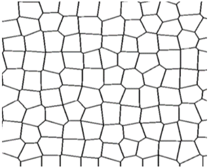
Skolkovo Institute of Science and Technology

The motivation

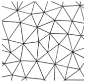
FDM: rectangular cells: fast and simple mesh construction; issues: discontinuous coefficients, complex geometries

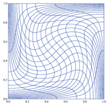


Finite Volumes Method: more flexible with respect to geometry

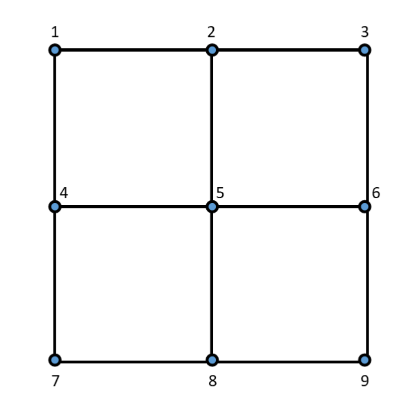


Finite Elements Method: Unstructured grids, complex geometries, stability



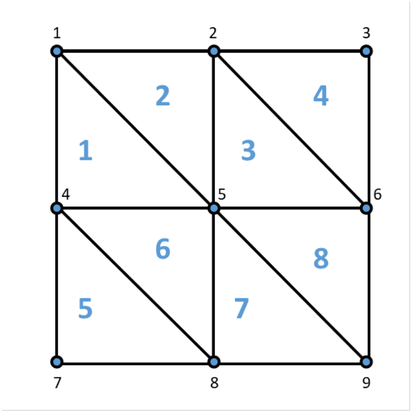


The FEM mesh notations



- The number of nodes: N;
- The nodes:
  $\mathbf{x}_i \in \mathbb{R}^d, \quad \text{for } i = 0, \dots, N-1.$

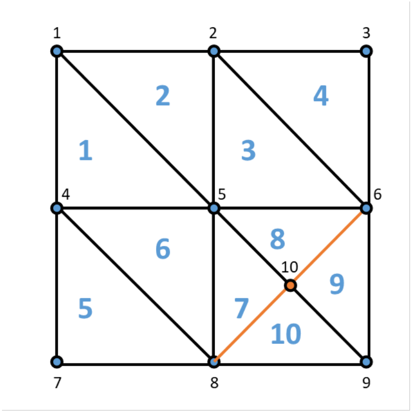
The FEM mesh notations



- The number of nodes: N (here $N = 9$);
- The nodes: $\mathbf{x}_i \in \mathbb{R}^d, \quad \text{for } i = 0, ..., N-1.$
- The Mesh Cell (Triangle, tetrahedron, etc.): $K_i, 0 = 1, ..., N_e - 1$
- The number of mesh cells: N_e (here $N_e = 8$);

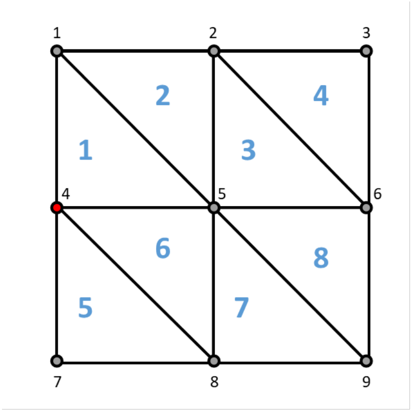
Skoltech
Skolkovo Institute of Science and Technology

The FEM mesh notations



- The number of nodes here: $N = 10$;
- The number of elements here: $N_e = 10$;
- Sometimes, we are unable to enumerate nodes with appropriate order. Such grids called **unstructured grids.**

The FEM mesh approximation

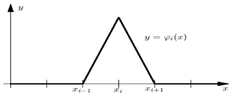


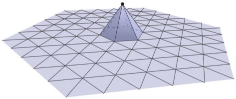
- Introduce **the basis:** the set of continuous functions $\varphi_i(\mathbf{x}), i = 0, ..., N-1$, associated with the nodes, such that:

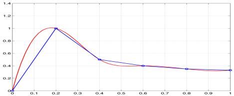
$$\varphi_i(\mathbf{x}) = \begin{cases} 1, & \text{at } i\text{-th node} \\ 0, & \text{at other nodes} \end{cases}$$

- Each basis function is connected to the node: $\varphi_j(\mathbf{x}), j = 0, ..., N-1$.
- The basis functions being sometimes called **Elements**

FEM approximation







- **The function $u(\mathbf{x})$ can be approximated as follows:**

$$u(\mathbf{x}) \approx \sum_{i=0}^{N-1} u_i \varphi_i(\mathbf{x})$$

- **The derivatives:** $\nabla u(\mathbf{x}) = \sum_{i=0}^{N-1} u_i \nabla \varphi_i(\mathbf{x})$
- **The integrals:**

$$\int_{\Omega} u(\mathbf{x}) d\mathbf{x} \approx \sum_{i=0}^{N-1} u_i \int_{\Omega} \varphi_i(\mathbf{x}) d\mathbf{x}, \quad \int_{\Omega} \nabla u(\mathbf{x}) d\mathbf{x} \approx \sum_{i=0}^{N-1} u_i \int_{\Omega} \nabla \varphi_i(\mathbf{x}) d\mathbf{x}$$

The weak formulation

Let F be a Hilbert space. The equation to be solved:

$$Au = f, \quad u, f \in F, A : F \to F.$$

The latter equation is an equivalent to finding $u \in F$ such that:

$$(Au, v)_F = (f, v)_F \quad \forall v \in F$$

The function v is called a test function. For Poisson equation it means, since $\Delta u \in L_2(\Omega)$, that:

$$(\Delta u, v)_{L_2(\Omega)} = -(f, v)_{L_2(\Omega)}.$$

The pipeline of the FEM

0. **Original equation:** $Au = f, \quad u \in U, f \in F$
1. **Weak formulation:** $(Au, v)_F = (f, v)_F, \quad \forall v \in F$
2. **Approximation (assuming linearity of A):**

$$u(\mathbf{x}) \approx \sum_{i=0}^{N-1} u_i \varphi_i(\mathbf{x}) \quad \Longrightarrow \quad \sum_{i=0}^{N-1} u_i \left(A\varphi_i, v\right)_F = (f, v)_F$$

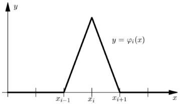
3. **The system:** since $v \in F$ is a certain function, we put $v = \varphi_j, \quad j = 0, 1, \ldots, N-1$ in order to obtain N equations with respect to unknown u_i:

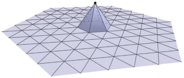
$$\sum_{i=0}^{N-1} u_i \left(A\varphi_i, \varphi_j\right)_F = (f, \varphi_j)_F$$

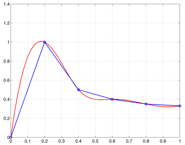
4. Solve the system above.
5. Profit!

Kinds of Finite Elements

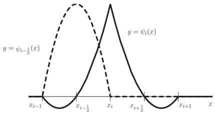
- Linear elements:







- Quadratic elements:



- Bicubic, exponential, etc.
- Since $v \in F$, the elements may imply requirements for U, F spaces!

Skolkovo Institute of Science and Technology

3D EEG FEM modelling example

3D EEG modelling: statement of the problem

The Neumann problem for Poisson equation in complex heterogeneous area

- The governing equation:

$$\nabla \cdot \big(\sigma(\mathbf{x})\nabla u(\mathbf{x})\big) = \nabla \cdot J(\mathbf{x}), \quad \mathbf{x} \in \Omega \subset \mathbb{R}^3$$

- The Neumann boundary condition:

$$\frac{\partial u}{\partial n}\Big|_{\partial\Omega} = 0.$$

3D EEG modelling: the weak formulation

The weak formulation of the problem

$$-\int\limits_{\Omega} \big(\sigma(\mathbf{x})\nabla u(\mathbf{x})\big) \cdot \nabla v(\mathbf{x}) d\mathbf{x} = \int\limits_{\Omega} \big(\nabla \cdot \mathbf{J}(\mathbf{x})\big) v(\mathbf{x}) d\mathbf{x}.$$

In order to manage the discontinuous function $\mathbf{J}$, we avoid usage of its derivative in the RHS using the known integral identity:

$$\int\limits_{\Omega} \big(\nabla \cdot \mathbf{J}(\mathbf{x})\big) v(\mathbf{x}) d\mathbf{x} + \int\limits_{\Omega} \mathbf{J}(\mathbf{x}) \cdot \nabla v(\mathbf{x}) d\mathbf{x} = \int\limits_{\partial\Omega} \big(\mathbf{J}(\mathbf{x}) \cdot \mathbf{n}\big) v(\mathbf{x}) d\mathbf{x}$$

Finally, the weak formulation takes the form:

$$\int\limits_{\Omega} \big(\sigma(\mathbf{x})\nabla u(\mathbf{x})\big) \cdot \nabla v(\mathbf{x}) d\mathbf{x} = \int\limits_{\Omega} \mathbf{J}(\mathbf{x}) \cdot \nabla v(\mathbf{x}) d\mathbf{x}$$

3D EEG modelling: the discretization

- Cover the domain Ω with tetrahedral mesh;
- Approximate the function $u(\mathbf{x})$ and its derivatives:

$$u(\mathbf{x}) \approx u_h(\mathbf{x}) = \sum_{i=0}^{N-1} u_i \varphi_i(\mathbf{x}) \quad \Longrightarrow \quad \nabla u_h(\mathbf{x}) = \sum_{i=0}^{N-1} u_i \nabla \varphi_i(\mathbf{x})$$

- Since the function v may be any function, we can use $v(\mathbf{x}) = \varphi_j(\mathbf{x}), j = 0, 1, ..., N-1$. Substituting the approximation to the weak formulation, we have:

$$\sum_{i=0}^{N-1} u_i \int_{\Omega} \sigma(\mathbf{x}) \nabla \varphi_i(\mathbf{x}) \cdot \nabla \varphi_j(\mathbf{x}) d\mathbf{x} = \int_{\Omega} \mathbf{J}(\mathbf{x}) \cdot \nabla \varphi_j(\mathbf{x}) d\mathbf{x},$$

  for $i, j = 0, 1, ..., N-1$.
- Additionally define the obtained system with the values of u on boundaries.

3D EEG modelling: the discretization

After discretization with linear finite elements we get the system to solve:

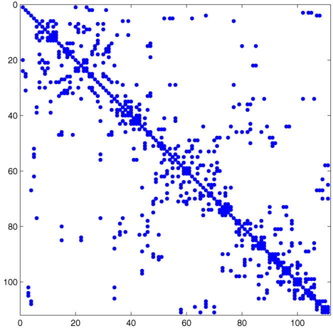
$$\tilde{A}u_h = \mathbf{b}, \quad \tilde{A}_{ij} = \int\limits_{\Omega} \sigma(\mathbf{x})\nabla\varphi_i(\mathbf{x}) \cdot \nabla\varphi_j(\mathbf{x})d\mathbf{x}, \quad b_j = \int\limits_{\Omega} \mathbf{J}(\mathbf{x}) \cdot \nabla\varphi_j(\mathbf{x}),$$

for $i, j = 0, 1, ..., N-1$. Thus,

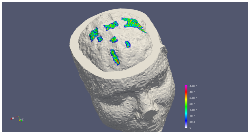
- $\tilde{A}.shape = (N, N)$, and
- $\mathbf{b}.shape = N$
- The gradients of elements $\nabla\varphi_i(\mathbf{x})$ depend only on mesh and can be pre-calculated.
- The matrix $\tilde{A}$ (called sometimes the *stiffness matrix*) depends only on the mesh and properties of volume (conductivity $\sigma(\mathbf{x})$), and can also be pre-calculated.
- The vector $\mathbf{b}$ depends on distribution $\mathbf{J}(\mathbf{x})$.

Properties of the matrix

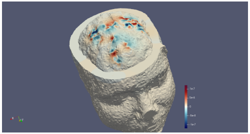
- Square matrix
- Sparse and symmetric
- $\kappa(A) = O\left(\frac{1}{h^2}\right)$



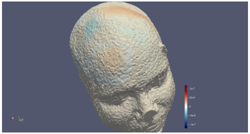
- The system contains $10^6 - 10^7$ equations and the same number of unknown variables;
- The matrix of the system is ill-conditioned;
- It's, however, symmetric and sparse;
- The suitable method to solve: generalized residual method with regularization.



$\mathbf{J}(\mathbf{x})$



$u(\mathbf{x})|_{\partial\Omega_2}$



$u(\mathbf{x})|_{\partial\Omega_1}$

Scientific Computing
Lecture 5

Galerkin approach,
Nikolay Koshev

October 14, 2023

Skoltech

Skolkovo Institute of Science and Technology

Differential operator

The operator equation for differential equations is often written as:

$$D\mathbf{u} = 0,$$

where D represents both mathematical and the right-hand side (observed data); the function $\mathbf{u}$ is an unknown function to be found.

- **Ordinary differential equation (ODE):** Let $x \in \mathbb{R}^1$. ODE of the k-th order can be represented with the operator:

$$D\mathbf{u} = F\Big(x, u(x), u'(x), u''(x), ..., u^{(k)}(x)\Big).$$

- **Partial differential equation (PDE):** Let $\mathbf{x} \in \mathbb{R}^n \equiv (x_1, x_2, ..., x_n)$. PDE of the k-th order can be represented with the following operator:

$$D\mathbf{u} = F\Big(\mathbf{x}, \mathbf{u}(\mathbf{x}), \frac{\partial \mathbf{u}}{\partial x_1}, ..., \frac{\partial \mathbf{u}}{\partial x_n}, \cdots \frac{\partial^2 \mathbf{u}}{\partial x_1^2}, ..., \frac{\partial^2 \mathbf{u}}{\partial x_n^2}, \cdots \frac{\partial^{(k)} \mathbf{u}}{\partial x_1^{(k)}}, \frac{\partial^{(k)} \mathbf{u}}{\partial x_n^{(k)}}\Big).$$

The Galerkin Method

- The Galerkin method may be considered an initial point - an **IDEA** - for linear DE/IE solution
- The Method is a base for wavelets analysis, FEM, FDM etc
- The Method shows one of the most general approaches in computational problems
- The Method allows to chose the concrete solution spaces with respect to the problem (may be considered a regularization too)
- Easy to understand
- Easy to research

Consider the Boundary Value problem for the linear differential operator D:

$$Du = 0, \quad \text{in } \Omega \subset \mathbb{R}^n, \quad S(u) = 0, \text{in } \partial\Omega$$

- Assume $u \in U, \quad D : U \to V$, where U, V are some Hilbert spaces.
- Let $\{\varphi_i(\mathbf{x})\}, i = 1, ...$ be some basis in U.
- Chose finite number N and approximate

$$u(\mathbf{x}) \approx u_a(\mathbf{x}) = u_0(\mathbf{x}) + \sum_{i=1}^{N} a_i \varphi_i(\mathbf{x}).$$

Consider the Boundary Value problem for the linear differential operator D:

$$Du = 0, \quad \text{in } \Omega \subset \mathbb{R}^n, \quad S(u) = 0, \text{in } \partial\Omega$$

- Assume $u \in U, \quad D : U \to V$, where U, V are some Hilbert spaces.
- Let $\{\varphi_i(\mathbf{x})\}, i = 1, ...$ be some basis in U.
- Chose finite number N and approximate the solution u
- Since N is finite, and u_a is only the approximation, we get:

$$D\Big(\sum_{i=1}^{N} a_i \varphi(\mathbf{x})\Big) + Du_0 = R(a_1, ..., a_n, \mathbf{x}).$$

Consider the Boundary Value problem for the linear differential operator D:

$$Du = 0, \quad \text{in } \Omega \subset \mathbb{R}^n, \quad S(u) = 0, \text{in } \partial\Omega$$

- Assume $u \in U, \quad D : U \to V$, where U, V are some Hilbert spaces.
- Let $\{\varphi_i(\mathbf{x})\}, i = 1, ...$ be some basis in U.
- Chose finite number N and approximate the solution u
- Since N is finite, and u_a is only the approximation, we get the residual $R(a_1, ..., a_n, \mathbf{x})$
- The function R is never zero... But we can minimize it with respect to the coefficients $\{a_i\}$!

Skoltech
Skolkovo Institute of Science and Technology

The idea

- Represent (or approximate) the solution u with some weighted sum of functions
- Substitute the approximation or representation into the original equations
- Minimize the residual with respect to weights
- After the weights are calculated, reconstruct the approximate (or, sometimes, exact) solution, substituting the weights into your representation (or approximation) of it
- Be careful: the residual rarely being equal to zero, but it should be small!

Let the problem be linear: $D, S(u)$ are linear.

$$Du = 0, \quad \text{in } \Omega \subset \mathbb{R}^n, \quad S(u) = 0, \text{in } \partial\Omega$$

- The weak formulation:

$$(Du, v) = 0, \quad v \in U.$$

- Approximate the solution:

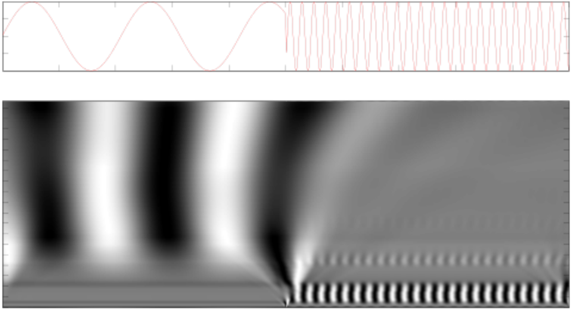
$$u = \sum_{i=0}^{N} u_i \varphi_i(\mathbf{x}), \quad \varphi_i(\mathbf{x}) = \begin{cases} 1, & \mathbf{x} = \mathbf{x}_i \\ 0, & \mathbf{x} = \mathbf{x}_j, j \neq i \\ \text{continuous} & \text{elsewhere} \end{cases}$$

- Substitute the approximation into the weak form:

$$\sum_{i=1}^{N} u_i (D\varphi_i, \varphi_j) = 0, \quad \sum_{l:\mathbf{x}_l \in \partial\Omega} u_l \varphi_l(\mathbf{x}) = 0.$$

With the wavelet we can analyse not only frequencies (like in Fourier analysis), but both frequencies and its locations in time (scaling and shifting)

$$U(a,b) = \frac{1}{|a|^{\frac{1}{2}}} \int\limits_{\mathbb{R}} u(x)\psi^*\left(\frac{t-b}{a}\right)dt$$



Galerkin and wavelets

- Continuous wavelet transform:

$$U(a,b) = \frac{1}{|a|^{\frac{1}{2}}} \int\limits_{\mathbb{R}} u(x)\psi^*\left(\frac{t-b}{a}\right)dt$$

- The continuous wavelet transform (the inverse one, 1D case):

$$u(x) = C \int\limits_{\mathbb{R}^2} \frac{1}{a^2} U(a,b) exp\left(i\frac{t-b}{a}\right) dbda$$

- In discrete case:

$$u(x) = C_\psi \sum_{i,j=-\infty}^{\infty} U_{ij}\psi_{ij}(t), \quad U_{ij} = \int\limits_{\mathbb{R}} u(x)\psi^*_{ij}dt$$

- In *n*-dimensional case we use separable wavelets: each for one dimension.

Multiresolution analysis

Let $\{V_j\}_{j=-\infty}^{\infty}$ is a sequence of spaces such that:

- $V_j \subset V_{j+1}$
- $f(t) \in V_j \Rightarrow f(2t) \in V_{j+1}, \quad f(t) \in V_j \Rightarrow f(t-k) \in V_j$
- The single-scaling function φ defines orthonormal basis in V_j:

$$\psi_{jk} = 2^{j/2}\psi(2^j t - k)$$

- Example: Haar's multiresolution analysis:

$$V_j = \{f \in L^2(\mathbb{R}); \forall k \in \mathbb{Z} : f|_{[2^j k, 2^j(k+1)]} = const\}$$

- After we choose the basis, then do the same: substitute the solution approximated with that basis!