Chapter 1

Discretization of the computational domain

1.1 Mesh generation

Simplicial meshes

Definition 1.1.1 (Simplicial Mesh)**.** Let Ω be polygonal ($d = 2$) or polyhedral ($d = 3$) subset of $\mathbb{R}^d$, we define the mesh $\mathcal{M} = \mathcal{T}_h(\Omega)$ obtained by a triangulation of Ω as a finite family $\mathcal{K}(\mathcal{M}) = \{K_i\}$ of disjoints non-empty subsets of Ω named cells; the topological dimension of the mesh is d and $N_{\mathcal{K}(\mathcal{M})} = \mathrm{card}(\mathcal{K}(\mathcal{M}))$ is the number of cells in the mesh. Moreover $\mathcal{V}(\mathcal{M}) = \{\mathrm{v}_j\}$ denotes the set a vertices of $\mathcal{M}$, $N_{\mathcal{V}(\mathcal{M})}$ is the number of cells in the mesh, and $n_{\mathrm{v}}(K) = N_{\mathcal{V}(K)}$ the number of vertices in a cell K.

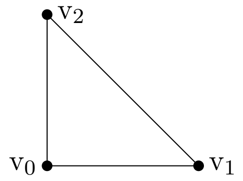
Whenever there is no possible confusion (only one mesh) notations will be shortened *e.g.* $\mathcal{K}(\mathcal{M})$ will be only written $\mathcal{K}$, $N_{\mathcal{K}}$ (resp. $N_{\mathcal{V}}$) will denote the number of cells (resp. vertices) in $\mathcal{M}$, and

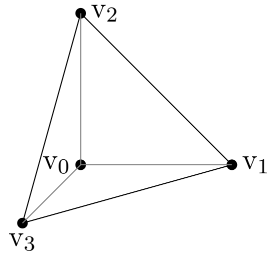
The *topology* of a mesh $\mathcal{M}$ is determined by the relations between elements of the set of cells $\mathcal{K}(\mathcal{M})$ and elements of the set of vertices $\mathcal{V}(\mathcal{M})$. The relations can be expressed in the form of a graph $\mathcal{C}_{d,0}(\mathcal{M}) = \big(\mathcal{K}(\mathcal{M}), \mathcal{V}(\mathcal{M})\big)$ called *connectivity.* The connectivity between cells and vertices defines the structure of the mesh and can be represented under the form of a vector of cell connectivities $\mathcal{C}_{d,0}(K_1)$.

$$\mathcal{C}_{d,0}(\mathcal{M}) = \begin{bmatrix} \mathcal{C}_{d,0}(K_1) \\ \vdots \\ \mathcal{C}_{d,0}(K_i) \\ \vdots \\ \mathcal{C}_{d,0}(K_N) \end{bmatrix}$$

Given that the mesh is uniform, any cell $K \in \mathcal{M}$ contains $n_{\mathrm{v}} = N_{\mathcal{V}(K)}$ vertices, so $\mathcal{C}_{d,0}(K_1)$ can be represented as a matrix of $\mathbb{N}^{N_{\mathcal{K}} \times n_{\mathrm{v}}}$; at minima, distributing the mesh topology consists of partitioning the matrix row-wise.

Example 1.1.2 (Simplices)**.** Reference triangle and tetrahedron cells are represented below with locally indexed vertices.





$$\mathcal{V}(\hat{K}) = [\mathrm{v}_0, \mathrm{v}_1, \mathrm{v}_2]$$

$$\begin{aligned} \mathrm{v}_0 &= (0,0) \\ \mathrm{v}_1 &= (1,0) \\ \mathrm{v}_2 &= (0,1) \end{aligned}$$

$$\mathcal{V}(\hat{K}) = [\mathrm{v}_0, \mathrm{v}_1, \mathrm{v}_2, \mathrm{v}_3]$$

$$\begin{aligned} \mathrm{v}_0 &= (0,0,0) \\ \mathrm{v}_1 &= (1,0,0) \\ \mathrm{v}_2 &= (0,1,0) \\ \mathrm{v}_3 &= (0,0,1) \end{aligned}$$

The set of vertices $\mathcal{V}(\mathcal{M})$ is provided with a set of point coordinates $\mathcal{P}(\mathcal{M}) = \{\mathbf{x}_j\}$ which define the *geometry* of the mesh. The position of the vertices in the geometric space may then change while the topology remains invariant. The topological dimension of the mesh is not the same as the dimension of geometric space if the mesh is embbeded in $\mathbb{R}^n$, $n > d$; this is for example the case of surface meshes. In general $\mathcal{P}(\mathcal{M})$ is then a family of vectors of $\mathbb{R}^n$ but usually $n = p$.

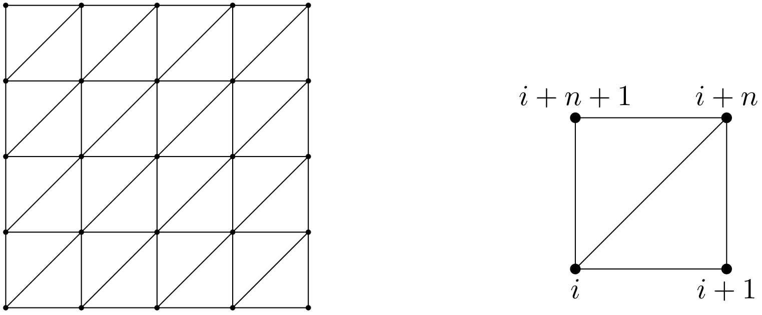
$$\mathcal{P}(\mathcal{M}) = \begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_i \\ \vdots \\ \mathbf{x}_N \end{bmatrix}$$

The geometry can be then represented by a matrix of $\mathbb{R}^{N_\mathcal{V} \times n}$; distributing the geometry amounts to partition the matrix row-wise.

Generating a mesh consists of the construction of $\mathcal{C}_{d,0}(\mathcal{M})$ and $\mathcal{P}(\mathcal{M})$ but if other entities (edges, faces, ...) are needed, then their connectivities can be created from $\mathcal{C}_{d,0}(\mathcal{M})$ and from the definition of the reference cell type together with rules for ordering the entities in the mesh.

In the case of vertices, the cell local ordering of connectivities will just be the natural ordering.

Example 1.1.3 (Unit square)**.** Generation of a structured triangle mesh with right-crossed squares.



Distribution of entities

Any distribution of a finite set $\mathcal{V}$ amounts to partition a subset of $\mathbb{N}$ in the form of a set of *global indices* $\mathcal{I}(\mathcal{V})$ ($\mathcal{I}$ defines a *global numbering* which is a one-to-one relation by definition), then represent additional relations if entities have dependencies. It is usually convenient to consider a contiguous numbering such that $\mathcal{I}(\mathcal{V}) = \{1, 2, \cdots, N_{\mathcal{V}}\}$ but this is not necessarily the case. In the following the numbering are 1-indexed but the same applies in the case of 0-indexed numberings.

Considering a partition such that $\mathcal{I}(\mathcal{V})$ is a disjoint sum of p index subsets,

$$\mathcal{I}(\mathcal{V}) = \cup_{k=1}^{\mathrm{p}} \mathcal{I}^k(\mathcal{V}), \quad \mathcal{I}^i(\mathcal{V}) \cap \mathcal{I}^j(\mathcal{V}) = \emptyset$$

such that in practice any *global index* only appears once.

For each subset of global indices $\mathcal{I}^k(\mathcal{V})$ containing $N_{\mathcal{V}}^k$ entities, a *local numbering* $i^k(\mathcal{V})$ defines a mapping from a contiguous numbering of the entities $i = 1, \dots, N_{\mathcal{V}}^k$ to $\mathcal{I}^k(\mathcal{V})$. In practice, $\mathcal{I}^k(\mathcal{V})$ may not be ordered, so $i^k(\mathcal{V})$ is merely a numbering of the entities in the order they are iterated over.

If the global numbering is ordered and contiguous, then any subset $\mathcal{I}^k(\mathcal{V})$ can be defined by a *range* r_k with parameters (o_k, s_k), with

1. the *range offset* o_k is the index of the first global index in the subset,
2. the *range size* s_k is the cardinal of $\mathcal{I}^k(\mathcal{V})$.

In a distributed setting the range size is the number of entities owned by the processor and the offset $o_k = \sum_{i<p} s_i$ can be computed with a scan reduction. The global numbering of $\mathcal{V}$ is a vector of $\mathbb{N}^{N_{\mathcal{V}}}$ distributed across processors.

Example 1.1.4. Let us consider the simple example of a set containing 16 entities uniformly partitioned (subsets have the same size) into 4 subsets.

$$\mathcal{I}(\mathcal{V}) = \Big[\big[\, 1 \; 2 \; 3 \; 4 \, \big] \big[\, 5 \; 6 \; 7 \; 8 \, \big] \big[\, 9 \; 10 \; 11 \; 12 \, \big] \big[\, 13 \; 14 \; 15 \; 16 \, \big] \Big]$$

then any local numbering will consist of a mapping from $\big[\, 1 \; 2 \; 3 \; 4 \, \big]$ to $\mathcal{I}^k(\mathcal{V})$, and ranges are defined by $r_1 = (0, 4)$, $r_2 = (5, 4)$, $r_3 = (9, 4)$, $r_4 = (13, 4)$.

In a distributed setting, whenever data attached to an entity is required on more than one processor then the entity is *shared* between a number of processors. Given the previous definition of a partitioning, entities can only be *owned* by one processor: there cannot be duplicate global indices. If a processor sees an entity but does not own it, it is called a *ghost*: the processor can read its data but is not responsible for computing it. On any processor k the set of entities $\bar{\mathcal{V}}^k$ seen by the processor is:

$$\bar{\mathcal{V}}^k = \mathcal{V}^k + \mathcal{V}^k_*$$

with the *local size* defined as $\mathrm{card}(\bar{\mathcal{V}}^k)$, $\mathcal{V}^k$ the set of owned entities, and $\mathcal{V}^k_*$ the set of ghost entities. Additionally, the set of shared entities is noted $\hat{\mathcal{V}}$ and has usually non-empty intersection with $\mathcal{V}^k$ and $\mathcal{V}^k_*$. The set of vertices of a mesh partition is defined like $\bar{\mathcal{V}}^k$.

Whenever data should be shared between processors, sets $\hat{\mathcal{V}}$ and $\mathcal{V}^k_*$ should be constructed: any elements of $\mathcal{V}^k_*$

Chapter 2

Linear Algebra

Matrix storage

There are two alternatives for storing a matrix A of $\mathbb{R}^{m\times n}$.

Row-major

$$\mathrm{A} = \begin{bmatrix} a_{11} & \dots & a_{1j} & \dots & a_{1n} \\ \\ a_{i1} & \dots & a_{ij} & \dots & a_{in} \\ \\ a_{m1} & \dots & a_{mj} & \dots & a_{mn} \end{bmatrix}$$

Flatten into a vector of $\mathbb{R}^{mn}$:

$$\mathrm{A} = \Big[\big[\, a_{11} \, \dots \, a_{1j} \, \dots \, a_{1n} \big] \dots \big[\, a_{i1} \, \dots \, a_{ij} \, \dots \, a_{in} \big] \dots \big[\, a_{m1} \, \dots \, a_{mj} \, \dots \, a_{mn} \big] \Big]$$

Column-major

$$\mathrm{A} = \begin{bmatrix} a_{11} & a_{1j} & a_{1n} \\ \vdots & \vdots & \vdots \\ a_{i1} & a_{ij} & a_{in} \\ \vdots & \vdots & \vdots \\ a_{m1} & a_{mj} & a_{mn} \end{bmatrix}$$

Flatten into a vector of $\mathbb{R}^{mn}$:

$$\mathrm{A} = \Big[\big[\, a_{11} \, \dots \, a_{i1} \, \dots \, a_{m1} \big] \dots \big[\, a_{1j} \, \dots \, a_{ij} \, \dots \, a_{mj} \big] \dots \big[\, a_{1n} \, \dots \, a_{in} \, \dots \, a_{mn} \big] \Big]$$

Parallelization

Given a distribution on p processors, for clarity in the following figures the ranks are considered index from zero to $\omega = \mathrm{p} - 1$.

The matrix M can be partitioned row-wise,

$$\mathrm{M} = \begin{bmatrix} [& \mathrm{M}_0 &] \\ & \vdots & \\ [& \mathrm{M}_k &] \\ & \vdots & \\ [& \mathrm{M}_\omega &] \end{bmatrix}$$

or column-wise

$$\mathrm{M} = \begin{bmatrix} \begin{bmatrix} \\ \mathrm{M}_0 \\ \\ \end{bmatrix} & \cdots & \begin{bmatrix} \\ \mathrm{M}_k \\ \\ \end{bmatrix} & \cdots & \begin{bmatrix} \\ \mathrm{M}_\omega \\ \\ \end{bmatrix} \end{bmatrix}$$

depending on the type of operation required.

In the case of Finite Element the matrix is partitioned row-wise, following the distribution of degrees of freedom: each row block corresponds to a range of degrees of freedom owned by the processor. In the case of linear Lagrange Finite Elements degrees of freedoms are located at the vertices which allows to reuse the same mapping for vertices, degrees of freedom, and algebraic unknowns.

Each row-block consists of submatrices corresponding to the column-wise distribution:

$$\mathrm{M} = \begin{bmatrix} \mathrm{M}_{00} & \cdots & \mathrm{M}_{0r} & \cdots & \mathrm{M}_{0\omega} \\ \vdots & \ddots & \vdots & \ddots & \\ \mathrm{M}_{q0} & \cdots & \mathrm{M}_{qr} & \cdots & \mathrm{M}_{q\omega} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \mathrm{M}_{\omega 0} & \cdots & \mathrm{M}_{\omega r} & \cdots & \mathrm{M}_{\omega\omega} \end{bmatrix}$$

The rectangular matrix on each row can be split in two sparse matrices:

1. D_k: the diagonal blocks containing only local contributions
2. $[\,\mathrm{M}_{kj}\,] : j \neq k$

$$\mathrm{M} = \begin{bmatrix} \mathrm{D}_{0} & \cdots & \mathrm{M}_{0r} & \cdots & \mathrm{M}_{0\omega} \\ \vdots & \ddots & \vdots & \ddots & \\ \mathrm{M}_{q0} & \cdots & \mathrm{D}_{qr} & \cdots & \mathrm{M}_{q\omega} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \mathrm{M}_{\omega 0} & \cdots & \mathrm{M}_{\omega r} & \cdots & \mathrm{D}_{\omega\omega} \end{bmatrix}$$

Each of these row blocks with m row of z non-zero entries can be stored using a CSR format:

1. $O \in \mathbb{N}^m$: containing the offset of column indices for each row,

2. $C \in \mathbb{N}^{m \times z}$: containing the column indices for each row,
3. $V \in \mathbb{R}^{m \times z}$: containing the corresponding entries.

Chapter 3

Finite elements

Weak formulation and Galerkin methods

The study of mathematical properties of PDE problems is usually performed on a general formulation called *abstract problem* which reads in this case:

$$\left|\begin{array}{l}\text{Find } u \in V\text{, such that:}\\[1em] a(u,v) = L(v) \quad , \forall\, v \in V\end{array}\right. \tag{3.1}$$

with $a(\,\cdot\,,\,\cdot\,)$ a continuous bilinear form on $V \times V$ and $L(\,\cdot\,)$ a continuous linear form on V.

For the weak formulation of the Helmoltz equation, the bilinear form reads

$$\begin{array}{rccl} a: & V \times V & \to & \mathbb{R} \\ & (u,v) & \mapsto & \displaystyle\int_\Omega u\, v \,\mathrm{d}\boldsymbol{x} + \int_\Omega \boldsymbol{\nabla} u \cdot \boldsymbol{\nabla} v \,\mathrm{d}\boldsymbol{x} \end{array}$$

and the linear form,

$$\begin{array}{rccl} L: & V & \to & \mathbb{R} \\ & v & \mapsto & \displaystyle\int_\Omega f\, v \,\mathrm{d}\boldsymbol{x} \end{array}$$

The weak formulation given by Problem (3.1) involves looking for a solution in a space V which has an infinite dimension as the function u is defined at any point of domain Ω. The approximation of Problem (3.1) by a Galerkin method consists of constructing a discrete space V_h which has a finite dimension. The dimension of V_h is the number of degrees of freedom, *i.e.* the number of unknowns of the problem. Solving the discrete problem consists of finding the values of real coefficients u_i, $i = 1, .., \dim(V_h)$ that define the discrete solution uniquely (if the discrete problem is well-posed):

$$u_h(\mathbf{x}) = \sum_{j=1}^{\dim(V_h)} u_j \varphi_j(\mathbf{x})$$

with $\{\varphi_j\}$ a basis of V_h.

Admissible mesh

Definition 3.0.1 (Mesh)**.** Let Ω be polygonal ($d = 2$) or polyhedral ($d = 3$) subset of $\mathbb{R}^d$, we define $\mathcal{T}_h$ (a triangulation in the simplicial case) as a finite family $\{K_i\}$ of disjoints non-empty subsets of Ω named cells. Moreover $\mathcal{V} = \{\mathrm{v}_i\}$ denotes the set a vertices of $\mathcal{T}_h$.

The Finite Element Method constructs the finite-dimensional space V_h based on a discretization of the geometric domain Ω. In the case of $\mathbb{P}_1$ elements, the domain Ω will be considered as a union of disjoint cells K_i which will be intervals in one dimension, triangles in two dimensions, and tetrahedra in three dimensions.

Only conforming mesh are considered, such that any facets of a cell is a facet for a neighbouring cell, or a facet located on the boundary of the domain: no hanging nodes are permitted.

Reference element

From a discretization of the domain Ω into cells, the construction of the discrete space V_h consists of choosing where to evaluate the degrees of freedom, and how to evaluate them. The definition of a Finite Element consists of a triple $(K, \mathcal{P}, \Sigma)$ that will provide the expansion of the discrete function $u_h \in V_h$ as:

$$u_h(\mathbf{x}) = \sum_{j=1}^{\dim(V_h)} u_j \varphi_j(\mathbf{x})$$

with u_j real coefficients (degrees of freedom), and φ_j a family of functions generating V. The three ingredients correspond to choosing a cell $\mathcal{K}$, constructing a basis for V_h, and finally deciding how to evaluate the degrees of freedom $\{u_j\}$.

Definition 3.0.2 (Finite Element)**.** A Finite Element consists of a triple $(K, \mathcal{P}, \Sigma)$, such that

- K is a compact, connected subset of $\mathbb{R}^d$ with non-empty interior and with regular boundary (typically Lipshitz continuous),
- $\mathcal{P}$ is a finite dimensional vector space, $\dim(\mathcal{P}) = N$, of functions $p : K \to \mathbb{R}$, which is the space of shape functions,
- Σ is a set $\{\sigma\}_j$ of linear forms,

$$\begin{array}{rccl} \sigma_j : & \mathcal{P} & \to & \mathbb{R} \qquad\qquad\qquad , \ \forall\, j \in [\![1, N]\!] \\ & p & \mapsto & p_j = \sigma_j(p) \end{array}$$

which is a basis of $\mathcal{L}(\mathcal{P}, \mathbb{R})$, the dual of $\mathcal{P}$.

In the case of $\mathbb{P}_1$ elements:

1. K are intervals, triangles, or tetrahedra.
2. $\mathcal{P}$ is constructed by the choice of piecewise linear functions φ_i.
3. Σ defines the degrees of freedom as nodal values $u_j = u(\mathbf{x}_j)$

The location of degrees of freedoms depends on the Finite Element: in the case of $\mathbb{P}_1$ they are located at vertices of the mesh, and for the sake of simplicity only structured meshes will be considered in two dimension.

The Finite Element can be expressed in the form of a Reference Finite Element $(\hat{K}, \hat{\mathcal{P}}, \hat{\Sigma})$ considered on a reference cell $\hat{K}$. For example, in the case of simplicial meshes the reference cell $\hat{K}$ in one dimension is the unit interval $[0, 1]$, in two dimension the unit triangle with vertices $\{(0, 0), (0, 1), (1, 0)\}$.

Shape functions φ_j for the $\mathbb{P}_1$ Reference Element on an interval:

$$\begin{aligned} \varphi_1(\hat{\mathbf{x}}) &= 1 - \hat{x} \\ \varphi_2(\hat{\mathbf{x}}) &= \hat{x} \end{aligned}$$

Shape functions φ_j for the $\mathbb{P}_1$ Reference Element on a triangle:

$$\begin{aligned} \varphi_1(\hat{\mathbf{x}}) &= 1 - \hat{x} - \hat{y} \\ \varphi_2(\hat{\mathbf{x}}) &= \hat{x} \\ \varphi_3(\hat{\mathbf{x}}) &= \hat{y} \end{aligned}$$

Transport of the Finite Element

For Lagrange elements the transformation between the Reference Finite Element $(\hat{K}, \hat{\mathcal{P}}, \hat{\Sigma})$ to the Finite element $(K, \mathcal{P}, \Sigma)$ on any cell of the mesh is achieved by an affine mapping $T_K : \hat{\mathbf{x}} \mapsto \mathbf{x}$ from the reference element $\hat{K}$ to cell K,

$$\int_K \psi(\mathbf{x}) \mathrm{d}\mathbf{x} = \int_{\hat{K}} \psi \circ T_K(\hat{\mathbf{x}}) |\det(J_K)| \mathrm{d}\hat{\mathbf{x}}$$

The contribution can be calculated on the reference element using

$$\varphi(\mathbf{x}) = \hat{\varphi}(\hat{\mathbf{x}})$$

and The contribution can be calculated on the reference element using

$$J_K(\hat{\mathbf{x}})^T \cdot \nabla\varphi(\mathbf{x}) = \nabla\hat{\varphi}(\hat{\mathbf{x}})$$

The one-dimensional and two-dimensional Jacobian matrices are,

$$J_K = \begin{bmatrix} x_1 - x_0 \end{bmatrix}$$

and

$$J_K = \begin{bmatrix} x_1 - x_0 & x_2 - x_0 \\ y_1 - y_0 & y_2 - x_0 \end{bmatrix}$$

Numerical Integration

Integration is performed with quadrature rules as the weighted sum over q Gauss points in the reference element,

$$\int_{\hat{K}} \hat{\psi}(\hat{\mathbf{x}}) \mathrm{d}\hat{\mathbf{x}} \approx \sum_q \hat{\psi}(\hat{\mathbf{x}}_q) \hat{\omega}_q$$

so that any contribution from cell $K \in \mathcal{M}$ is

$$\int_K \psi(\mathbf{x}) \mathrm{d}\mathbf{x} \approx \sum_q \psi(\hat{\mathbf{x}}_q) \circ T_K(\hat{\mathbf{x}}_q) \hat{\omega}_q |\det(J_K)(\mathbf{x}_q)|$$

Assembly

For each cell $K \in \mathcal{M}$ terms encountered in the Helmoltz equation are under the form of a stiffness matrix A,

$$\int_K \varphi_j(\mathbf{x}) \varphi_i(\mathbf{x}) \mathrm{d}\mathbf{x}$$

and a mass matrix M

$$\int_K \nabla\varphi_j(\mathbf{x}) \nabla\varphi_i(\mathbf{x}) \mathrm{d}\mathbf{x}$$