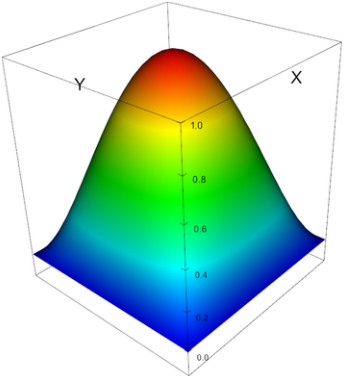
Specialized Numerical Methods for Transport Phenomena

Interpolation in the context of the Finite Element method

Bruno Blais

Associate Professor
Department of Chemical Engineering
Polytechnique Montréal

January 29, 2024



Outline

The role of interpolation

Defining Lagrange Polynomials

Continuous piecewise Lagrange polynomials

Implementation in the context of FEM

Extension to 2D and 3D

Code example

Outline

Interpolation

Let's make a recap of what we have seen thus far:

- Triangulating a domain Ω into smaller elements (cells)
- Integrating over a domain by integrating over cells

Interpolation

Let's make a recap of what we have seen thus far:

- Triangulating a domain Ω into smaller elements (cells)
- Integrating over a domain by integrating over cells

Interpolation

Interpolation defines the basis in which we will find our solution. Formally, from a function $f(x)$ known at $n+1$ points of the form $(x_i, f(x_i))$ we will construct an approximation $f(x)$ $\forall x \in [\min(x_i), \max(x_i)]$. The points x_i are the collocation (or interpolation) points.

Outline

Lagrange Polynomial

Lagrange polynomial are a common way to define a polynomial interpolation. Given $(n+1)$ points $(x_i, f(x_i))$ for $i = 0, 1, \cdots, n$, we suppose that we are able to construct $(n+1)$ polynomials $\phi_i(x)$ of degree n which satisfy:

$$L_i(x_j) = \begin{cases} 1, & j = i \\ 0, & j \neq i \end{cases} \tag{1}$$

Using these functions and the collocation points, we define the Lagrange polynomial:

$$L(x) = \sum_{i=0}^{n} f(x_i) L_i(x)$$

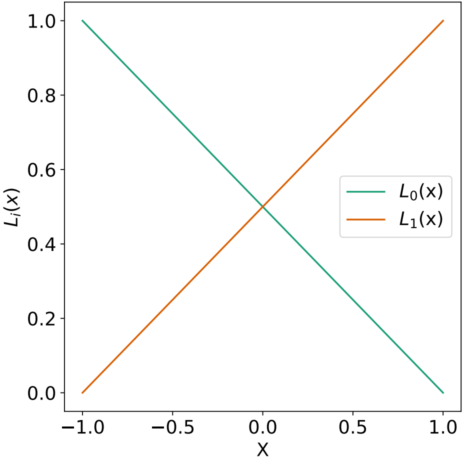
Let's construct L in the case where we have 2, 3 and $n+1$ points.

Degree 1

The polynomials $L_i(x)$ are:

$$L_0(x) = \frac{x - x_1}{x_0 - x_1}$$

$$L_1(x) = \frac{x - x_0}{x_1 - x_0}$$

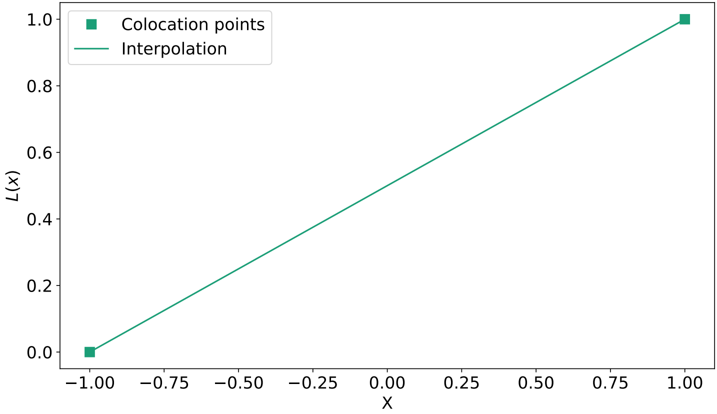


Degree 1

The interpolation polynomial is:

$$L(x) = f(x_0)L_0(x) + f(x_1)L_1(x)$$

$$L(x) = f(x_0)\frac{x - x_1}{x_0 - x_1} + f(x_1)\frac{x - x_0}{x_1 - x_0}$$



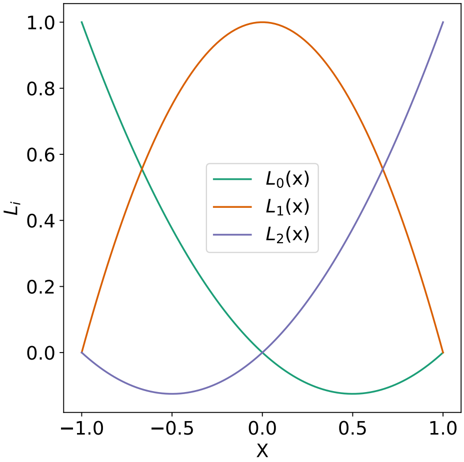
Degree 2

The polynomials $L_i(x)$ are:

$$L_0(x) = \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)}$$

$$L_1(x) = \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)}$$

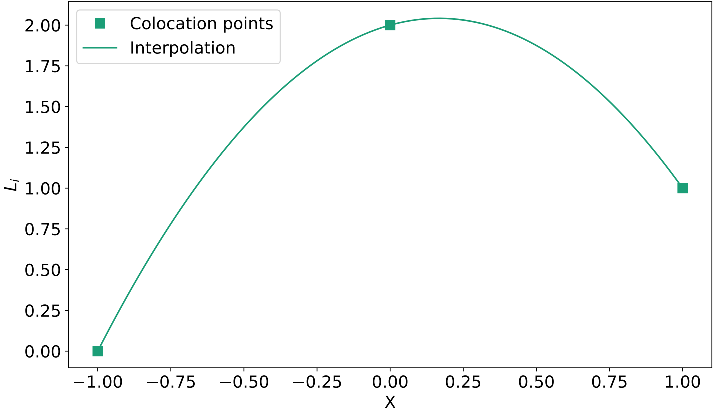
$$L_2(x) = \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)}$$



Degree 2

The interpolation polynomial is:

$$L(x) = f(x_0)L_0(x) + f(x_1)L_1(x) + f(x_2)L_2(x)$$



Generalization

The polynomials $L_i(x)$ are:

$$L_0(x) = \frac{(x - x_1)\,(x - x_2)\cdots(x - x_n)}{(x_0 - x_1)\,(x_0 - x_2)\cdots(x_0 - x_n)}$$

$$L_1(x) = \frac{(x - x_0)\,(x - x_2)\cdots(x - x_n)}{(x_1 - x_0)\,(x_1 - x_2)\cdots(x_1 - x_n)}$$

$$L_i(x) = \frac{(x - x_0)\cdots(x - x_{i-1})\,(x - x_{i+1})\cdots(x - x_n)}{(x_i - x_0)\cdots(x_i - x_{i-1})\,(x_i - x_{i+1})\cdots(x_i - x_n)}$$

Using 5 points

We have 5 polynomials $L_i(x)$:

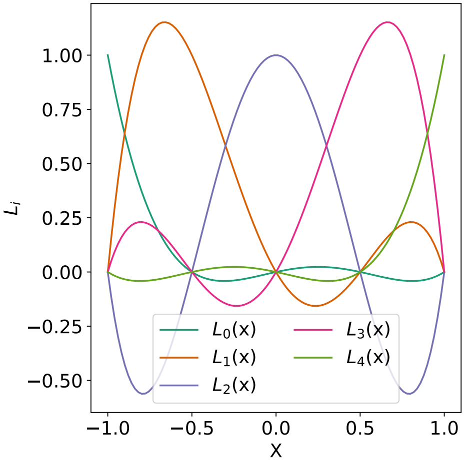
$$L_0(x)$$

$$L_1(x)$$

$$L_2(x)$$

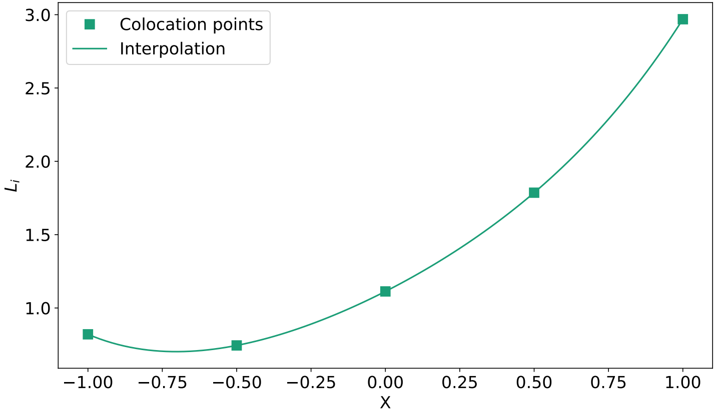
$$L_3(x)$$

$$L_4(x)$$



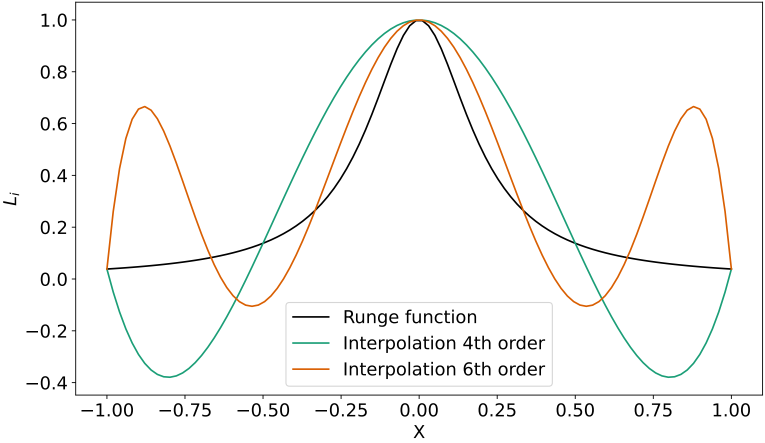
Example with 5 points

For fairly regular data like this, high degree interpolation performs visually well.



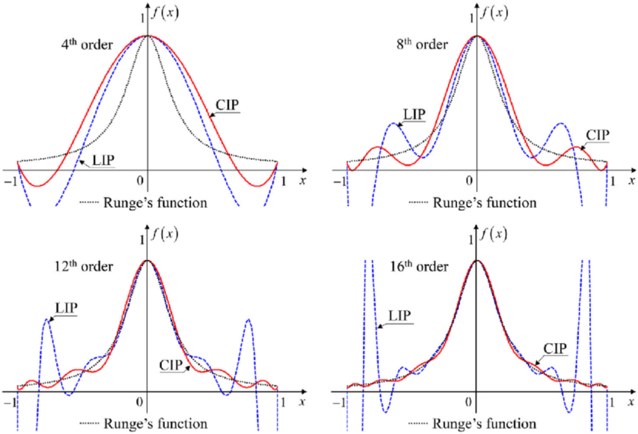
Runge Phenomenon

Interpolation with high degree polynomials using regularly spaced points can lead to large interpolation errors near the edges of the interval. This phenomenon is called the Runge phenomenon.



Runge Phenomenon

Using an alternative spacing of the points (e.g. Chebyshev's or Lobatto's) can mitigate this phenomena.



Outline

Continuous piecewise polynomials

Motivation

Lagrange polynomials allow us to define an interpolation polynomial using collocation points. In the end, we wish to define one interpolation polynomial per cell.

Continuous piecewise polynomials

Motivation

Lagrange polynomials allow us to define an interpolation polynomial using collocation points. In the end, we wish to define one interpolation polynomial per cell.

Continuous piecewise polynomials

Continuous piecewise Lagrange polynomials allow us to define a polynomial space for the union of all elements. The idea is relatively simple: for every cell, we will define a Lagrange polynomial using collocation points within that cell.

Notation

Let $\Omega = [x_0, x_n]$ be an interval partitioned by $n+1$ points $\{x_i\}_{i=0}^{n}$. We define the elements $\Omega_h = \Omega_1 \cup \Omega_2 \cup ... \cup \Omega_n$ such that:

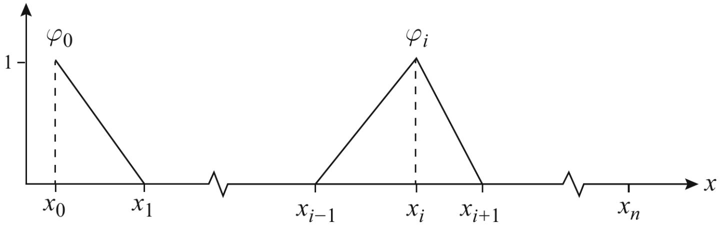
$$\Omega_h : x_0 < x_1 < x_2 < ... < x_{n-1} < x_n$$

With for example $\Omega_{i=2} = [x_1, x_2]$ and $h_2 = x_2 - x_1$. We define the space of piecewise continuous linear functions V_h in the triangulation Ω_h. In this space, the interpolation of any function $f(x)$ can be defined by the **linear** combination of hat functions $\{\varphi_i(x)\}_{i=0}^{n}$ and coefficients $\{f(x_i)\}_{i=0}^{n}$

Degree 1

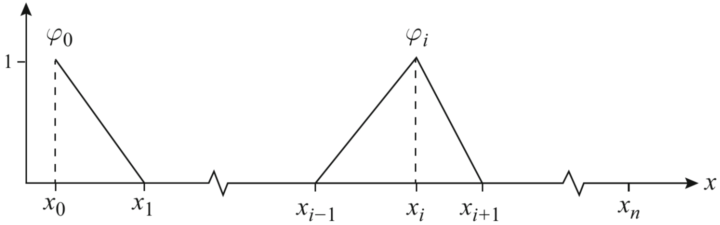
$$f_p(x) = \sum_{i=0}^{n} f(x_i)\varphi_i(x)$$

$$\varphi_i = \begin{cases} (x - x_{i-1})/h_i, & \text{if } x \in \Omega_i \\ (x_{i+1} - x)/h_{i+1}, & \text{if } x \in \Omega_{i+1} \\ 0, & \text{otherwise} \end{cases}$$



Properties

- φ_i is 1 at node i.
- φ_i is non-zero only on the elements touching the node i.
- φ_i is null at all the other nodes of the mesh.



Generalization of the degree

For each interval Ω_i consisting of $(p+1)$ points (where p is the degree of the interpolation), we construct a Lagrange polynomial.

Number of points used

- Degree 1 ($p = 1$) 2 points per interval
- Degree 2 ($p = 2$) 3 points per interval
- Degree 3 ($p = 3$) 4 points per interval
- Etc

Outline

In practice

Coordinates

Thus far, we have always defined our polynomials using the *real* coordinates x. In practice, this is not what is done.

- Requires redefining polynomial for every cell
- Difficult to apply in 2D and 3D

In practice

Coordinates

Thus far, we have always defined our polynomials using the *real* coordinates x. In practice, this is not what is done.

- Requires redefining polynomial for every cell
- Difficult to apply in 2D and 3D

Reference element

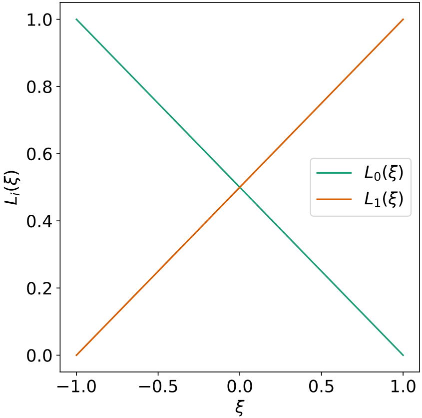
Instead, we define the Lagrange polynomial once within the reference element [-1,1] and we will transform each element Ω_i to the reference element using a change of variables.

Q1 element

Q1 elements use linear interpolation within the element. Thus we have two Lagrange polynomials:

$$L_0 = \frac{1}{2}(1 - \xi)$$

$$L_1 = \frac{1}{2}(1 + \xi)$$



Changing to the reference element

Once we have defined the Lagrange polynomial within the reference element,

$$L_0 = \frac{1}{2}(1-\xi)$$

$$L_1 = \frac{1}{2}(1+\xi)$$

We only need to transform from the real position to the reference position by defining $\xi(x)$:

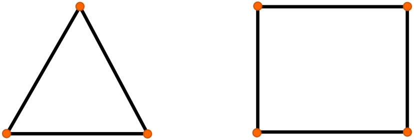
$$\xi = \frac{x - x_0}{x_0 - x_1}$$

Outline

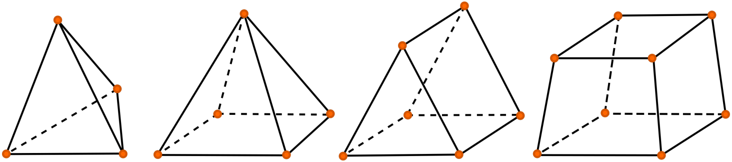
Cells in 1D to 3D

In 1D, a domain Ω will be a line. It is discretized using smaller segments:

In 2D, we can choose between triangles and quadrilaterals:



In 3D, the spectrum is larger. Ranges from tetrahedron to hexahedron:



2D and 3D

In this class, we will mainly work using tensor elements (lines, quadrilaterals and hexahedra). For these elements, defining the 2D and 3D Lagrange polynomial is direct by using tensor products. We define a set of Lagrange polynomial per coordinate:

- Along the ξ coordinate:

$$
\begin{aligned}
L_{0,\xi} &= \frac{1}{2}(1-\xi) \\
L_{1,\xi} &= \frac{1}{2}(1+\xi)
\end{aligned}
$$

- Along the η coordinate:

$$
\begin{aligned}
L_{0,\eta} &= \frac{1}{2}(1-\eta) \\
L_{1,\eta} &= \frac{1}{2}(1+\eta)
\end{aligned}
$$

Tensor product

$$L_{0,\xi} = \frac{1}{2}\left(1-\xi\right)$$
$$L_{1,\xi} = \frac{1}{2}\left(1+\xi\right)$$
$$L_{0,\eta} = \frac{1}{2}\left(1-\eta\right)$$
$$L_{1,\eta} = \frac{1}{2}\left(1+\eta\right)$$

The 2D interpolation polynomials are :

$$L_k\left(\xi,\eta\right) = L_{i,\xi}L_{j,\eta}$$

with $k = i + 2j$.

Result

$$L_0(\xi,\eta) = L_{0,\xi}L_{0,\eta} = \frac{1}{2}(1-\xi)\frac{1}{2}(1-\eta)$$
$$L_1(\xi,\eta) = L_{1,\xi}L_{0,\eta} = \frac{1}{2}(1+\xi)\frac{1}{2}(1-\eta)$$
$$L_2(\xi,\eta) = L_{0,\xi}L_{1,\eta} = \frac{1}{2}(1-\xi)\frac{1}{2}(1+\eta)$$
$$L_3(\xi,\eta) = L_{1,\xi}L_{1,\eta} = \frac{1}{2}(1+\xi)\frac{1}{2}(1+\eta)$$

The same approach applies to 3D or to higher order. The only challenge is to be able to go from the reference location to the physical location.

More than just interpolation

Although we have only used Lagrange polynomials to interpolate, we can also extend this concept to calculate other values:

- Gradient
- Hessians
- etc.

This is achieved by deriving the interpolation!

Outline

What does it look like?

deal.II provides abstraction for interpolating within cells using two classes.

```
FE_Q<2> fe(1);
QGauss<2> quadrature_formula(2);
FEValues<2> fe_values(fe, quadrature_formula, update_values |
    update_JxW_values | update_gradients);
const unsigned int n_q_points = fe_values.n_quadrature_points;

for (auto cell: triangulation.active_cell_iterators())
{
  fe_values.reinit(cell);
}
```

The FE_Q class generates the finite element interpolation support.
The FEValues object takes care of coupling the interpolation object (FE) and the quadratures. It automatically calculates everything related to the geometrical transformation.