Plugins
Generic programming
Expression templates

Paolo Joseph Baioni

April 30, 2025

Exercise 1: quadrature rules with plugins

- Implement a code that enables the user to compute the integral of a scalar function by selecting the quadrature rule as the name of a dynamically loadable object;
- The dynamically loadable object should define a function with the following signature: `double integrate(std::function<double (double)>, double a, double b)`.
- Implement plugins for midpoint and trapezoidal rule.
- Implement a plugin for quadrature with adaptive refinement.

Exercise 2: 1D interpolator using generic programming

This exercise (inspired by `Examples/src/Interp1D`) shows an example of a generic piecewise linear 1D interpolator, with an application to vectors of couples of values (representing the interpolation nodes and the corresponding values) and to two separate vectors (one for the nodes and one for the values).
It is rather generic and accepts as an input iterators to any container with bi-directional iterators. Bi-directionality is indeed only needed for extrapolation on the right.

Exercise 3: expression templates

This exercise is inspired by `https://www.modernescpp.com/index.php/expression-templates`.

The starting code provided implements a custom `MyVector` class.

- Given two vectors **x** and **y**, implement the sum and the multiplication operators to compute the quantity $2\mathbf{x} + \mathbf{y}^2$, where the operations are intended component-wise.
- Provide an expression template implementation and compare the computational costs with respect to a non-template implementation.