Generic factory and code refactoring: A Runge-Kutta-Fehlberg solver using traits and concepts (part II)

Matteo Caldana

27/04/2023

Refactoring the RKF solver

Starting from the solution to the previous lab:

1. define a class to handle the input options for the `RKF` class, provide the corresponding setter method and a method to parse options from a GetPot file;
2. implement the Fehlberg12 and the Dormand-Prince methods (`https://en.wikipedia.org/wiki/List_of_Runge%E2%80%93Kutta_methods#Embedded_methods`) and use them to solve the Lorenz system (`https://en.wikipedia.org/wiki/Lorenz_system`). Plot the solution both as a function of time and in the phase space;
3. implement a custom factory for the `ButcherArray` class, so that the actual method can be polymorphically selected at runtime from the GetPot file;
4. replace your custom-defined factory using the generic factory provided in the `generic-factory` folder, and provide a proxy for registering all RKF methods implemented.