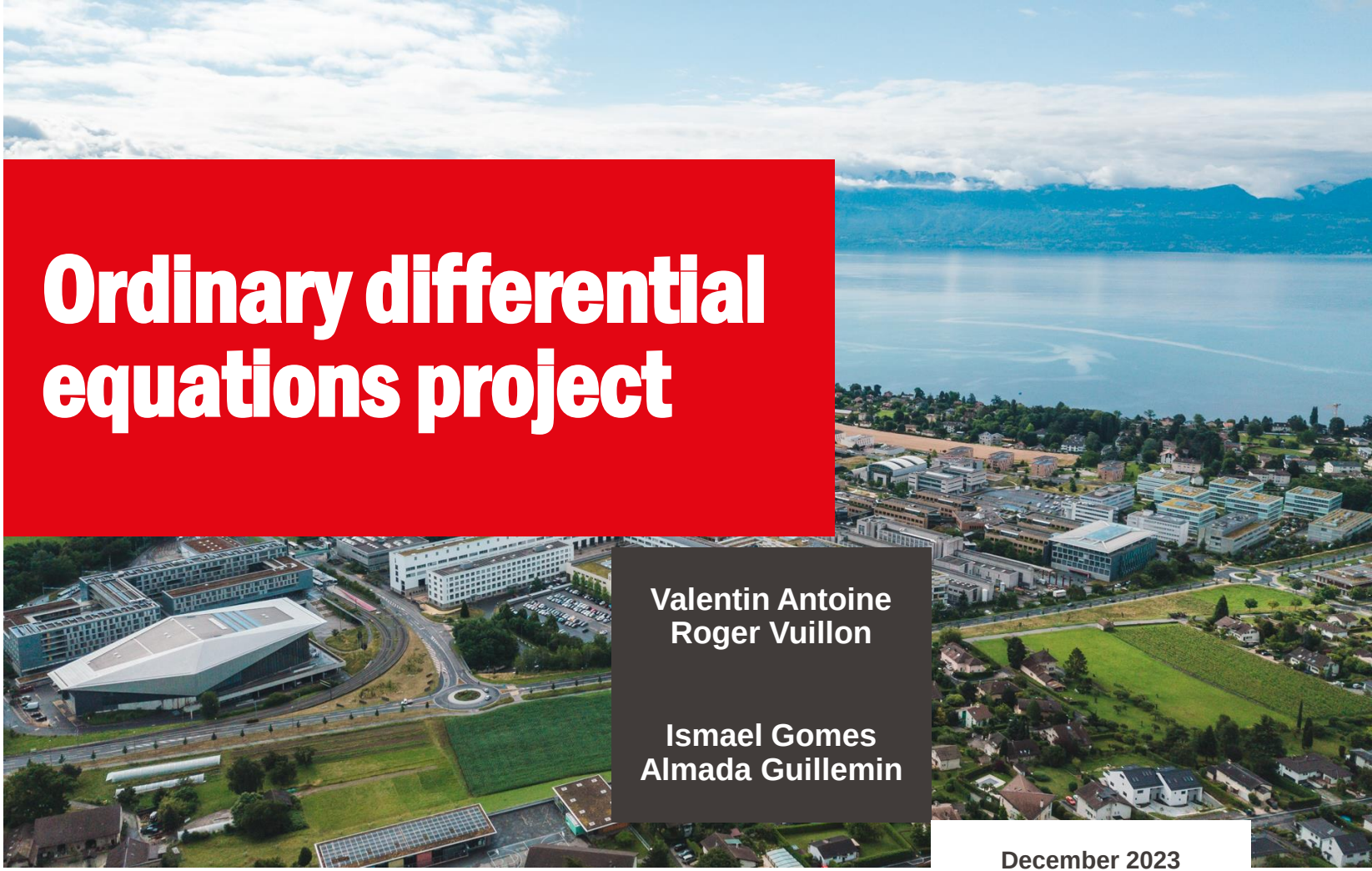
Ordinary differential equations project



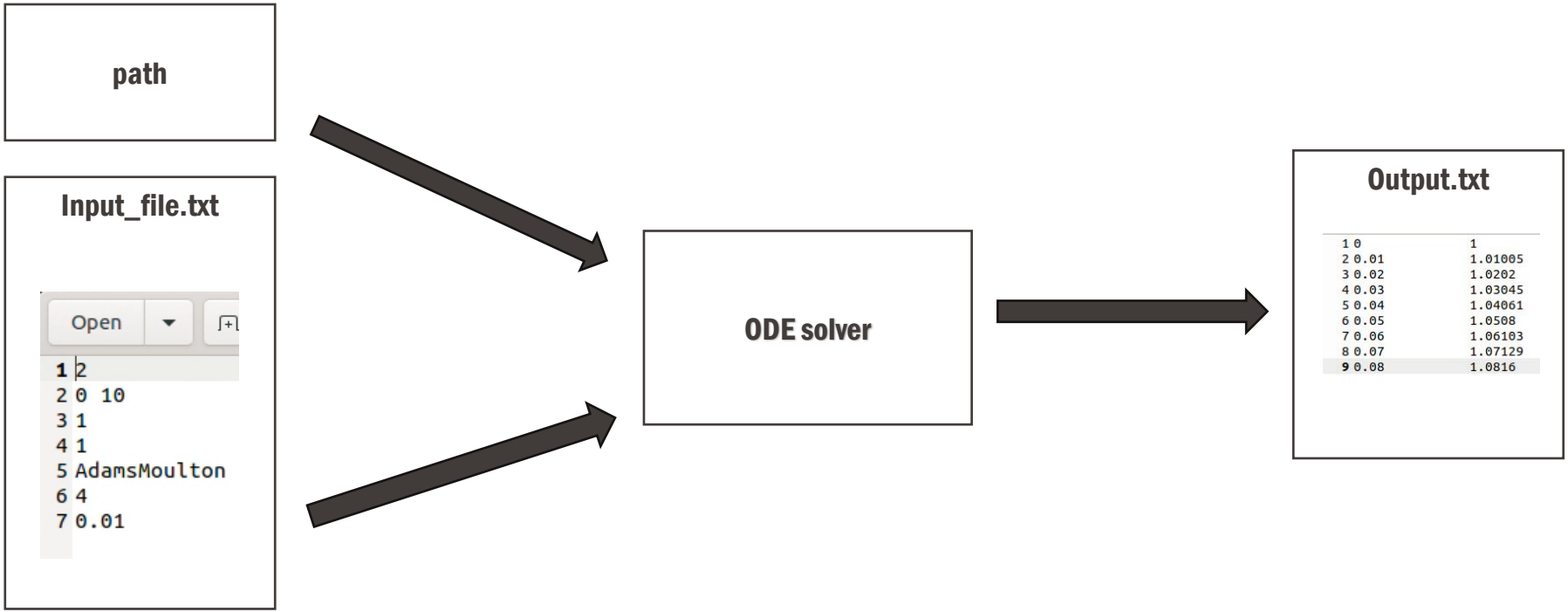
Valentin Antoine
Roger Vuillon

Ismael Gomes
Almada Guillemin

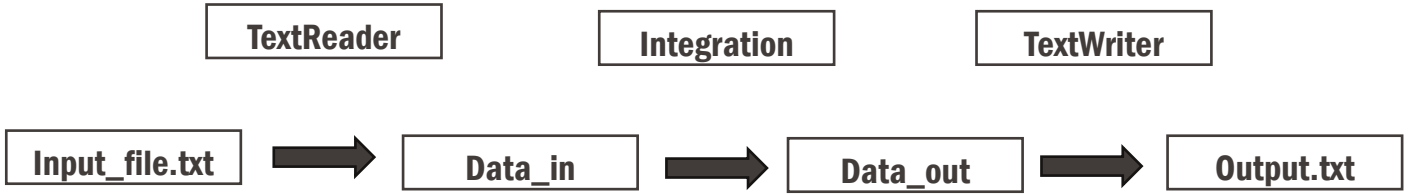
December 2023

École
polytechnique
fédérale
de Lausanne

The program flow



The program flow



Polymorphism

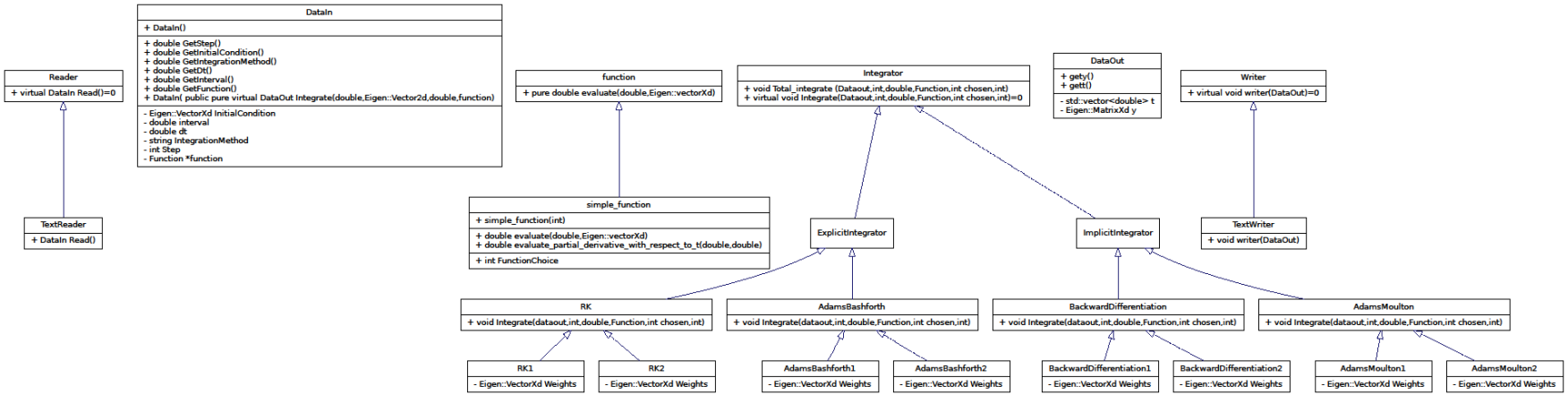
- Reader class with .read method: could read with text file, terminal…
- Total integrate can adapt to any integration method*
- f.evaluate() a general way to evaluate functions: SimpleFunction for instance
- Integrate adapts for RK to RK1, RK2, RK3, RK4 and this is the case for every integration methods
- Writer class with .write method: a general way to write data

The conception: Data structures

- Data_in class: Eigen::VectorXd Initial condition, double Interval [2], double dt, int number_of_nodes, string integration method, int step, Function *f, int dimension

- Data_out class: std::vector<double>, Eigen::MatrixXd y;

The conception: class diagram



The conception: features

Can solve equations of the type

$dy1/dt(t)=f1(t,y1(t),y2(t),y3(t)...)$

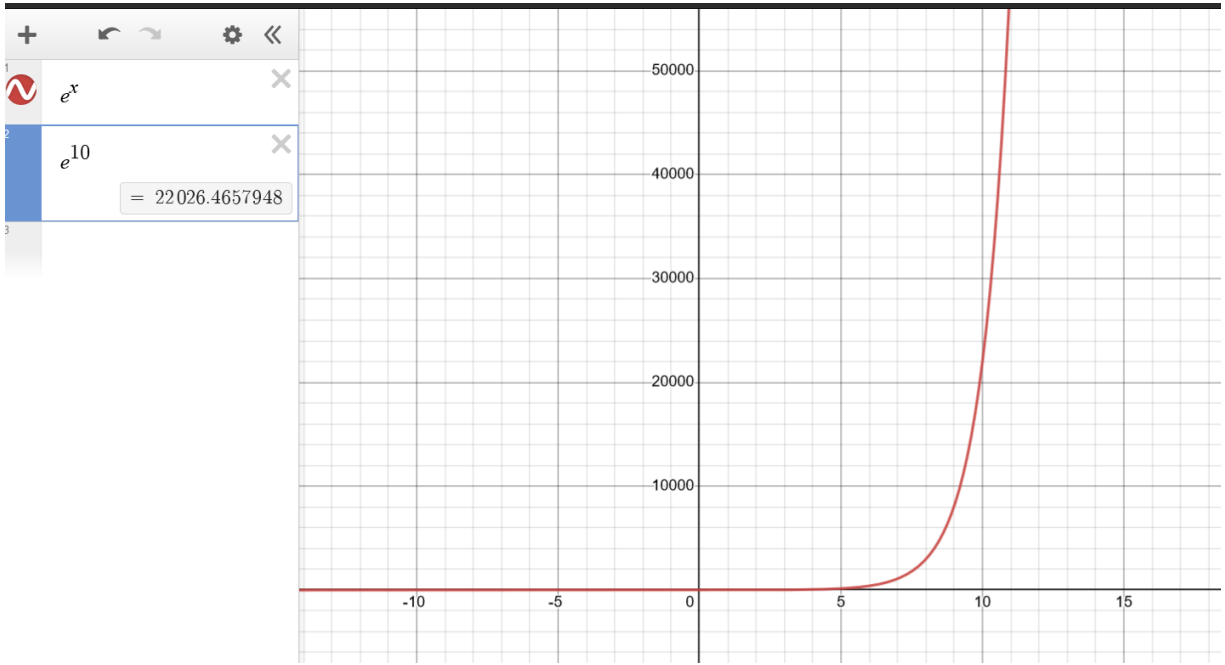
$dy2/dt(t)=f2(t,y1(t),y2(t),y3(t)...)$

$dy3/dt(t)=f3(t,y1(t),y2(t),y3(t)...)$

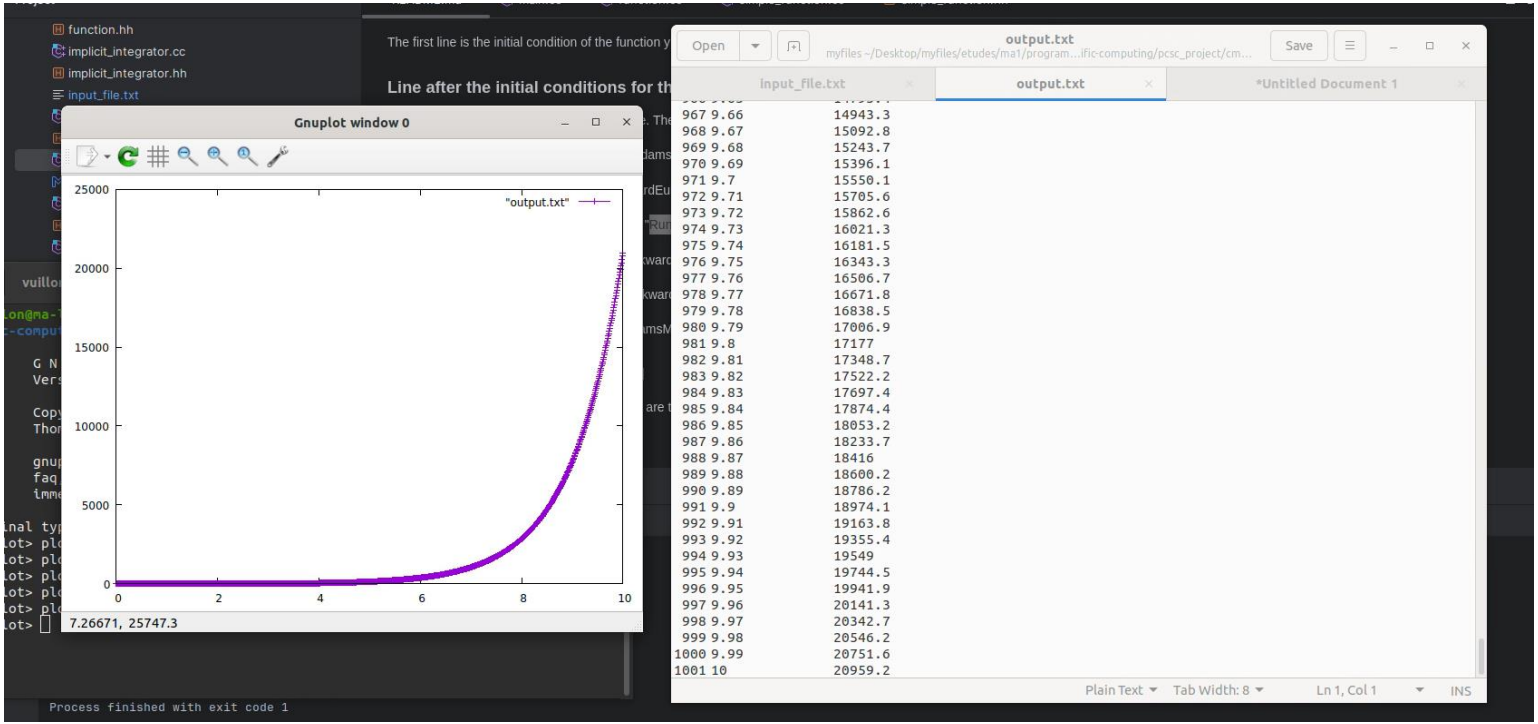
...

With set of simple functions

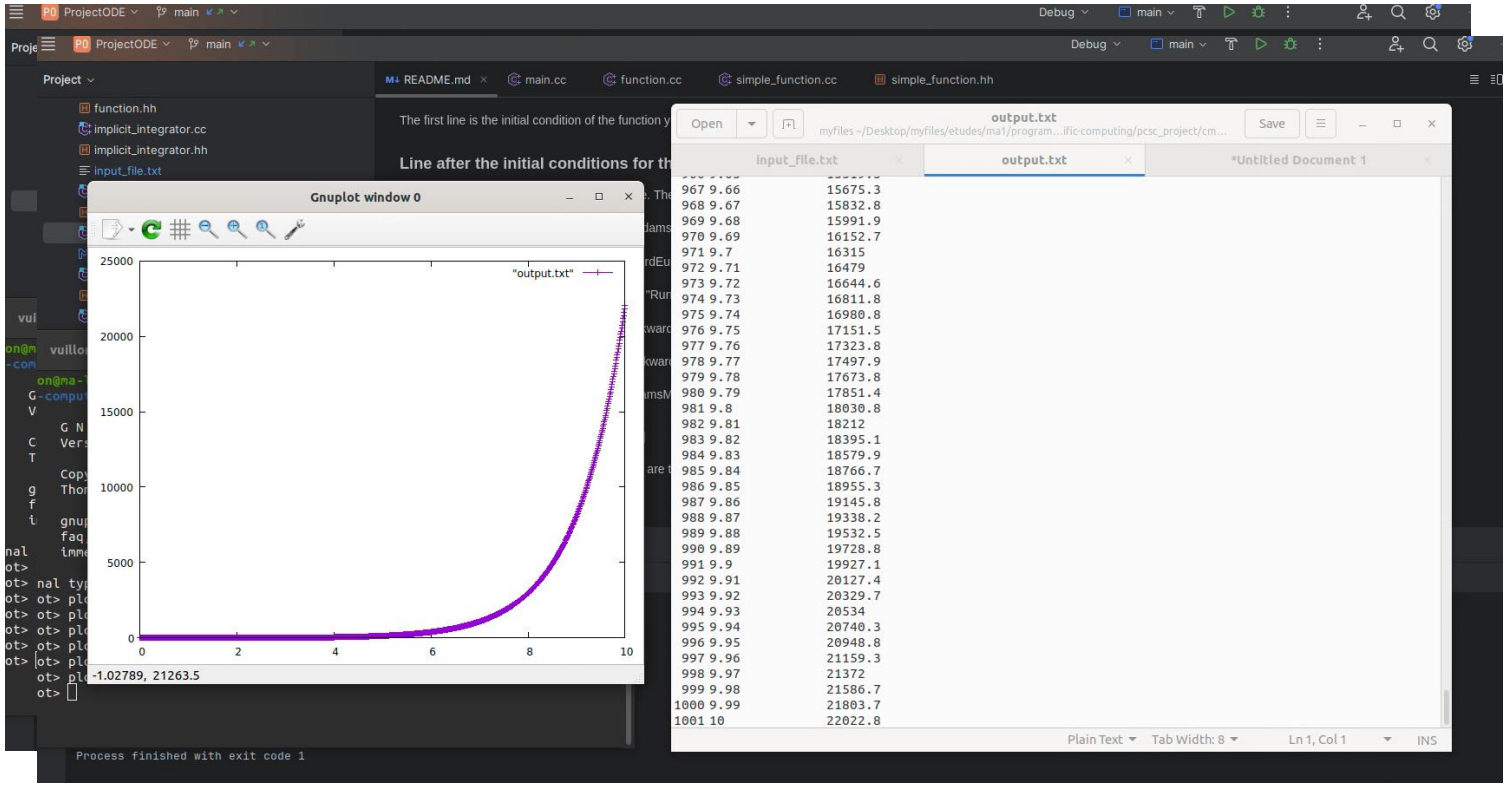
The conception: tests



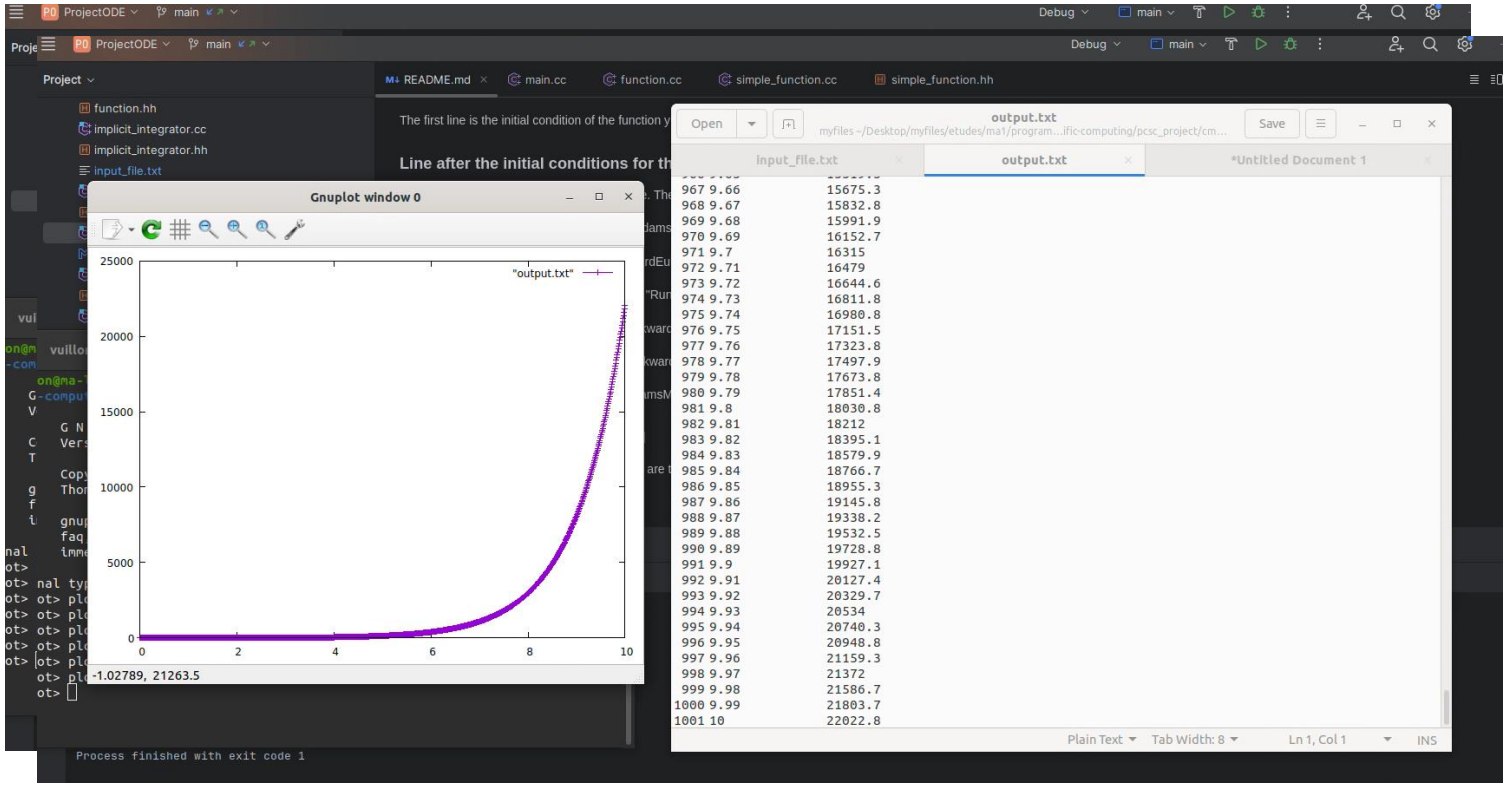
The conception: tests (RK1)



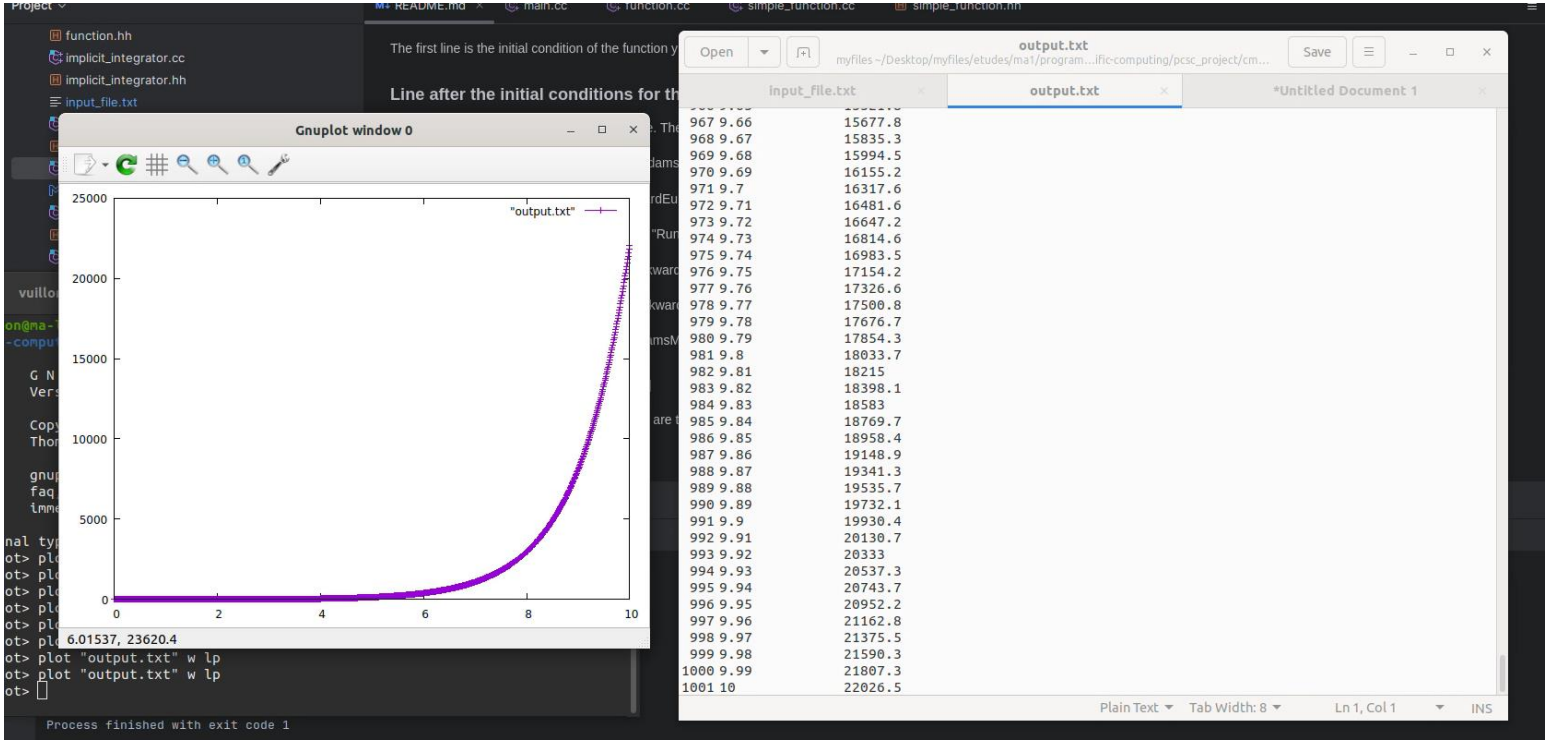
The conception: tests (RK2)



The conception: tests (RK3)



The conception: tests (RK4)



Issues and possible improvements

- Issues with the generalization to systems of equations
- Issue with Adam Moulton Method, missing evaluation of function f

$$
\begin{aligned}
y_n &= y_{n-1} + hf(t_n, y_n), \\
y_{n+1} &= y_n + \frac{1}{2}h\left(f(t_{n+1}, y_{n+1}) + f(t_n, y_n)\right), \\
y_{n+2} &= y_{n+1} + h\left(\frac{5}{12}f(t_{n+2}, y_{n+2}) + \frac{8}{12}f(t_{n+1}, y_{n+1}) - \frac{1}{12}f(t_n, y_n)\right), \\
y_{n+3} &= y_{n+2} + h\left(\frac{9}{24}f(t_{n+3}, y_{n+3}) + \frac{19}{24}f(t_{n+2}, y_{n+2}) - \frac{5}{24}f(t_{n+1}, y_{n+1}) + \frac{1}{24}f(t_n, y_n)\right), \\
y_{n+4} &= y_{n+3} + h\left(\frac{251}{720}f(t_{n+4}, y_{n+4}) + \frac{646}{720}f(t_{n+3}, y_{n+3}) - \frac{264}{720}f(t_{n+2}, y_{n+2}) + \frac{106}{720}f(t_{n+1}, y_{n+1}) - \frac{19}{720}f(t_n, y_n)\right).
\end{aligned}
$$

- All the methods except Runge Kutta methods are Linear Multistep Methods -> code could have been factorized thanks to this
- Only a set of hard coded function. Implementation of formula parser would be the only way to have a useful ODE solver for scientific applications

Thank you

Valentin Antoine
Roger Vuillon

Ismael Gomes Almada Guillemin

École polytechnique fédérale de Lausanne