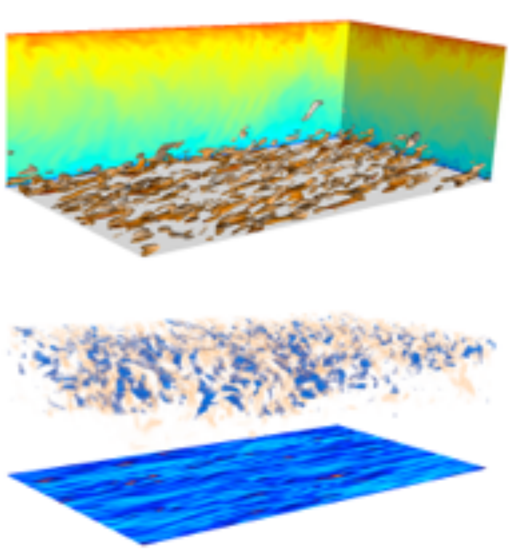
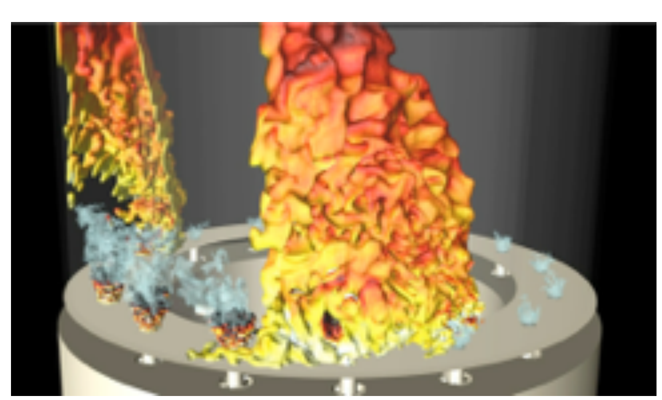
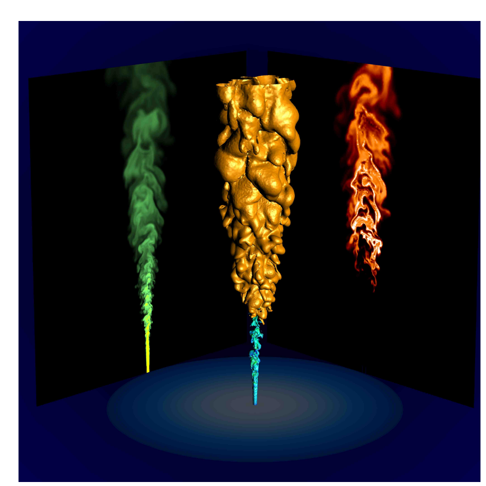
Numerical Methods in Engineering Applications

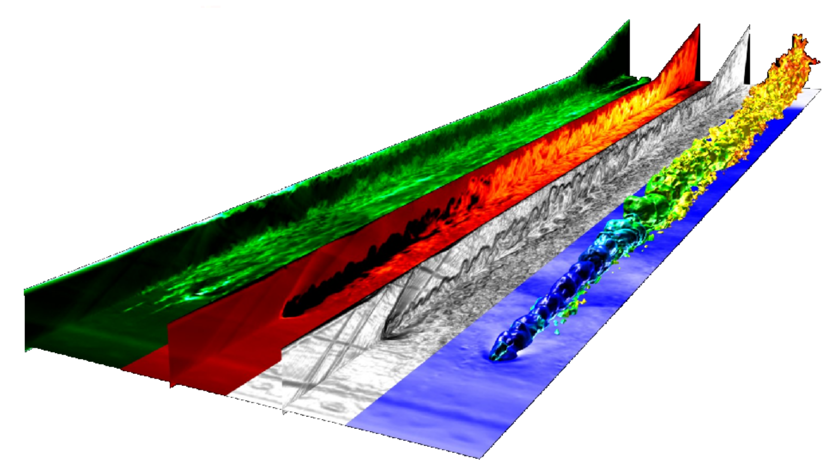
Session #3
Elliptic PDEs (1)

Ronan.Vicquelin@centralesupelec.fr
Aymeric.Vie@centralesupelec.fr









Course contents

● Theoretical lecture
■ Problem-solving workshop

I. Basics on numerical approximations

- Introduction and Finite Differences. ● ■ ■
- Numerical solution of ordinary differential equations. ● ■

II. Solving large linear equations systems: Applications to steady heat equation.

- Elliptic PDE 1. ● ■ ■
- Elliptic PDE 2. ● ■

III. Methods for unsteady advection/diffusion problems

- Hyperbolic and parabolic PDE: Explicit methods. ● ■
- Characterization of numerical errors. ● ■
- Hyperbolic and parabolic PDE: Implicit methods. ● ■

IV. Towards computational fluid dynamics

- Methodology in numerical computations. ■ ■
- Incompressible Flow equations. ● ■
- Semi-Implicit method for incompressible flows. ● ■
- Final project on incompressible flow. ■ ■

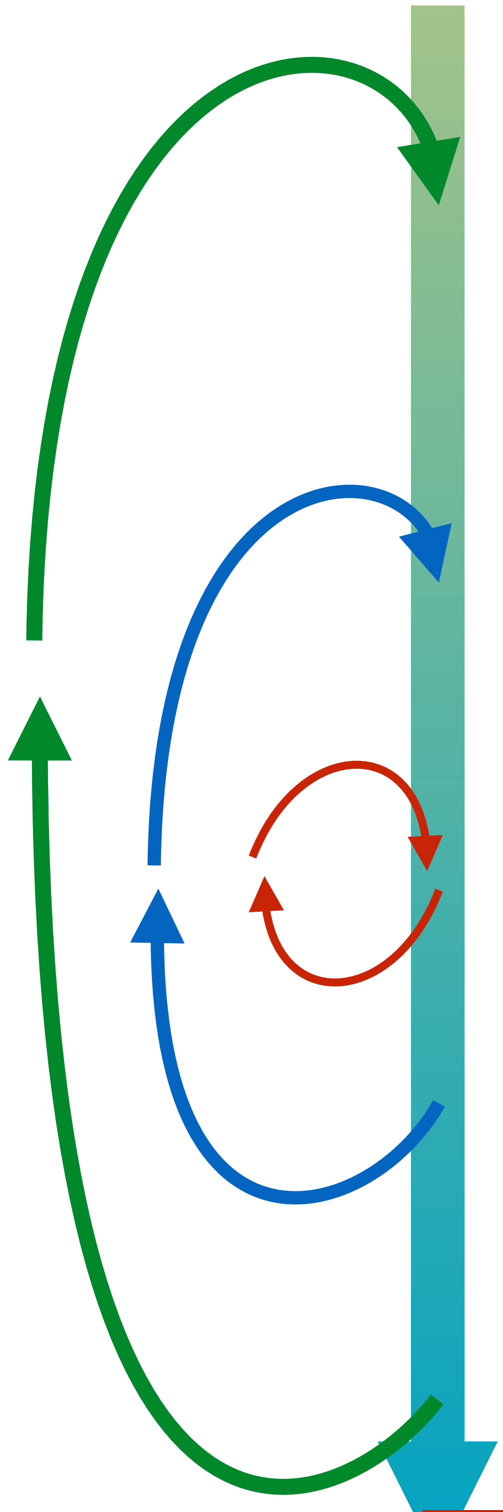
Today's contents

- ODE
  - Runge-Kutta and Multistep methods

- Elliptic PDEs
  - Resolution of Elliptic PDEs
  - Discretized 2D Poisson Equation
  - Direct vs Iterative methods
  - Jacobi method
  - Gauss-Seidel method

Next time: Beyond 19^{th} century ! : SOR, CG

Methodology



- Set the physical problem
- Formalization: Equations and Hypotheses
- Simplifications, symmetries
- Discretization (structures, unstructured mesh)
- Boundary conditions. Initial conditions.
- **Numerical resolution** (finite differences, finite volumes, finite elements, ...)
- Characterization of numerical error (accuracy of the numerical scheme, mesh convergence/adaptation)
- Data post-processing
- Model validation, Verification of hypotheses

The problem answer

ODE Integration methods

- We have seen : Forward Euler, Backward Euler, Trapezoidal method

- Higher order methods

  - One-step methods : Runge-Kutta
    Higher accuracy by evaluating f(y,t) several time between t_n and t_{n+1}

  - Multi-step methods
    Use knowledge of previous steps $y_{n-1},\ y_{n-2},\ \ldots,\ f(y_{n-1}),\ f(y_{n-2})$

ODE Integration methods

- We have seen : Forward Euler, Backward Euler, Trapezoidal method
- Higher order methods

	One-step	Multi-step
Explicit	Explicit Runge-Kutta methods	Adams-Bashforth, Leap-Frog, …
Implicit	Implicit Runge-Kutta methods	Adams-Moulton, BDF, …

Example of Runge-Kutta methods

RK2

2-stage, explicit, 2nd-order

$$k_1 = f(t_n, y_n)$$

$$k_2 = f(t_n + \frac{h}{2}, y_n + \frac{1}{2}hk_1)$$

$$y_{n+1} = y_n + hk_2$$

RK4

4-stage, explicit, 4th-order

$$k_1 = f(t_n, y_n)$$

$$k_2 = f(t_n + \frac{1}{2}h, y_n + \frac{1}{2}hk_1)$$

$$k_3 = f(t_n + \frac{1}{2}h, y_n + \frac{1}{2}hk_2)$$

$$k_4 = f(t_n + h, y_n + hk_3)$$

$$y_{n+1} = y_n + \frac{h}{6}k_1 + \frac{h}{3}k_2 + \frac{h}{3}k_3 + \frac{h}{6}k_4$$

General explicit Runge-Kutta methods

s-stage explicit Runge-Kutta

$$
\begin{aligned}
k_1 &= f(x_0, y_0) \\
k_2 &= f(x_0 + c_2 h, y_0 + h a_{21} k_1) \\
k_3 &= f\left(x_0 + c_3 h, y_0 + h\,(a_{31} k_1 + a_{32} k_2)\right) \\
&\ldots \\
k_s &= f\left(x_0 + c_s h, y_0 + h\,(a_{s1} k_1 + \ldots + a_{s,s-1} k_{s-1})\right) \\
y_1 &= y_0 + h\,(b_1 k_1 + \ldots + b_s k_s)
\end{aligned}
$$

General explicit Runge-Kutta methods

Expressed in Butcher table

$$
\begin{array}{c|ccccc}
0 & & & & & \\
c_2 & a_{21} & & & & \\
c_3 & a_{31} & a_{32} & & & \\
\vdots & \vdots & \vdots & \ddots & & \\
c_s & a_{s1} & a_{s2} & \cdots & a_{s,s-1} & \\
\hline
 & b_1 & b_2 & \cdots & b_{s-1} & b_s
\end{array}
$$

RK2 (standard)

$$
\begin{array}{c|cc}
0 & & \\
1/2 & 1/2 & \\
\hline
 & 0 & 1
\end{array}
$$

RK3 (Heun)

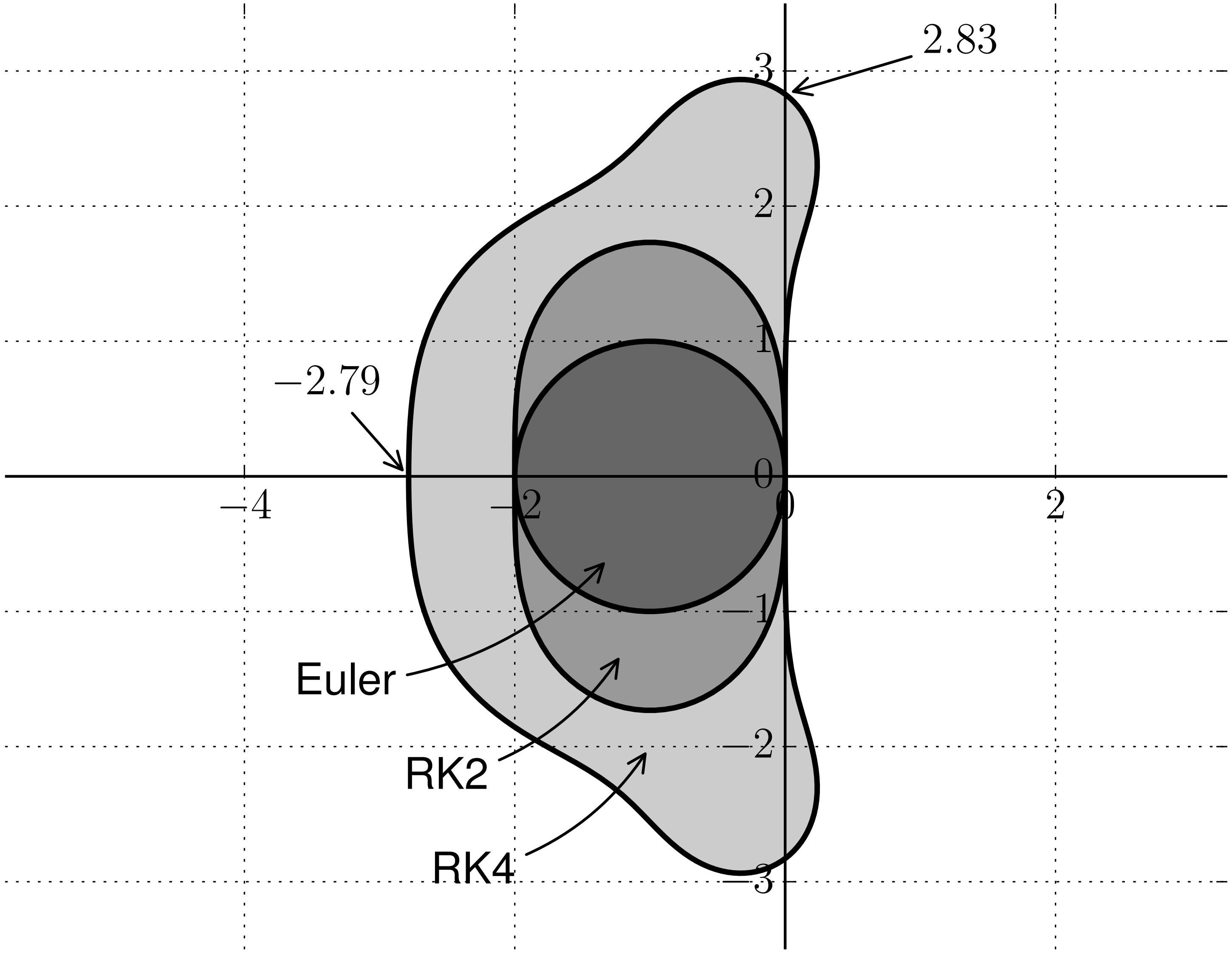
$$
\begin{array}{c|ccc}
0 & & & \\
1/3 & 1/3 & & \\
2/3 & 0 & 2/3 & \\
\hline
 & 1/4 & 0 & 3/4
\end{array}
$$

RK4 (standard)

$$
\begin{array}{c|cccc}
0 & & & & \\
1/2 & 1/2 & & & \\
1/2 & 0 & 1/2 & & \\
1 & 0 & 0 & 1 & \\
\hline
 & 1/6 & 2/6 & 2/6 & 1/6
\end{array}
$$

Comparaison of stability regions

For the conditionally stable methods :
Euler, RK2, RK4



Example of multi-step methods

- Adams-Bashforth, explicit, 2nd-order

$$y_{n+1} = y_n + \frac{3}{2}hf_n - \frac{1}{2}f_{n-1}$$

- Leap-Frog (aka Midpoint-rule), explicit, 2nd order

$$y_{n+1} = y_{n-1} + 2hf_n$$