HPCCamp 2017 - ECAR 2017

Introduction to computational mechanics in multiphysics

Mariano Vázquez

Barcelona Supercomputing Center

September 2017
FCEN - UBA
Buenos Aires
Argentina

Background

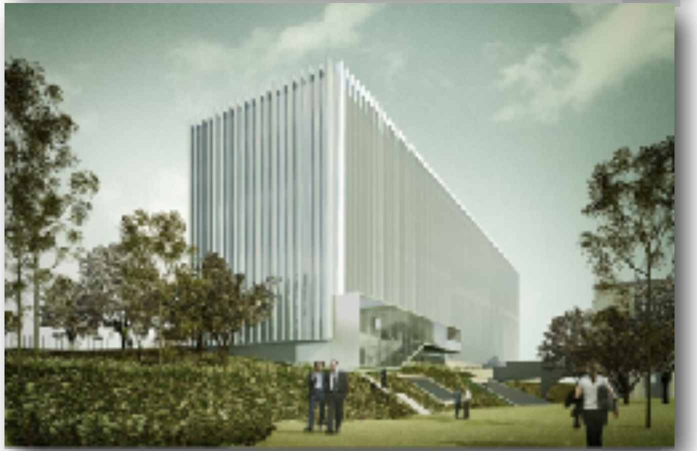
Barcelona Supercomputing Center

BSC-CNS is the Barcelona Supercomputing Center – Centro Nacional de Supercomputación, the Spanish national supercomputing center

Director: Prof. Mateo Valero

Established in 2005. Upgrades the former parallelisation research center CEPBA

It is a **public center**, co-financed by the Spanish Ministry of Science, the regional government of Catalonia and the UPC (Technical University of Catalonia)



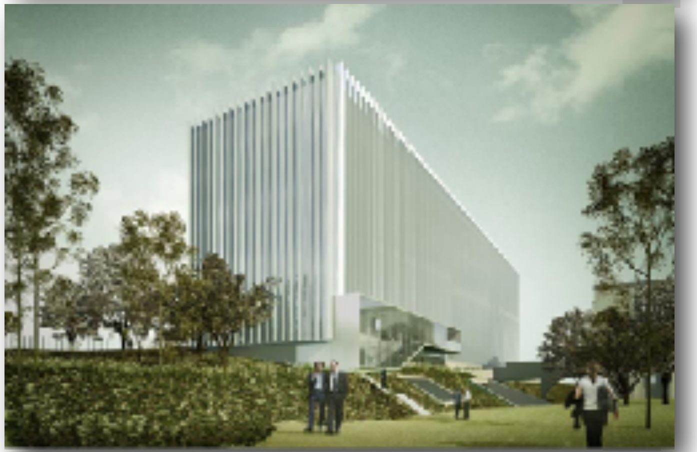
Barcelona Supercomputing Center

Around 500 researchers from several disciplines

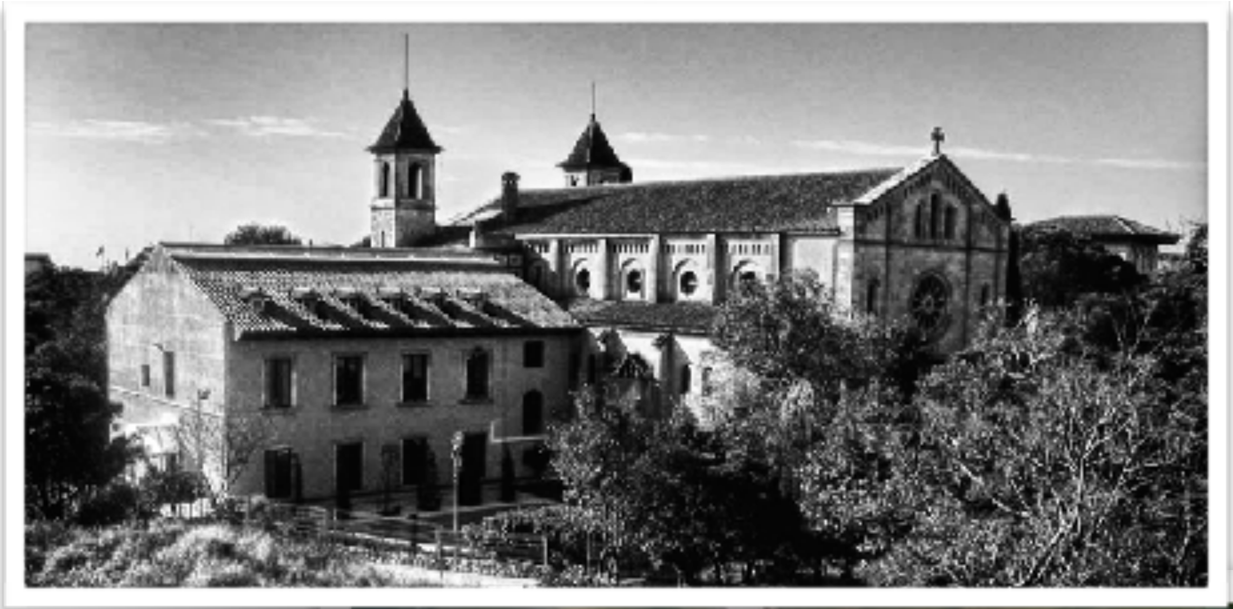
It hosts the **MareNostrum**, former largest European supercomputer (2005 and 2007), former 4th and 5th in the World.

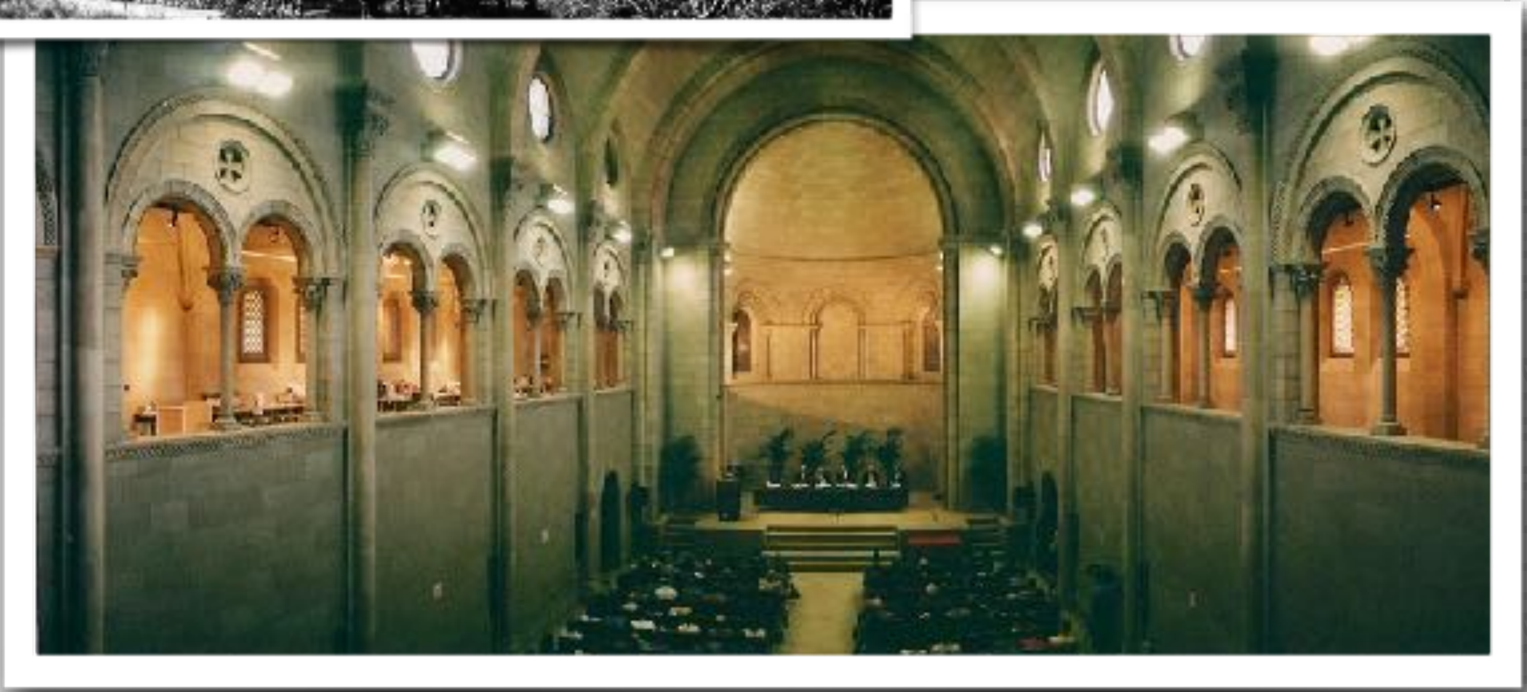
Manage the **Spanish Supercomputing Network** (RES)

Tier 0 of **PRACE-IP** European supercomputing infrastructure project



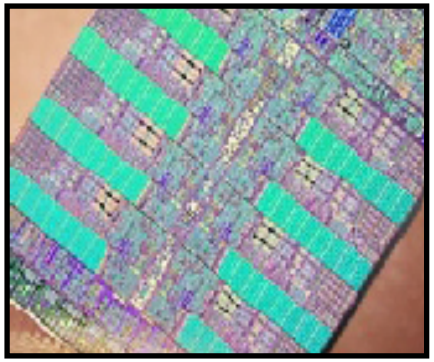
Barcelona Supercomputing Center





Barcelona Supercomputing Center

BSC-CNS Research Departments

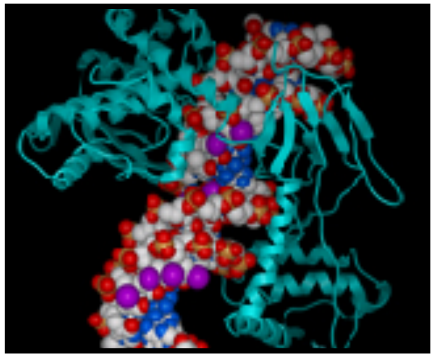


Computer Science

Performance tools

Computer architectures

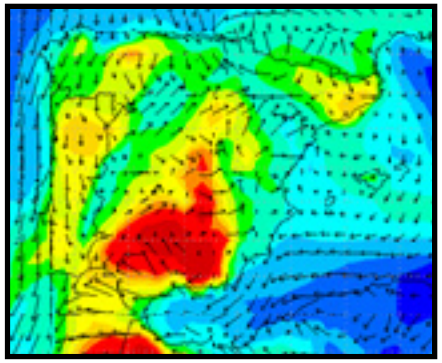
Programming models



Life Science

Genomics

Proteomics



Earth Science

Air quality

Climate

Computer Applications in Science and Engineering

CASE

CASE Department – Objectives

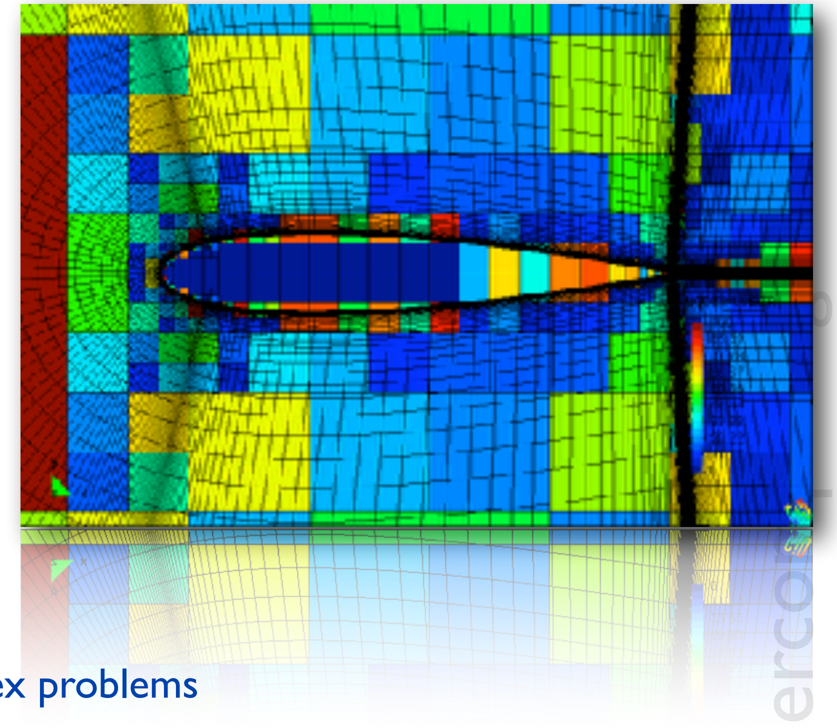
Computer Applications in Science

and Engineering (CASE) Department

Computational Physics and Engineering

Interdisciplinary research unit of

the BSC-CNS



Our mission:

To develop computational tools to simulate highly complex problems

seamlessly adapted to run onto high-end parallel supercomputers

Around 90 researchers:

Post-docs, students, programmers

Computer Science, Physicists, Mathematicians, Engineers

Research lines

Physical and Numerical Modeling

Numerical Solution Algorithms: from stabilisation to solvers

Multi-physics and multi-scale coupling

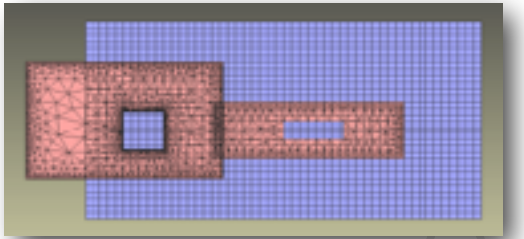
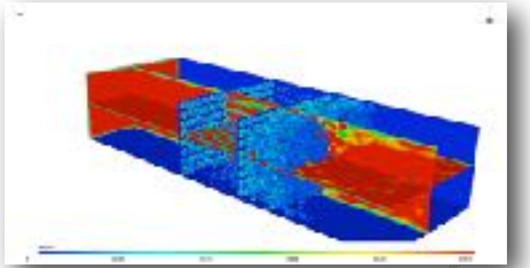
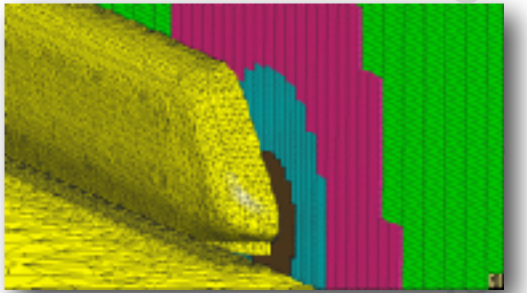
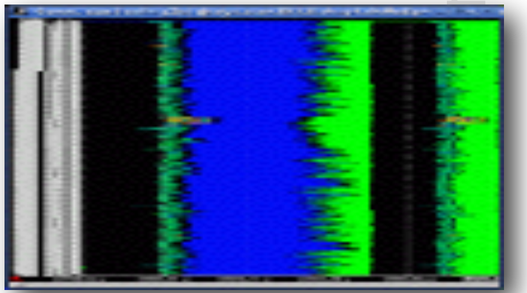
High Performance Computing in CM (HPCM)

Parallelisation in Distributed and Shared memory machines

Mesh Generation

Scientific Visualisation & Big Data

Optimisation

Application lines

Environment

Energy

Aerospace

Trains and Automotive

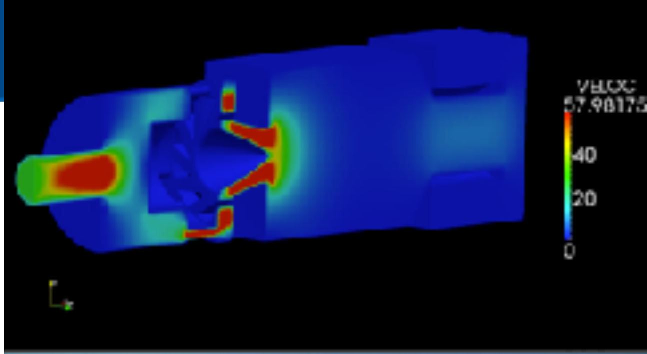
Oil and Gas

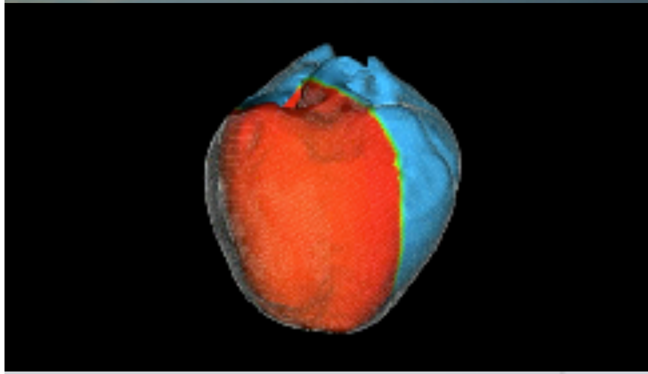
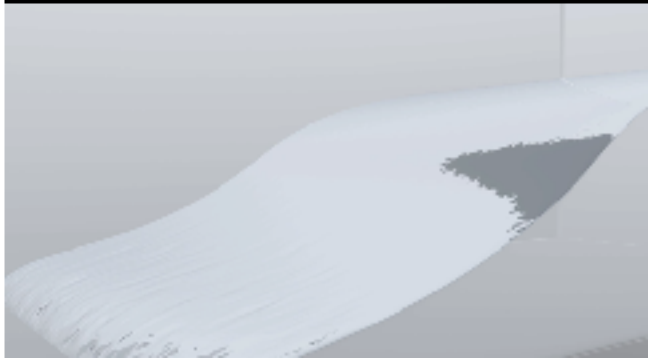
Smart Cities

High Energy Physics

Materials Sciences

Biomechanics



CASE Department Action

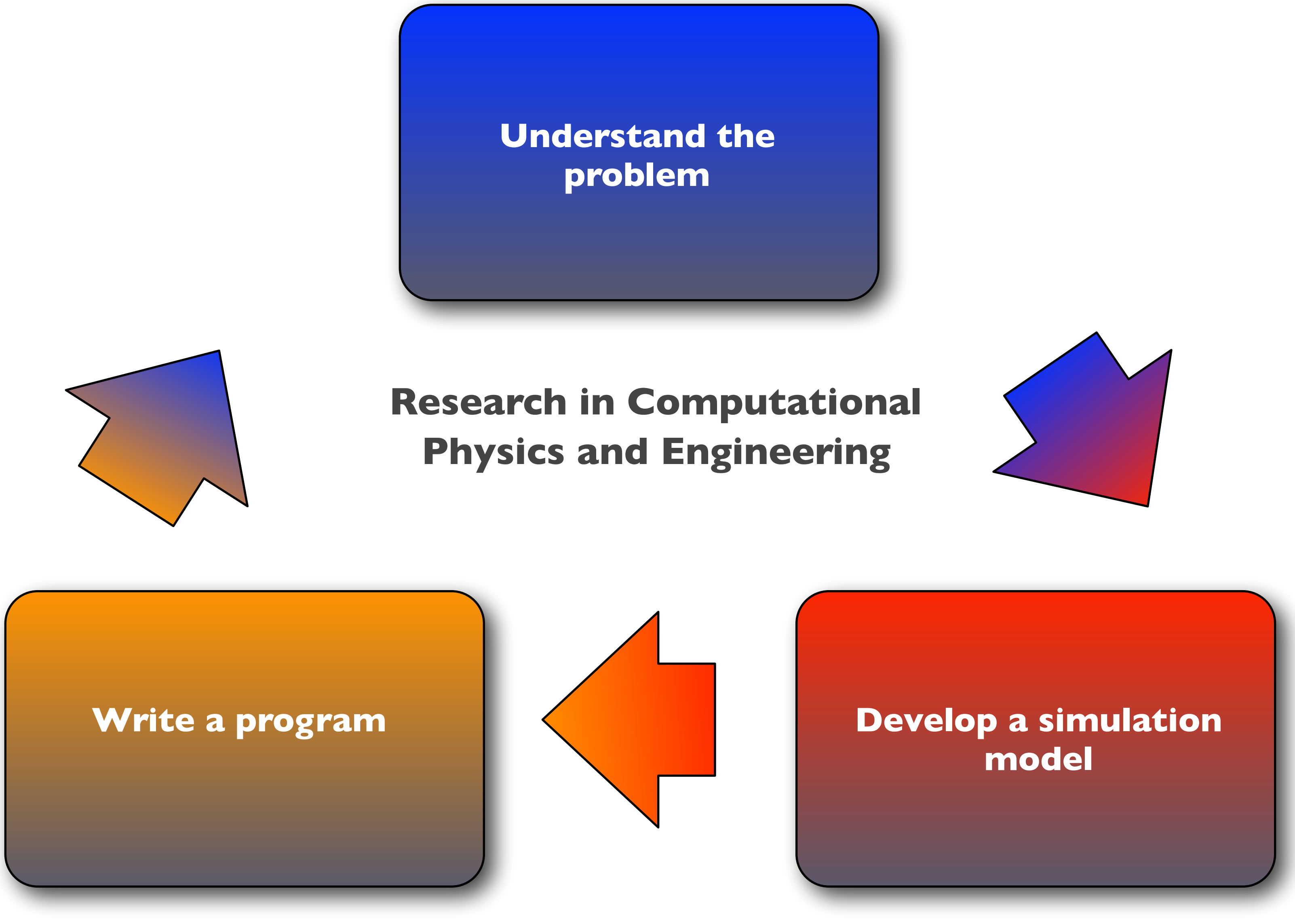
Physics

Research in Computational Physics and Engineering

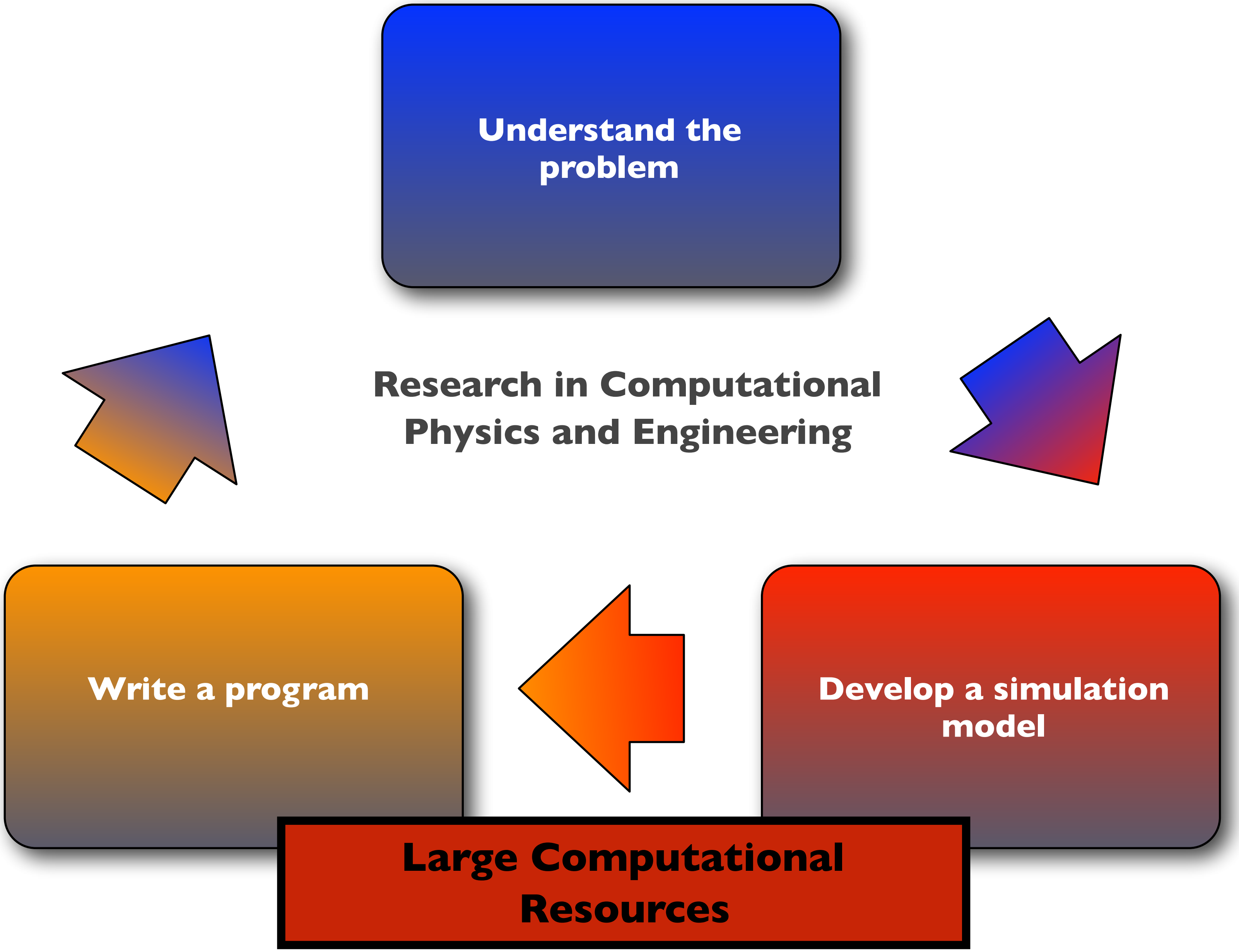
Computer Science

Mathematics

CASE Department Action


Understand the problem
Research in Computational Physics and Engineering
Develop a simulation model
Write a program

CASE Department Action


Understand the problem
Research in Computational Physics and Engineering
Develop a simulation model
Write a program
Large Computational Resources

Course Goal:

Show bridges over the gaps between:

- Computer Science and Computational Mechanics
- Computational Mechanics and Engineering
- Academia and Industry

What is behind a parallel simulation code?

Scarce information on how to program things

Most of the time… no information at all!

You only understand it when you program it.

What is behind a parallel simulation code?

⇩

Complex Multi-Physics Applications

The real world

Motivation

Introduction: Motivation

Physical Understanding:

We deal with Physical systems

Mathematical Models:

Governed by Differential Equations...

... Numerically solved

What is behind a simulation code?

Computer Science:

... and translated in a Computational Model

Introduction: Motivation

Computational Mechanics:
A definition as a discipline on its own

Oden, Belytschko, Babuska and Hughes (2003):

> **Theoretical and applied mechanics (TAM)** is the branch of applied science concerned with the study of **mechanical phenomena**: the behavior of fluids, solids, and complex materials under the actions of forces.[...]
>
> **Computational mechanics (CM**) is that sub-discipline of TAM concerned with the use of **computational methods and devices** to study events governed by the principles of mechanics.

High Performance Computational Mechanics:

A CM sub-discipline

Efficient use of **HPC resources, no matter the size**

Introduction: Motivation

Computational Science or Scientific Computing:
A definition as a discipline on its own (again)

Computational Science constructs **mathematical models** and uses **computers** to analyse and solve scientific problems.

Its main products are **computer simulations** and other forms of computation from numerical analysis and theoretical **computer science** to problems in **various scientific disciplines**.

Shortly, **Computational Physics or Computational Engineering**

High Performance Computational Science /Scientific Computing

Introduction: Motivation

Theoretical Physics

Experimental Physics

The 2 pillars of Physics

Introduction: Motivation

Theoretical Physics

Experimental Physics

Computational Physics

The 3 pillars of Physics

Introduction: Motivation

Theoretical Physics

Physical model definition
Classical, Quantum, Relativistic...
Fluid, solids, electromagnetism, gravitation, rigid body...

Experimental Physics

Experiments and observation
Wind tunnels, particle accelerators, meteorology, astrophysics...

Computational Physics

Numerical methods and programming

Application areas:
Engineering, Aeronautics, Meteorology, Biology, Astrophysics...

Introduction: Motivation

Mature

Deep mathematical basis, reliable (not just finite differences)

Very flexible and powerful programming tools (languages, compilers, ...)

The only way to attack some problems (**a lot** of problems indeed...)

It allows to verify and improve the theory and design new experiments

Large (and growing) computers available

Computational Physics

Introduction: Motivation

Computational Mechanics:
A course

My own taste… a Physicist's Manifesto.

Even if you write a code or not, you need to know the basics (the real basics!)

Mathematics, physics and programming are deeply entangled:

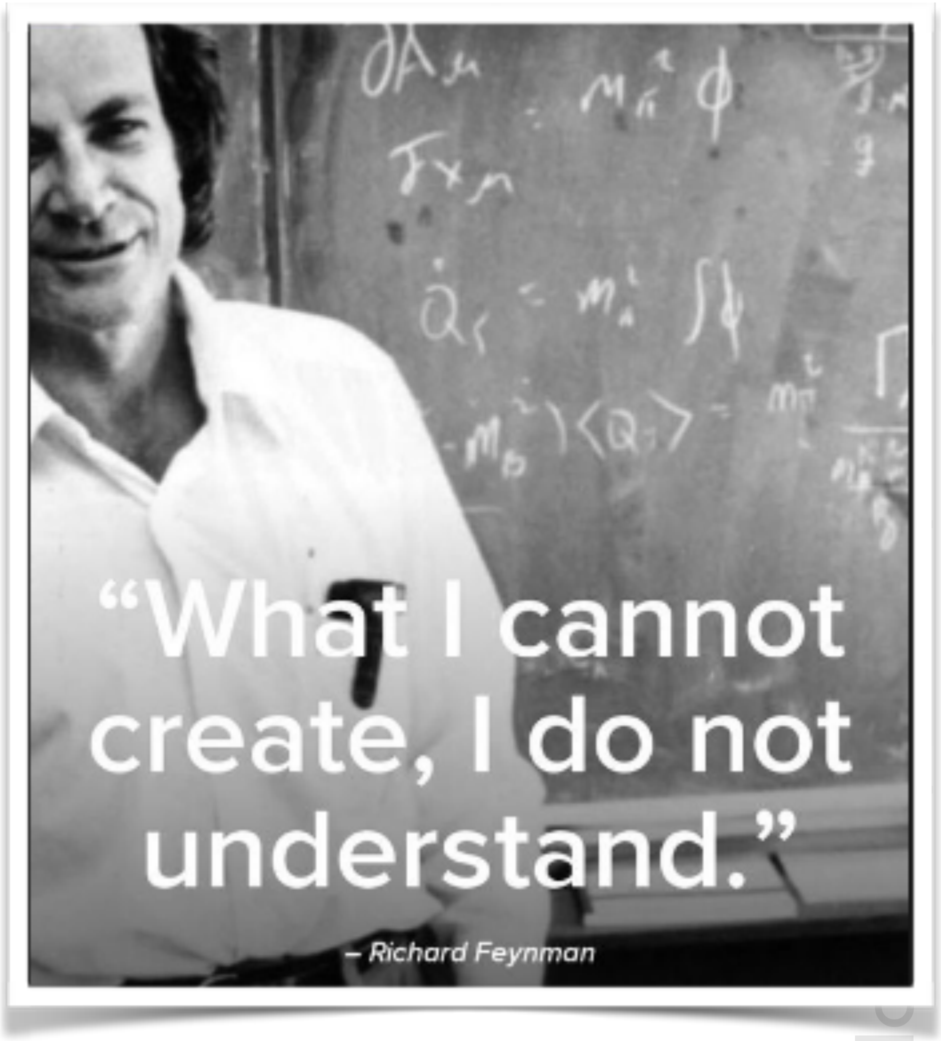
Maths: know at least the basics on why and how to interpret what Nature is saying

Physics: understand both ends, i.e., what is the motivation and how a simulation project develops, from the beginning to the end

Programming: try to use a "programmer's mind". Today, plenty of good tools to do it: Matlab, Python, PETSc, … Tutorials, hands-ons, …

Introduction: Motivation

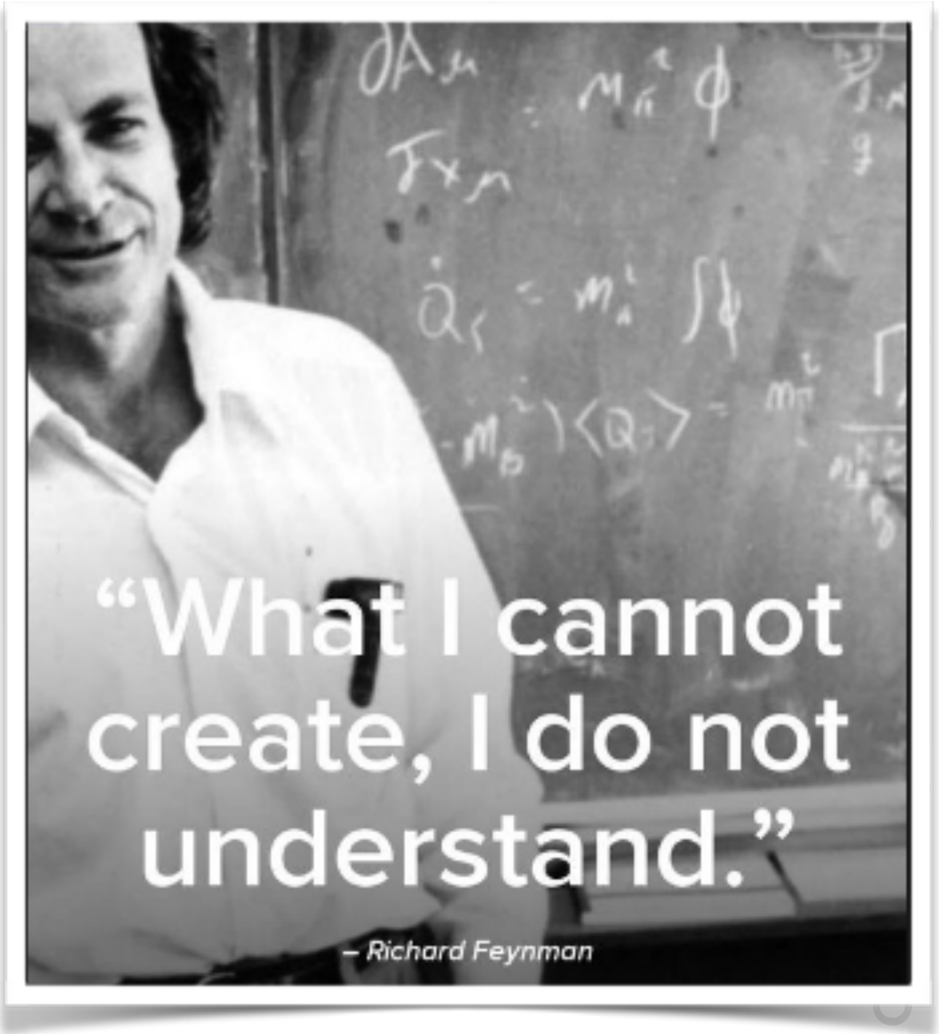
Computational Mechanics:
A course



Introduction: Motivation

Computational Mechanics:
A course

“... and programming is creating.”



The Physical System and its Mathematical Description

The Physical System

Revisiting the definition of CM...

Governed by forces, or more generally by conservation principles.

Usually modelled by Partial or Ordinary Differential Equations (PDE - ODE) but maybe...

... many of them and coupled (i.e. combustion, species transport...)

... multi-physics (Fluid-structure Interaction -FSI-, multi-scale modelling...)

... non-local phenomena ("all against all" connectivities, infinite speed...)

... design variables exist (optimisation problem)

Usually highly non-linear and highly transient

Usually many of these features appears at the same time... i.e. **large problems**.

The Physical System

All these features strongly condition the method you use:

Either if you use your own code

or

if you use another one's code (commercial or open source)

The consequence: know deeply the Physics!

The Physical System

All these features strongly condition the method you use:

- Physical modeling
  - ✓ Transient vs. Stationary: *time step? what's that? how do I set it?*
  - ✓ Incompressible vs. Compressible: *Mach number in water? Shock waves?*
  - ✓ RANS vs. LES turbulence models: *what scales do I need to solve? application ranges?*
  - ✓ Material design: *what material do I have to use?*
  - ✓ Fluid-Structure Interaction: *is fluid deforming the structure? Transient?*
  - ✓ Boltzman, SPH, ...: *what is this? what are the limitations? where are they coming from?*

The Physical System

All these features strongly condition the method you use:

Physical modeling

- ☑ Transient vs. Stationary: *robustness? convergence?*
- ☑ Incompressible vs. Compressible: *convergence? discontinuities?*
- ☑ RANS vs. LES turbulence models: *how do I couple the problem? convergence?*
- ☑ Material design: *what solution scheme? do I have the tangent matrix?*
- ☑ Fluid-Structure Interaction: *how do I couple the problem? added mass?*
- ☑ Boltzman, SPH, ...: *how do they behave with respect to other methods?*

● Mathematical modeling

- ☑ ... same as above, but with mathematical perspective
- ☑ Optimization problem? Adjoint vs. Genetic Algorithms. Surrogate models.

The Physical System

All these features strongly condition the method you use:

Physical modeling

- ✓ Transient vs. Stationary
- ✓ Incompressible vs. Compressible
- ✓ RANS vs. LES turbulence models
- ✓ Material design
- ✓ Fluid-Structure Interaction
- ✓ Boltzman, SPH, ...

Mathematical modeling

- ✓ ... same as above, but with mathematical perspective
- ✓ Optimization problem? Adjoint vs. Genetic Algorithms

● Discretization

- ✓ FEM, FV, FD, Spectral method, Lattice Boltzmann, Clustering in SPH...
- ✓ XFEM
- ✓ Stabilization problems
- ✓ Time Integration: Explicit - Implicit

The Physical System

All these features strongly condition the method you use:

- Solution Algorithm
  - ✓ Time advance
  - ✓ Space solver: Direct vs. Iterative
  - ✓ Non-linear solver: Jacobi, Newton...
  - ✓ Preconditioners
  - ✓ Coupling strategies

The Physical System

All these features strongly condition the method you use:

Solution Algorithm

- ☑ Time advance
- ☑ Space solver: Direct vs. Iterative
- ☑ Non-linear solver: Jacobi, Newton...
- ☑ Preconditioners
- ☑ Coupling strategies

● Implementation

- ☑ FEM, FV, FD, Spectral method
- ☑ Lattice Boltzmann, Clustering in SPH...
- ☑ Parallel vs. sequential
- ☑ OpenMP vs. MPI

The Physical System

All these features strongly condition the method you use:

Solution Algorithm

- ✓ Time advance
- ✓ Space solver: Direct vs. Iterative
- ✓ Non-linear solver: Jacobi, Newton...
- ✓ Preconditioners
- ✓ Coupling strategies

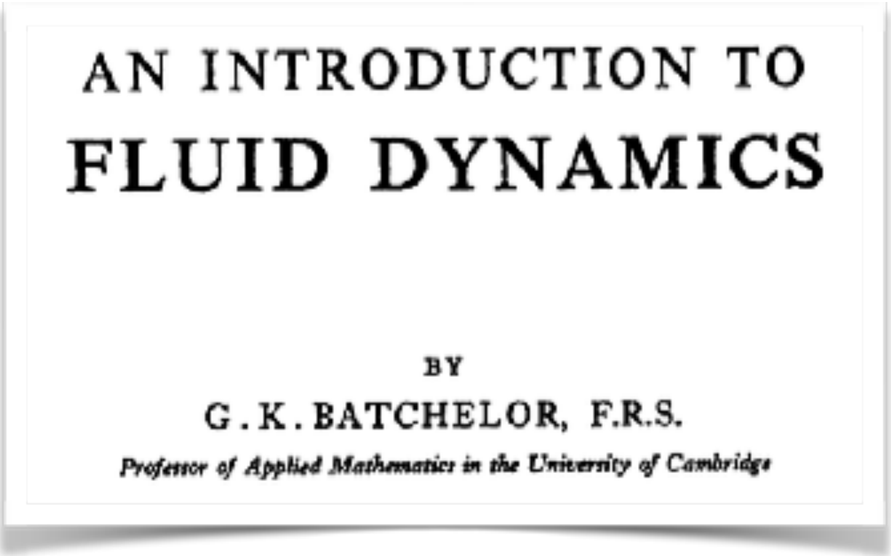
Implementation

- ✓ FEM, FV, FD, Spectral method
- ✓ Lattice Boltzmann, Clustering in SPH...
- ✓ Parallel vs. sequential
- ✓ OpenMP vs. MPI

● Validation

- ✓ How can I be sure of what I am simulating?
- ✓ Study the application ranges

The Physical System

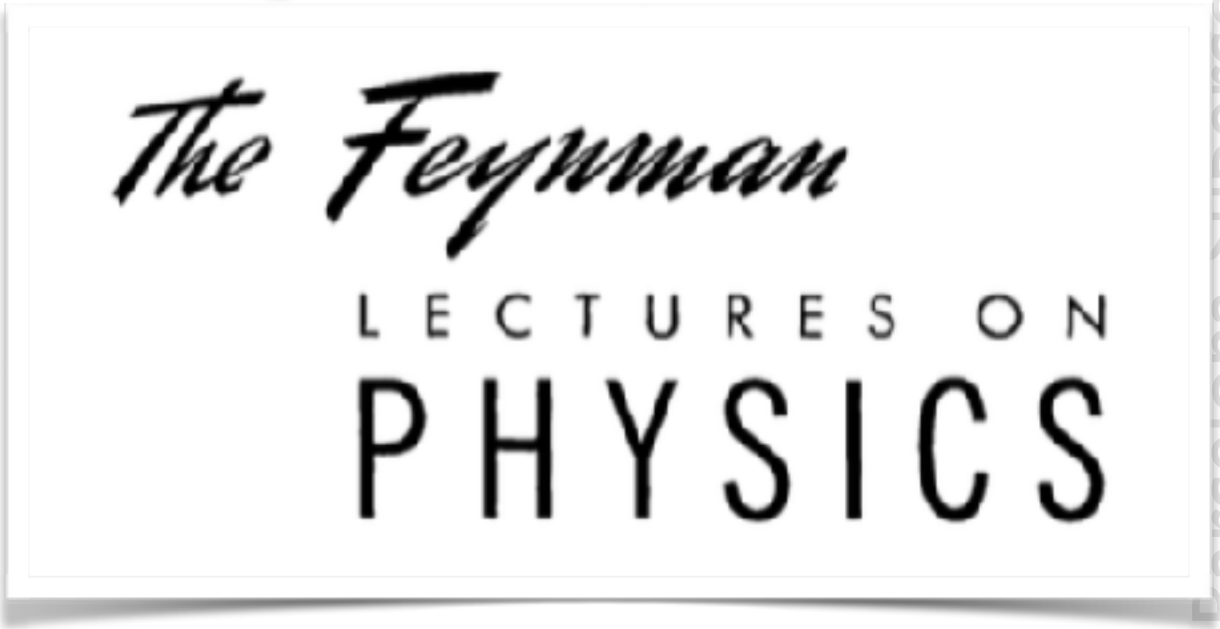

AN INTRODUCTION TO

FLUID DYNAMICS

BY

G.K. BATCHELOR, F.R.S.

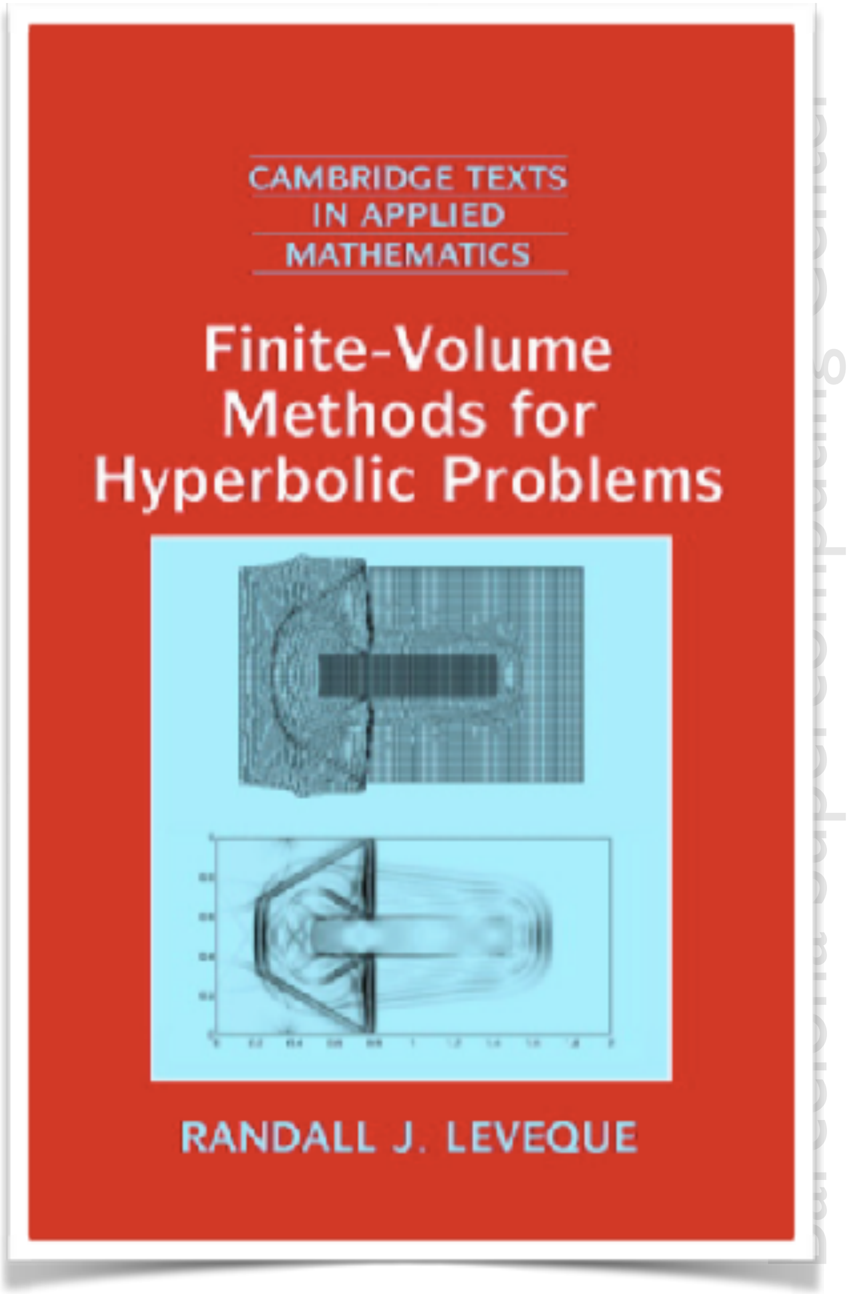
Professor of Applied Mathematics in the University of Cambridge


The Feynman

LECTURES ON

PHYSICS

The Physical System



CAMBRIDGE TEXTS IN APPLIED MATHEMATICS

Finite-Volume Methods for Hyperbolic Problems

RANDALL J. LEVEQUE

The Mathematical Description: Conservation Laws

We will focus in systems governed by **Conservation Laws**

Conservation Laws come from basic Physical principles

Examples: Mass, Energy, Linear or Angular Momentum, Spin, People...

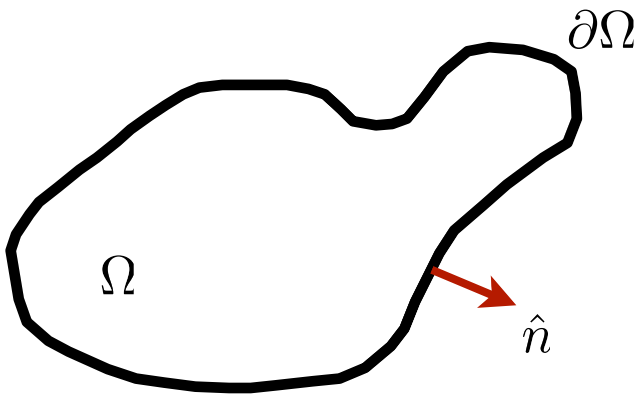
Based in:

$q^\beta(\bar{x},t)$	A number of certain evolving quantities
$\Omega \subset R^3$	A closed domain
$\partial\Omega$	and its boundary
$0 < t < T$	The time interval

The Mathematical Description: Conservation Laws

Let us suppose that the domain is not changing form with time.

Then



The Integral Form (IF)

$F_i^\alpha(q^\beta(\bar{x},t))$ The quantities flux

- ✓ Latin index: cartesian dimensions
- ✓ Greek index: variables that define the system
- ✓ Einstein convention on repeated indices

The Mathematical Description: Conservation Laws

$\forall \alpha$

This equation is **the fundamental mathematical interpretation of a conservation principle**

What do we mean by a **conservation principle** ?
Example: persons in this room, money in your pocket...

The principle is basic... what are the conserved quantities is not so basic. What are the fluxes is not so basic neither.
Examples: mass, mechanical energy (1st. law of thermodynamics), linear momentum (2nd. law of Newton)

Indeterminacies: fluxes definition, quantities meaning, boundary and initial values, material properties...

The conservation law gives the idea of how extensive quantities behave

Sources and drains can be included

The system can be divided in parts (volume/surface) and quantities transferred back and forth

The Mathematical Description: Conservation Laws

$\forall \alpha$

If the domain is changing in time, no problem: *Reynolds transport theorem*

Exercise

The domain could be extended to the infinite or reduced to infinitesimal

Applying the *Gauss theorem, and requiring some continuity properties on the fluxes*

Exercise

$\forall \alpha$

Therefore

$$\frac{\partial q^\alpha}{\partial t} + \frac{\partial F_i^\alpha(q)}{\partial x_i} = 0 \qquad \forall \alpha$$

The Mathematical Description: Differential equations

The target:

Systems governed by PDEs **coming from conservation principles**

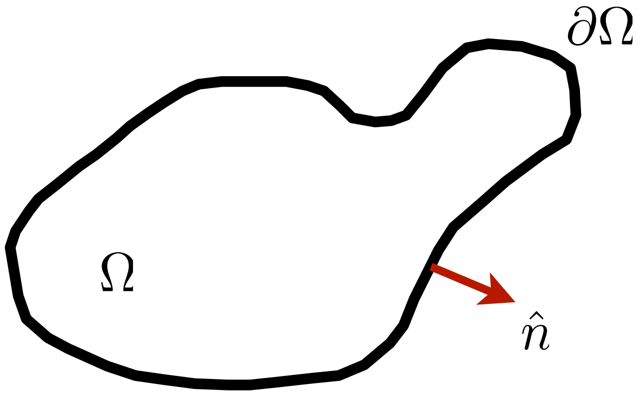
Forces, Energy, Mass... + Boundary and initial conditions

The Differential Conservative Form (DCF)

Fields of unknowns

Fluxes

$S(x_i, t)$ Sources



The Mathematical Description: Differential equations

The target:

Systems governed by PDEs coming from conservation principles

Forces, Energy, Mass... + Boundary and initial conditions

The Differential Conservative Form (DCF)

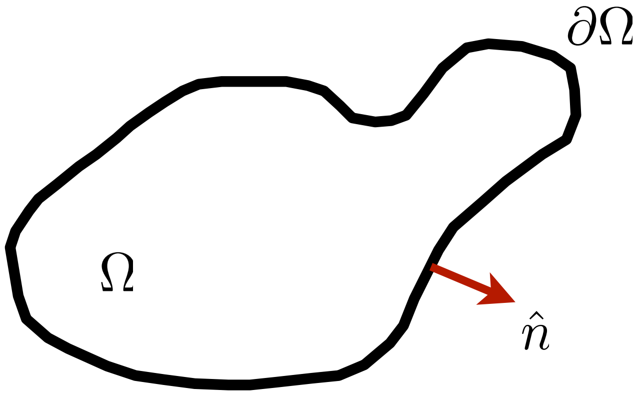
Complex Physics, even for the very simple systems

Complex geometries

Complex numerical issues

Fine resolution in time and space

Coupling



No analytical solutions!

The Mathematical Description: Differential equations

... but remember that:

The DCF is not as fundamental as the IF because:

will be scarcely used for getting the analytical solution

passing from IF to DCF involves some doubtful steps

However, we will see that the DCF (and other differential equations derived from IF) is very useful to design good numerical methods.

But always keep in mind that the IF lies in the core of every numerical method... let us see why...

Discretising the System

What is a simulation code?

"Divide and conquer"

Heavily reducing the dimensionality of the unknowns

Going from time-space continuum to a discrete subset

How to choose this **discrete subset** lies in the core of Computational Mechanics

Discretising the System

What is a simulation code?

"Divide and conquer"

Heavily **reducing the dimensionality** of the unknowns

Going from time-space continuum to a **discrete subset**

How to **choose** this discrete subset lies in the core of Computational Mechanics

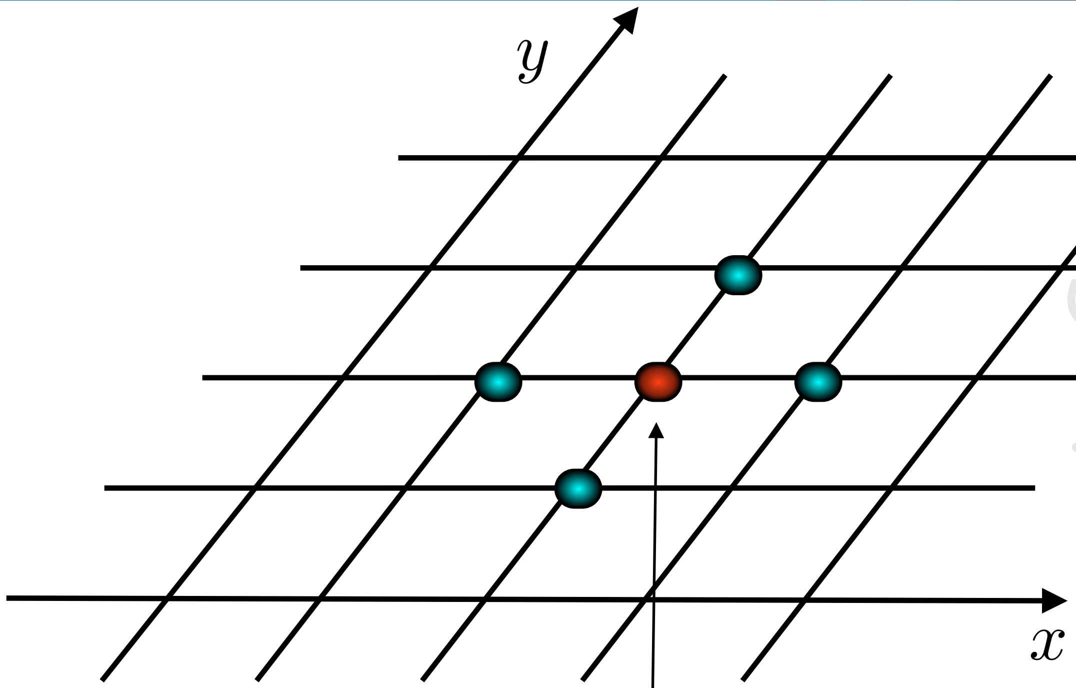
For instance:

- ☑ Replace ∂ by Δ in the original continuum equations
- ☑ Discretise the physical domain in small cells and solve individual conservation problems
- ☑ Discretise the solution space to find an approximate solution

Discretising the System

First strategy:

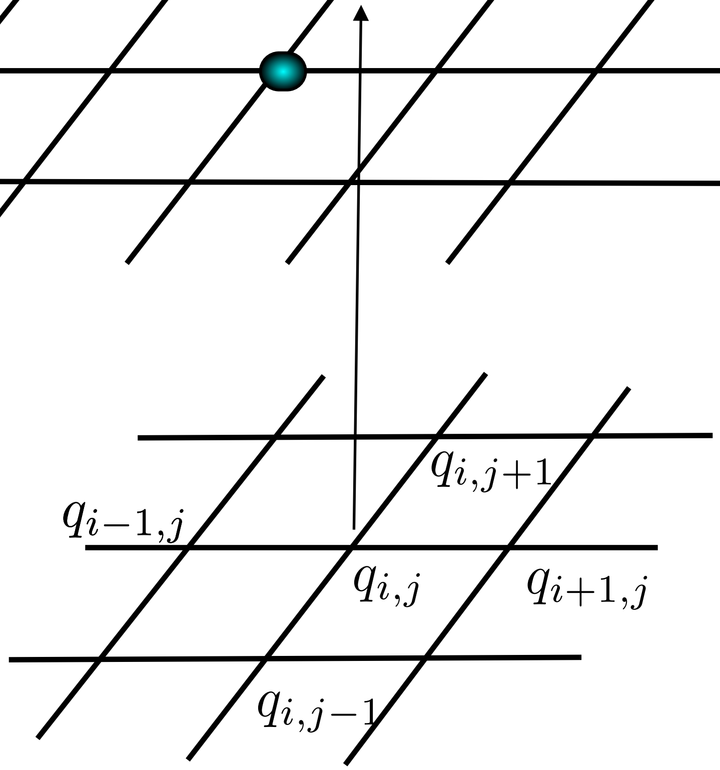
Finite Differences



Suppose convective and diffusive fluxes:

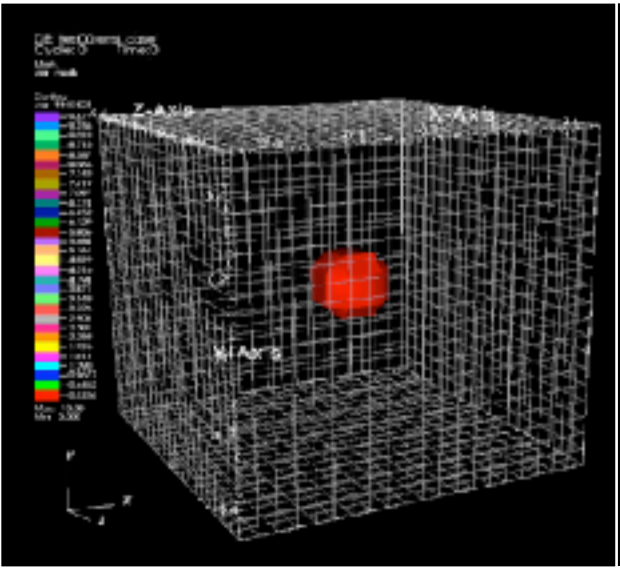
$$\frac{\Delta q_{i,j}}{\Delta x} = \frac{q_{i+1,j} - q_{i,j}}{\Delta x}$$

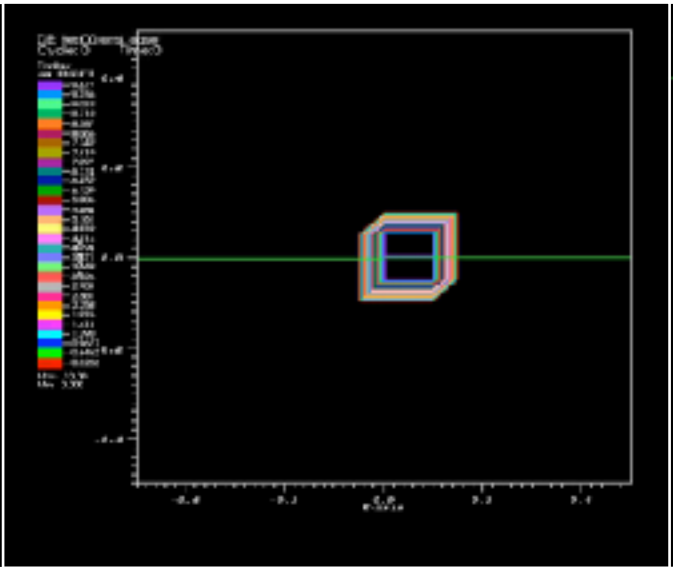
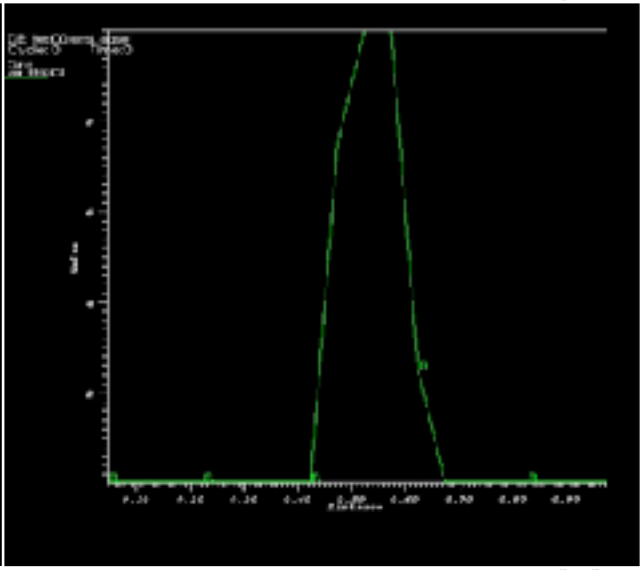
$$\frac{\Delta}{\Delta x}\left(\frac{\Delta q_{i,j}}{\Delta x}\right) = \frac{q_{i+1,j} - 2q_{i,j} + q_{i-1,j}}{\Delta x^2}$$



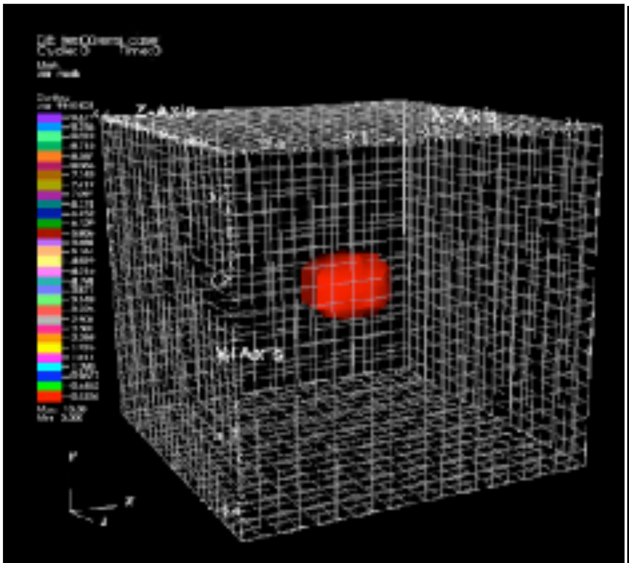
The End?

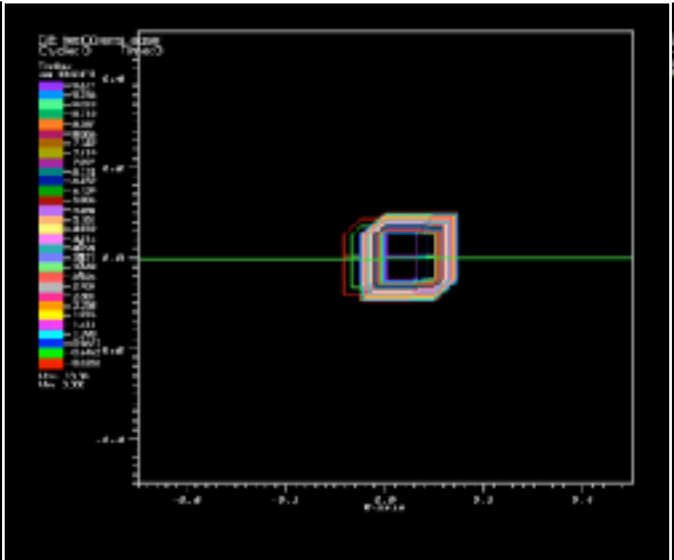
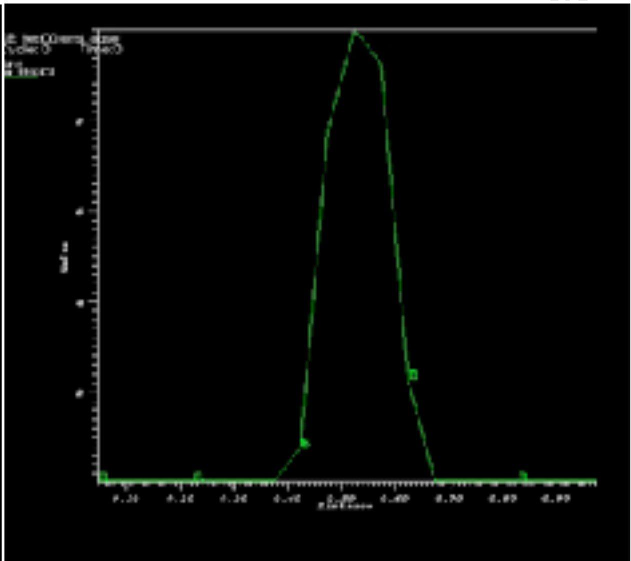
Case I



Case II: smaller dt



The Physical System
and its
Mathematical Description

-II-

Back to the The Physical System

The Integral Form (IF)

$$\frac{\partial q^{\alpha}}{\partial t} + \frac{\partial F_i^{\alpha}(q)}{\partial x_i} = 0$$

The Differential Conservative Form (DCF)

Back to the The Physical System

Always recall that we started at IF.

To go from IF to DCF some "more" must be asked for…

… meaning that not all solutions of IF accomplish DCF!

So let us try to always base our methods in IF

The Integral Form (IF)

The Differential Conservative Form (DCF)

Back to the The Physical System

$$\frac{\partial q^\alpha}{\partial t} + \frac{\partial F_i^\alpha(q)}{\partial x_i} = 0$$

The Differential Conservative Form (DCF)

This is the conservative form of the differential equation.

It requires that the **time derivative** of the variable and the **divergence of the flux** exist, we needed to derive it.

Using the **flux Jacobians**, the DCF becomes the DJF:

$$\frac{\partial q^\alpha}{\partial t} + A_i^{\alpha\beta}\frac{\partial q^\beta}{\partial x_i} = 0$$

The Differential Jacobian Form (DJF)

Where the Jacobians are defined as follows

$$A_i^{\alpha\beta} = \frac{\partial F_i^\alpha}{\partial q^\beta}$$

Back to the The Physical System

Fluid flows: The Navier-Stokes equations

$$\frac{\partial U_j}{\partial t} + \frac{\partial}{\partial x_i}(u_i U_j) + \frac{\partial}{\partial x_i}(\delta_{ij} p - \tau_{ij}) + \rho g_j = 0$$

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_i}(U_i) = 0$$

$$\frac{\partial E}{\partial t} + \frac{\partial}{\partial x_i}(u_i E) + \frac{\partial}{\partial x_i}\left(u_i p - k\frac{\partial T}{\partial x_i} - \tau_{ij} u_j\right) + \rho(u_i g_i + r) = 0$$

$U_i = \rho u_i, E = \rho(C_v T + u^2/2)$ are the momentum and the total energy

τ_{ij} is the viscous stress tensor

Back to the The Physical System

Fluid flows: The Navier-Stokes equations

Momentum: forces balance

Density: continuity equation

Energy: energy balance

$$\frac{\partial U^\alpha}{\partial t} = \frac{\partial F_i^\alpha}{\partial x_i} + S$$

$$U^\alpha = (U_j, \rho, E)$$

The Mathematical Description

Let us see an example on a complete form of a system.

$$\frac{\partial q^\alpha}{\partial t} + \frac{\partial F_i^\alpha(q)}{\partial x_i} = 0 \qquad \frac{\partial q^\alpha}{\partial t} + A_i^{\alpha\beta} \frac{\partial q^\beta}{\partial x_i} = 0$$

can be re-written as

$$\frac{\partial q^\alpha}{\partial t} + \frac{\partial F_x^\alpha}{\partial x} + \frac{\partial F_y^\alpha}{\partial y} = 0$$

and also as

$$\frac{\partial q^\alpha}{\partial t} + A_x^{\alpha\beta} \frac{\partial q^\beta}{\partial x} + A_y^{\alpha\beta} \frac{\partial q^\beta}{\partial y} = 0$$

The Mathematical Description

$$\frac{\partial q^\alpha}{\partial t} + \frac{\partial F_x^\alpha}{\partial x} + \frac{\partial F_y^\alpha}{\partial y} = 0 \qquad \frac{\partial q^\alpha}{\partial t} + A_x^{\alpha\beta}\frac{\partial q^\beta}{\partial x} + A_y^{\alpha\beta}\frac{\partial q^\beta}{\partial y} = 0$$

For instance, for a 2D problem with 2 variables $q^\alpha = \begin{pmatrix} q_1 \\ q_2 \end{pmatrix}$

$$F_x^\alpha = \begin{pmatrix} F_x^1(q_1, q_2) \\ F_x^2(q_1, q_2) \end{pmatrix} \qquad A_i^{\alpha\beta} = \begin{pmatrix} \frac{\partial F_x^1}{\partial q_1} & \frac{\partial F_x^1}{\partial q_2} \\ \frac{\partial F_x^2}{\partial q_1} & \frac{\partial F_x^2}{\partial q_2} \end{pmatrix}$$

Note that:

There is one ***A*** for each space dimension (here only shown for "x")

The dimensions of ***A*** depends on how many coupled variables you have

The Mathematical Description

Remarks:

✓ Tensors "***A***" (one for each space dimension) couples in a very explicit way all the system equations.

$$\frac{\partial}{\partial t}\begin{pmatrix} q_1 \\ q_2 \end{pmatrix} + \begin{pmatrix} \frac{\partial F_x^1}{\partial q_1} & \frac{\partial F_x^1}{\partial q_2} \\ \frac{\partial F_x^2}{\partial q_1} & \frac{\partial F_x^2}{\partial q_2} \end{pmatrix}\begin{pmatrix} \frac{\partial q_1}{\partial x} \\ \frac{\partial q_2}{\partial x} \end{pmatrix} + \begin{pmatrix} \frac{\partial F_y^1}{\partial q_1} & \frac{\partial F_y^1}{\partial q_2} \\ \frac{\partial F_y^2}{\partial q_1} & \frac{\partial F_y^2}{\partial q_2} \end{pmatrix}\begin{pmatrix} \frac{\partial q_1}{\partial y} \\ \frac{\partial q_2}{\partial y} \end{pmatrix} = 0$$

The Mathematical Description

Remarks:

☑ DJF comes from DCF so it will be different for each **set** of variables

For instance, in the Navier - Stokes equations, the heat transport equation can be used instead of the total energy one:

$$\frac{\partial T}{\partial t} + u_i \frac{\partial T}{\partial x_i} + \frac{1}{C_v \rho}\left(p\frac{\partial u_k}{\partial x_k} - \tau_{ij}\frac{\partial u_i}{\partial x_j} - k\frac{\partial^2 T}{\partial x_i \partial x_i}\right) = 0$$

$$\frac{\partial E}{\partial t} + \frac{\partial}{\partial x_i}\left(u_i(E+p) - u_j\tau_{ij} - k\frac{\partial T}{\partial x_i}\right) = 0$$

$$E = \rho(C_v T + u^2/2)$$

The Mathematical Description

Remarks:

☑ DJF comes from DCF so it will be different for each **set** of variables

... which means that the Jacobian form can be defined as coming from either a conservative or an non-conservative set of equations.

Forms can be transformed in this way

$$\frac{\partial q^\alpha}{\partial t} + A_x^{\alpha\beta}\frac{\partial q^\beta}{\partial x} + A_y^{\alpha\beta}\frac{\partial q^\beta}{\partial y} = 0$$

$$\frac{\partial q}{\partial x_i} \longrightarrow \frac{\partial q}{\partial q'}\frac{\partial q'}{\partial x_i}$$

$$A_i^{\alpha\beta} = \begin{pmatrix} \frac{\partial F_x^1}{\partial q_1} & \frac{\partial F_x^1}{\partial q_2} \\ \frac{\partial F_x^2}{\partial q_1} & \frac{\partial F_x^2}{\partial q_2} \end{pmatrix}$$

The Mathematical Description

Remarks:

☑ The Jacobian tensor "A" couples in a very explicit way the system equations.

Now... what if "A" could be diagonalisable?

The Mathematical Description

Remarks:

☑ The Jacobian tensor "A" couples in a very explicit way the system equations.

Now... what if "A" could be diagonalisable?

The system is called hyperbolic if it is **diagonalisable** with **real** eigenvalues.

$$\frac{\partial q^\alpha}{\partial t} + A^{\alpha\beta} \frac{\partial q^\beta}{\partial x_i} = 0$$

$$\frac{\partial q^\alpha}{\partial t} + R^{\alpha\gamma} \Lambda^{\gamma\theta} R^{\theta\beta}_{inv} \frac{\partial q^\beta}{\partial x_i} = 0$$

$\Lambda^{\gamma\theta} = \lambda_{1,2,\ldots} \delta\gamma\theta$ eigenvalues matrix

$R^{\alpha\gamma}$ right eigenverctors matrix

The Mathematical Description

Remarks:

✓ The Jacobian tensor "A" couples in a very explicit way the system equations.

$$\frac{\partial q^\alpha}{\partial t} + A^{\alpha\beta}\frac{\partial q^\beta}{\partial x_i} = 0$$

$$\frac{\partial q^\alpha}{\partial t} + R^{\alpha\gamma}\Lambda^{\gamma\theta}R^{\theta\beta}_{inv}\frac{\partial q^\beta}{\partial x_i} = 0$$

$$A = R\Lambda R^{-1} \qquad R^{-1}AR = \Lambda$$

$$R^{-1}q_t + R^{-1}\ A\ RR^{-1}q_x = 0$$

$$w_t + \Lambda w_x = 0$$

The Mathematical Description

Remarks:

✓ The Jacobian tensor "A" couples in a very explicit way the system equations.

The system is called hyperbolic if it is **diagonalisable** with **real** eigenvalues.

If "A" is symmetric, the system is always (symmetric) hyperbolic

If all eigenvalues are different, it is strictly hyperbolic
The solution will become a superposition of linear waves, each one with its own characteristic speed, **the corresponding eigenvalue:**

$$w_t^\alpha + \Lambda w_x^\alpha = 0$$

But maybe we have solved the problem… have we?

R^{-1} is the change of variables matrix.

The Mathematical Description

Remarks:

☑ The Jacobian tensor "A" couples in a very explicit way the system equations.

Now... what if "A" could be diagonalisable?

The system could be decoupled… but recall that,

$$\frac{\partial q^\alpha}{\partial t} + \underset{\uparrow}{A_i^{\alpha\beta}} \frac{\partial q}{\underset{\uparrow}{\partial x_i}}^{\beta} = 0$$

Diagonalization should be done simultaneously for all dimensions…!

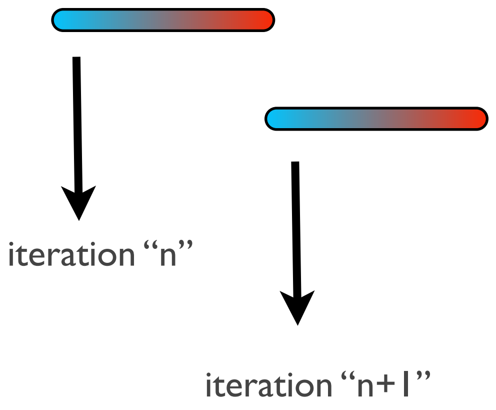
This is capital issue for building good numerical methods.

The Mathematical Description

Remarks:

☑ If the problem is non-linear, "A" could be a very intuitive "linearisator" for an iterative scheme:

$$A_i^{\alpha\beta}(q^n) \quad \frac{\partial q^{n+1}}{\partial x_i}$$



Quasilineal form

The Mathematical Description

Remarks:

☑ This is the convective derivative of the quantity "q":

$$A_i^{\alpha\beta}(q^n) \; \frac{\partial q^{n+1}}{\partial x_i}$$

☑ It is, indeed. But the Jacobian is the Jacobian of the complete flux.

This means that it could include diffusion terms too.
It is as a "generalised velocity"
It can include a hyperbolic part and a non-hyperbolic part

☑ Very important: always remember that DJF is derived from IF providing that some restrictions on the continuity of the variables...

The Mathematical Description

Let us study some examples of increasing complexity...

First the simplest one: **1D advection equation**

The Mathematical Description

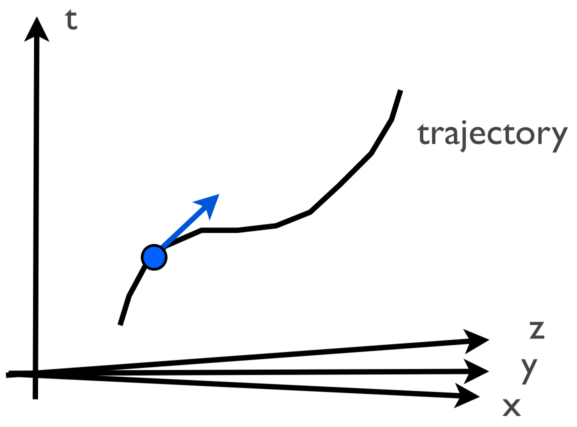
Advection equation

It represents the convective transport of a quantity

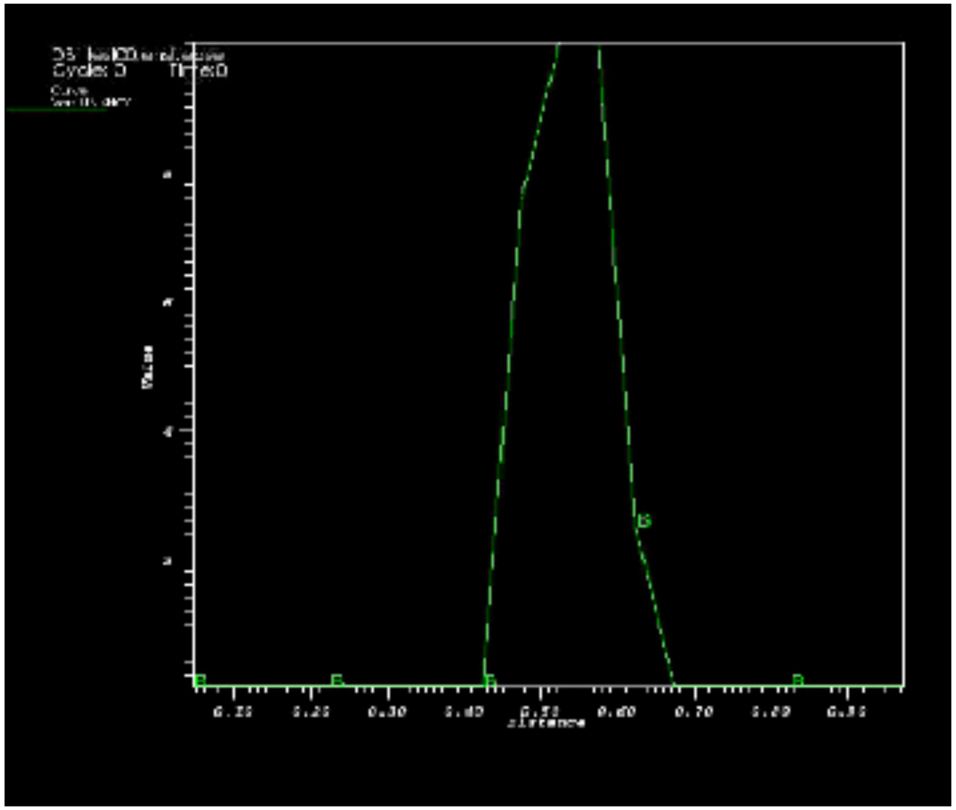
$$\underbrace{\frac{\partial q}{\partial t}}_{\text{variation if don't move}} + \underbrace{u_i \frac{\partial q}{\partial x_i}}_{\text{variation if moving}} = 0 \longrightarrow \underbrace{\frac{Dq}{Dt} = 0}_{\textbf{Material Derivative}}$$

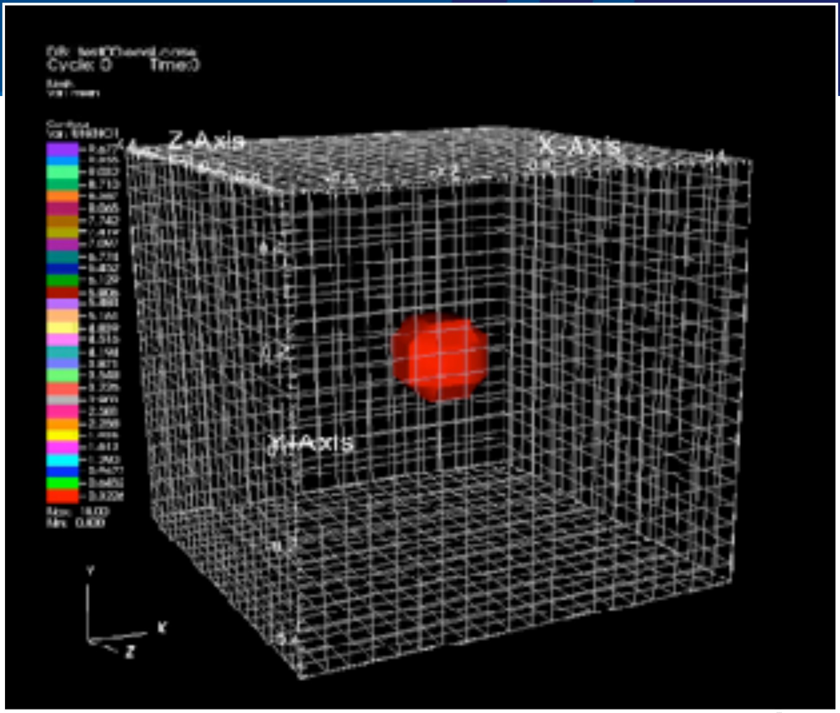
Giving an initial value for "q", it remains constant if we move following the trajectory

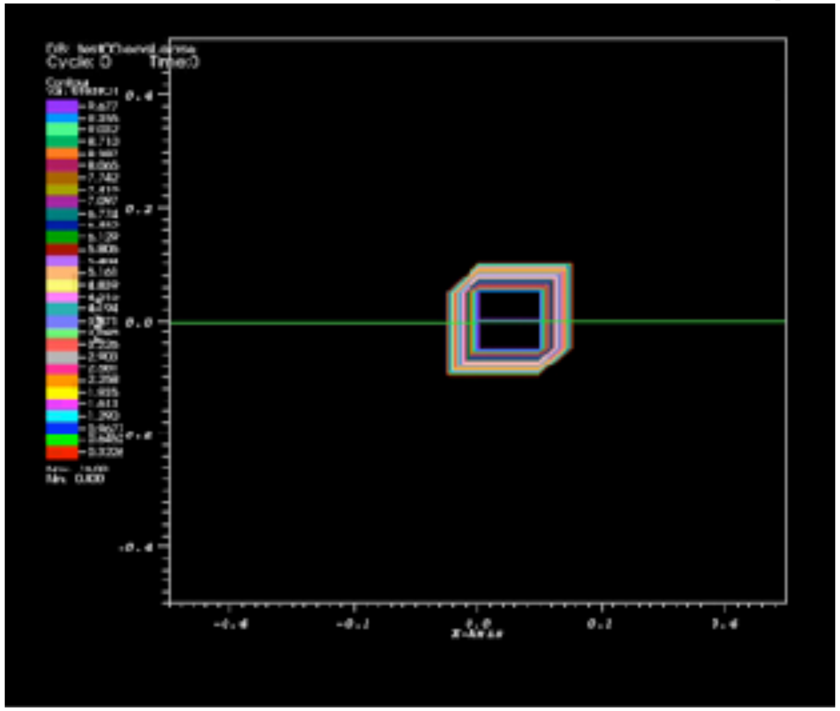
$$\frac{Dq}{Dt}(X(t), t) = 0$$



The Mathematical Description







$$\frac{Dq}{Dt}(X(t), t) = 0$$

The Mathematical Description

Advection equation, 1D

If you can "reconstruct" the trajectories you solve the problem!

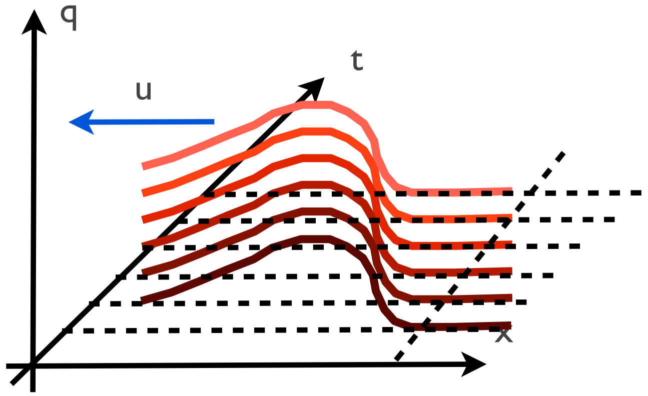
The equation complexity is transferred to the trajectory reconstruction.
Think that the "blob" is a rigid body…!

If you know the speed, you know where will the blob be at a certain time
Let us consider the 1D problem (partial derivatives replaced by "d"):

$$\frac{dq}{dt} + u\frac{dq}{dx} = 0$$

Now, suppose the **velocity** is defined as

$$-u = \frac{dx}{dt}$$



The Mathematical Description

If "u" is constant, the trajectory can be trivially integrated

$$dx = -u\, dt$$

$$x - x_0 = -u\, (t - t_0)$$

$$x = \underline{(x_0 + u\, t_0)} - u\, t = \underline{C} - u\, t$$

So the trajectory equation is

$$x + u\, t = C$$

$$q := q(x,t) = q(C - u\, t, t)$$

The Mathematical Description

Which means that, if

$$q := q(x,t) = q(C - u\,t, t)$$

then, the equation is verified:

$$\frac{dq}{dt} = -u\frac{dq}{dx}$$

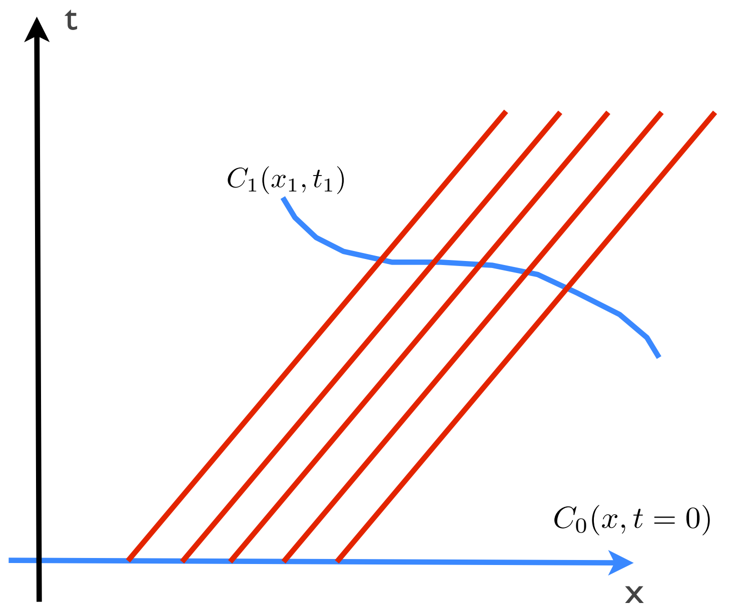
By defining initial data on one of these curves

$C_0(x, t=0)$

$C_1(x_1, t_1)$

data will move unchanged on each of the trajectories

$x + at = x_0$



The Mathematical Description

Suppose you chose

$$C_1(x_1, t_1)$$

then, if you travel horizontally fixing a time "t", then data (i.e. "q") will change.

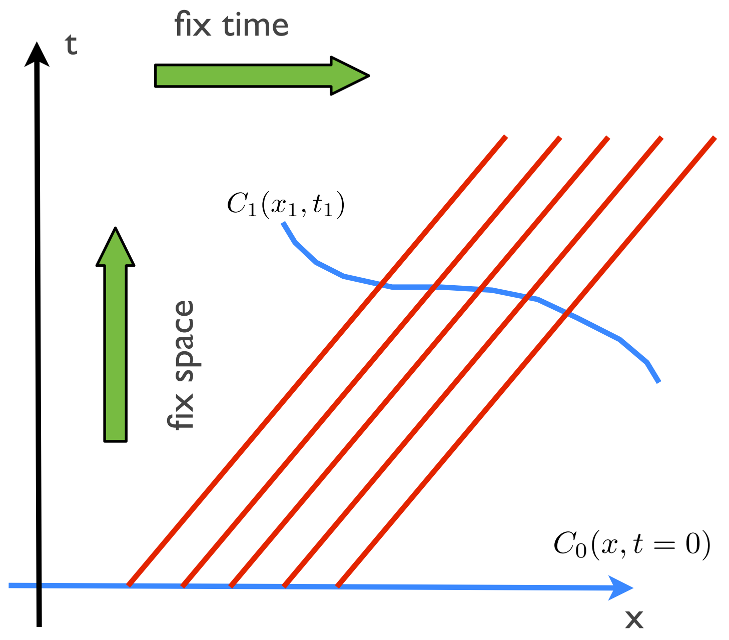
If you do it vertically by fixing a position "x", data will also change.

But if you do it "in a synchronized way" q does not change.

The x-t description is the **Eulerian** description.

On the other hand, if you travel with the particle, data will not change and the description is **Lagrangian**

These trajectories in the space-time are the **Characteristics**



The Mathematical Description

Characteristic curves can be written in parametric form:

$$\frac{dt}{ds} = 1 \quad , \quad \frac{dx}{ds} = u$$

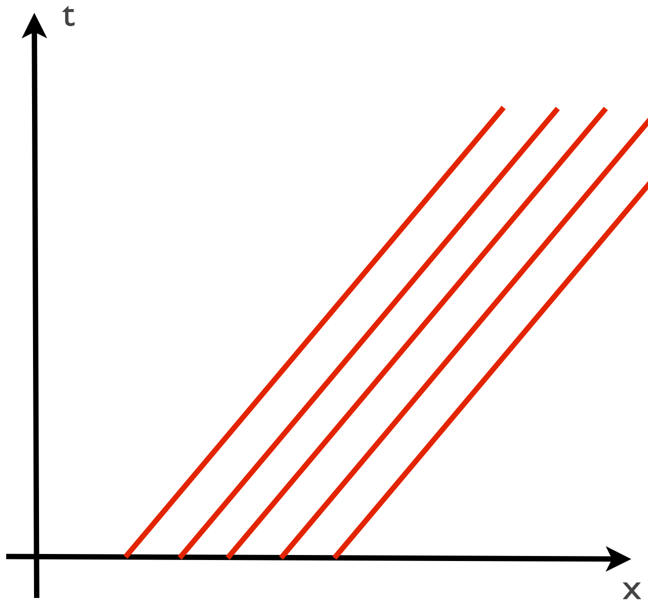
This transformation is just a scaling, where "u" and "1" are the scale factors.

This scaling gives the precise gauge by enlarging or shortening one or the other axe to adjust the curve as a characteristic.

In this way, we travel with the particle, where

$$\frac{Dq}{Ds} = 0$$

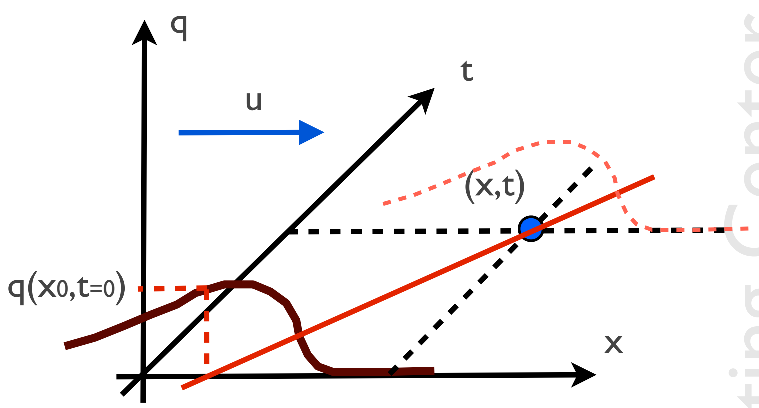
$$\frac{Dq}{Ds} = \frac{dt}{ds}\frac{dq}{dt} + \frac{dx}{ds}\frac{dq}{dx}$$



The Mathematical Description

Then, the problem could be solved for every (x,t)

Initial distribution is propagated forwards without change

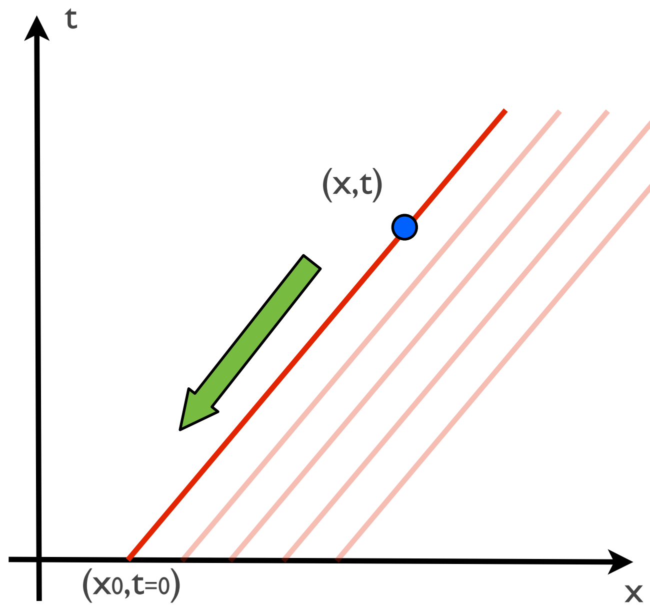


The tracks are the characteristic curves

To know what is “q” at a certain pair (x,t), i.e. **to solve the problem**:

☑ you **identify** the characteristic through the pair (x,t)

☑ you travel **backwards** to see the initial value of “q”



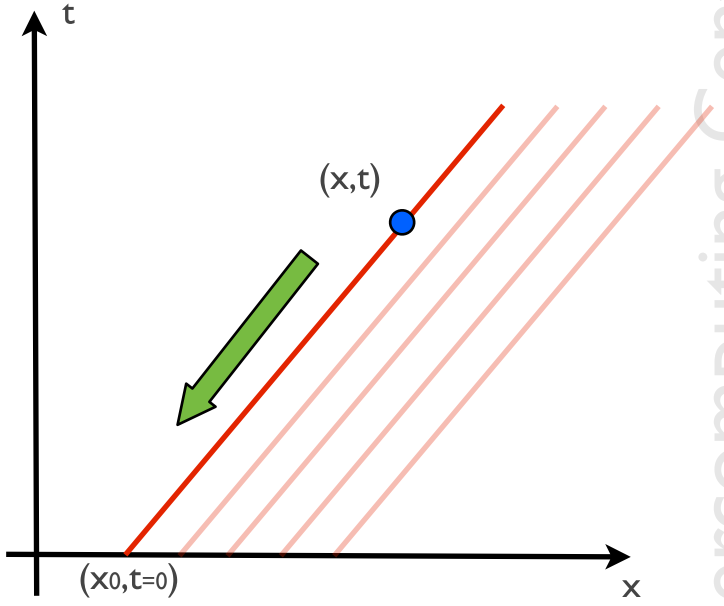
The Mathematical Description

This is the rate of change following the characteristics, which should be zero.

$$\frac{Dq}{Ds} = 0$$

We can set up now a definition of a characteristic:

The original differential equation determines the value of "q" over a characteristic **C** **only** from values on the same characteristic **C**.

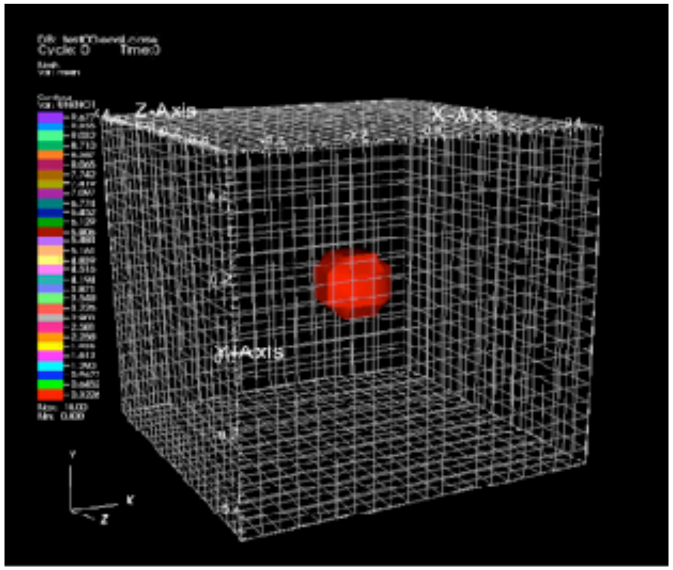


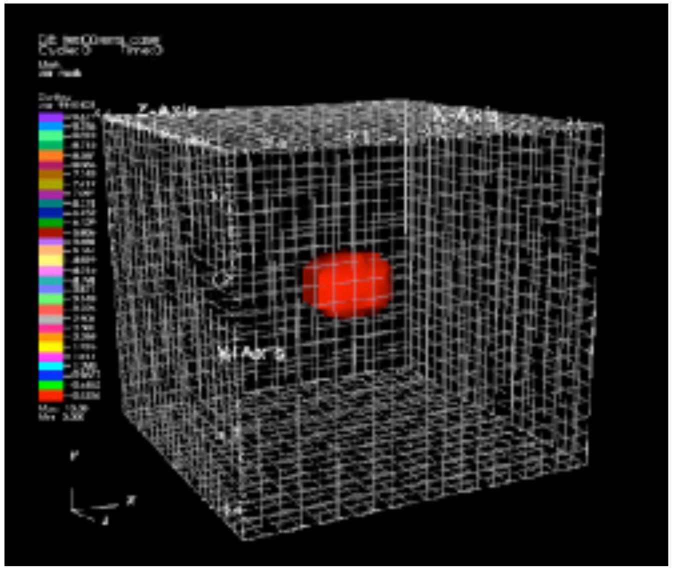
It means that the value of q for a point (x,t) must come determined **only** by tracking back the characteristic to (x_o, t_o).

It **cannot** be determined by all neighboring points, but **only** by those lying on the characteristic.

This particular set of points where to look for the value of "q" is **fixed by the equation.**

The Mathematical Description





But what is happening with the blob?

Why can a change in the time increment arise catastrophic consequences?

If we know the speed, can't we just move properly the blob?

What is wrong here?

The Mathematical Description

Let us first study some examples of increasing complexity...

Now, suppose the **speed is not constant**

The Mathematical Description

If the velocity depends on time and space

$$u := u(x,t) \qquad \frac{dq}{dt} = -u\frac{dq}{dx}$$

Trajectory integration is not that straightforward, so the solution is not trivially found, because it must be discretized somewhat...
However, the same analysis still holds.

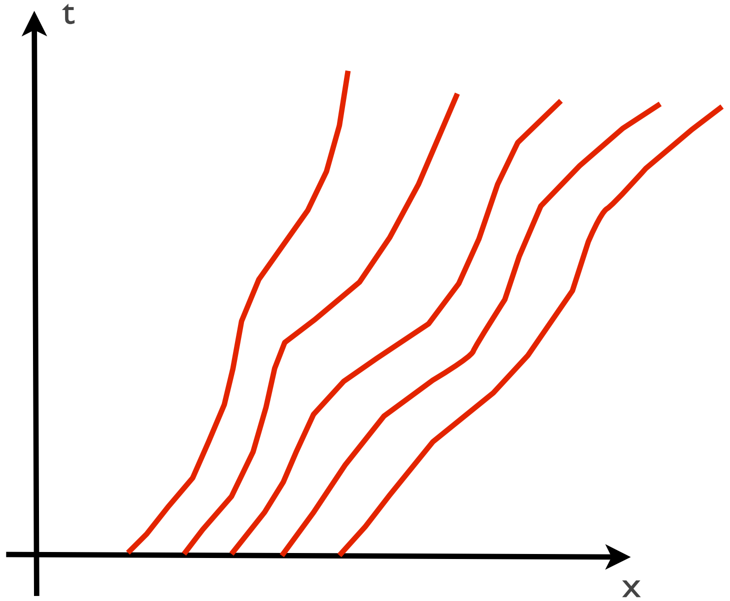
Then

$$\frac{dt}{ds} = 1 \quad , \quad \frac{dx}{ds} = u(x,t)$$

Which allows us to write

$$\frac{Dq}{Ds} = \frac{dt}{ds}\frac{dq}{dt} + \frac{dx}{ds}\frac{dq}{dx}$$

$$\frac{Dq}{Ds} = \frac{dq}{dt} + u(x,t)\frac{dq}{dx} = 0$$

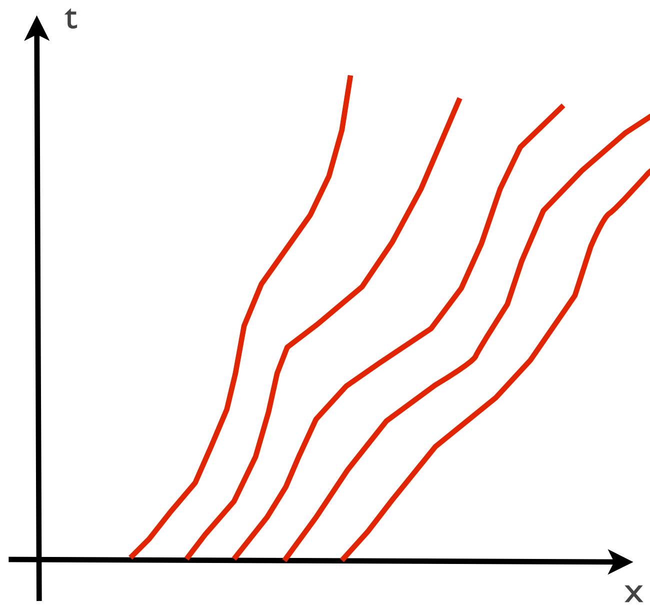


The Mathematical Description

If the velocity depends on time and space

$$u := u(x,t)$$

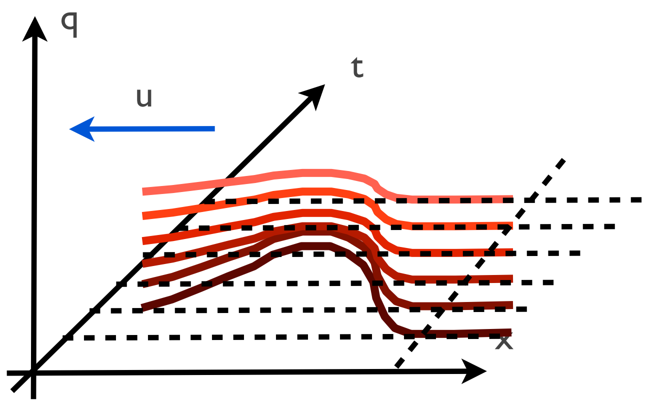
Which means that we can do the same procedure, but the trajectories are not constant in time.



The Mathematical Description

If the problem has a source term,

$$\frac{Dq}{Dt} = \frac{dq}{dt} + u(x,t)\frac{dq}{dx} = b(x,t)$$



The characteristics are the same as before.

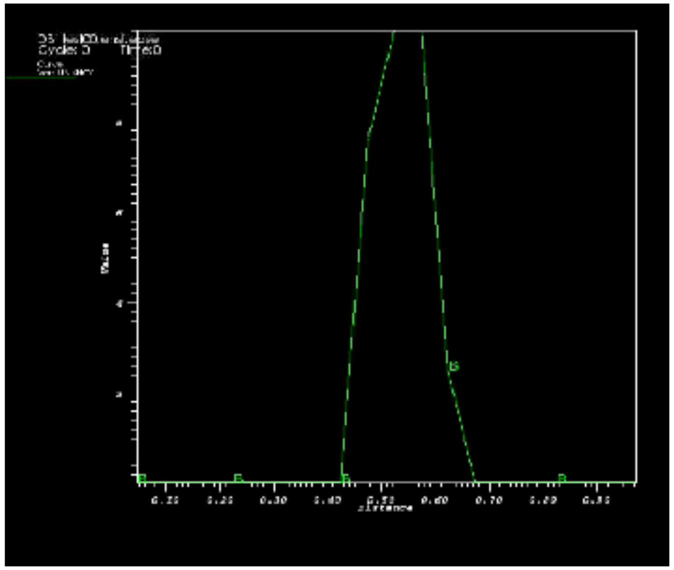
However, "q" is different:

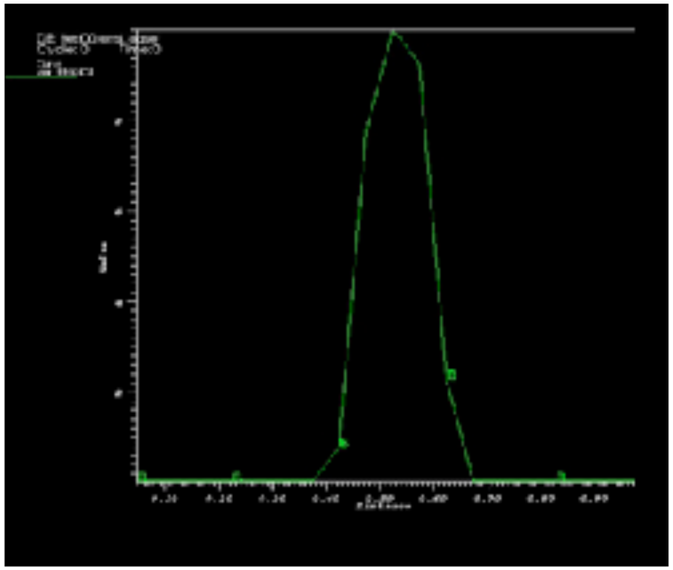
$$q(x(s),t(s)) = q(s) = q(s_0) + \int_{s_0}^{s} b(x(m),t(m))dm$$

This means that "q" is progressively deformed (like being "eroded") due to the amount of "b" which accumulates traveling along the characteristic.

The kind of "erosion" depends on how the source ***b*** is defined.

The Mathematical Description





This could give us a hint on what's happening...

The Mathematical Description

Now, the problem is **non-linear**

The Mathematical Description

If the problem is non-linear, suppose that

$$\frac{Dq}{Dt} = \frac{dq}{dt} + q\frac{dq}{dx} = 0$$

q takes now the place of ***u***

Taking the derivatives,

$$\frac{Dq}{Ds} = \frac{dt}{ds}\frac{dq}{dt} + \frac{dx}{ds}\frac{dq}{dx}$$

$$\frac{Dq}{Ds} = \underbrace{\frac{dt}{ds}}_{1}\frac{dq}{dt} + \underbrace{\frac{dx}{ds}}_{q}\frac{dq}{dx}$$

The characteristics are always,

$$\frac{dt}{ds} = 1 \quad , \quad \frac{dx}{ds} = q$$

The Mathematical Description

The value of “q” is constant over them, but the characteristics can cross.

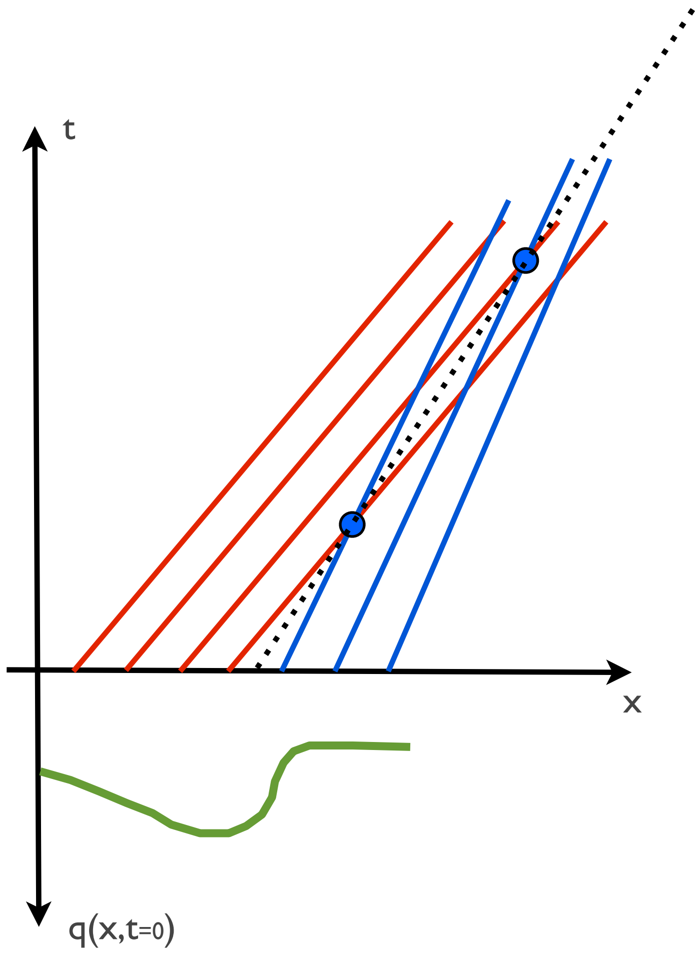
The meaning of characteristics crossing is subtle.

Points over the dotted line have associated more than one characteristic... although there is only one unknown.

This means that its value is different when coming from one side or from the other: it is **discontinuous**.

Even if you start with a smooth distribution, it can became non-smooth.

Example: the Burgers equation and the breaking waves... we will come back to this



The Mathematical Description

Now, consider a **1D system**

The Mathematical Description

Let us consider the general problem, defined with a system

$$\frac{dq^\alpha}{dt} + A^{\alpha\beta}\frac{dq^\beta}{dx} = 0$$

Now let us take derivatives to define the characteristics

$$\frac{dq^\alpha}{ds} = \frac{dt}{ds}\frac{dq^\alpha}{dt} + \frac{dx}{ds}\frac{dq^\alpha}{dx}$$

and recall that along them, this derivative must remain zero. Now

$$\frac{dq^\alpha}{ds} = \frac{dt}{ds}\underbrace{\left(-A^{\alpha\beta}\frac{dq^\beta}{dx}\right)}_{\text{from the system equation}} + \frac{dx}{ds}\frac{dq^\alpha}{dx}$$

The Mathematical Description

This expression can be rewritten using Kroenecker's tensor as

$$\frac{dq^\alpha}{ds} = \frac{dt}{ds}(-A^{\alpha\beta}\frac{dq^\beta}{dx}) + \frac{dx}{ds}\frac{dq^\beta}{dx}\delta^{\alpha\beta}$$

where

$$\delta^{\alpha\beta} = \begin{cases} 1 & \alpha = \beta \\ 0 & \text{otherwise} \end{cases}$$

Then, we establish a proportionality relation between the two derivatives:

$$\frac{dq^\alpha}{ds} = \frac{dq^\beta}{dx}(-A^{\alpha\beta}\frac{dt}{ds} + \delta^{\alpha\beta}\frac{dx}{ds})$$

The Mathematical Description

$$\frac{dq^\alpha}{ds} = \frac{dq^\beta}{dx}(-A^{\alpha\beta}\frac{dt}{ds} + \delta^{\alpha\beta}\frac{dx}{ds})$$

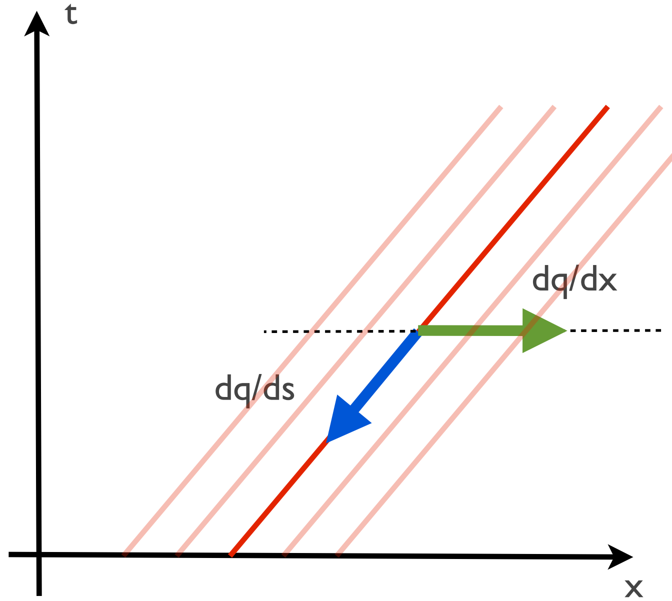
This is the rate of change following the characteristics, which should be **zero**.

Recall the definition of characteristics:

The original differential equation determines the value of "q" over a characteristic **C** **only** from values on the same characteristic **C**.

The Mathematical Description

$$\frac{dq^\alpha}{ds} = \frac{dq^\beta}{dx}\left(-A^{\alpha\beta}\frac{dt}{ds} + \delta^{\alpha\beta}\frac{dx}{ds}\right)$$



It means that the value of q for a point (x,t) must come determined **only** by tracking back the characteristic to (x_o, t_o).

It **cannot** be determined by all neighbouring points, but **only** by those lying on the characteristic.

This particular set of points where to look for the value of "q" is **fixed** by the equation.

The Mathematical Description

$$\frac{dq^\alpha}{ds} = \frac{dq^\beta}{dx}\left(-A^{\alpha\beta}\frac{dt}{ds} + \delta^{\alpha\beta}\frac{dx}{ds}\right)$$

In other words, rewriting this expression in compact form

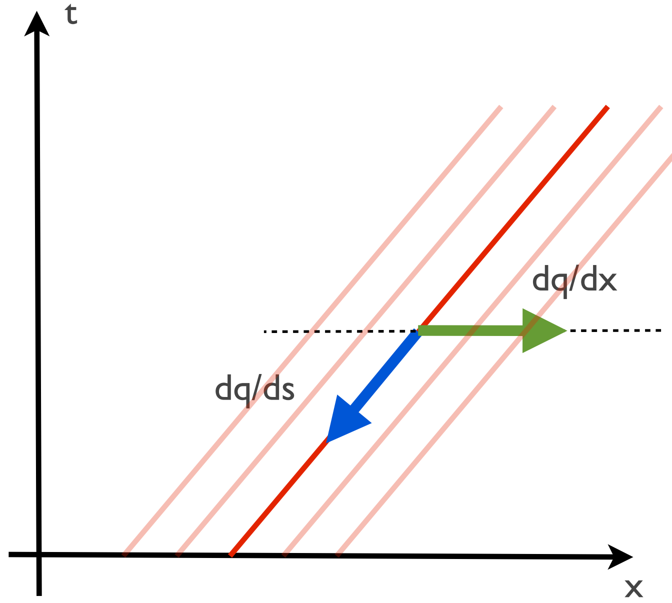
$$\frac{dq^\alpha}{ds} = K^{\alpha\beta}\frac{dq^\beta}{dx}$$

But a variation of "q" w.r.t. "x" **should not be** expressed in terms of the variation of "q" w.r.t. "s".

For that reason,

$$K^{\alpha\beta}$$

should be **non-invertible**, i.e. **singular**.



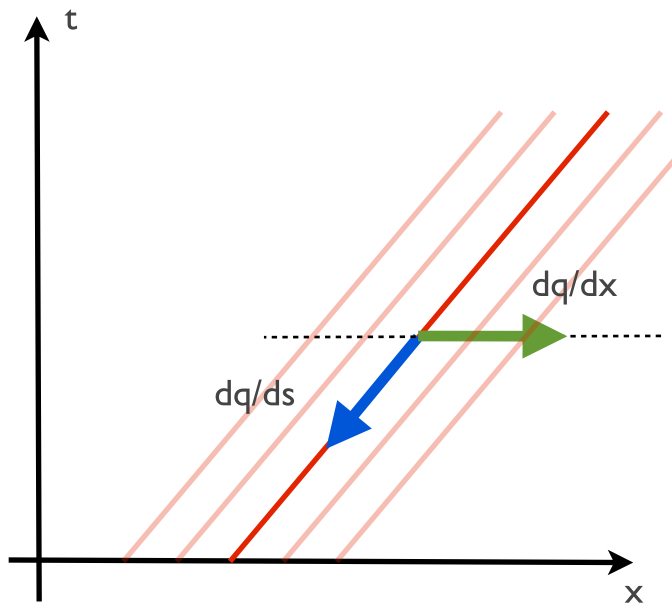
The Mathematical Description

$$K^{\alpha\beta} = -A^{\alpha\beta}\frac{dt}{ds} + \delta^{\alpha\beta}\frac{dx}{ds}$$

The matrix is singular if

$$\frac{dt}{ds} = 1 \quad , \quad \frac{dx}{ds} = \underbrace{\lambda(x,t,q^{\alpha})}_{\downarrow}$$

Eigenvalues matrix of $A^{\alpha\beta}$



Compare with the requirements for the **scalar problem:**

$$\frac{dt}{ds} = 1 \quad , \quad \frac{dx}{ds} = u$$

The Mathematical Description

And now, again: the system is then called **Hyperbolic**.

Each of "A" eigenvalues has an associated **eigenvector**.
This eigenvector sets the **direction in the space-time of each of the characteristics.**

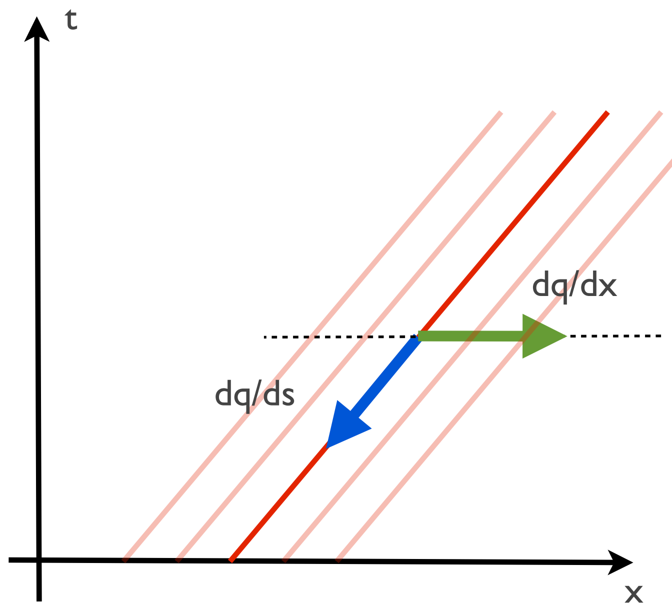
$$\alpha = (\quad 1 , \ldots , \; m \quad)$$

So for each equation in the system, there is one eigenvector, i.e., one characteristic.

In particular, if "A" is diagonal, the characteristics are co-linear with the canonical base of the space-time

$$r^\alpha = e^\alpha$$

Then, the equations are uncoupled, each one with its own canonical characteristic.



The Mathematical Description

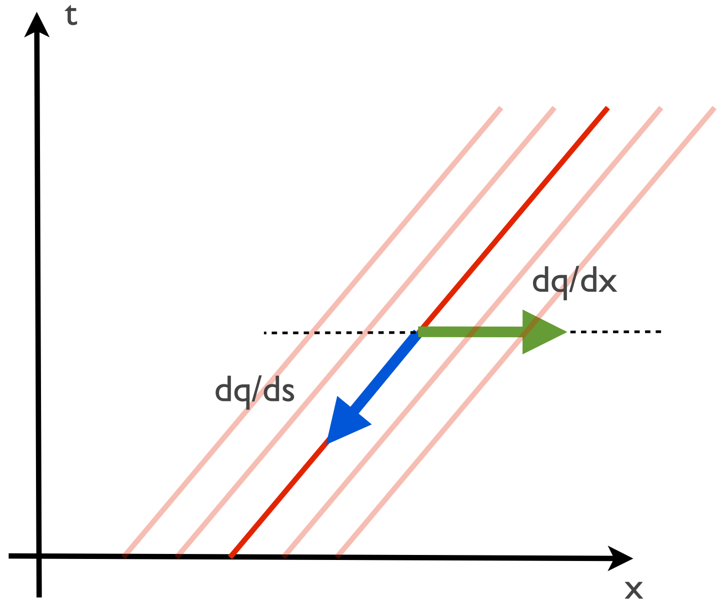
The system is then called **Hyperbolic**.

Remarks:

If "A" is **linear**, if it is hyperbolic, it will remain **hyperbolic for ever**.

If "A" **depends on (x,t)**, it could **lose** its hyperbolic character.

Even worse, if "A" is **non-linear**...

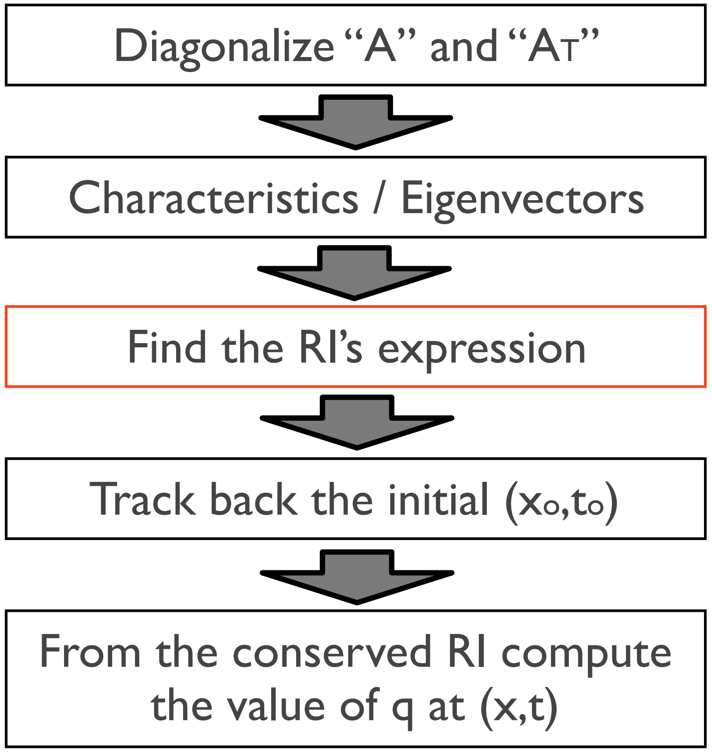


The Mathematical Description

The procedure is to compute the characteristics at each (x,t) by diagonalizing "A".

But once diagonalized, it is very likely that "q" will not be the quantity conserved when moved along the characteristic track.

We can then try to find functions of "q" that are conserved along the characteristics: the **Riemann Invariants**.


Diagonalize "A" and "A$_T$"
Characteristics / Eigenvectors
Find the RI's expression
Track back the initial (x_o,t_o)
From the conserved RI compute the value of q at (x,t)

$$\frac{dq^\alpha}{dt} + A^{\alpha\beta}\frac{dq^\beta}{dx} = 0$$

Warning: sometimes the RI's do not exist...

The Mathematical Description

Let us analyse a case:

One dimensional gas dynamics

$$\frac{\partial}{\partial t}\begin{pmatrix}\rho\\ u\end{pmatrix} + A\,\frac{\partial}{\partial x}\begin{pmatrix}\rho\\ u\end{pmatrix} = 0$$

$$A = \begin{pmatrix} u & \rho \\ c^2/\rho & u \end{pmatrix} \quad , \quad \lambda^{\pm} = u \pm c$$

This equation governs the **propagation of sound** in compressible flow.

Sound is a low intensity pressure wave that propagates at **constant speed c**.

Remark I: This equation can be re-written as a wave equation for the pressure.

Remark II: A similar equation can be obtained for sound propagation in solids.

The Mathematical Description

Let us analyse a case:

One dimensional gas dynamics

$$\frac{\partial}{\partial t}\begin{pmatrix}\rho \\ u\end{pmatrix} + A\,\frac{\partial}{\partial x}\begin{pmatrix}\rho \\ u\end{pmatrix} = 0$$

$$A = \begin{pmatrix} u & \rho \\ c^2/\rho & u \end{pmatrix} \quad , \quad \lambda^{\pm} = u \pm c$$

Then, the two characteristics are the curves

$$\eta^{\pm} = \frac{dx}{dt} = u \pm c$$

The Mathematical Description

This means that, for the **one dimensional** problem, at each point of the x-coordinate (i.e. the space coordinate) there are **two characteristics emerging at two different speeds**.

$$\eta^{\pm} = \frac{dx}{dt} = u \pm c$$

The problem can be seen as evolving surface water.

Sound propagation is the uniform and constant motion at speed ***c*** of surface gravity waves produced by some perturbation.

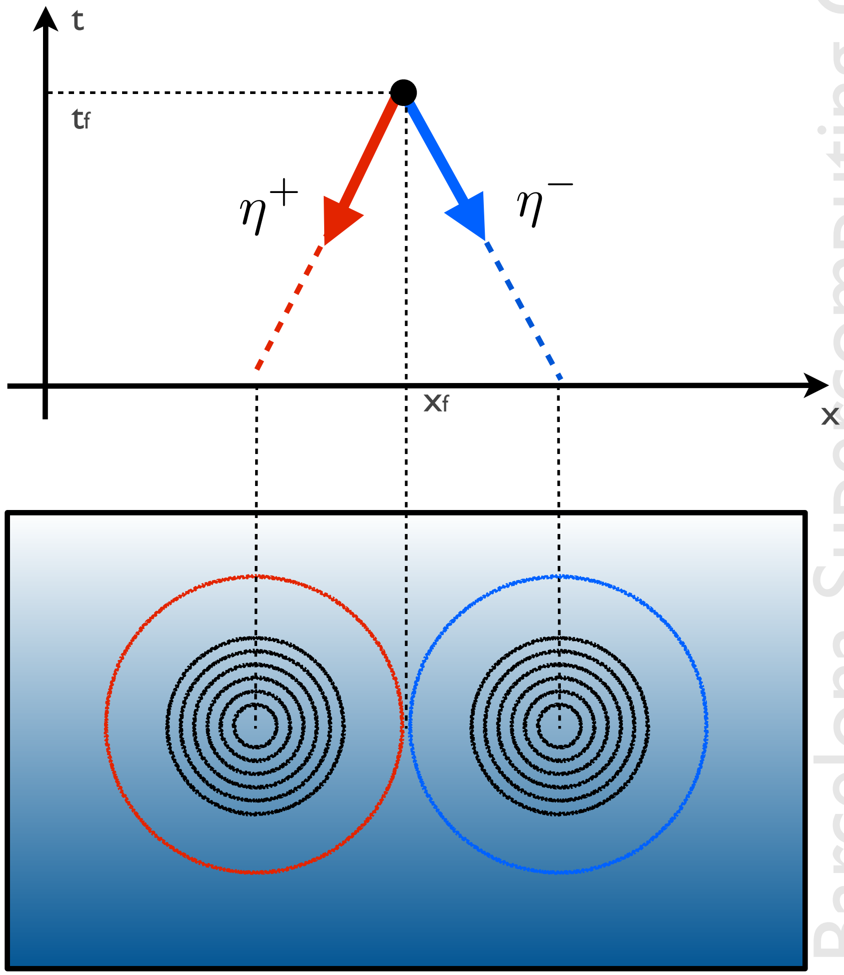
On top of that, **flow speed *u*** is any constant motion of the water pool with respect to a fixed reference frame.

The Mathematical Description

There are the three possible situations:

1. No flow speed

$$\eta^{\pm} = \frac{dx}{dt} = u \pm c$$



✓ If "u=0" there are two characteristics of opposite signs.

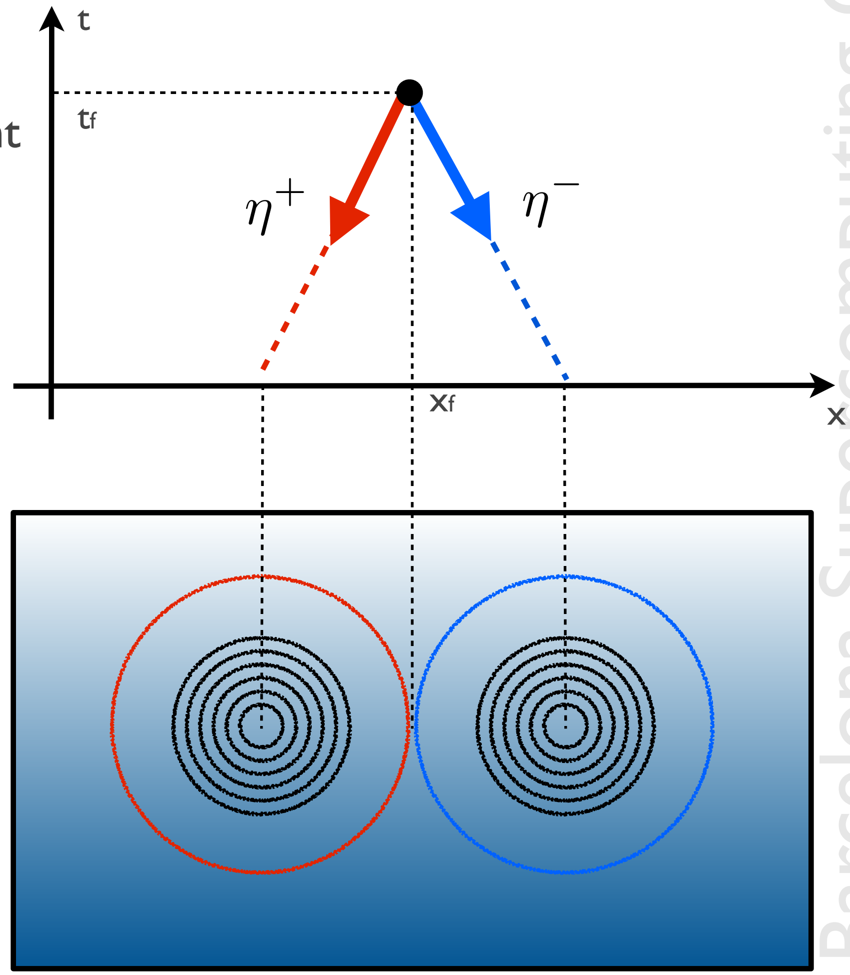
I.e., one goes left, the other goes right

The Mathematical Description

There are the three possible situations:

1. No flow speed

Characteristics are tracked back to positions ahead and behind, which are equidistant to (x_f,t_f)



The Mathematical Description

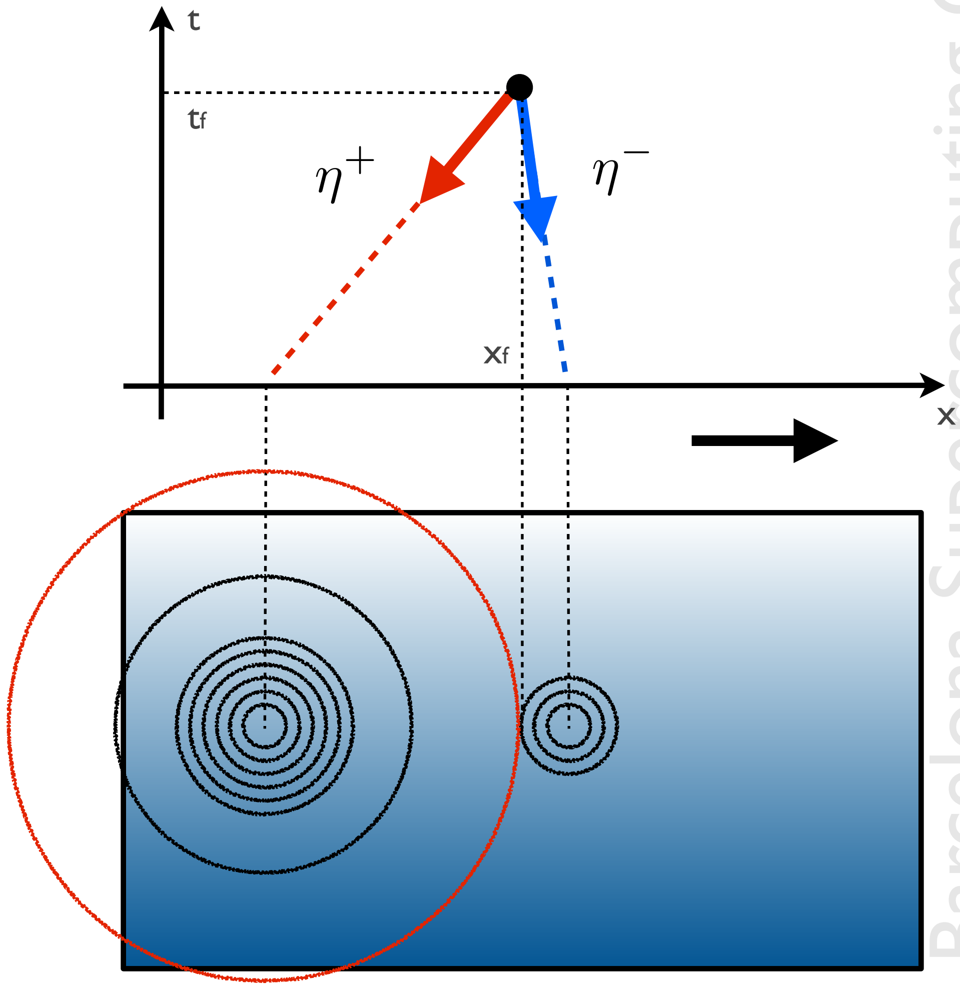
There are the three possible situations:

2. Low flow speed (subsonic)

$$\eta^{\pm} = \frac{dx}{dt} = u \pm c$$

✓ If the sound speed "c" is larger than the velocity "u", one is positive and the other one is negative.

I.e., one goes left and the other one goes right.



The Mathematical Description

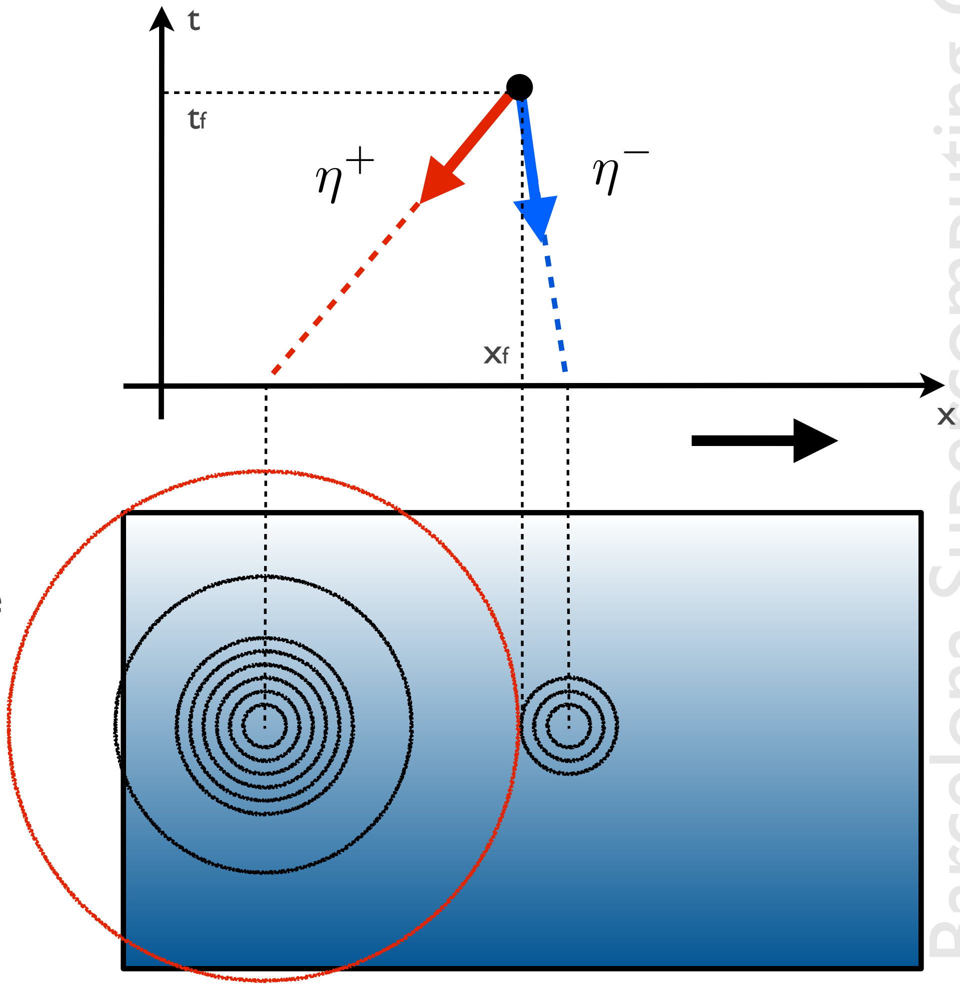
There are the three possible situations:

2. Low flow speed (subsonic)

If the water pool is moving, speeds are summed up so both sound sources are dragged behind (x_f,t_f)

The blue characteristic (downwind) brings information from a closer point, because u counteracts c.

The red one (upwind) does it from a more distant point, because u adds to c.



The Mathematical Description

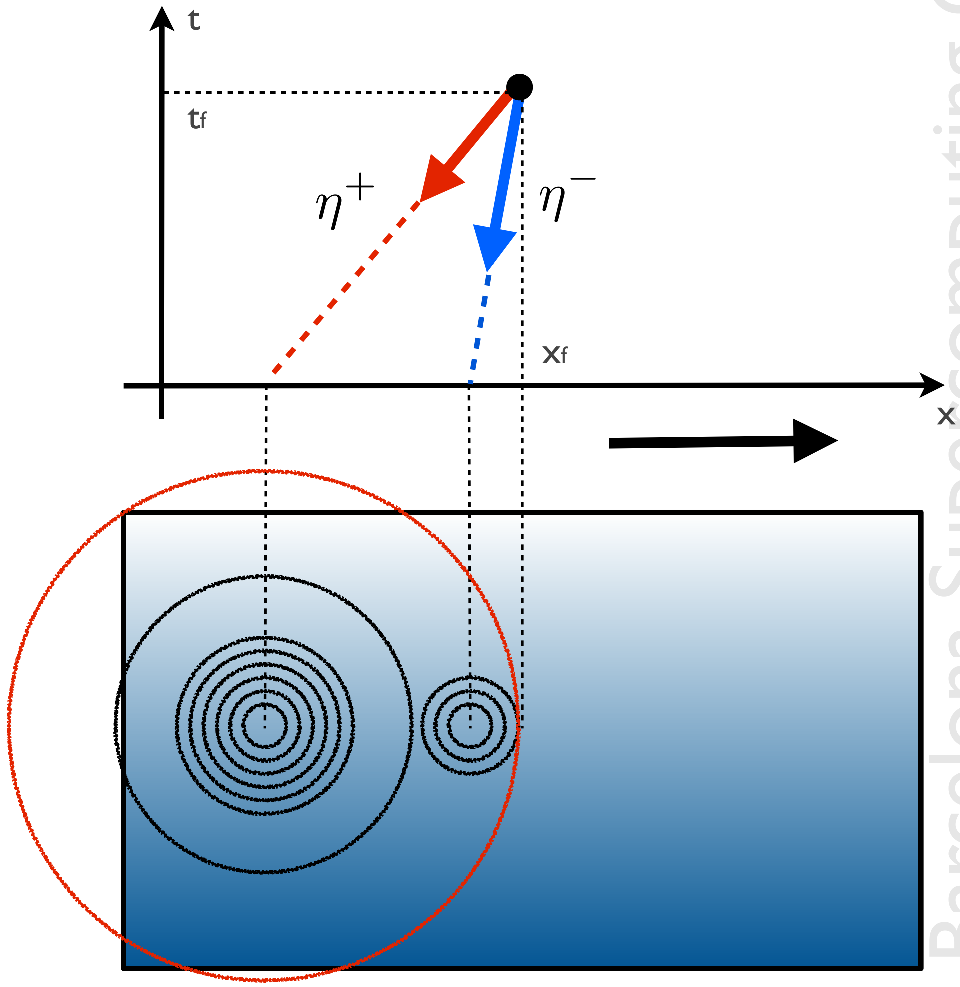
There are the three possible situations:

3. High flow speed (supersonic)

$$\eta^{\pm} = \frac{dx}{dt} = u \pm c$$

✓ If the sound speed "c" is smaller than the velocity "u", both have the same sign, but one is larger than the other.

I.e., either both go left or both go right.



The Mathematical Description

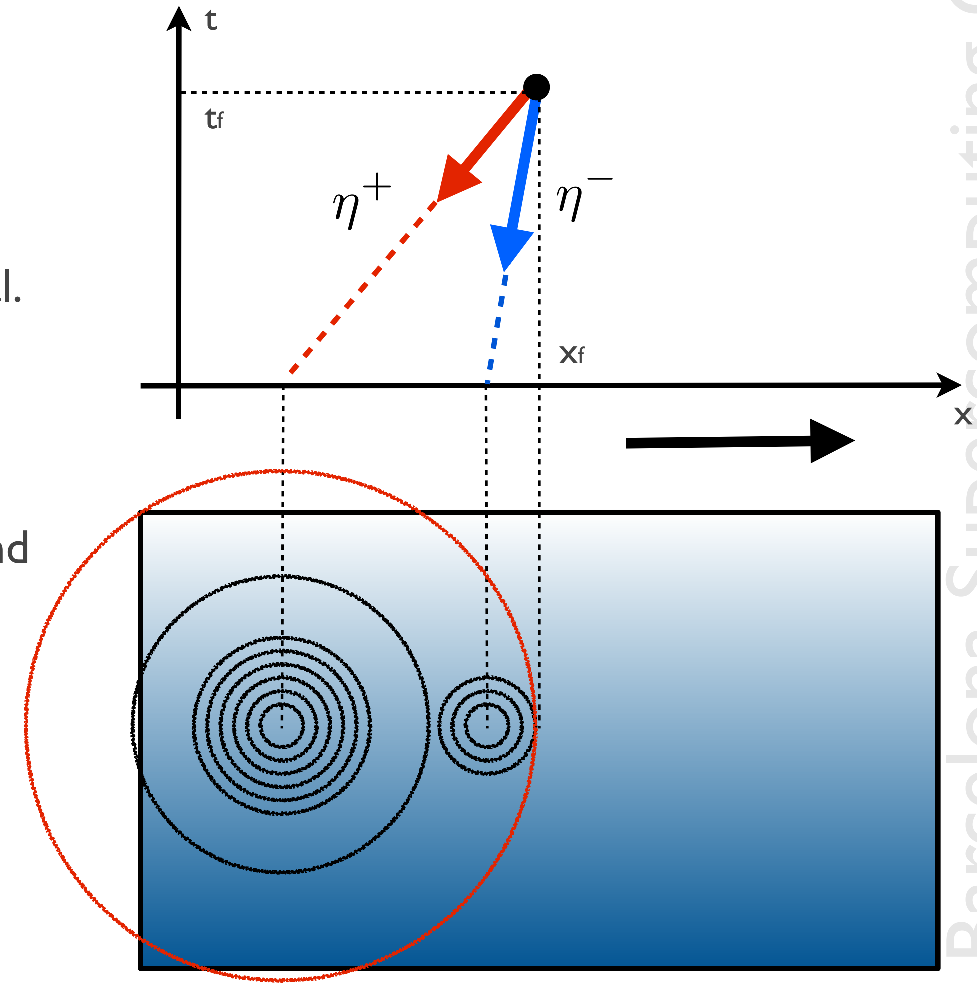
There are the three possible situations:

3. High flow speed (supersonic)

If the water pool is moving at an even higher speed we reach the limiting point when it moves at the same speed that sound, so the blue characteristic is vertical.

Going even further, both sources are left behind (x_f,t_f)

No information comes from the downwind direction.



The Mathematical Description

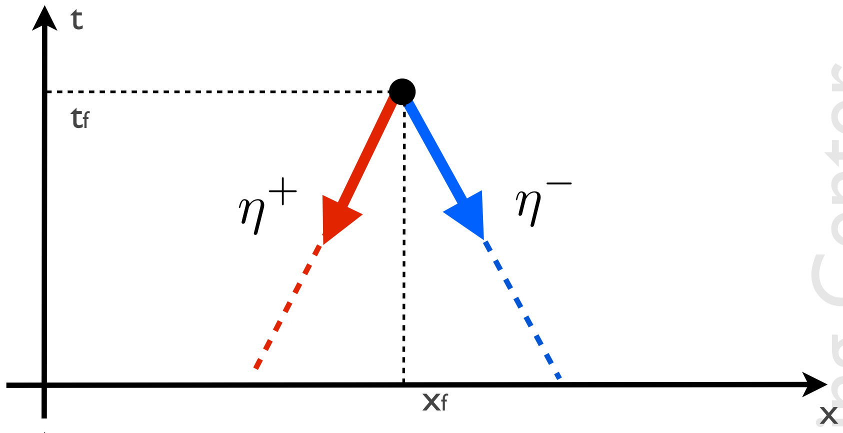
Morale:

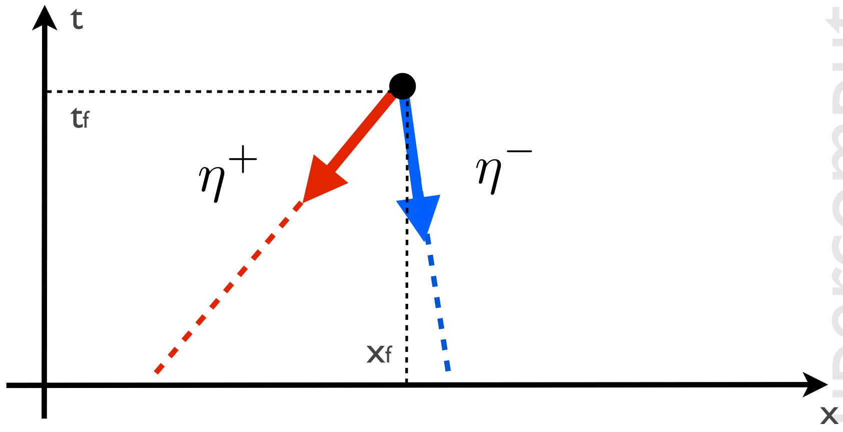
To compute what happens at (x_f,t_f), we must know

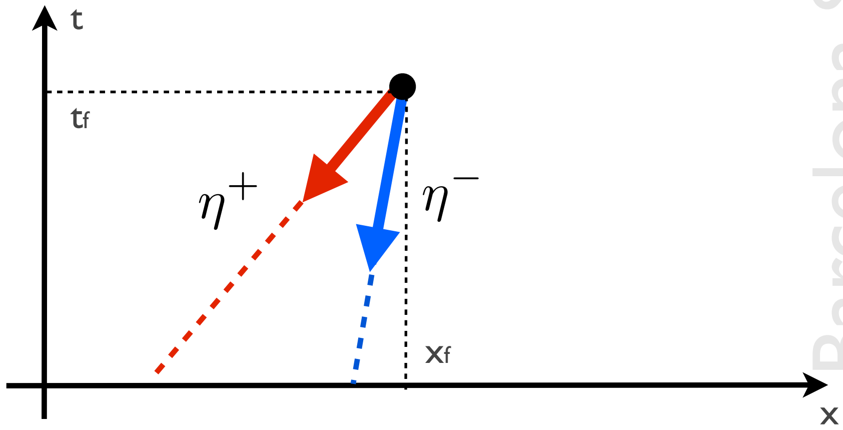
$$\eta^{\pm} = \frac{dx}{dt} = u \pm c$$

where to go and get the information that set the value of **q** there...

... and nowhere else!!!!







The Mathematical Description

Now, consider a **1D system with diffusion**

The Mathematical Description: Diffusion

Coming back to the original problem, let us see the effect of **diffusion**

$$\frac{dq^\alpha}{dt} + A^{\alpha\beta}\frac{dq^\beta}{dx} = 0$$

where "K" is determined by the system

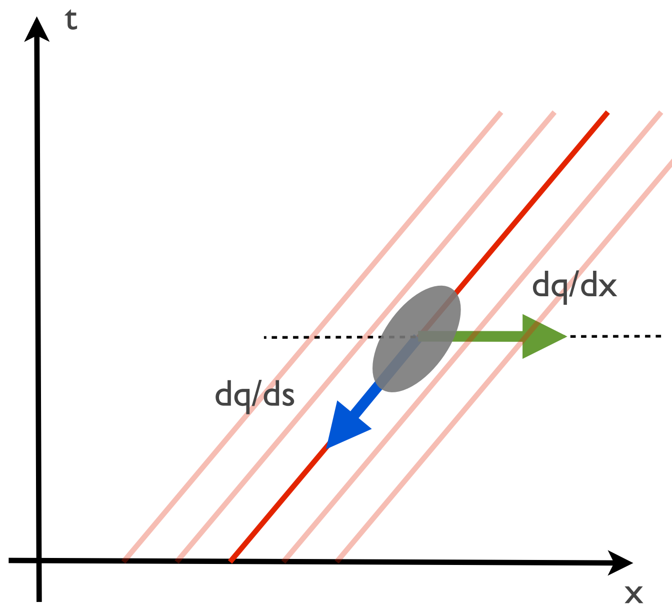
$$\frac{dq^\alpha}{ds} = K^{\alpha\beta}\frac{dq^\beta}{dx}$$

$$K^{\alpha\beta} = -A^{\alpha\beta}\frac{dt}{ds} + \delta^{\alpha\beta}\frac{dx}{ds}$$

If now "K" **is not singular**, then it could be inverted.

Then, there is **no privileged direction** at any point (x,t) and the value of "q" there is influenced from a surrounding environment.

Let us analyze this key aspect as follows...



If "K" is invertible, a relationship can be established among both differentials…!

The Mathematical Description: Diffusion

Suppose that we take an "A" that defines a hyperbolic problem and we add a non-diagonalizable tensor "B":

$$\frac{dq^\alpha}{dt} + (A^{\alpha\beta} + B^{\alpha\beta})\frac{dq^\beta}{dx} = 0$$

Then

$$\frac{dw^\alpha}{dt} + \Lambda^{\alpha\beta}\frac{dw^\beta}{dx} \quad = \quad -(R_{\alpha\gamma}B^{\gamma\epsilon}R^{-1}_{\epsilon\beta})\frac{dw^\beta}{dx}$$

where

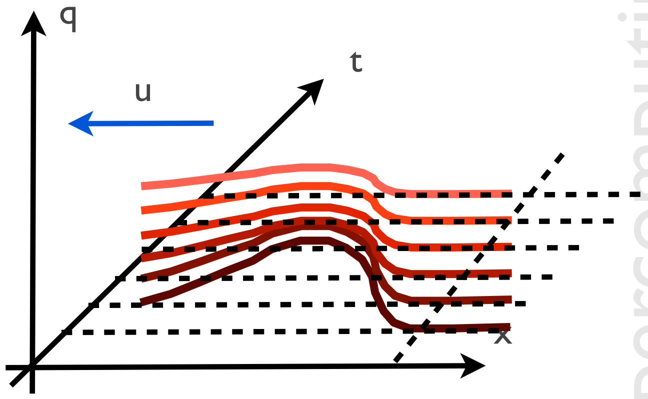
$\Lambda^{\alpha\beta}$	is the diagonal matrix whose entries are "A" eigenvalues
$R_{\alpha\gamma}$	is the matrix whose columns are "A" eigenvectors
w^α	is the vector of unknowns in the diagonalized basis

The Mathematical Description: Diffusion

Recall that the effect of a source term in the hyperbolic equation

$$\frac{Dq}{Dt} = \frac{dq}{dt} + u(x,t)\frac{dq}{dx} = b(x,t)$$

was to "erode" the initial condition for "q" or the Riemann Invariants.

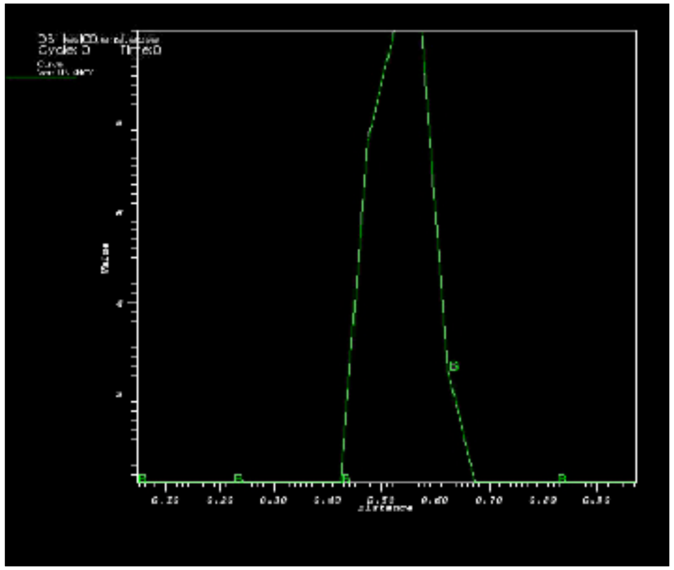


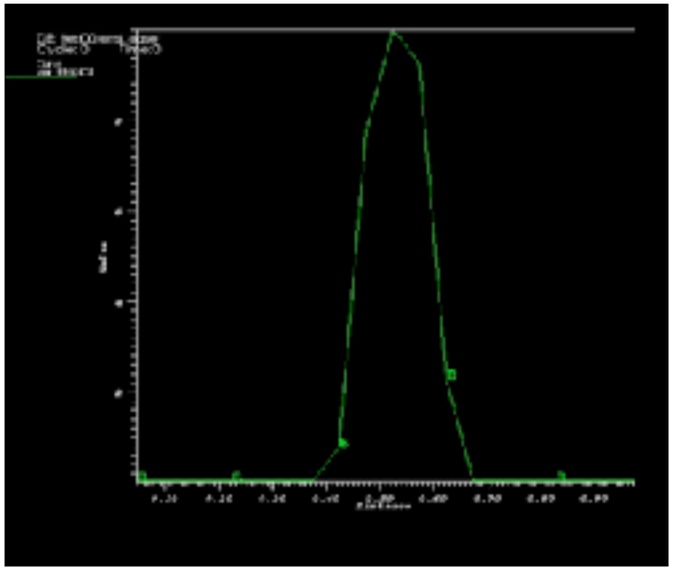
Then, the action of the right hand side term is to act as a "diffusion".

The diffusion can be seen as a mixing action on the unknowns "w":

$$\frac{dw^\alpha}{dt} + \Lambda^{\alpha\beta}\frac{dw^\beta}{dx} \quad = \quad -(R_{\alpha\gamma}B^{\gamma\epsilon}R^{-1}_{\epsilon\beta})\frac{dw^\beta}{dx}$$

The Mathematical Description





This could give us a hint on what's happening... again!

It is like to a purely **convective equation**, numerical discretisation has introduced **diffusion**!!

The Mathematical Description

Let us summarise the most important issues:

Not all of the problems described by PDEs are hyperbolic...

... however, hyperbolic behaviour exposes a subtle feature that is of utmost importance.

It is so important that convective instabilities can arise even when diffusion is present, but it is not dominant.

Hyperbolic behaviour means that the value of the unknowns at certain (x,t) is **completely determined** by what's happening **along the space-time trajectory (i.e. characteristic).**

What lies outside these curves **must not influence at all** what is happening in (x,t).

However, it happens…

This fact has a decisive consequence for **discretising** any equation...

Discretisation:

Divide and Conquer

Discretising the System

Suppose a 1-D convection equation:

$$\frac{dq}{dt} + u\frac{dq}{dx} = 0$$

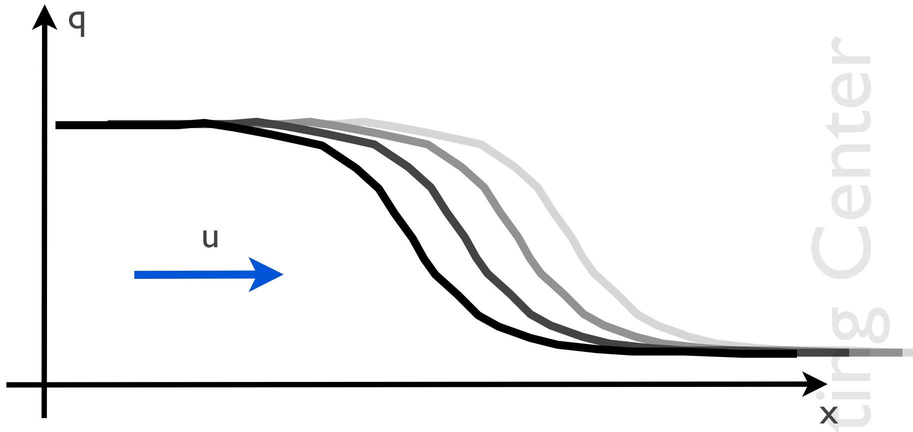
With a certain initial data distribution q(x, t=0).

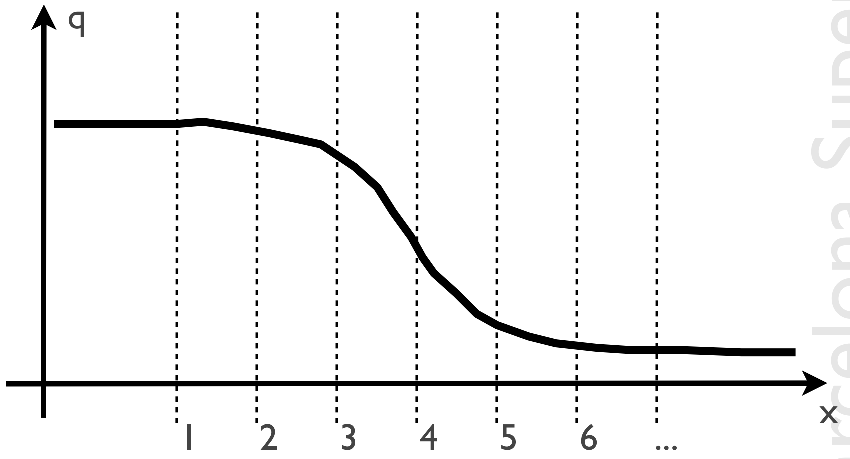
We already know that it must propagate rightwards as it is.

Discretising the equation,

$$\frac{\Delta q}{\Delta t} = -u\frac{\Delta q}{\Delta x}$$

$$\Delta q = -u\Delta t\frac{\Delta q}{\Delta x}$$

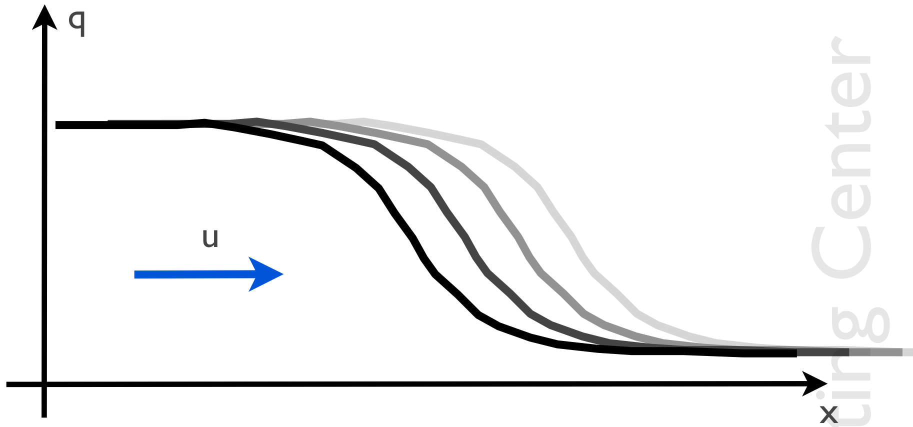




Discretising the System

At 1st order, let us discretise time **backwards**:

$$\frac{\Delta q^{n+1}}{\Delta t} = \frac{q^{n+1} - q^n}{\Delta t} = \frac{q^{n+1} - q}{\Delta t}$$

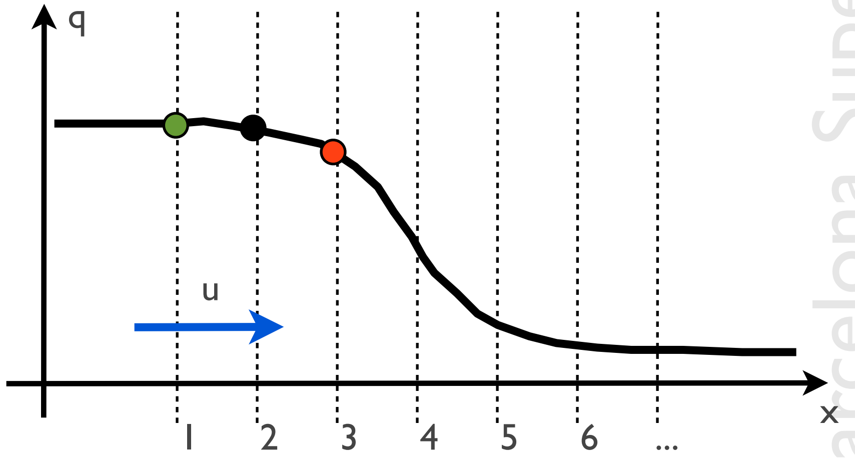


At first order, space is discretised in one of these three different ways, considering the velocity direction (positive in this case):

$$\frac{\Delta q_2}{\Delta x} = \frac{q_2 - q_1}{\Delta x}$$ **upwind**

$$\frac{\Delta q_2}{\Delta x} = \frac{q_3 - q_2}{\Delta x}$$ **downwind**

$$\frac{\Delta q_2}{\Delta x} = \frac{q_3 - q_1}{\Delta x}$$ **centered**



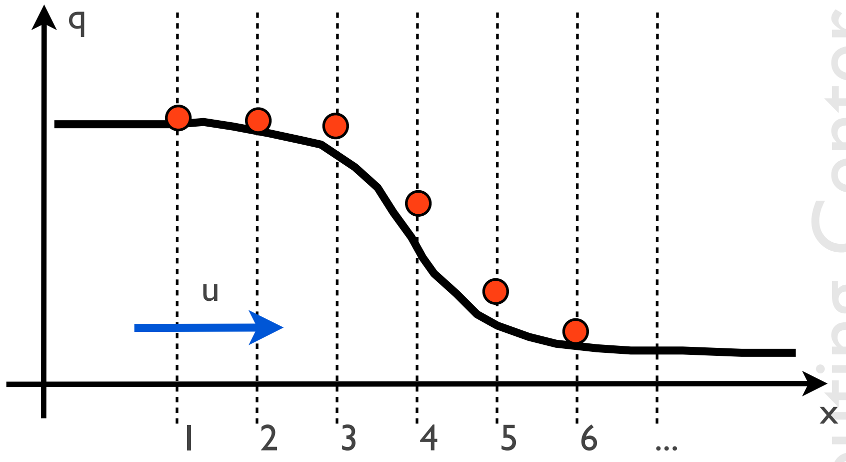
Discretising the System

Upwind

$$q_3 = q_3 - C(q_3 - q_2)$$

$$q_4 = q_4 - C(q_4 - q_3)$$

$$q_5 = q_5 - C(q_5 - q_4)$$



Upwind discretisation produces a proper wave propagation although it is diffusive specially the higher the gradient.... **unless what is explained a few slides later happens!**

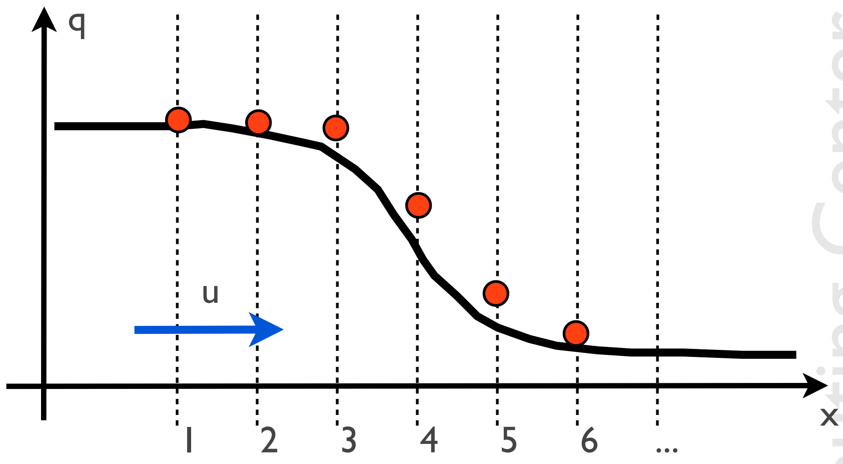
$$\Delta q = -u\Delta t \frac{\Delta q}{\Delta x}$$

Numerical diffusion is co-related with the size of space-time discretisation.

$$C = \frac{u\Delta t}{\Delta x}$$

Discretising the System

Downwind discretisation produces oscillations that makes the scheme **unstable**, no matter how small discretisation is.



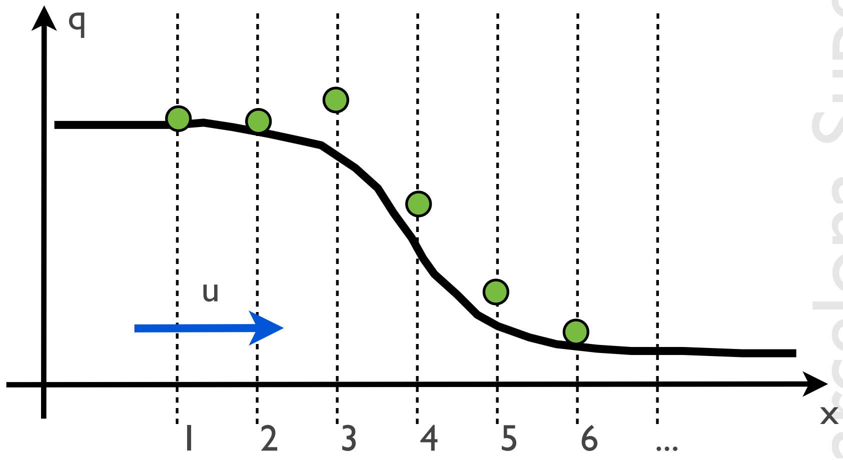
$$C = \frac{u\Delta t}{\Delta x}$$

Downwind

$$q_3 = q_3 - C(q_4 - q_3)$$

$$q_4 = q_4 - C(q_5 - q_4)$$

$$q_5 = q_5 - C(q_6 - q_5)$$

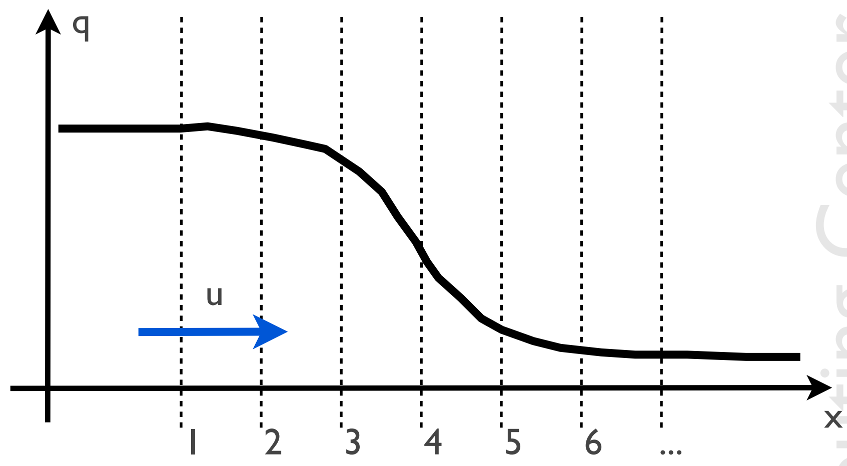


Discretising the System

Centered

Exercise

Centred it is even worse...



Discretising the System

Why is upwind a stable discretisation scheme...?

Discretising the System

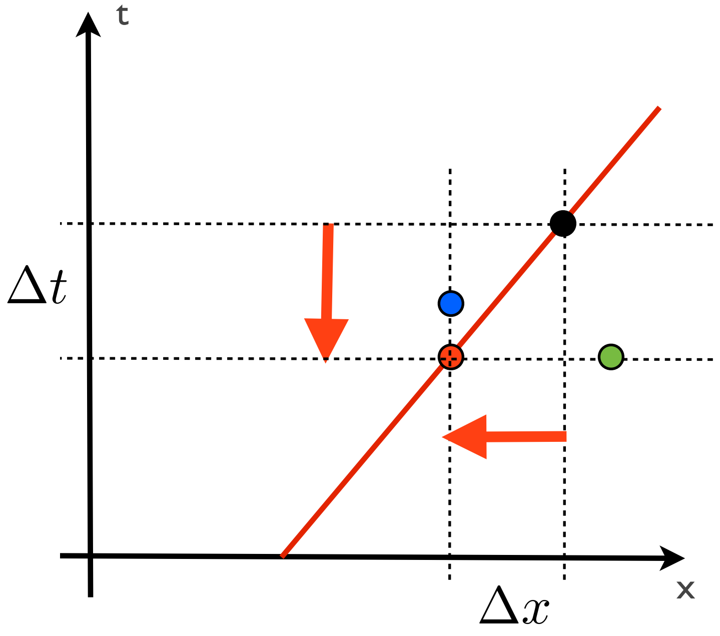
By combining the proper time and space integration schemes, we move backwards along the characteristic.

Current values must be computed only from previous ones and at a **specific location**: on the characteristic.

Time increment and space discretisation must have a relationship, both can´t be fixed independently.

Backwards (i.e. explicit) schemes stability depends on a limiting value of the time step:

CAUSALITY CANNOT BE VIOLATED



Discretising the System

... and finally, the upwind explains what was happening to our blob:

Let us set

$$C = \frac{u\Delta t}{\Delta x}$$

$$q_3 = q_3 - C(q_3 - q_2)$$

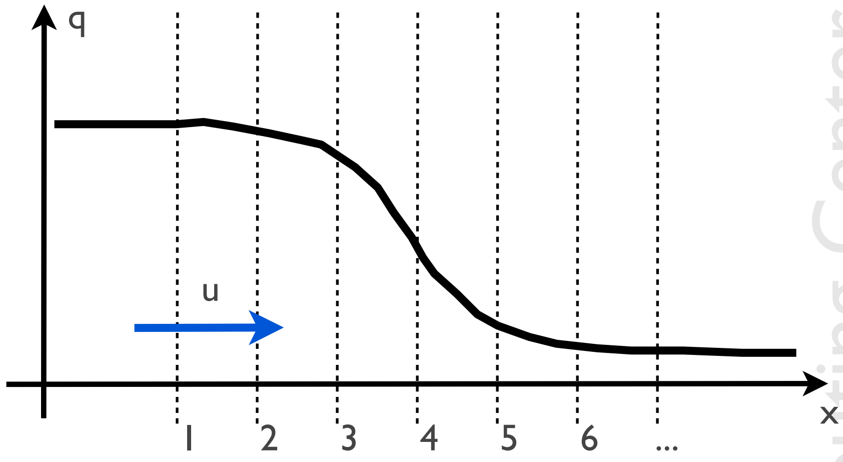
1. If we take **exactly**

$$\Delta t = \frac{\Delta x}{u}$$

then, *C*=1 and

$$\begin{aligned} q_3 &= q_3 - C(q_3 - q_2) & & & q_3 &= q_2 \\ q_4 &= q_4 - C(q_4 - q_3) & &\Rightarrow & q_4 &= q_3 \\ q_5 &= q_5 - C(q_5 - q_4) & & & q_5 &= q_4 \end{aligned}$$

which is a perfect transport!!!



Discretising the System

... and finally, the upwind explains what was happening to our blob:

Let us set

$$C = \frac{u\Delta t}{\Delta x}$$

$$q_3 = q_3 - C(q_3 - q_2)$$

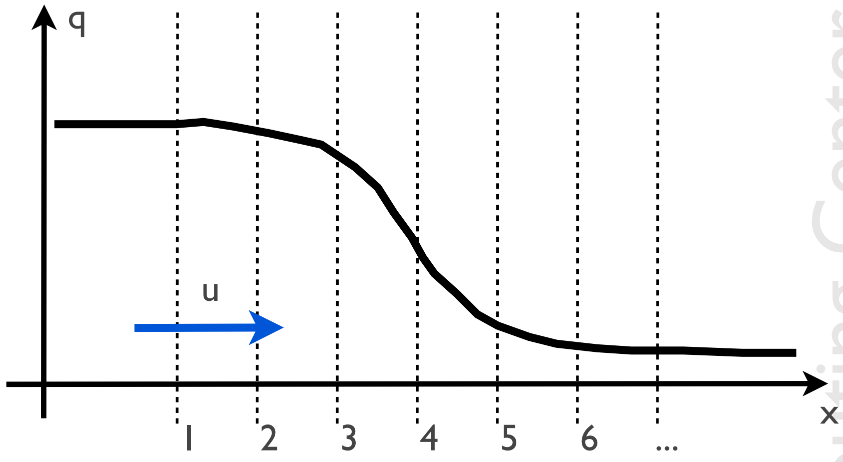
1. If we take **exactly**

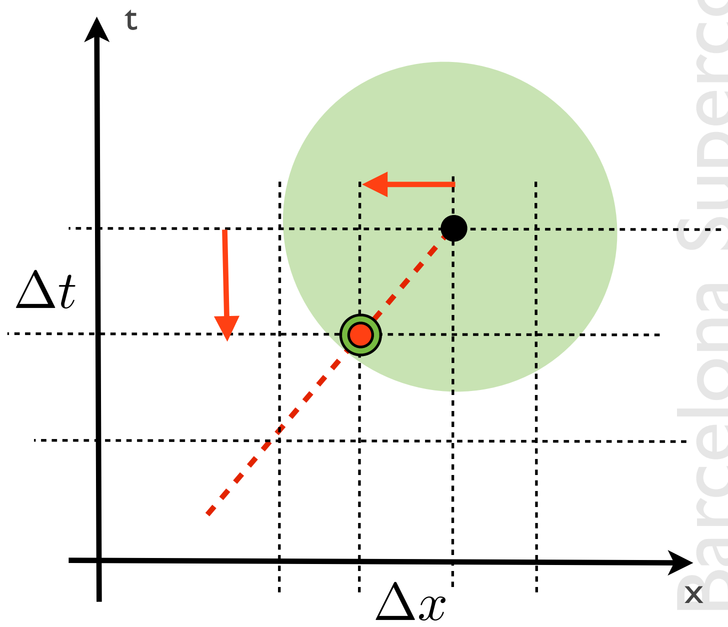
$$\Delta t = \frac{\Delta x}{u}$$

then, C=1 and

$$\begin{aligned} q_3 &= q_3 - C(q_3 - q_2) & & & q_3 &= q_2 \\ q_4 &= q_4 - C(q_4 - q_3) & & \Rightarrow & q_4 &= q_3 \\ q_5 &= q_5 - C(q_5 - q_4) & & & q_5 &= q_4 \end{aligned}$$

which is a perfect transport!!!





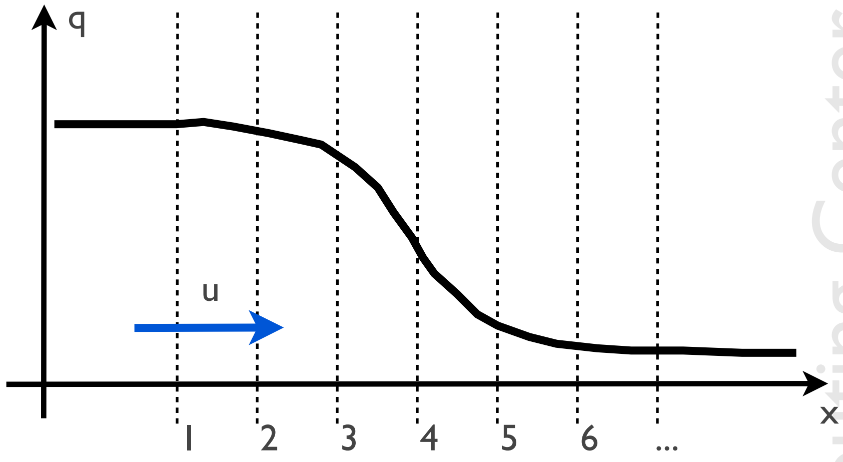
Discretising the System

... and finally, the upwind explains what was happening to our blob:

Let us set

$$C = \frac{u\Delta t}{\Delta x}$$

$$q_3 = q_3 - C(q_3 - q_2)$$



2. If C is **different than one**

$$q_3 = q_3 - C(q_3 - q_2)$$
$$q_4 = q_4 - C(q_4 - q_3)$$
$$q_5 = q_5 - C(q_5 - q_4)$$

and every value is a mean of itself and that of the upwind neighbour.

This introduces the numerical diffusion.

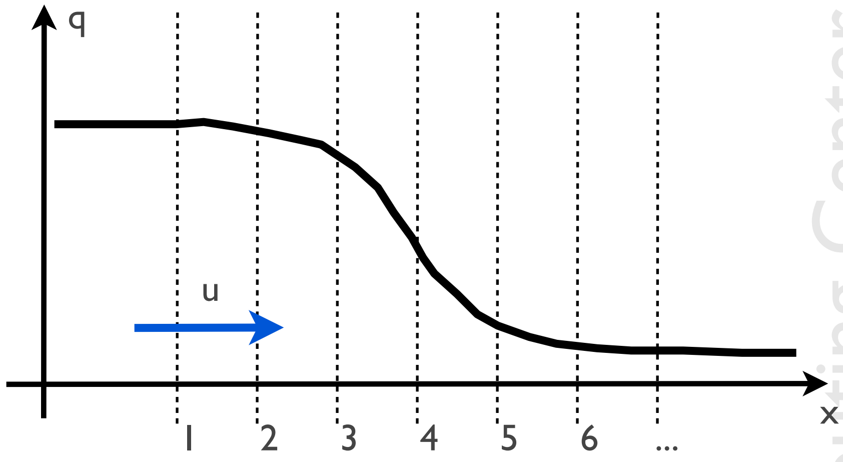
Discretising the System

... and finally, the upwind explains what was happening to our blob:

Let us set

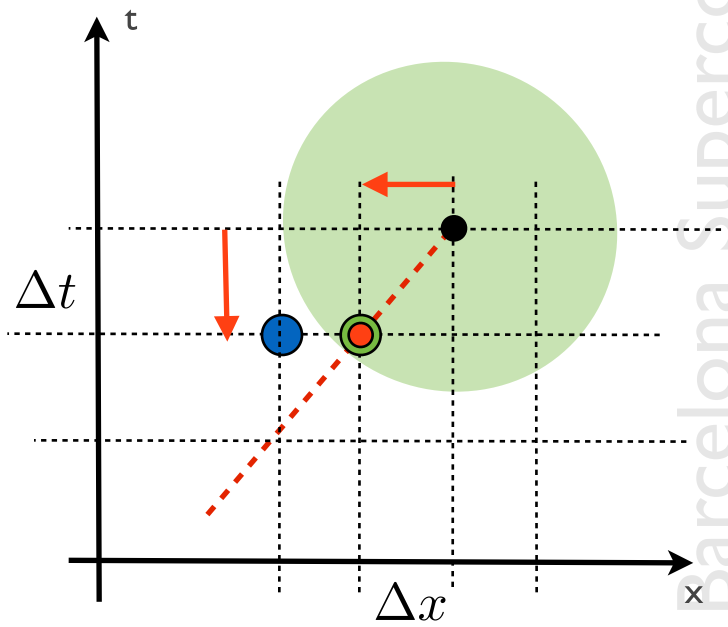
$$C = \frac{u\Delta t}{\Delta x}$$

$$q_3 = q_3 - C(q_3 - q_2)$$



2. If C is **different than one**

$$q_3 = q_3 - C(q_3 - q_2)$$
$$q_4 = q_4 - C(q_4 - q_3)$$
$$q_5 = q_5 - C(q_5 - q_4)$$



and every value is a mean of itself and that of the upwind neighbour.

This introduces the numerical diffusion.

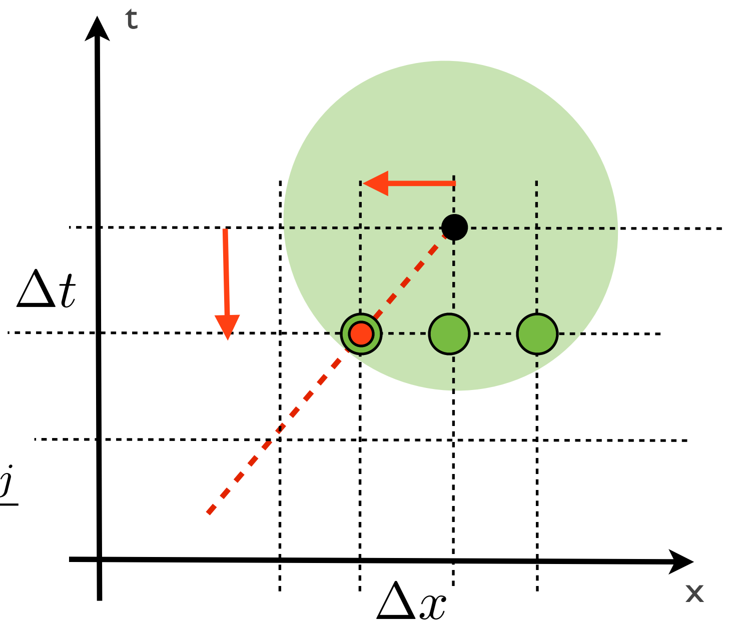
Discretising the System

Let's go back to the upwind problem...

The same can be applied to equations with non-hyperbolic terms, like diffusion.

Now the hyperbolic terms can be computed using the characteristics (i.e. with upwind) and the non-hyperbolic ones using the closest neighbours.

$$\frac{\Delta}{\Delta x}\left(\frac{\Delta q_{i,j}}{\Delta x}\right) = \frac{q_{i+1,j} - 2q_{i,j} + q_{i-1,j}}{\Delta x^2}$$



Recall that diffusion information propagation should include data from outside the characteristics.

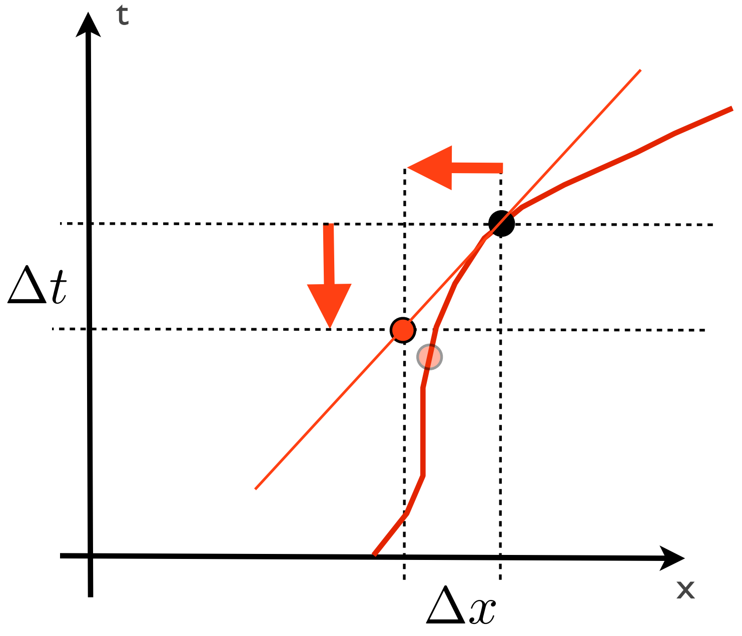
Discretising the System

The same analysis can be applied to the linear u(x,t) case

Accuracy depends on the time step and space increment

The same can be also applied to the non-linear u(q,x,t) case

Non-linearity could require a second level of discretisation strategy: **linearisation iterations**



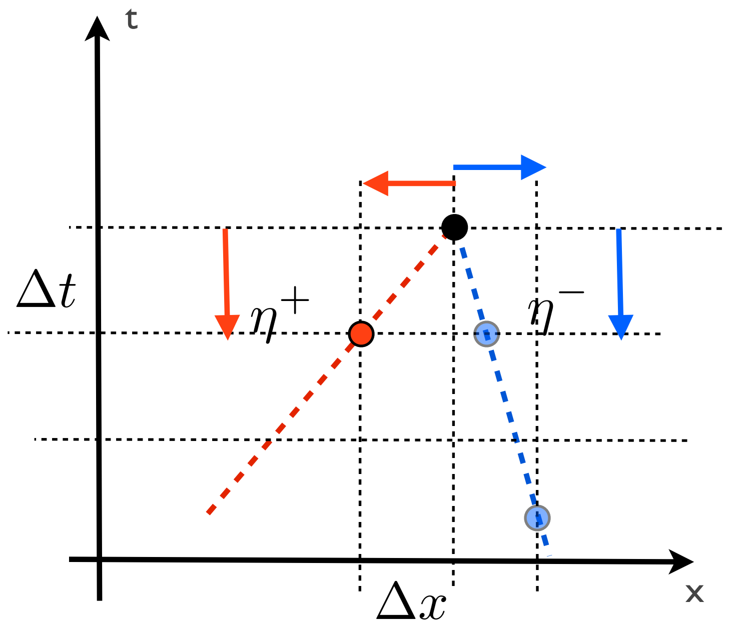
Discretising the System

The same can be applied for systems of equations (suppose 1D gas dynamics)

Now there are two unknowns and two characteristics, one for each equation

Both of them go backwards in time but at opposite directions in space...

...but we find a new discretisation problem:



The space-time discretisation does not fit well simultaneously for both characteristics!

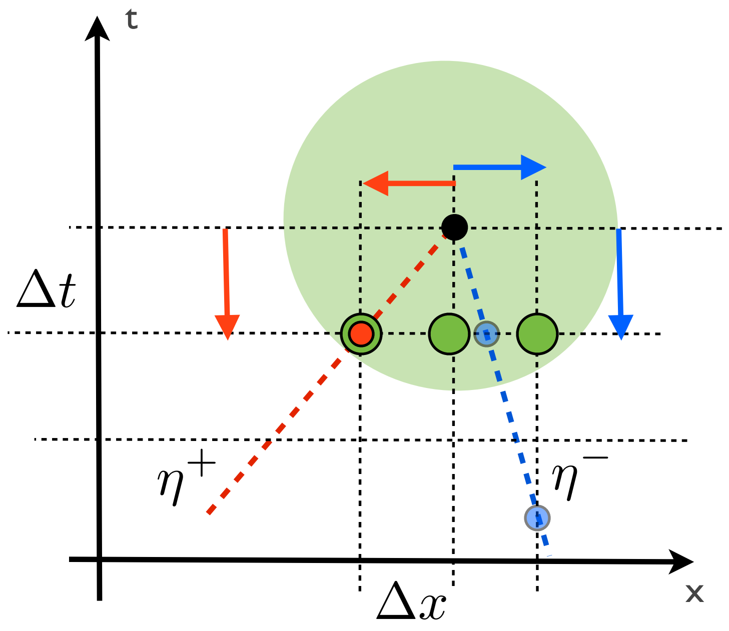
Discretising the System

The same can be applied to systems with non-hyperbolic terms.

Now the hyperbolic terms can be computed using the characteristics (i.e. with upwind) and the non-hyperbolic ones using the closest neighbors

But still remains the discretization problem with the convection for the blue characteristic.

$$\frac{dq^\alpha}{dt} + (A^{\alpha\beta} + B^{\alpha\beta})\frac{dq^\beta}{dx} = 0$$



Discretising the System

All cases share the same problem:

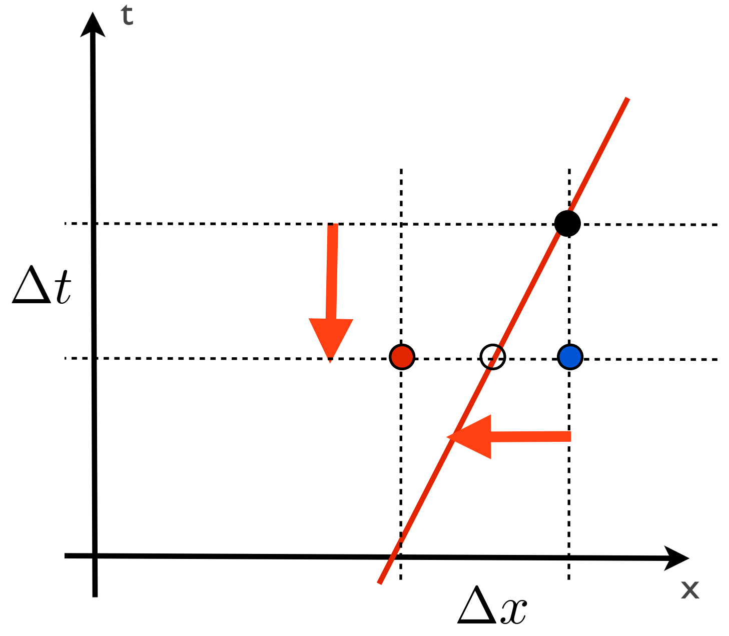
You cannot fix **independently** time and space

Time is flexible enough (up to causality)...

... but space discretisation is not!

Eventually, you could interpolate, but the blue dot is downwind!!!

And the problems pile up...



Discretising the System

These are discretisation problems we have encounter so far:

- ✓ For ALL hyperbolic cases, if no upwind is used, the scheme is unstable
- ✓ Backwards (i.e. explicit) schemes stability depends on a limiting value of the time step
- ✓ Accuracy depends on the time step and space increment
- ✓ Non-linearity requires a second level of discretization strategy: linearization iterations
- ✓ The space-time discretization **does not fit well** to the characteristics
- ✓ When non-hyperbolic terms are present, the discretization problem with the convection for the blue characteristic remains
- ✓ ... and more to come.

Discretising the System

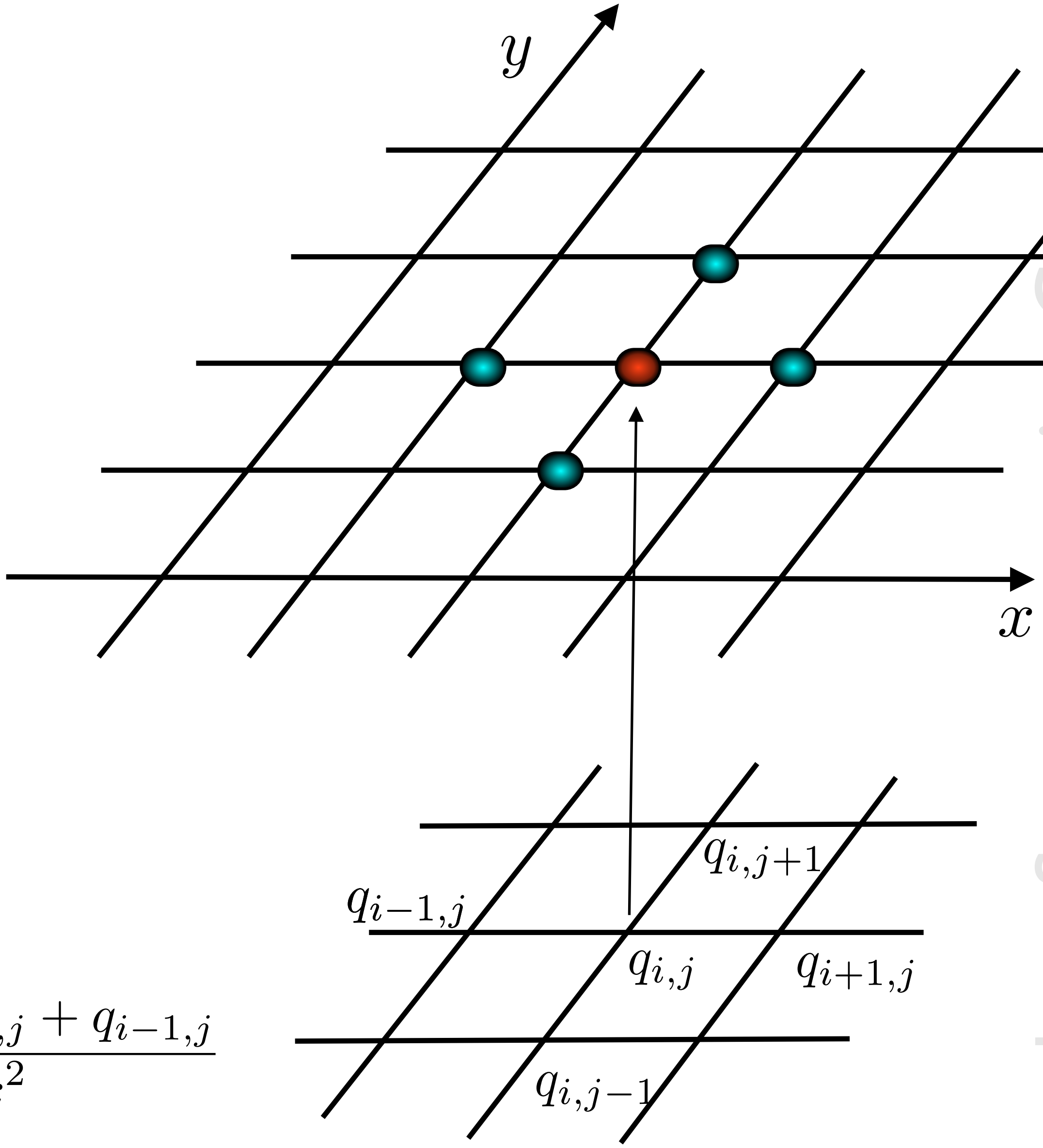
First strategy:
Finite Differences

All this means that you cannot expect that this will work on a general basis!!!

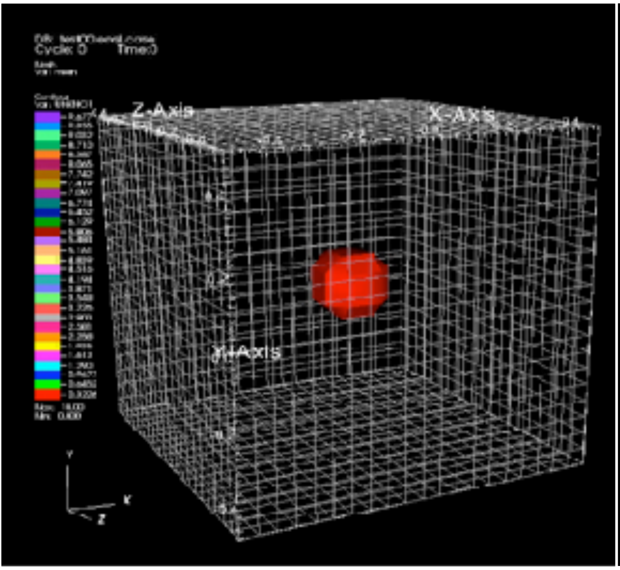
Suppose convective and diffusive fluxes:

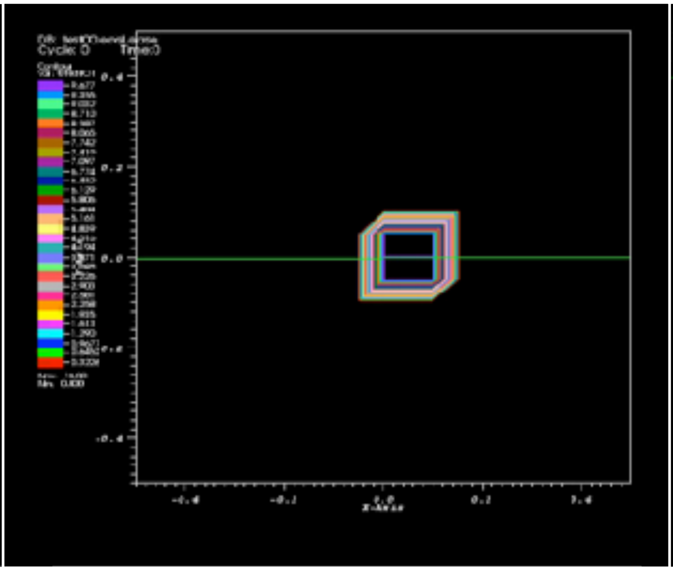
$$\frac{\Delta q_{i,j}}{\Delta x} = \frac{q_{i+1,j} - q_{i,j}}{\Delta x}$$

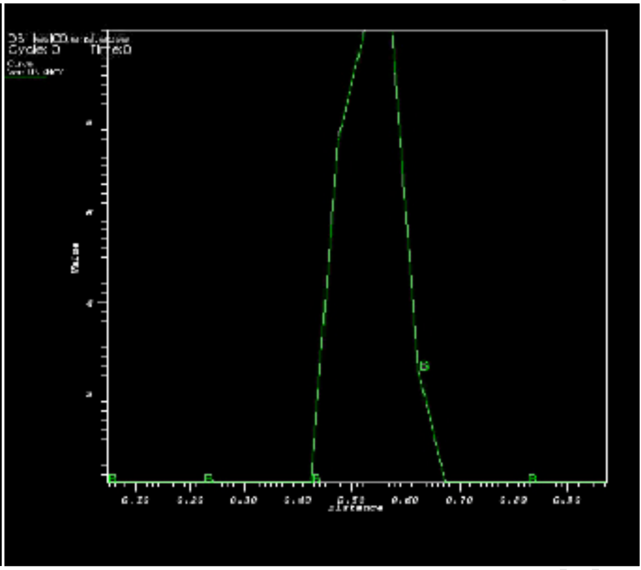
$$\frac{\Delta}{\Delta x}\left(\frac{\Delta q_{i,j}}{\Delta x}\right) = \frac{q_{i+1,j} - 2q_{i,j} + q_{i-1,j}}{\Delta x^2}$$



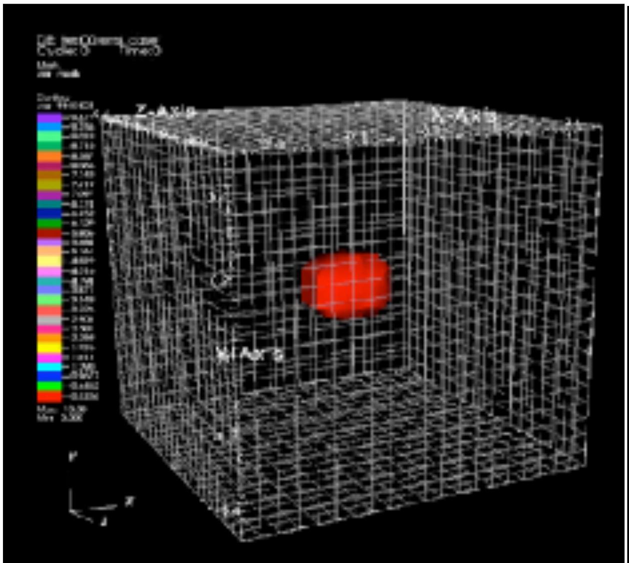
Case I

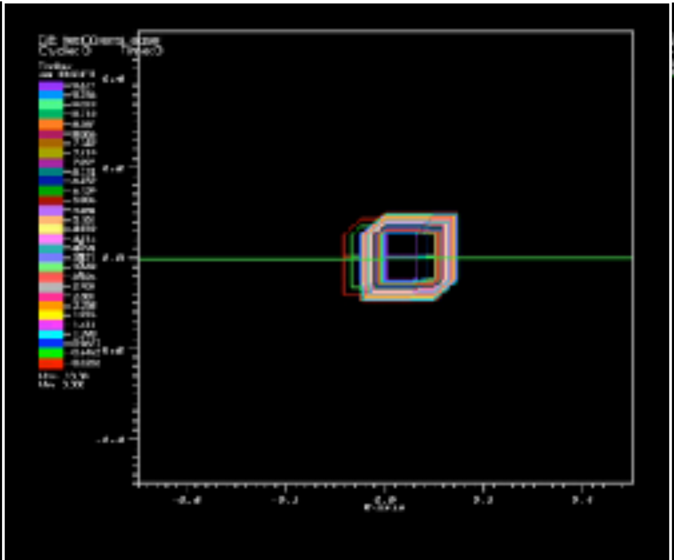


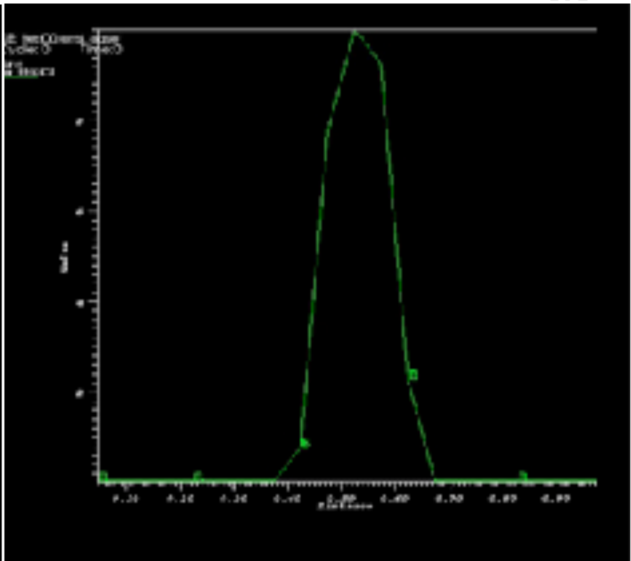




Case II: smaller dt







Discretising the System

So how all these problems can be attacked?

Convective stabilisation issue:

Convective stabilisation is a traditional issue in numerical analysis.
Different solutions proposed, depending on the context.
What context? Finite Elements or Finite Volumes.

- ✓ FV: Low order: upwind for first derivatives (linear version of Godunov method)
- ✓ FV: Higher order schemes: increase the time order
- ✓ FV: Limiters and Total Variation Diminishing (TVD) to treat discontinuities
- ✓ FE: Artificial Diffusion (AD), Streamline Diffusion (SD), Streamline Upwind Petrov- Galerkin (SUPG)
- ✓ FE: Characteristic Galerkin (CG)
- ✓ FE: Galerkin Least Squares (GLS), Variational Multiscale (VM or VMS), Preconditioned stabilisation
- ✓ FE: Shock capturing to treat discontinuities

Discretising the System

So how all these problems can be attacked?

Convective stabilisation issue:

Convective stabilisation is a traditional issue in numerical analysis.
Different solutions proposed, depending on the context.
What context? Finite Elements or Finite Volumes.

The idea is basically the same:

Try to stay as much as possible on the characteristics

Add a (numerical) diffusion along the characteristics

Increase the order of the scheme to diminish numerical diffusion: LOW ORDER IS MORE DIFFUSION (degrades faster with time / space)

Numerical diffusion must be smart enough to cope with mixed convection-diffusion-reaction problems

Recall that this problem comes from the hyperbolic character of the equations, so it will be present in both explicit or implicit schemes.

Discretising the System

So how all these problems can be attacked?

Convective stabilisation issue:

Convective stabilisation is a traditional issue in numerical analysis.
Different solutions proposed, depending on the context.
What context? Finite Elements or Finite Volumes.

Limiting time step for explicit schemes:

If the **critical** time step is too small, the efficiency of the scheme is compromised (low Mach, high aspect ratios, contact problems...).

Accuracy:

Time higher order schemes: Runge-Kutta, alpha-generalized, Crank-Nicholson,...
Space higher order schemes: limiters (FE), higher interpolation order (FE), larger stencils (FD)...

Non-linear iterative schemes (for implicit schemes):

Jacobi iterations, Newton iterations with tangent moduli or secants
Newton-Krilov matrix free

Discretising the System

To have a deeper idea on the difficulties of stabilising convection, let us go to a higher order scheme

Consider

$$\frac{\partial q}{\partial t} = u \frac{\partial q}{\partial x}$$

A 2nd order Taylor expansion gives

$$q^{n+1} = q \; + \; \Delta t \frac{\partial q}{\partial t} \; + \; \frac{1}{2} \Delta t^2 \frac{\partial^2 q}{\partial t^2} \; + \; \cdots$$

Considering that

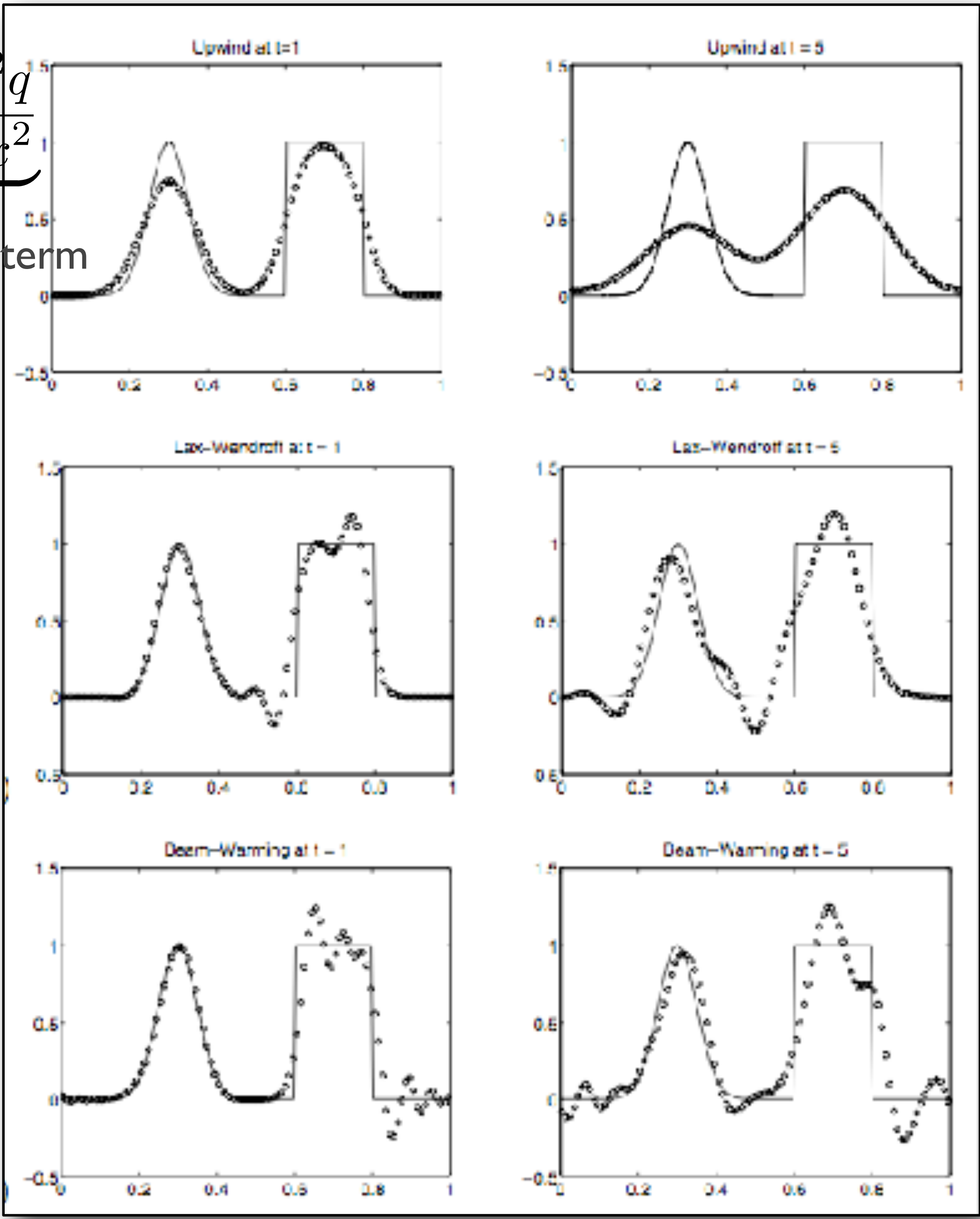
$$\frac{\partial^2 q}{\partial t^2} = u \frac{\partial}{\partial t}\left(\frac{\partial q}{\partial x}\right) = u \frac{\partial}{\partial x}\left(\frac{\partial q}{\partial t}\right) = u \frac{\partial}{\partial x}\left(u \frac{\partial q}{\partial x}\right) = u^2 \frac{\partial^2 q}{\partial x^2}$$

Then, the so called **Lax-Wendroff** scheme is

$$q^{n+1} = q \; + \; \Delta t \, u \frac{\partial q}{\partial x} \; + \; \underbrace{\frac{1}{2} \Delta t^2 \, u^2 \frac{\partial^2 q}{\partial x^2}}_{\text{Diffusion-like term}}$$

Discretising the System

$$q^{n+1} = q + \Delta t\, u \frac{\partial q}{\partial x} + \underbrace{\frac{1}{2}\Delta t^2\, u^2 \frac{\partial^2 q}{\partial x^2}}_{\text{Diffusion-like term}}$$



There are several ways of computing the first and second derivatives: upwind, centred, projecting the fluxes, ... so we obtain different behaviours and different schemes.

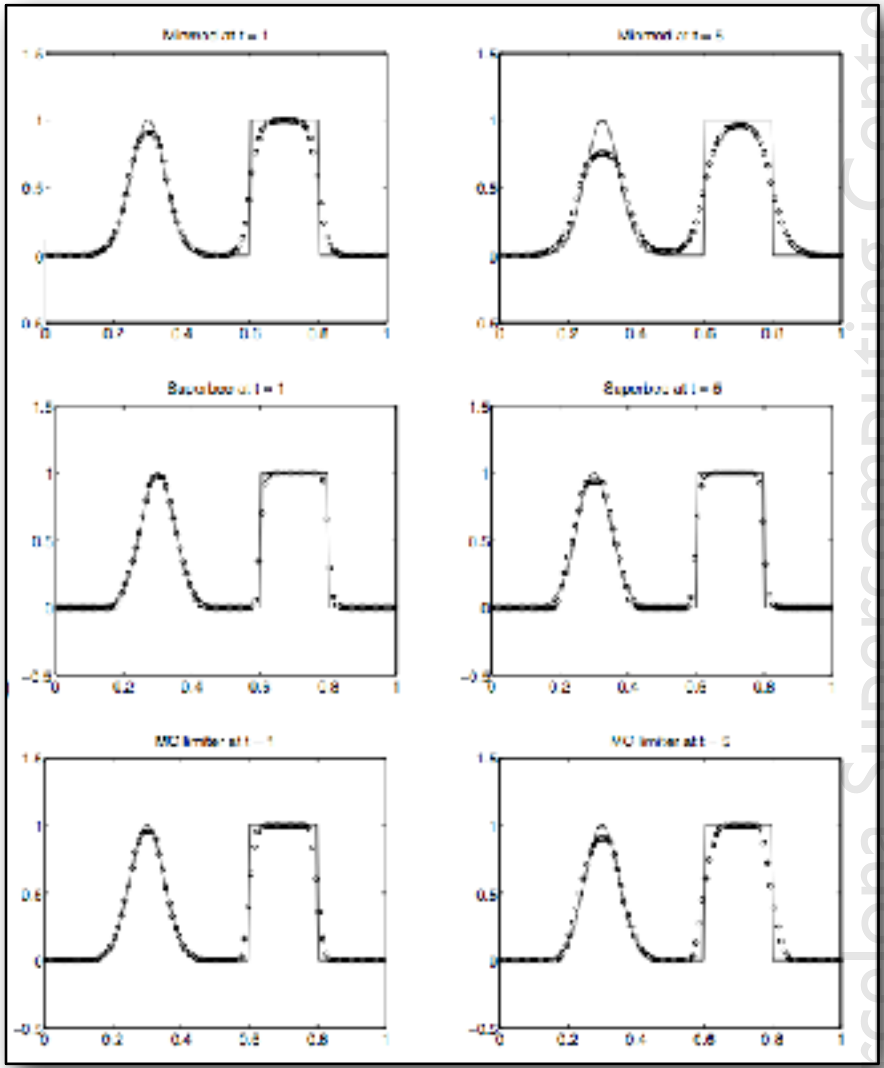
Lax Wendroff: all centered

Beam Warming: all upwinded

And discontinuities make them worse!

Discretising the System

$$q^{n+1} = q + \Delta t\, u \frac{\partial q}{\partial x} + \underbrace{\frac{1}{2}\Delta t^2\, u^2 \frac{\partial^2 q}{\partial x^2}}_{\text{Diffusion-like term}}$$



Limiters:
A FV-based stabilisation strategy.

They are sophisticated ways of limiting the steepness of the variations.

Sophisticated, yes...

... but computationally expensive and too artisanal.

Discretising the System

Trying to solve one issue…

we find another one!!

Discontinuities

Discretising the System

Discontinuities or very strong gradients are everywhere...!

Fluids:

Boundary layers at high Reynolds

Mixing layers

Shocks

Combustion, in regions of strong active behavior

Flows with free surface

Discontinuities in initial / boundary conditions

Discretising the System

Discontinuities or very strong gradients are everywhere...!

Solids:

Fracture

Abrupt changes in material properties

Complex materials and composites

Plasticity

Discretisation:

Divide and Conquer

(at last)

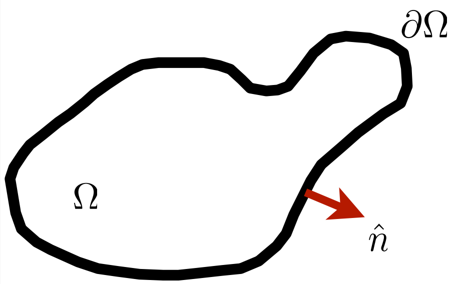
Discretising the System

The Integral Form (IF) ⇨ **The Differential Conservative Form (DCF)** ⇨ **The Differential Jacobian Form (DJF)**

The Integral Form (IF)

The basic form of a conservation principle

No continuity assumptions



$$\frac{\partial}{\partial t}\int_{\Omega} q^{\alpha} - \int_{\partial\Omega} F_i^{\alpha} n_i = 0$$

The Differential Conservative Form (DCF)

The partial differential equation derived from IF

Additional regularity assumptions required

$$\frac{\partial q^{\alpha}}{\partial t} + \frac{\partial F_i^{\alpha}(q)}{\partial x_i} = 0$$

The Differential Jacobian Form (DJF)

A little algebra to derive it from DCF

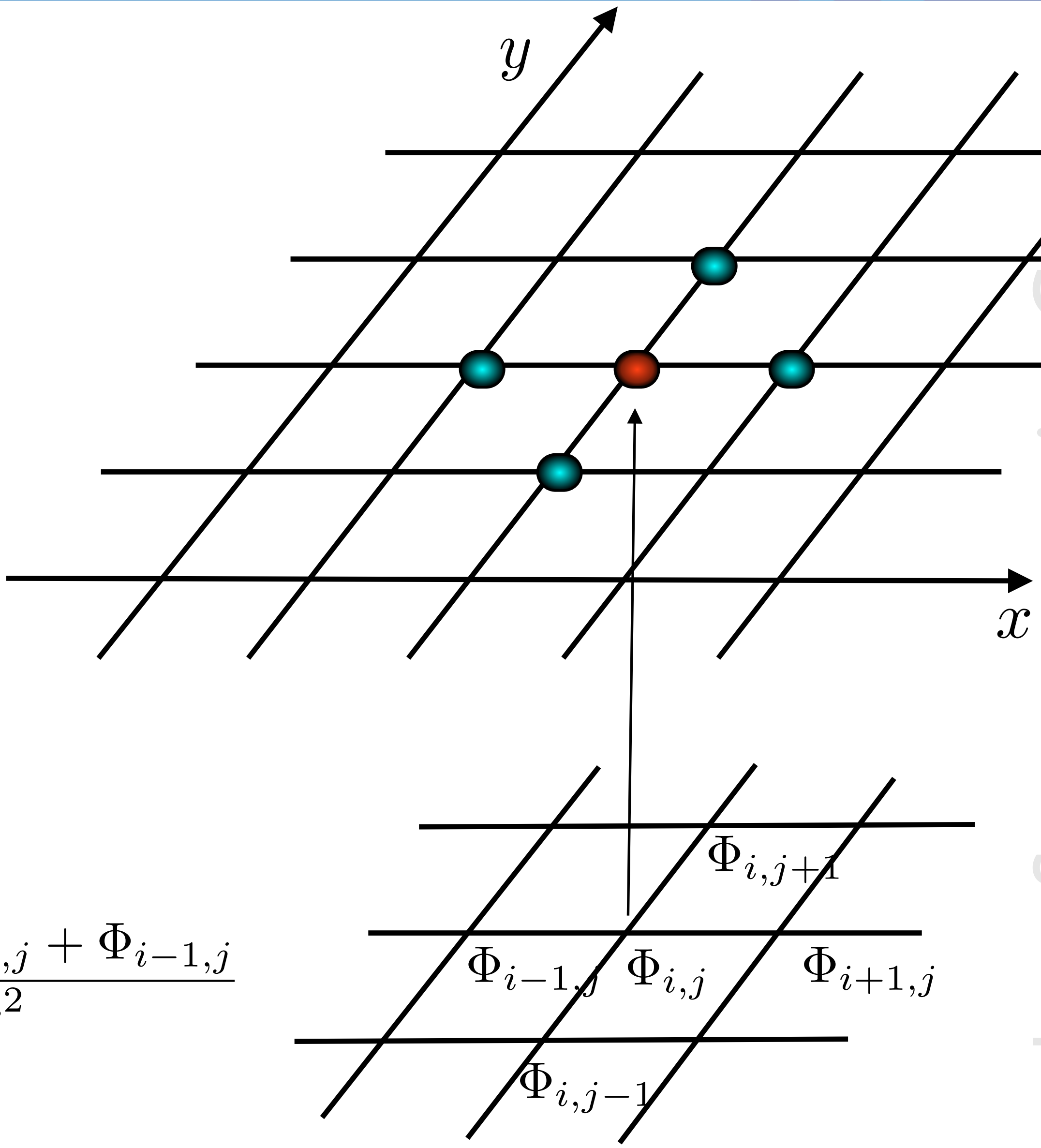
More regularity assumptions required

A form that exposes the deepest features of the problem

$$\frac{\partial q^{\alpha}}{\partial t} + A_i^{\alpha\beta}\frac{\partial q^{\beta}}{\partial x_i} = 0$$

Discretising the System

First strategy:
Finite Differences



Suppose convective and diffusive fluxes:

$$\frac{\Delta\Phi_{i,j}}{\Delta x} = \frac{\Phi_{i+1,j} - \Phi_{i,j}}{\Delta x}$$

$$\frac{\Delta}{\Delta x}\left(\frac{\Delta\Phi_{i,j}}{\Delta x}\right) = \frac{\Phi_{i+1,j} - 2\Phi_{i,j} + \Phi_{i-1,j}}{\Delta x^2}$$

Discretising the System

Finite Differences

Very intuitive

Simple coding

Very efficient and well suited for translation to a computer code

Structured meshes

Problems with Neuman boundary conditions (but solvable, if you pay the price)

Lack of scalability for high order schemes

Humble numerics: stabilisation, boundary conditions, adaptivity...

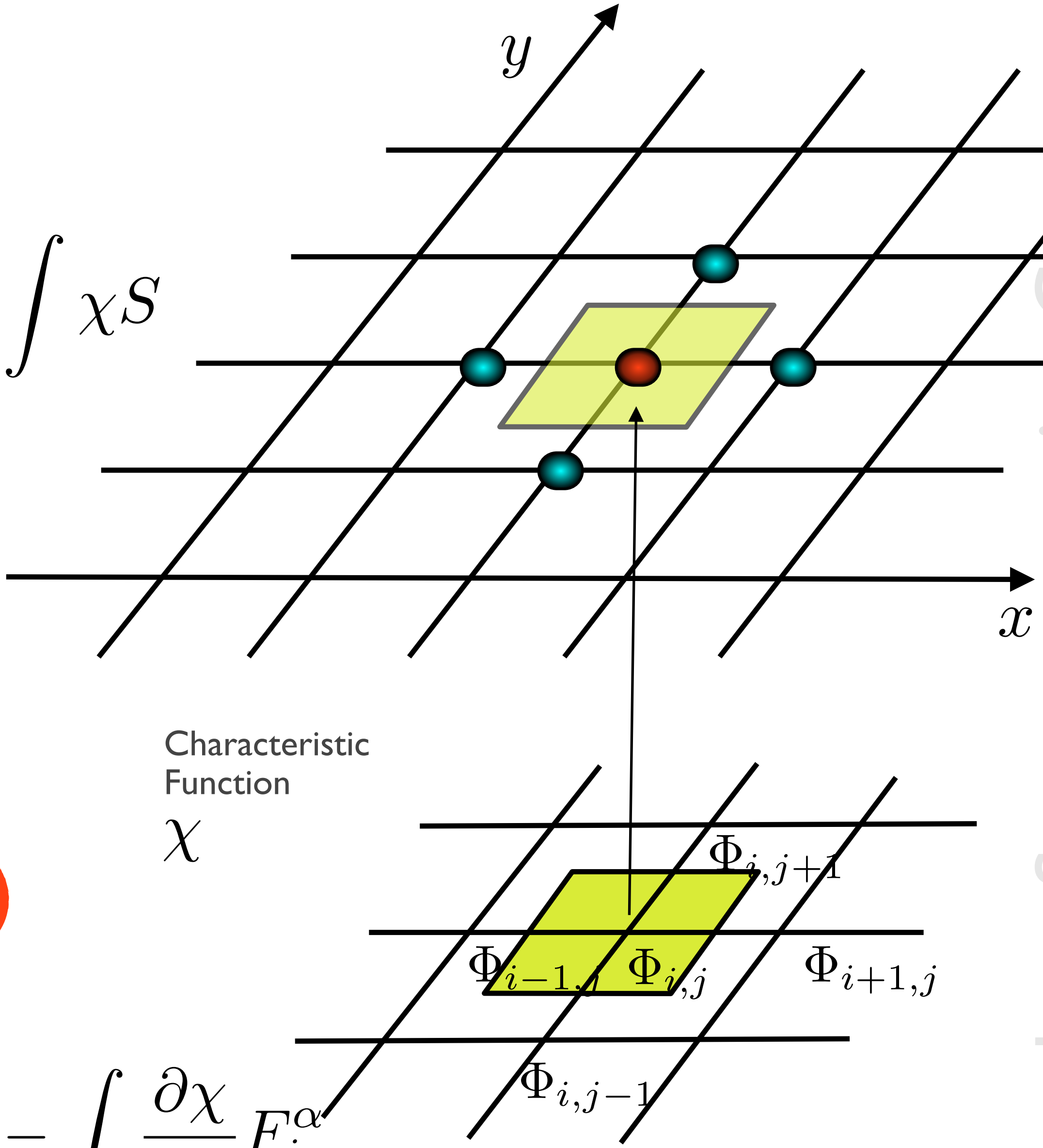
Discretising the System

Second strategy:
Finite Volumes

$$\int \chi \frac{\partial \Phi^\alpha}{\partial t} + \int \chi \frac{\partial F_i^\alpha}{\partial x_i} = \int \chi S$$

Divergence Gauss Theorem

$$-\oint \chi F_i^\alpha n_i$$



$$\int \chi \frac{\partial F_i^\alpha}{\partial x_i} = \int \frac{\partial}{\partial x_i}\left(\chi F_i^\alpha\right) - \int \frac{\partial \chi}{\partial x_i} F_i^\alpha$$

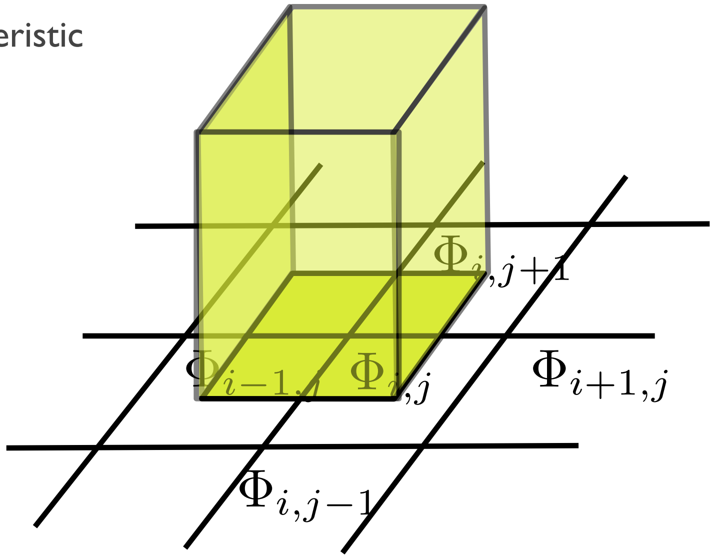
Discretising the System

Second strategy:
Finite Volumes

$$-\oint \chi F_i^{\alpha} n_i$$

Characteristic Function

χ



The characteristic function is a "filter" that focuses the equation only in the filter's support.

When you project, the rest of the domain disappears.

Is like the IF, but filtered on small cells.

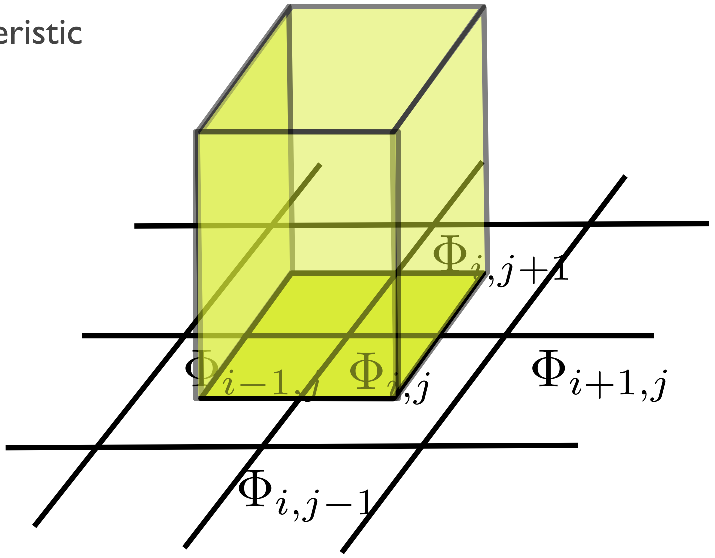
You only have to compute the fluxes through the limits of the cell:
the numerical fluxes

Discretising the System

Second strategy:
Finite Volumes

Characteristic
Function
χ

$$-\oint \chi F_i^\alpha n_i$$



Numerical fluxes

The cells' limits are discretised in faces or edges following the space discretisation.

The fluxes on faces are function of the values of the unknown at both sides of the face.

There are many many many ways of constructing these fluxes

The scheme's order depends on how the fluxes are computed and how far should we go to get the information required

(discussed below…)

Discretising the System

Finite Volumes

Intuitive, specially for Physicists

More complex codification

Very robust, with large story of success and theoretical developments

Unstructured meshes

Convective fluxes, ok. Diffusive fluxes, more bricolage is needed (but doable)

Lack of scalability for high order schemes

Sophisticate numerics: stabilisation, boundary conditions, adaptivity...

Long tradition, particularly for CFD compressible flows

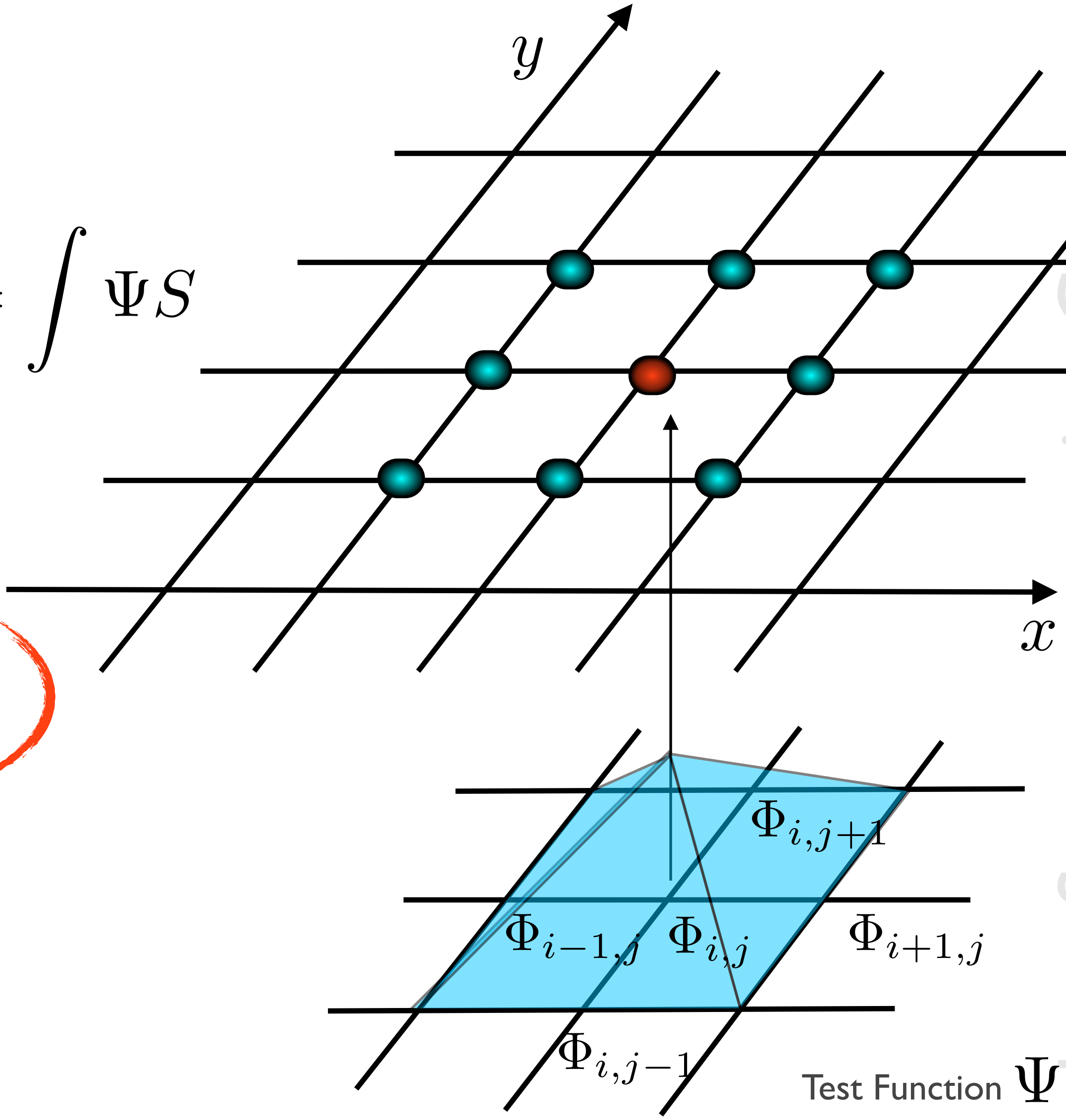
Discretising the System

Third strategy:
Finite Elements

$$\int \Psi \frac{\partial \Phi^\alpha}{\partial t} + \int \Psi \frac{\partial F_i^\alpha}{\partial x_i} = \int \Psi S$$

Divergence Gauss Theorem

$$-\int \frac{\partial \Psi}{\partial x_i} F_i^\alpha$$



$$\int \Psi \frac{\partial F_i^\alpha}{\partial x_i} = \int \frac{\partial}{\partial x_i}\left(\Psi F_i^\alpha\right) - \int \frac{\partial \Psi}{\partial x_i} F_i^\alpha$$

Discretising the System

Third strategy:
Finite Elements

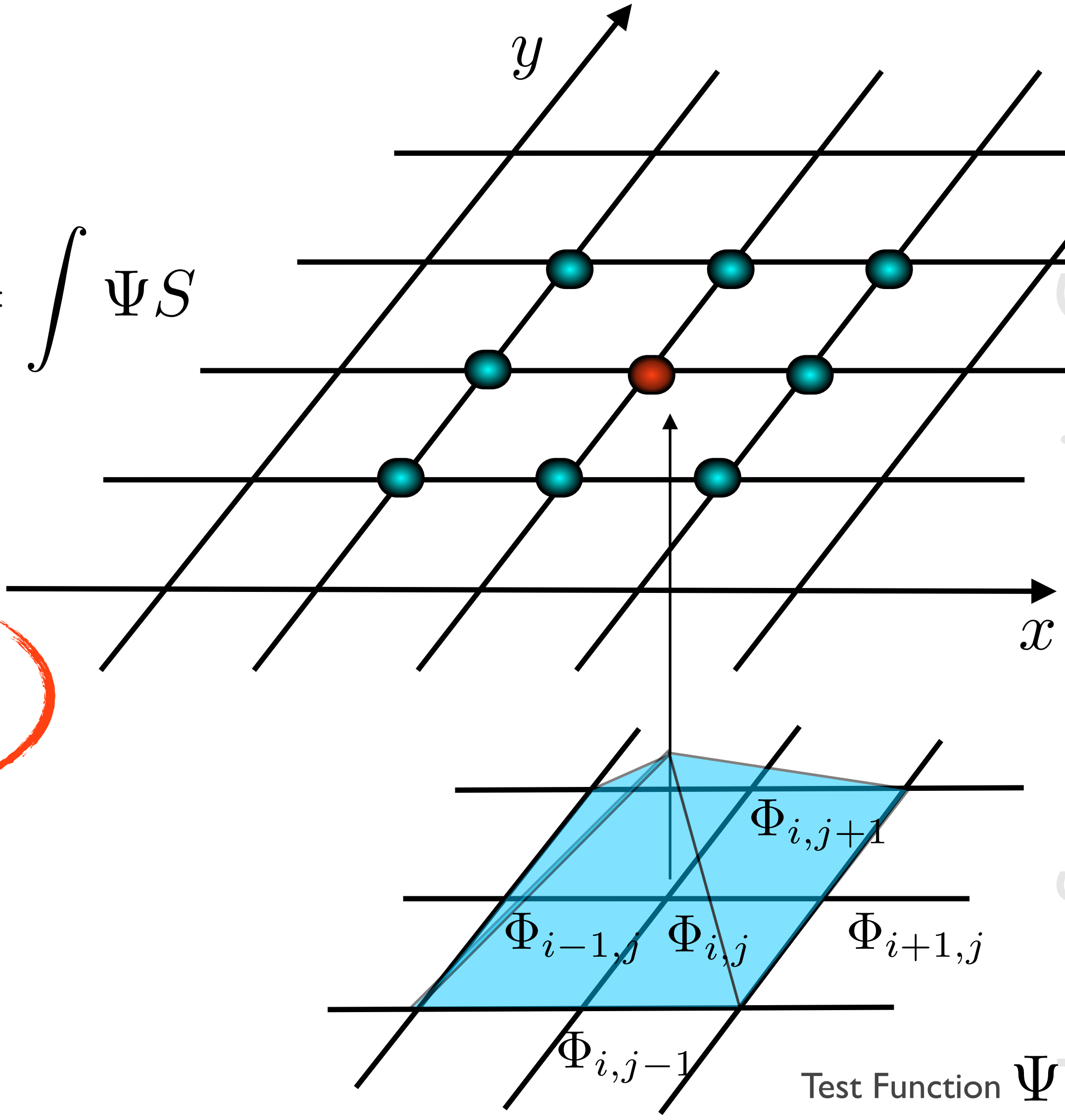
$$\int \Psi \frac{\partial \Phi^\alpha}{\partial t} + \int \Psi \frac{\partial F_i^\alpha}{\partial x_i} = \int \Psi S$$

Divergence Gauss Theorem

$$-\int \frac{\partial \Psi}{\partial x_i} F_i^\alpha$$

Recall that for FV:

$$-\oint \chi F_i^\alpha n_i$$

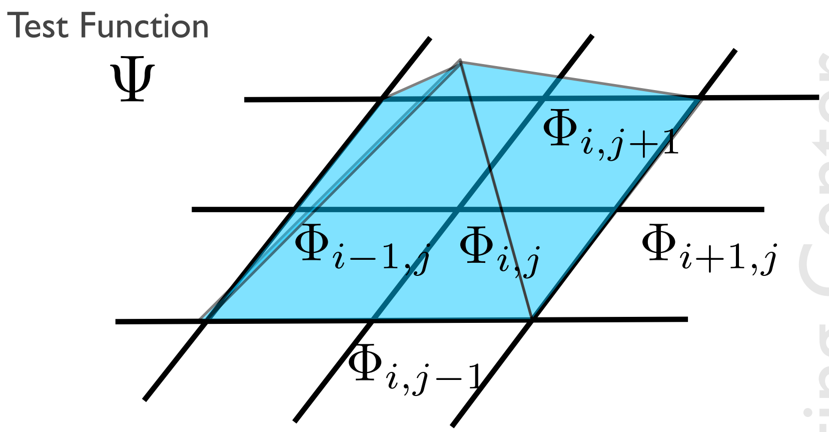


$$\int \Psi \frac{\partial F_i^\alpha}{\partial x_i} = \int \frac{\partial}{\partial x_i}\left(\Psi F_i^\alpha\right) - \int \frac{\partial \Psi}{\partial x_i} F_i^\alpha$$

Discretising the System

Third strategy:
Finite Elements

$$-\int \frac{\partial \Psi}{\partial x_i} F_i^\alpha$$



As in FV, the test function is a “filter” that focuses the equation only in the filter’s support.

When you project, the rest of the domain disappears.

It is like the IF, but filtered on small cells, the elements.

You only have to compute the fluxes through the limits of the cell or pass the derivatives (convolution) to the filter

The integrals are computed **numerically** as function of the values at the nodes of the elements

How to increase the order (see below)

Discretising the System

Finite Elements

Not very intuitive

Apparent complex codification, but not that much...

Profound mathematical background

Unstructured meshes, hybrid meshes

High order schemes naturally implemented

Good scalability

Not necessary expensive: must be programmed with care

Very sophisticate numerics: stabilisation, boundary conditions, adaptivity...

Long tradition in all fields of PDEs

Discretising the System

Finite Elements and the Navier-Stokes equations

Finite Elements and the Navier-Stokes equations

$$\frac{\partial U_j}{\partial t} + \frac{\partial}{\partial x_i}(u_i U_j) + \frac{\partial}{\partial x_i}(\delta_{ij} p - \tau_{ij}) + \rho g_j = 0$$

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_i}(U_i) = 0$$

$$\frac{\partial E}{\partial t} + \frac{\partial}{\partial x_i}(u_i E) + \frac{\partial}{\partial x_i}(u_i p - k\frac{\partial T}{\partial x_i} - \tau_{ij} u_j) + \rho(u_i g_i + r) = 0$$

$p = \rho R T$ is the ideal gas law

$U_i = \rho u_i, E = \rho(C_v T + u^2/2)$ are the momentum and the total energy

$\tau_{ij} = \mu(\frac{\partial u_j}{\partial x_i} + \frac{\partial u_i}{\partial x_j} - \frac{2}{3}\delta_{ij}\frac{\partial u_i}{\partial x_i})$ is the viscous stress tensor

Finite Elements and the Navier-Stokes equations

$$\frac{\partial U_j}{\partial t} + \frac{\partial}{\partial x_i}(u_i U_j) + \frac{\partial}{\partial x_i}(\delta_{ij} p - \tau_{ij}) + \rho g_j = 0$$

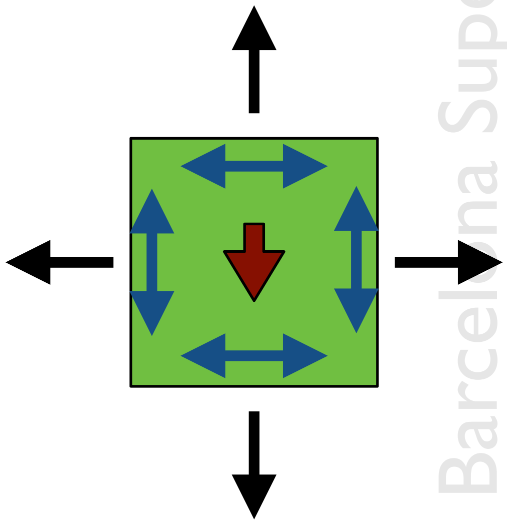
Momentum convection

Pressure forces

Viscous forces

Gravity forces

$$\tau_{ij} = \mu(\frac{\partial u_j}{\partial x_i} + \frac{\partial u_i}{\partial x_j} - \frac{2}{3}\delta_{ij}\frac{\partial u_i}{\partial x_i})$$



Finite Elements and the Navier-Stokes equations

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_i}(U_i) = 0$$

$$\frac{\partial \rho}{\partial t} + u_i \frac{\partial \rho}{\partial x_i} + \rho \underbrace{\frac{\partial u_i}{\partial x_i}}_{} = 0$$

Mass in a volume of fluid changes by crossing boundaries and by compression / expansion

Compressibility

Compressibility relates pressure, temperature and density through the equation of state

$p = \rho R T$ is an example

$$c = \sqrt{\frac{\partial p}{\partial \rho}|_s} = \sqrt{\gamma p/\rho}$$ speed of sound

Finite Elements and the Navier-Stokes equations

$$\frac{\partial E}{\partial t} + \frac{\partial}{\partial x_i}(u_i E) + \frac{\partial}{\partial x_i}\left(u_i p - k\frac{\partial T}{\partial x_i} - \tau_{ij} u_j\right) + \rho(u_i g_i + r) = 0$$

Energy convection	Pressure work	Thermal diffusion	Joule effect	Gravity work	Heat sources

$E = \rho(C_v T + u^2/2)$ is the total energy

Alternative, heat transport equation

$$\frac{\partial T}{\partial t} + u_i \frac{\partial T}{\partial x_i} + \frac{1}{C_v \rho}\left(p\frac{\partial u_k}{\partial x_k} - \tau_{ij}\frac{\partial u_i}{\partial x_j} - k\frac{\partial^2 T}{\partial x_i \partial x_i}\right) = 0$$

Finite Elements and the Navier-Stokes equations

Momentum: forces balance

Density: continuity equation

Energy: energy balance

$$\frac{\partial U^\alpha}{\partial t} = \frac{\partial F_i^\alpha}{\partial x_i} + S$$

$$U^\alpha = (U_j, \rho, E)$$

Three main regimes, determined by two non-dimensional numbers:

- ✓ Compressible and Incompressible: Mach number (M)
- ✓ Viscous and Inviscid: Reynolds number (Re)
- ✓ Laminar and Turbulent (Re)

Mach number:

The state equation couples thermodynamics with mechanics
The Mach number represents the ratio between the "mechanic speed" and the "thermodynamic speed"

When **strictly** incompressible:

- ✓ Energy decouples from mechanics, no transfer from internal to kinetic
- ✓ Speed of sound becomes infinite
- ✓ Pressure is only mechanical
- ✓ Temperature (i.e. heat) is a transported scalar
- ✓ To be deeper analyzed below…

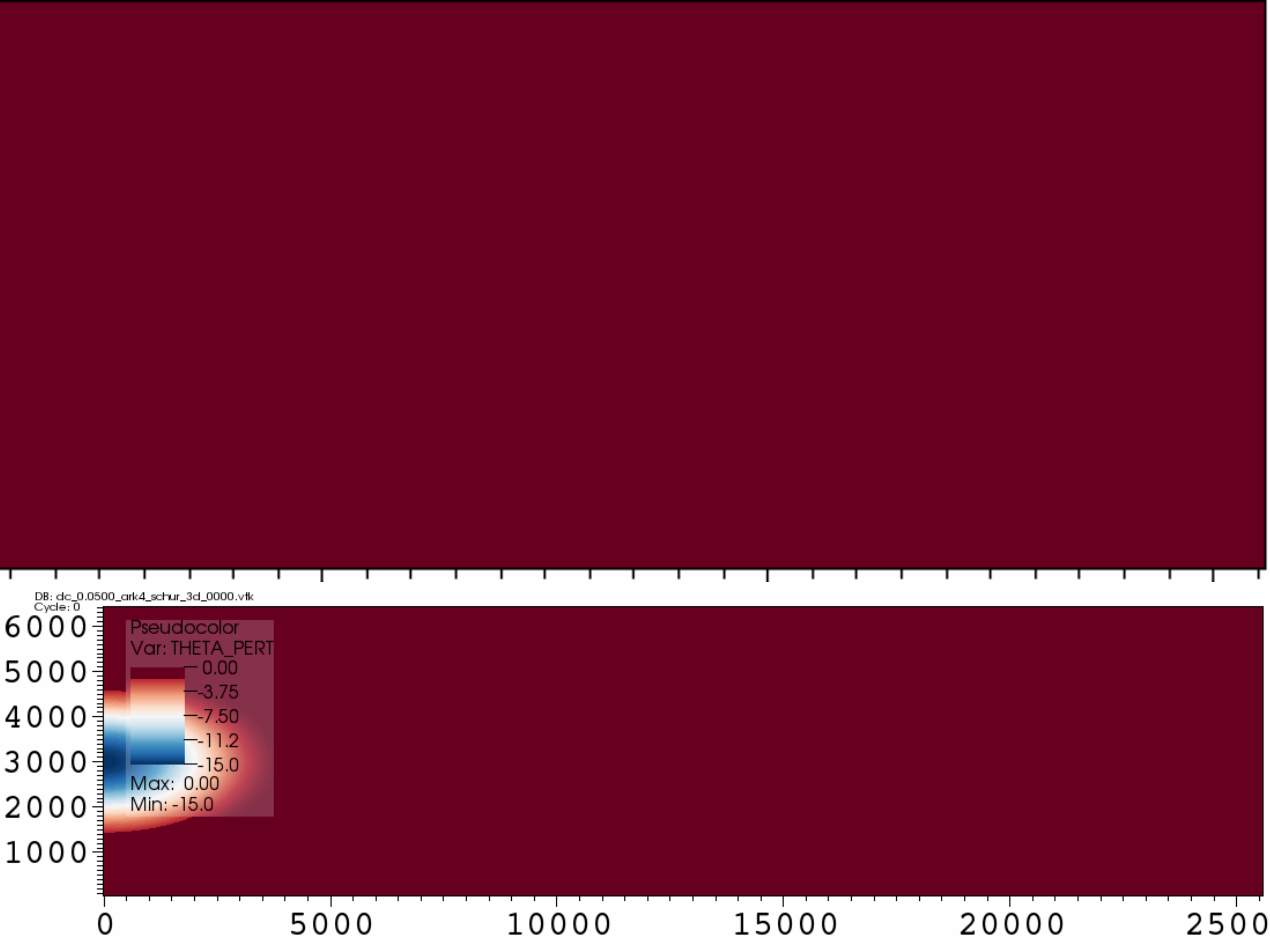
$$\frac{\partial u_i}{\partial x_i} = 0$$

The state equation couples thermodynamics with mechanics

When **slightly** compressible:

- ✓ Very low transfer from internal to kinetic by any meaning
- ✓ Speed of sound becomes very large, very low Mach numbers
- ✓ Pressure is mostly mechanical
- ✓ System matrix becomes very ill-conditioned

$$A_i^{\alpha\beta} \frac{\partial U^\beta}{\partial x_i}$$


DB: dc_0.0500_ark4_schur_3d_0000.vtk
Cycle: 0
Pseudocolor
Var: THETA_PERT
0.00
-3.75
-7.50
-11.2
-15.0
Max: 0.00
Min: -15.0
6000
5000
4000
3000
2000
1000
0
5000
10000
15000
20000
2500

Finite Elements and the Navier-Stokes equations

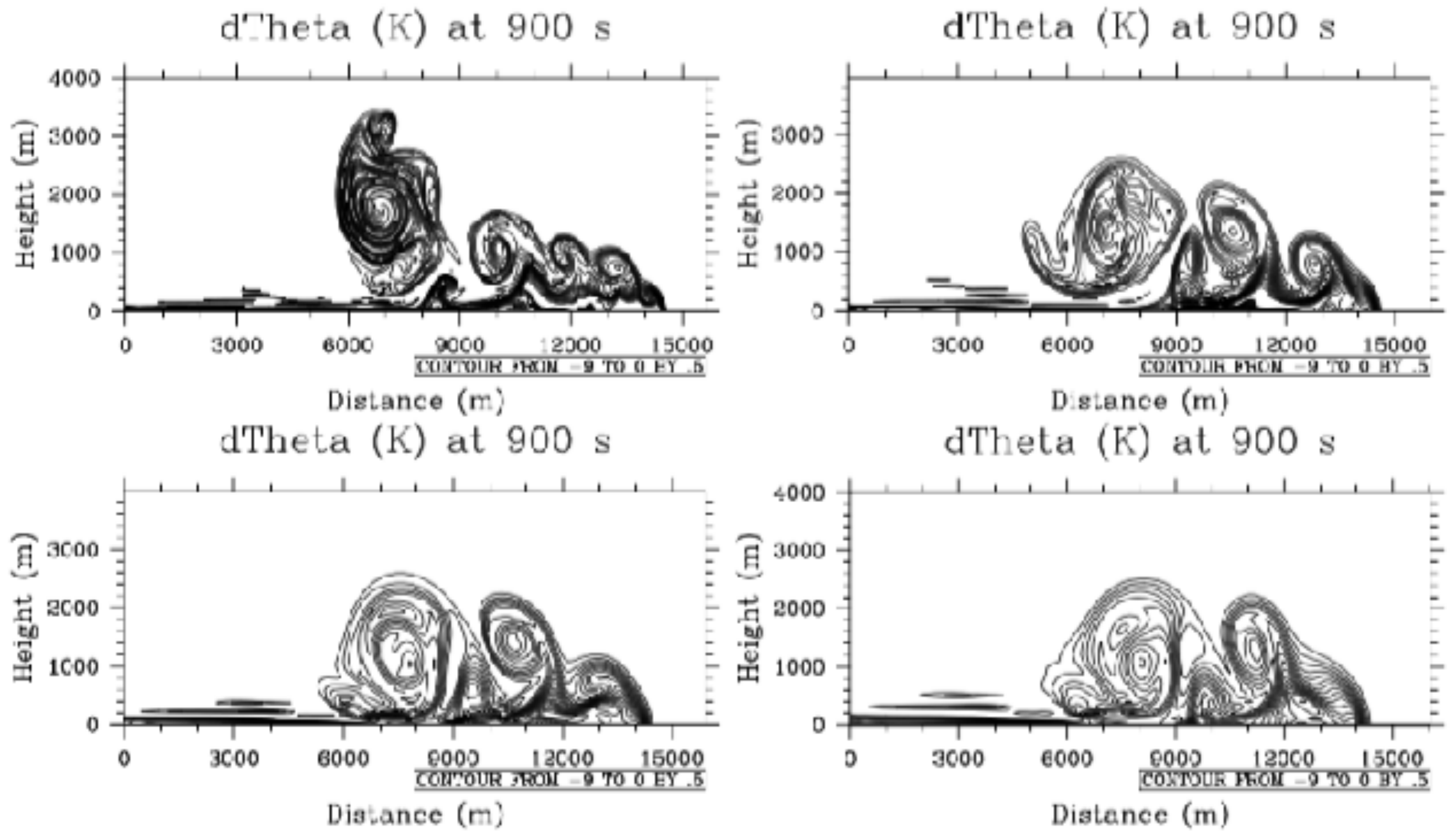


FIGURE 3.3: Density current. Contours of θ' at $T = 900\,s$ for four different grid resolutions. Top row: $\Delta x = \Delta z = 25\,m$ and $\Delta x = \Delta z = 50\,m$ resolution. Bottom row: $\Delta x = \Delta z = 75\,m$ and $\Delta x = \Delta z = 100\,m$ resolution.

Finite Elements and the Navier-Stokes equations

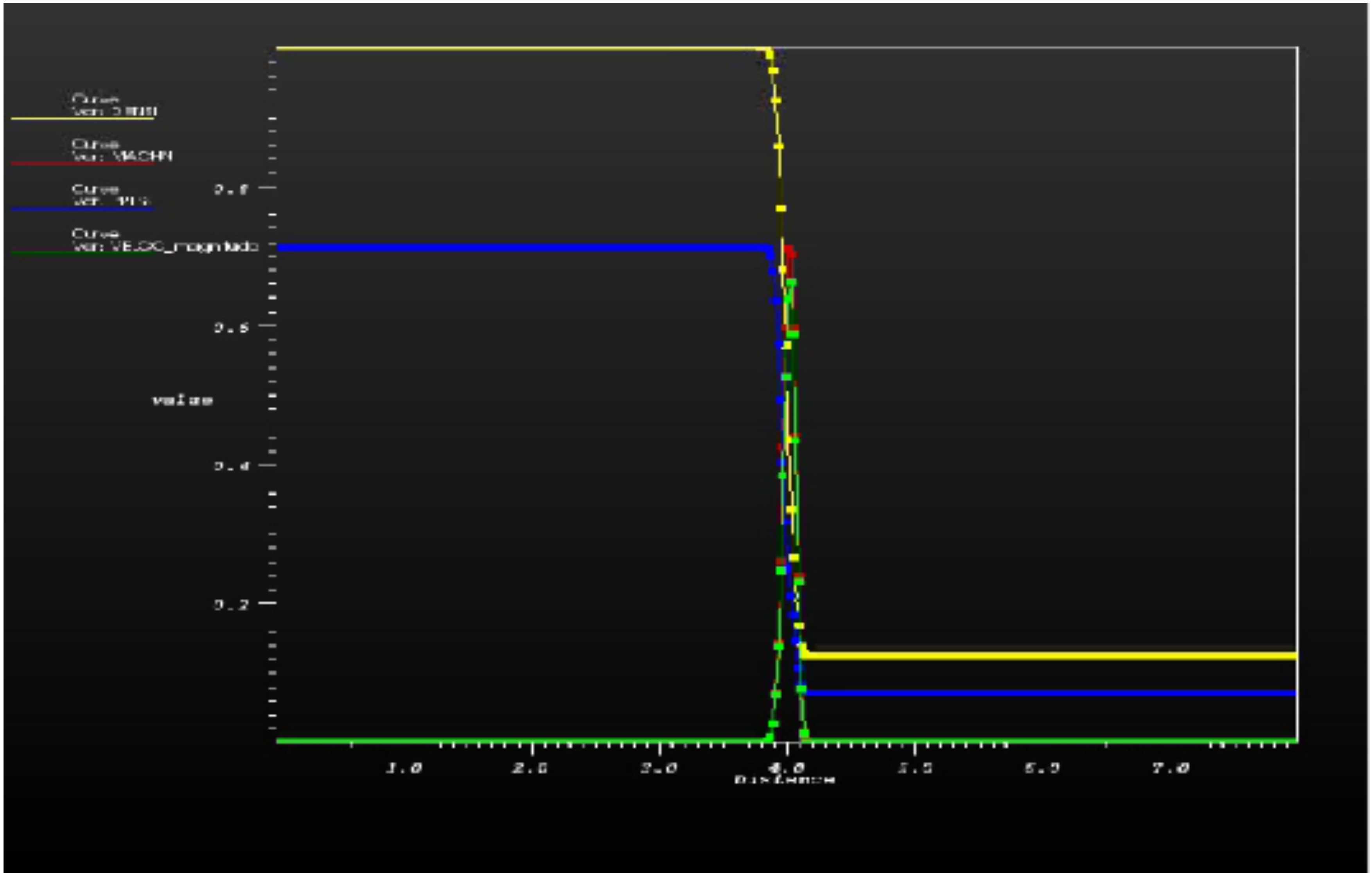
The state equation couples thermodynamics with mechanics

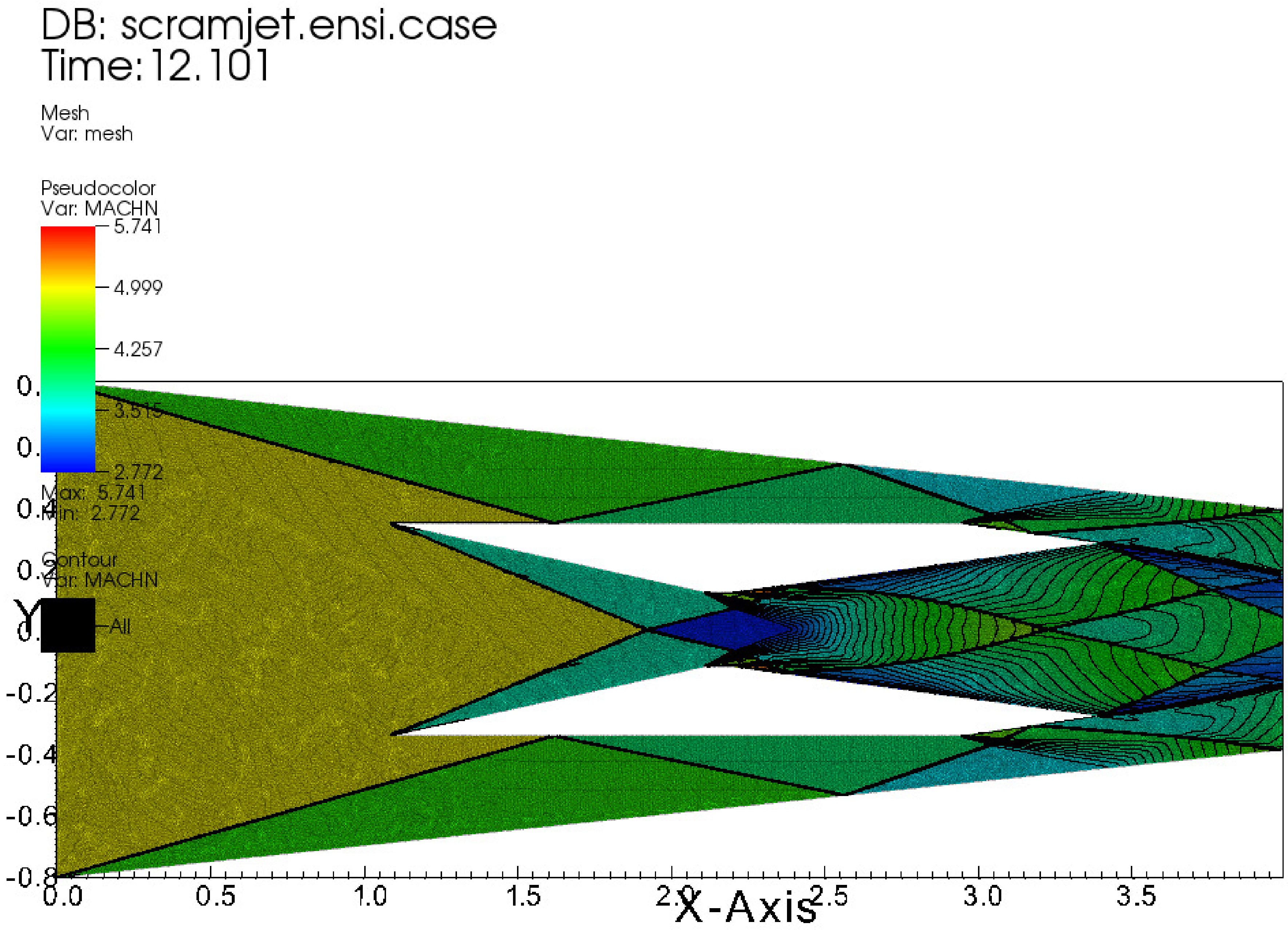
When compressible:

- ✓ Internal and kinetic heavily interchanged
- ✓ Speed of sound comparable to the speed of fluids
- ✓ Pressure works related to internal energy (temperature)
- ✓ Appearance of shock waves
- ✓ Low-Mach modelling could be a good compromise

$$A_i^{\alpha\beta} \frac{\partial U^\beta}{\partial x_i}$$

Finite Elements and the Navier-Stokes equations




DB: scramjet.ensi.case
Time:12.101
Mesh
Var: mesh
Pseudocolor
Var: MACHN
5.741
4.999
4.257
3.515
2.772
Max: 5.741
Min: 2.772
Contour
Var: MACHN
All
Y
0.8
0.6
0.4
0.2
0.0
-0.2
-0.4
-0.6
-0.8
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
X-Axis

Reynolds number:
As you will recall, viscous terms has a decisive effect
The Reynolds number represents the ratio between convection and viscous effects.

When viscosity is zero,

- ✓ Energy is not dissipated through heat
- ✓ Circulation is conserved, although vortices could appear (as seen above)
- ✓ Entropy is conserved, unless shocks appear

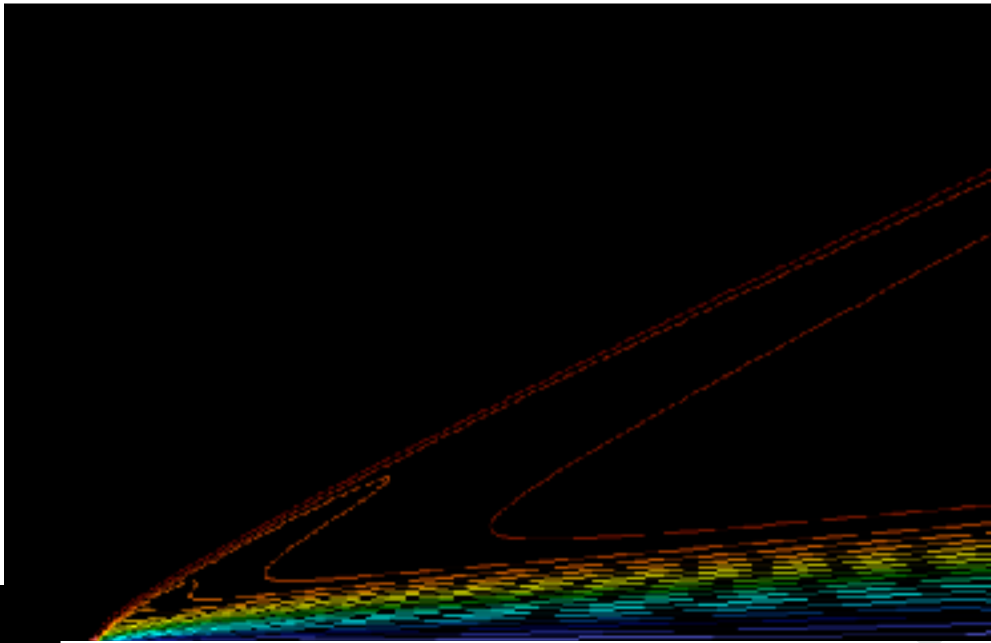
When viscosity is non-zero,

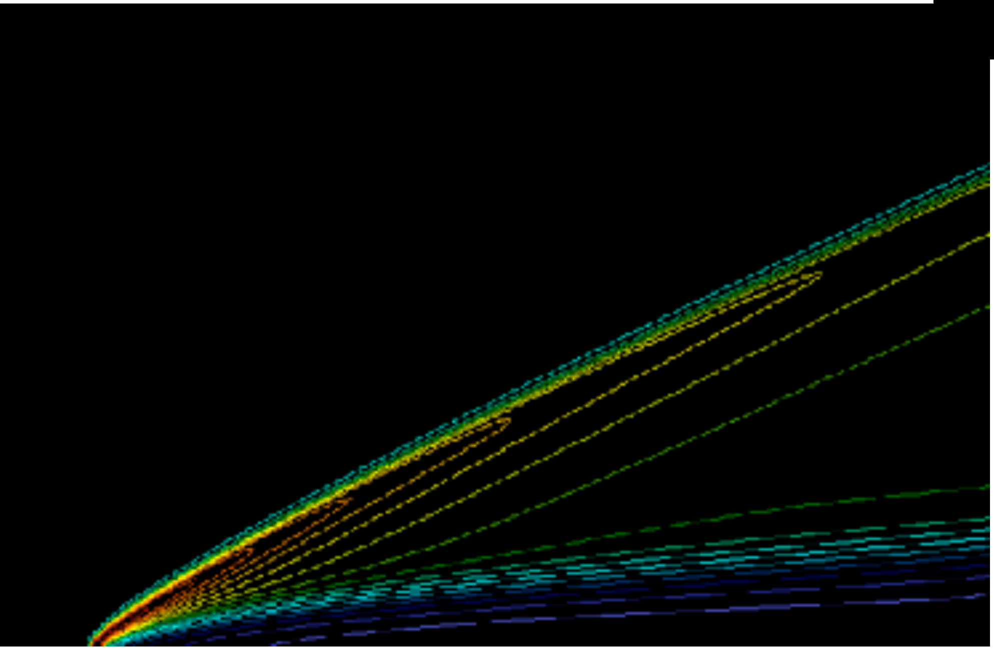
- ✓ Boundary layers appears
- ✓ Complex shock behaviours (lambda systems)

Finite Elements and the Navier-Stokes equations

Carter's problem:

Flow over a plate
Re = 1000, Ma= 3





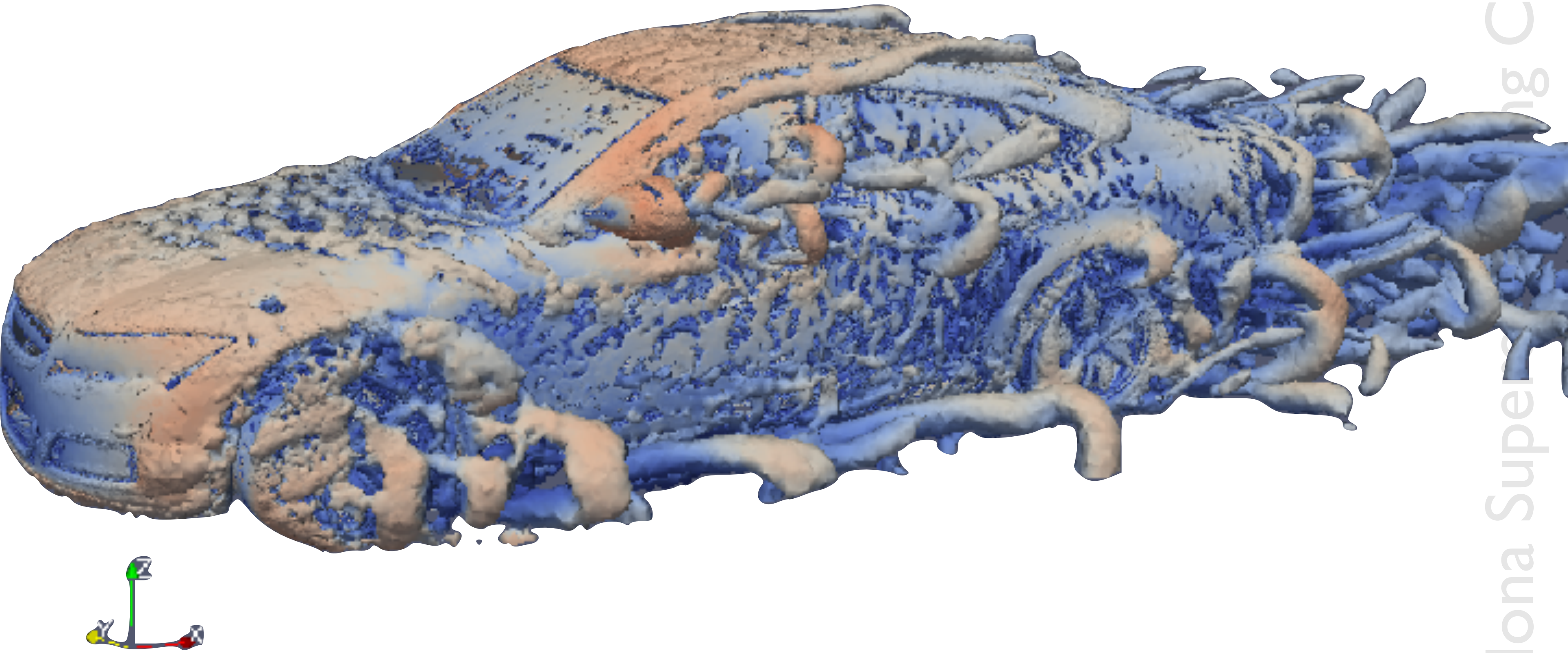
High Reynolds number:

The onset of turbulence and modelling subscales

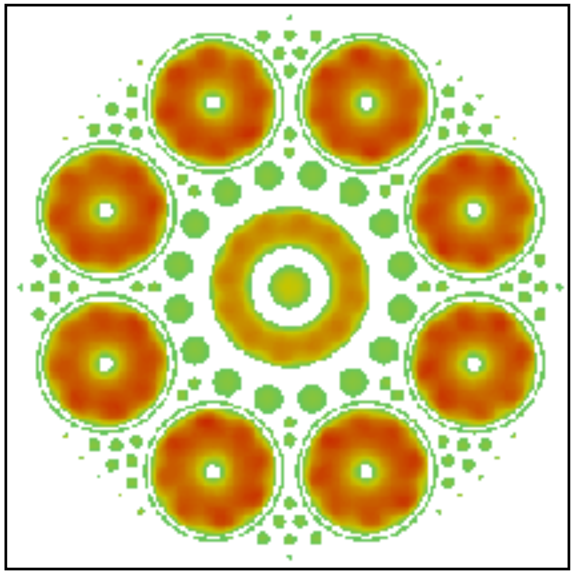
When Reynolds is high,

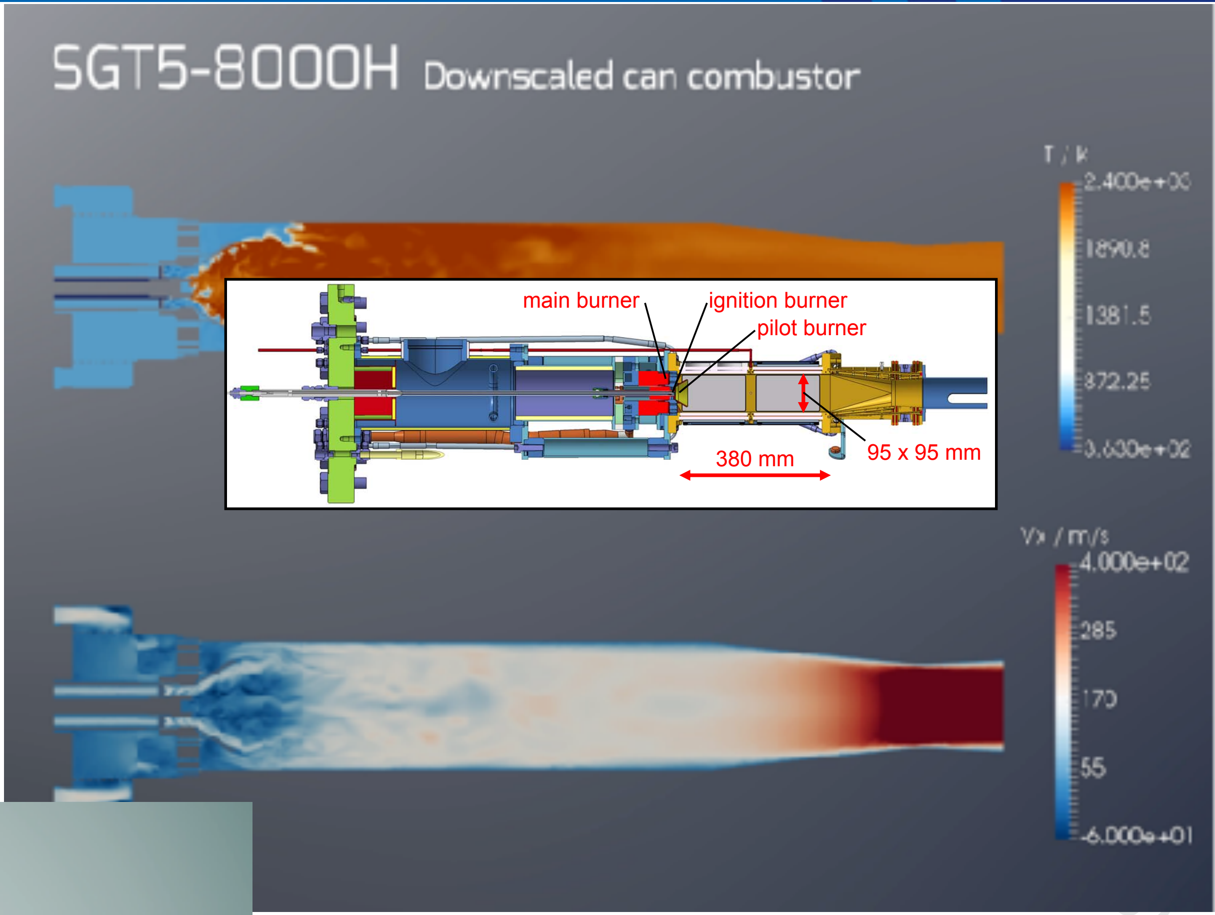
- ✓ Physical subscales cannot be simulated, so they are modelled
- ✓ DNS, simulating when possible
- ✓ RANS (over ensembles), LES (over space) and mixed modelling
- ✓ Compressible turbulence is even more difficult (for high compressibility)

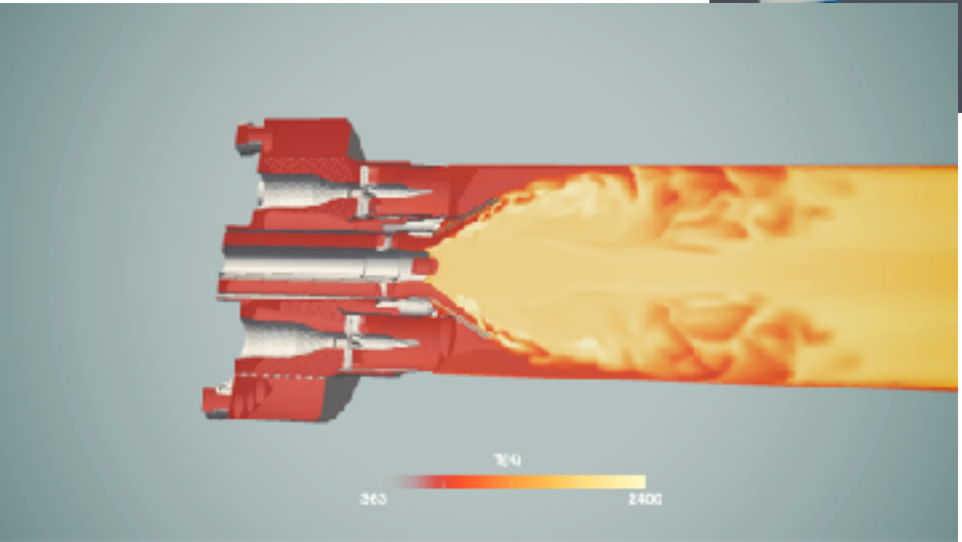
Finite Elements and the Navier-Stokes equations



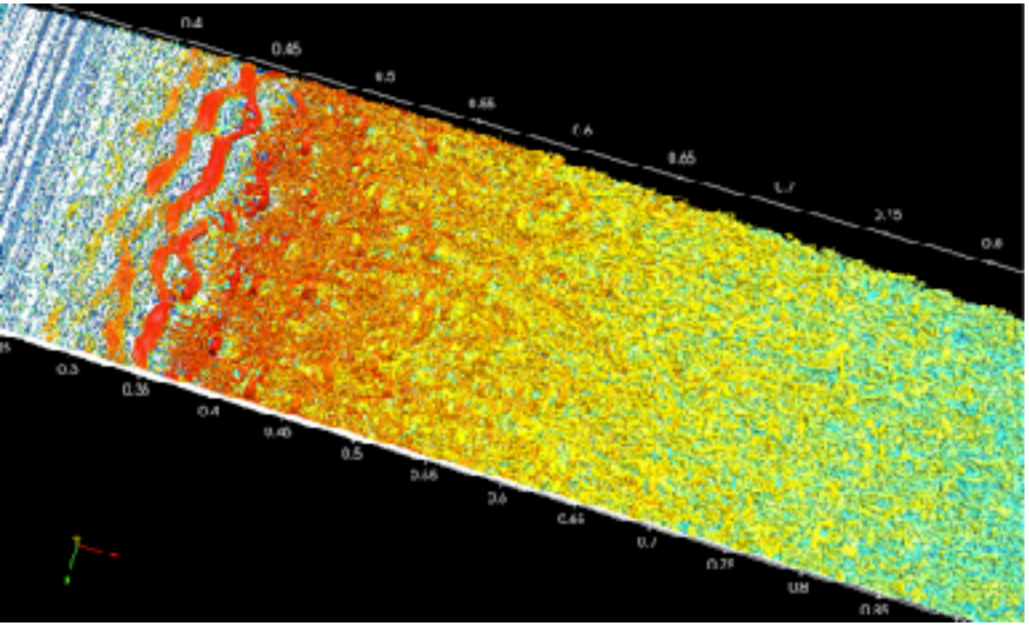
Finite Elements and the Navier-Stokes equations

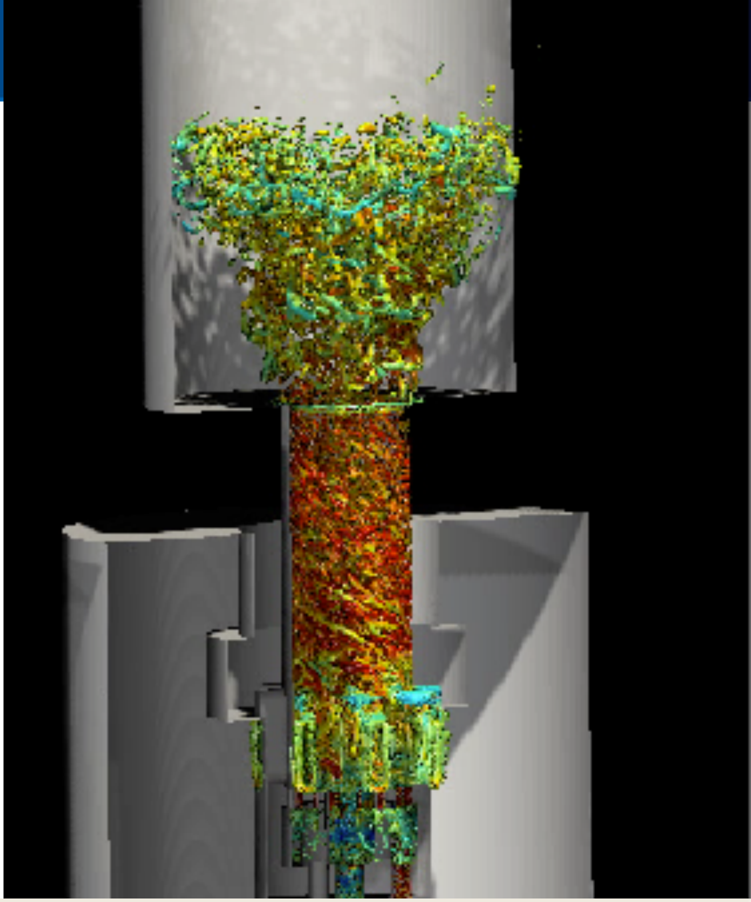


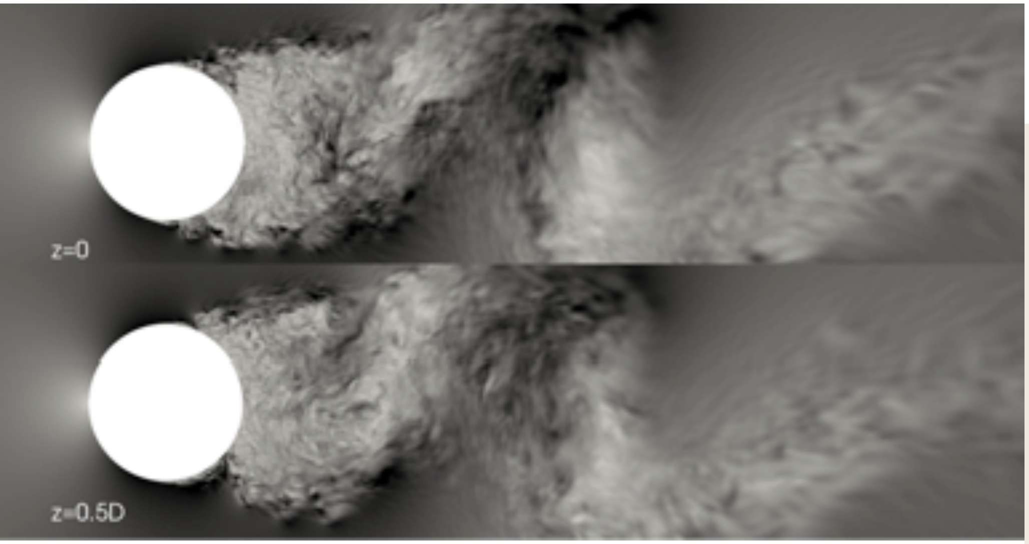


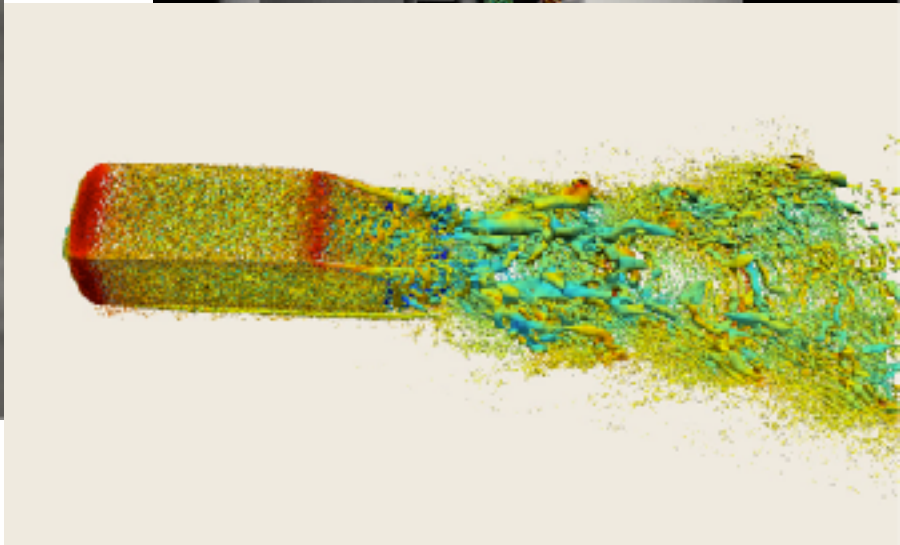


Finite Elements and the Navier-Stokes equations









Once discretized, our system looks like this:

$$\int \Psi \frac{\partial U^\alpha}{\partial t} = \int \Psi \frac{\partial F_i^\alpha}{\partial x_i} \quad \text{or} \quad \int \Psi \frac{\partial U^\alpha}{\partial t} = \int \Psi A_i^{\alpha\beta} \frac{\partial U^\beta}{\partial x_i}$$

$$\frac{\mathbf{M}}{\Delta t}\Delta\mathbf{U} + \mathbf{K}\mathbf{U}^* = 0$$

$$\Delta\mathbf{U} = \mathbf{U}^{n+1} - \mathbf{U}^n$$

* means **when** it is computed

* = n	is explicit
* = n+1	is implicit

M is the **mass** matrix
K is the **system** matrix

Discretising the System

The elementary system matrix is (the same can be said about the mass matrix):

$$K_e^{IJ} = K_e^{ab,\alpha\beta} \qquad M_e^{IJ} = M_e^{ab,\alpha\beta}$$

a, b label two different nodes

Elementary matrices size is:
ndofn x nnode

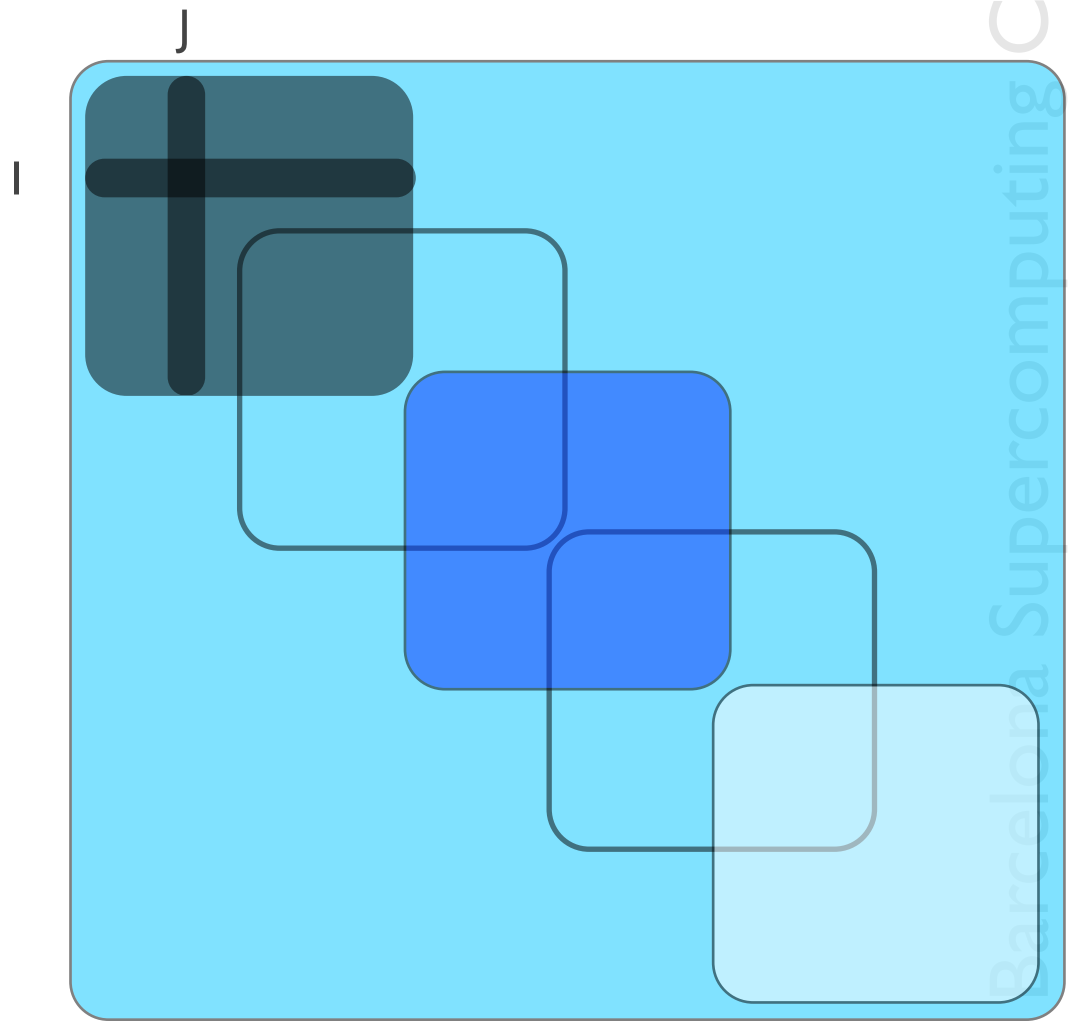
For instance, supposing tetrahedra:

Density for node 3 in element e is (3,4) or

J = 5*(3-1) + 4

X-momentum for node 2 in element e is (2,1) or

J = 5*(2-1) + 1



Discretising the System

The elementary system matrix is (the same can be said about the mass matrix):

$$K_e^{IJ} = K_e^{ab,\alpha\beta} \qquad M_e^{IJ} = M_e^{ab,\alpha\beta}$$

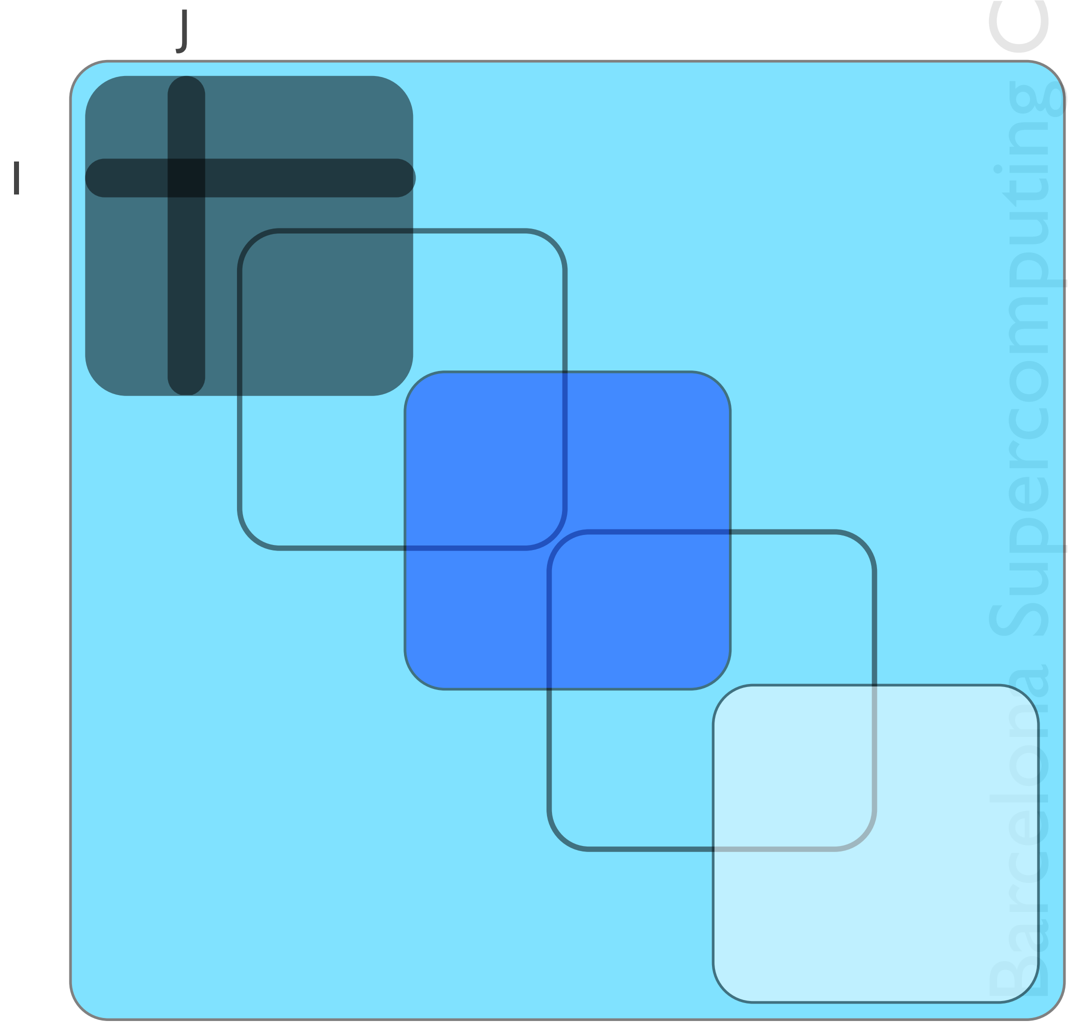
a, b label two different nodes

Elementary matrices size is:
ndofn x nnode

Once computed at each element, system or mass matrices are **assembled** in the global ones

$$K^{IJ} = \sum_e^{N_{\rm el}} K_e^{IJ}$$

$$M^{IJ} = \sum_e^{N_{\rm el}} M_e^{IJ}$$



Discretising the System

Finite Elements

Stabilization

Finite Elements
Stabilization

Typical FEM-based discretization use a residual weighted stabilization.

The idea is to maintain the order of the Galerkin scheme while adding a stabilization term that vanishes for sufficiently smooth solutions.

Variational Multiscale Stabilization (VMS) is the mother of all: SUPG, GLS, CG, …

Finite Elements
Stabilization

Typical FEM-based discretization use a residual weighted stabilization.

The idea is to maintain the order of the Galerkin scheme while adding a stabilization term that vanishes for sufficiently smooth solutions.

Variational Multiscale Stabilization (VMS) is the mother of all: SUPG, GLS, CG, …

$$\int \Psi \frac{\partial \Phi^\alpha}{\partial t} + \int \Psi A_i^{\alpha\beta} \frac{\partial \Phi}{\partial x_i}^\beta + \int \frac{\partial}{\partial x_i} (\Psi A_i^{*\alpha\beta}) \tilde{\Phi}^\beta = \int \Psi S$$

$$\int \Psi \frac{\partial \Phi^\alpha}{\partial t} + \int \Psi A_i^{\alpha\beta} \frac{\partial \Phi}{\partial x_i}^\beta + \underbrace{\int \frac{\partial \Psi}{\partial x_i} A_i^{*\alpha\beta} \tilde{\Phi}^\beta}_{\text{Stabilization term}} = \int \Psi S$$

$\tilde{\Phi}^\beta = \tau^{\beta\gamma} r^\gamma$ Subscale

Stabilization term

Discretising the System

Finite Elements
Stabilization

Typical FEM-based discretization use a residual weighted stabilization.

It looks complex, but it can be efficiently implemented
Very flexible and robust
Well-suited for multi-scale problems

As it depends on the residual, it vanishes if the equation is satified

(Moragues PhD Thesis, 2016)

Finite Elements and the Navier-Stokes equations

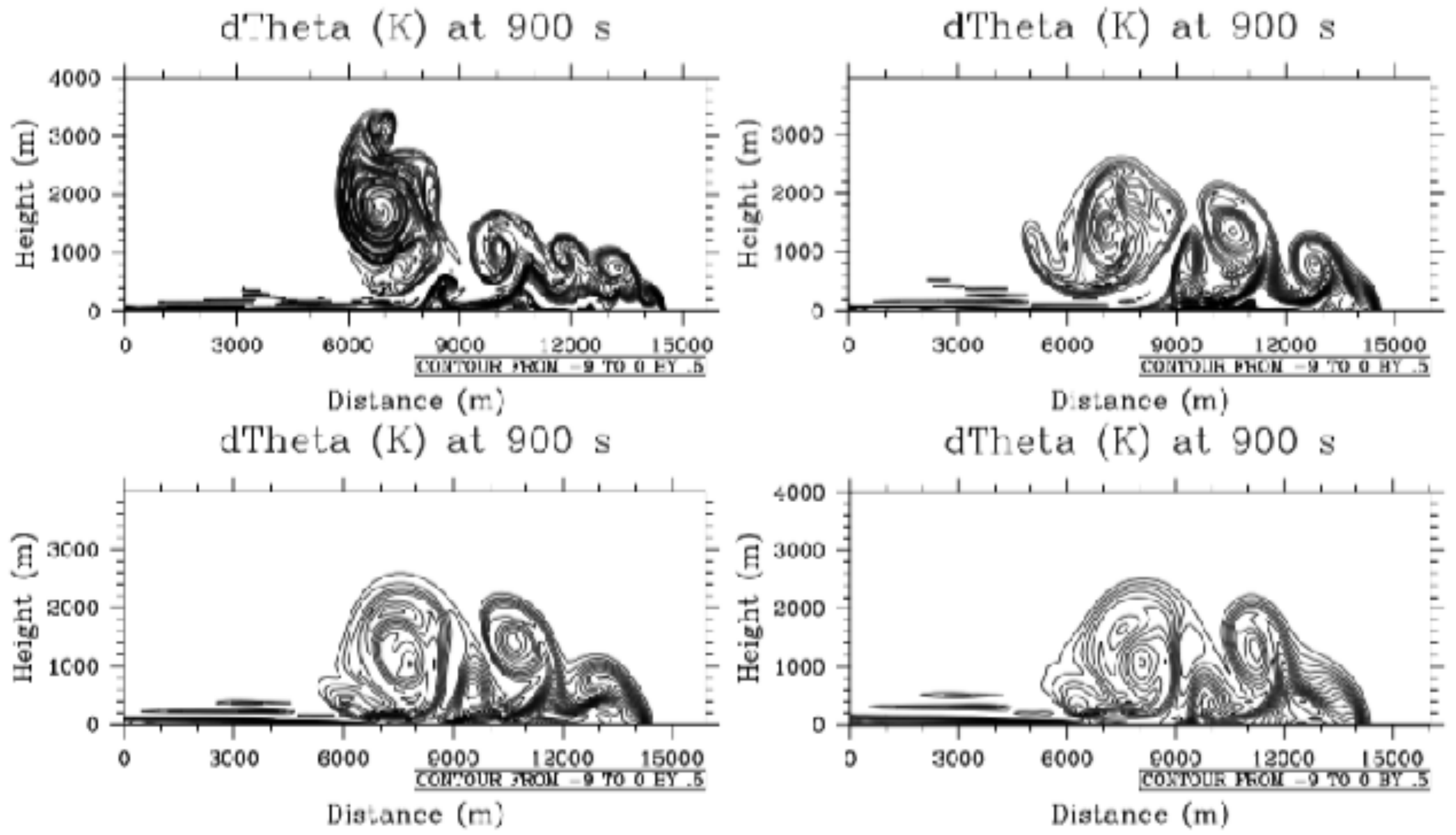
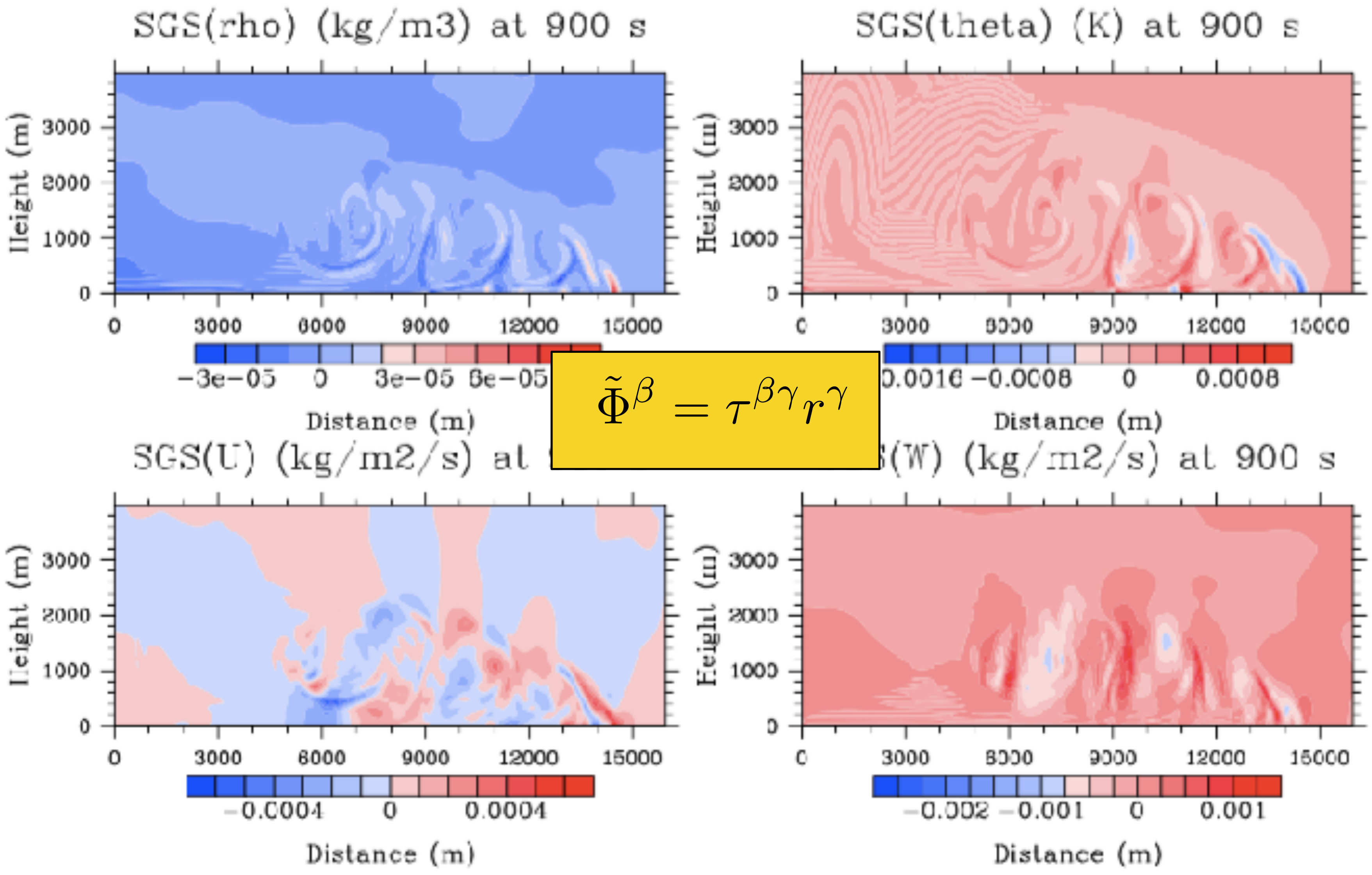


FIGURE 3.3: Density current. Contours of θ' at $T = 900\,s$ for four different grid resolutions. Top row: $\Delta x = \Delta z = 25\,m$ and $\Delta x = \Delta z = 50\,m$ resolution. Bottom row: $\Delta x = \Delta z = 75\,m$ and $\Delta x = \Delta z = 100\,m$ resolution.

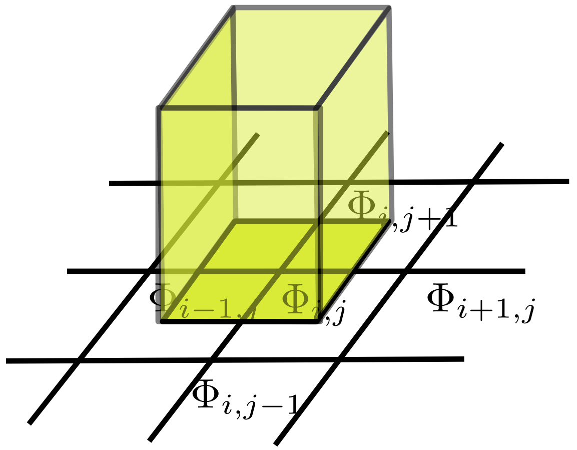
Finite Elements and the Navier-Stokes equations



$$\tilde{\Phi}^{\beta} = \tau^{\beta\gamma} r^{\gamma}$$

FIGURE 3.4: Density current. Filled contours of the normalized subscales at $T = 900\,s$ and $\Delta x = \Delta z = 50\,m$ resolution. Top row: $\tilde{\rho}/\rho_{max}$, $\tilde{\theta}/\theta_{max}$. Bottom row: $\tilde{U}/U_{max}$, $\tilde{W}/W_{max}$.

Discretising the System

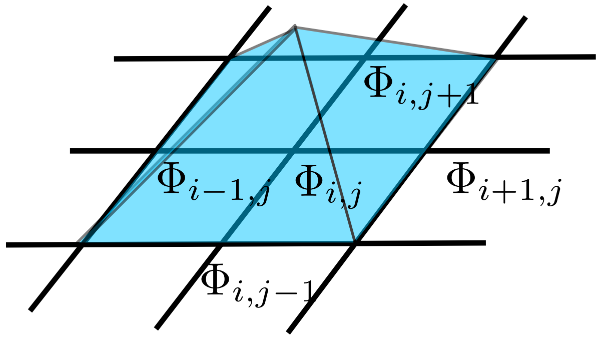


Unstructured meshes

Definition

How to construct the characteristic function or the test function?

How to increase the order?



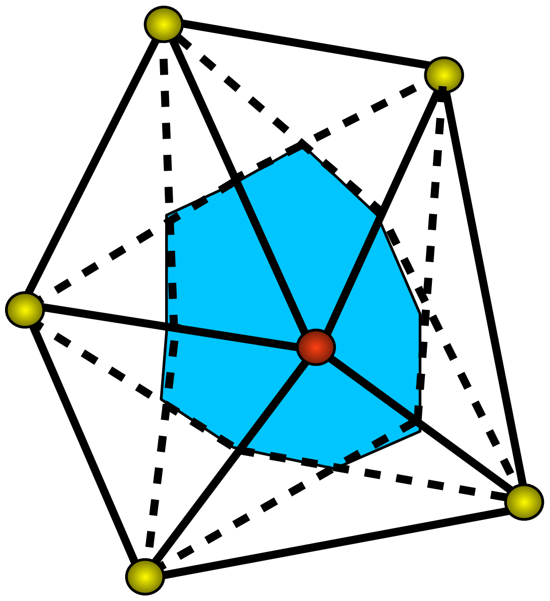
Discretising the System

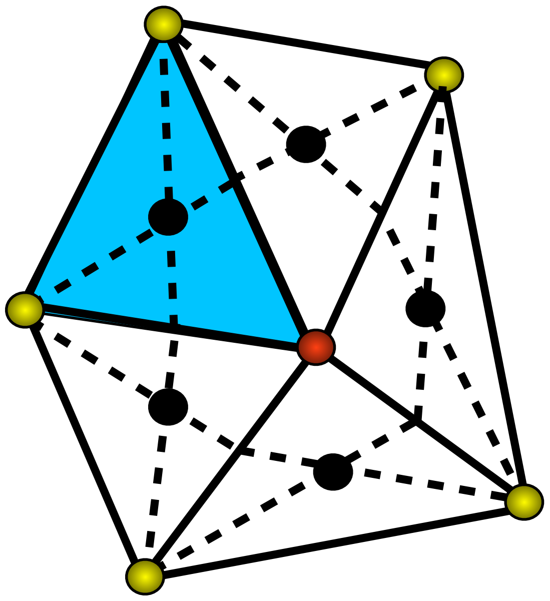
Unstructured meshes

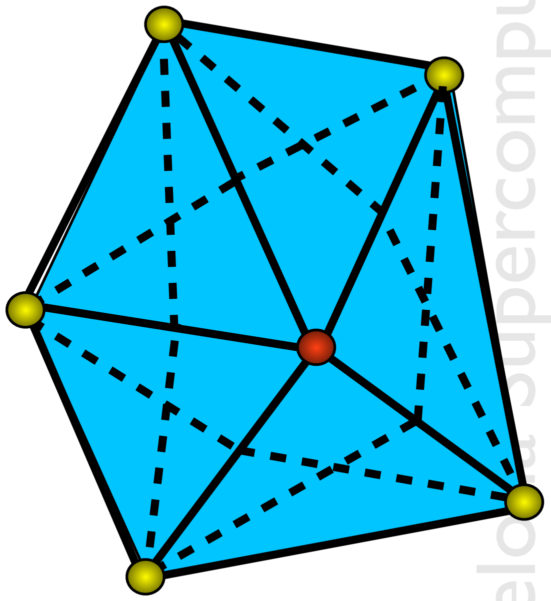
FV
Node centered

FV
Cell centered

FEM
Element based, but nodal unknowns







Discretising the System

Unstructured meshes

FV
Node centered

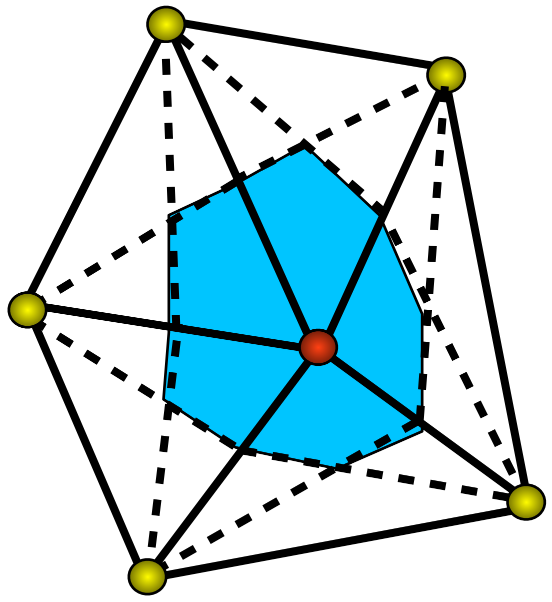
FEM
Element based, but nodal unknowns

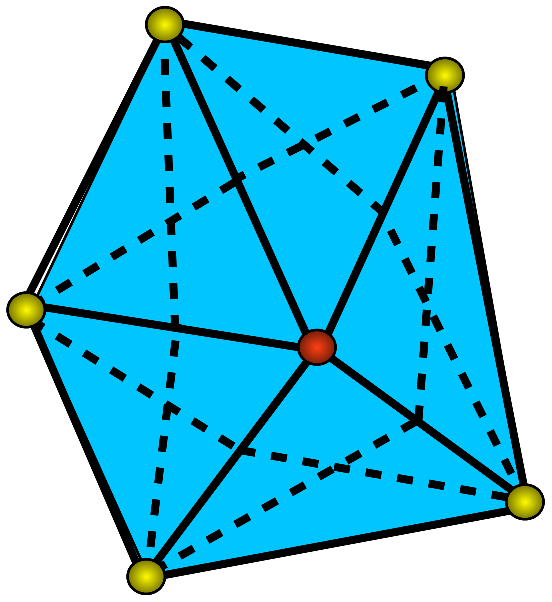
Share a very similar connectivity structure

Local connectivity can be high

(Highly) variable stencils

DoF on nodes

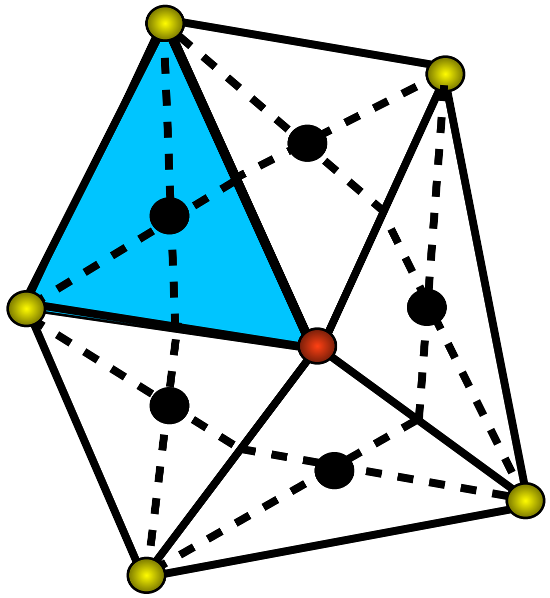




Discretising the System

Unstructured meshes

FV
Cell centered



Local connectivity is generally low

Stencils fixed

DoF on elements (usually more than nodes)

Discretising the System

Unstructured meshes

Increasing the order: Finite Volumes

Larger "friends" range

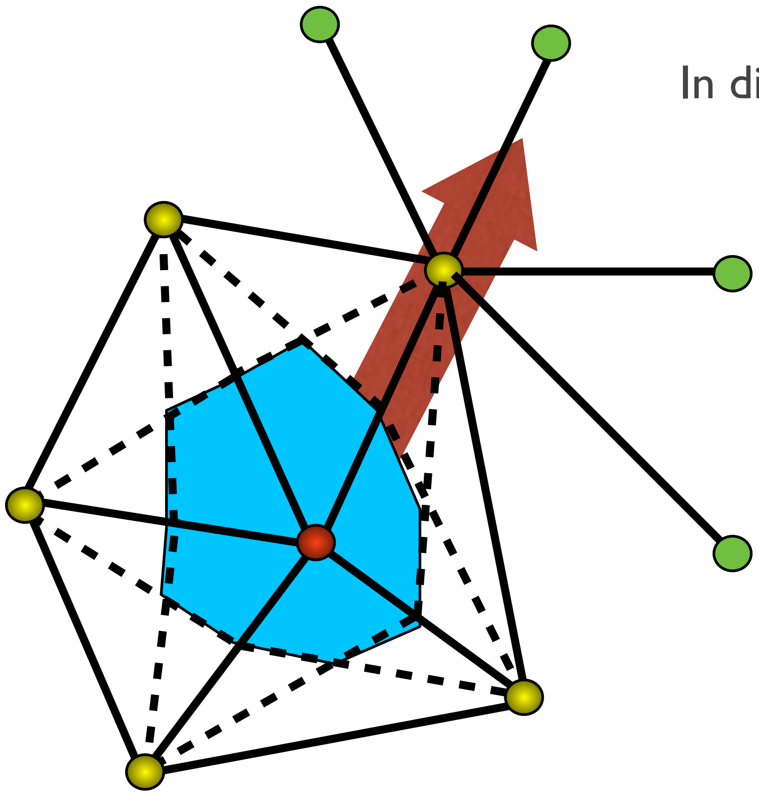
Increased matrix bandwidth

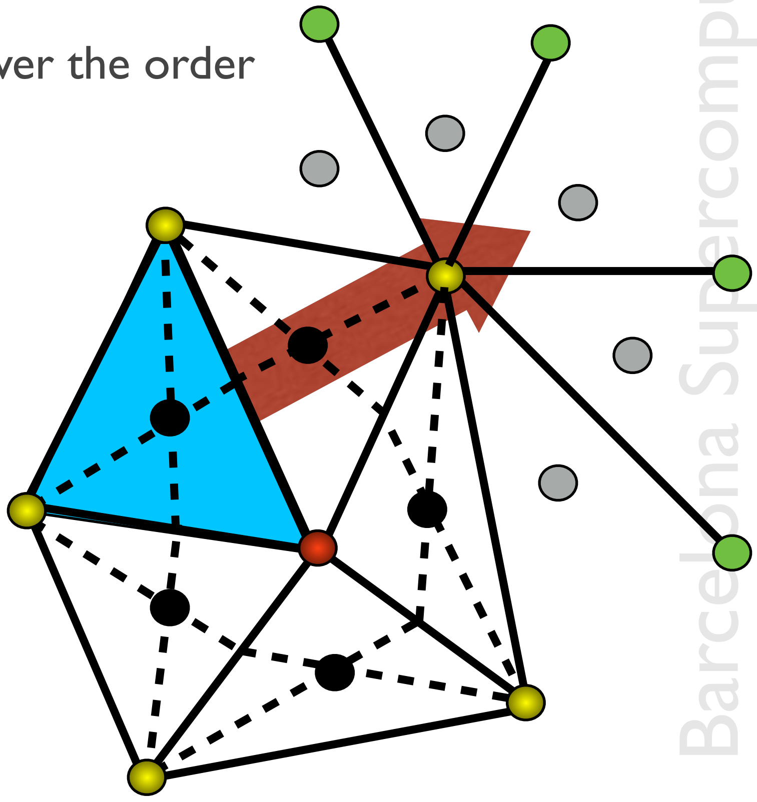
Heavier matrix assembly

New fluxes required, not a simple "extension"

Less nodes required to have the same accuracy

In discontinuities, lower the order





Discretising the System

Unstructured meshes

Increasing the order: Finite Volumes

Larger "friends" range

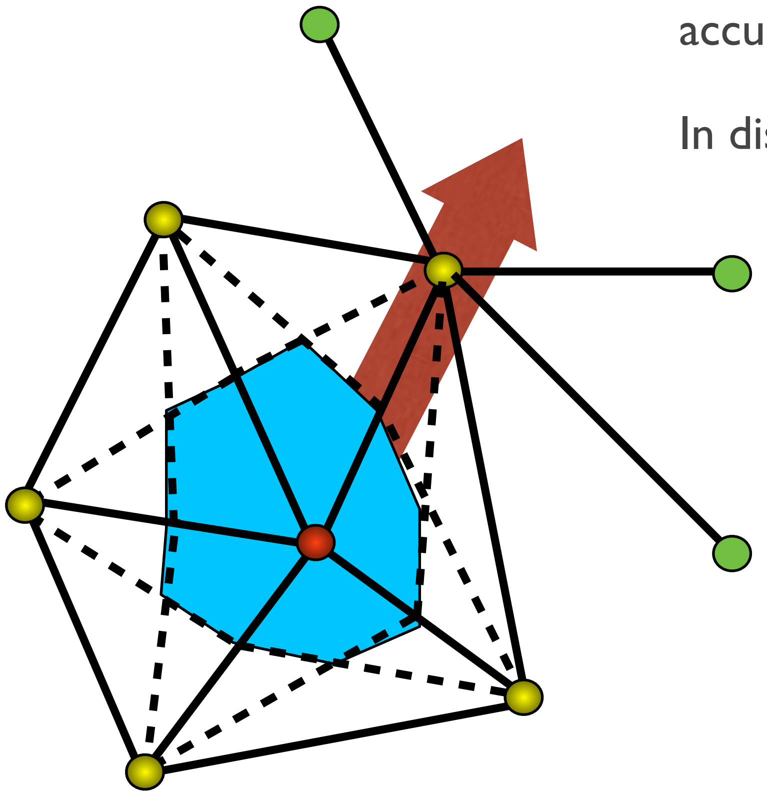
Increased matrix bandwidth

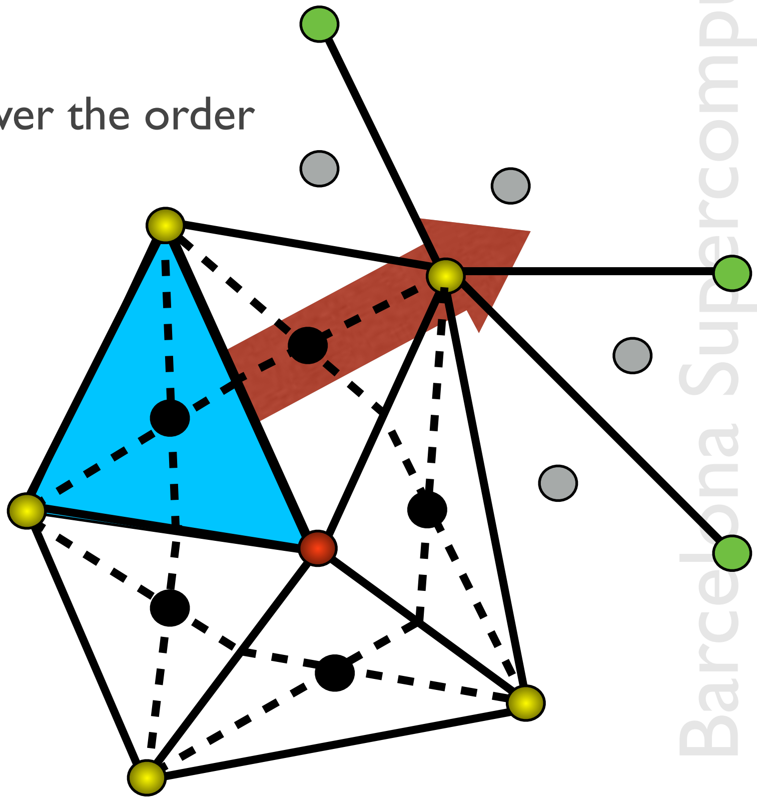
Heavier matrix assembly

New fluxes required, not a simple "extension"

Less nodes required to have the same accuracy

In discontinuities, lower the order

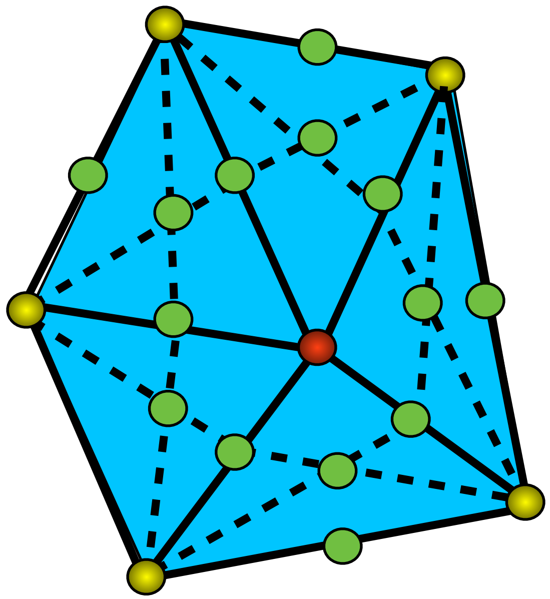




Discretising the System

Unstructured meshes

Increasing the order: Finite Elements



Higher polinomials’ order

Increased matrix bandwidth

Heavier matrix assembly

Simple extension

Less nodes required to have the same accuracy

Mesh generation issues

Not so good for discontinuities

Another possibility: spectral elements

Discretising the System

Finite Elements Stabilization

$$\int \Psi \frac{\partial \Phi^\alpha}{\partial t} + \int \Psi A_i^{\alpha\beta} \frac{\partial \Phi}{\partial x_i}^\beta + \int \frac{\partial \Psi}{\partial x_i} A_i^{*\alpha\beta} \tilde{\Phi}^\beta = \int \Psi S$$

$$K_e^{IJ} = K_e^{ab,\alpha\beta} = C_e^{ab,\alpha\beta} + S_e^{ab,\alpha\beta} + D_e^{ab,\alpha\beta}$$

Remember that, within an element and for a certain position, the value of a function and its derivative is interpolated from its values in the nodes in the following way:

PIZARRA:
- integración numérica
- funciones de forma
- funciones de test
- orden de elementos
- 1, 2, 3 dimensiones
- espacio paramétrico
- …

$$g(\zeta) = N^a(\zeta) g^a$$

$$\frac{\partial g}{\partial x_i}(\zeta) = \frac{\partial N}{\partial x_i}^a(\zeta) g^a$$

Discretising the System

Finite Elements Stabilization

$$\int \Psi \frac{\partial \Phi^{\alpha}}{\partial t} + \int \Psi A_i^{\alpha\beta} \frac{\partial \Phi}{\partial x_i}^{\beta} + \int \frac{\partial \Psi}{\partial x_i} A_i^{*\alpha\beta} \tilde{\Phi}^{\beta} = \int \Psi S$$

$$K_e^{IJ} = K_e^{ab,\alpha\beta} = C_e^{ab,\alpha\beta} + S_e^{ab,\alpha\beta} + D_e^{ab,\alpha\beta}$$

Mass matrix

$$M^{ab,\alpha\beta} = \delta_{\alpha\beta} N^b \; N^a$$

Tangent matrix, later…

$$D_e^{ab,\alpha\beta}$$

Discretising the System

Finite Elements Stabilization

$$\int \Psi \frac{\partial \Phi^\alpha}{\partial t} + \int \Psi A_i^{\alpha\beta} \frac{\partial \Phi}{\partial x_i}^\beta + \int \frac{\partial \Psi}{\partial x_i} A_i^{*\alpha\beta} \tilde{\Phi}^\beta = \int \Psi S$$

$$K_e^{IJ} = K_e^{ab,\alpha\beta} = C_e^{ab,\alpha\beta} + S_e^{ab,\alpha\beta} + D_e^{ab,\alpha\beta}$$

Galerkin term:
Even if there is no integration by parts, it is so-called by extension

$$C^{ab,\alpha\beta} = A_i^{\alpha\beta} \frac{\partial N^b}{\partial x_i} N^a$$

Discretising the System

Finite Elements Stabilization

$$\int \Psi \frac{\partial \Phi^\alpha}{\partial t} + \int \Psi A_i^{\alpha\beta} \frac{\partial \Phi}{\partial x_i}^\beta + \int \frac{\partial \Psi}{\partial x_i} A_i^{*\alpha\beta} \tilde{\Phi}^\beta = \int \Psi S$$

$$K_e^{IJ} = K_e^{ab,\alpha\beta} = C_e^{ab,\alpha\beta} + S_e^{ab,\alpha\beta} + D_e^{ab,\alpha\beta}$$

Stabilization term

$$S^{ab,\alpha\beta} = \tilde{L}^{a,\alpha\gamma}\, T^{b,\gamma\beta} \qquad \tilde{\Phi}^\beta = \tau^{\beta\gamma} r^\gamma$$

Adjoint operator

$$\tilde{L}^{a,\alpha\gamma} = A_i^{\alpha\gamma} \frac{\partial N^a}{\partial x_i}$$

Subscale

$$T^{b,\gamma\beta} = A_i^{\gamma\beta} \frac{\partial N^{(b)}}{\partial x_i} \tau_{(b)}$$

Discretising the System

We have completely separated **assembly** from **solver**.

$$K^{IJ} = \sum_{e}^{N_{\mathrm{el}}} K_e^{IJ}$$

Assembly can be done with **any** method: FVM, FEM, FDM, …

On **any** discretization

$$M^{IJ} = \sum_{e}^{N_{\mathrm{el}}} M_e^{IJ}$$

With **any** stabilization

Even boundary conditions can be integrated in the assembly

$$\frac{\mathbf{M}}{\Delta t}\Delta\mathbf{U} + \mathbf{K}\mathbf{U}^* = 0$$

$$\Delta\mathbf{U} = \mathbf{U}^{n+1} - \mathbf{U}^{n}$$

Then, the assembled matrix (or RHS) can be passed to the solver.

The solution scheme can be direct or iterative of any kind

With any preconditioner

Discretising the System: The solver

A very useful scheme is based on the "delta form"

$$\frac{\mathbf{M}}{\Delta t}\Delta\mathbf{U} + \mathbf{K}\mathbf{U}^* = 0 \qquad \left(\frac{\mathbf{M}}{\Delta t} + \mathbf{K}\right)\Delta\mathbf{U} = -\mathbf{K}\mathbf{U}^n$$

$$\Delta\mathbf{U} = \mathbf{U}^{n+1} - \mathbf{U}^n \qquad \Delta\mathbf{U} = \mathbf{U}^{n+1} - \mathbf{U}^n$$

Discretising the System: The solver

A very useful scheme is based on the "delta form"

$$\frac{\mathbf{M}}{\Delta t}\Delta\mathbf{U} + \mathbf{K}\mathbf{U}^* = 0 \qquad \left(\frac{\mathbf{M}}{\Delta t} + \mathbf{K}\right)\Delta\mathbf{U} = -\mathbf{K}\mathbf{U}^n$$

$$\Delta\mathbf{U} = \mathbf{U}^{n+1} - \mathbf{U}^n \qquad \Delta\mathbf{U} = \mathbf{U}^{n+1} - \mathbf{U}^n$$

Depending on *, we get:

$$\left(\frac{\mathbf{M}}{\Delta t}\right)\Delta\mathbf{U} = -\mathbf{K}\mathbf{U}^n$$

1st order Forward Euler

$$\left(\frac{\mathbf{M}}{\Delta t} + \mathbf{K}\right)\Delta\mathbf{U} = -\mathbf{K}\mathbf{U}^n$$

1st order Backward Euler

$$\left(\frac{\mathbf{M}}{\Delta t} + \frac{\mathbf{K}}{2}\right)\Delta\mathbf{U} = -\mathbf{K}\mathbf{U}^n$$

2nd order Crank-Nicholson

Discretising the System: The solver

A very useful scheme is based on the "delta form"

$$\frac{\mathbf{M}}{\Delta t}\Delta\mathbf{U} + \mathbf{K}\mathbf{U}^* = 0$$

$$\Delta\mathbf{U} = \mathbf{U}^{n+1} - \mathbf{U}^n$$

$$\left(\frac{\mathbf{M}}{\Delta t} + \mathbf{K}\right)\Delta\mathbf{U} = -\mathbf{K}\mathbf{U}^n$$

Remember that K depends on U…

For each time step the system is solved iteratively

$$\left(\frac{\mathbf{M}}{\Delta t} + \theta\mathbf{K}_i\right)\Delta\mathbf{U} = -\mathbf{K}\mathbf{U}^n \qquad \theta = 0, \quad 1, \quad 0.5$$

$$\mathbf{U}_{i+1} = \mathbf{U}^n + \Delta\mathbf{U}$$

Discretising the System: The solver

$$\left(\frac{\mathbf{M}}{\Delta t} + \theta \mathbf{K}\right) \Delta \mathbf{U} = -\mathbf{K}\mathbf{U}^n$$

On iterations $\Delta \tilde{\mathbf{U}} = \mathbf{U}_{i+1} - \mathbf{U}_i$

On time steps $\Delta \mathbf{U} = \mathbf{U}_{i+1} - \mathbf{U}^n$

$$\Delta \mathbf{U} = \Delta \tilde{\mathbf{U}} + \mathbf{U}_i - \mathbf{U}^n$$

$$\left(\frac{\mathbf{M}}{\Delta t} + \theta \mathbf{K}\right) (\Delta \tilde{\mathbf{U}} + \mathbf{U}_i - \mathbf{U}^n) = -\mathbf{K}\mathbf{U}^n$$

$$\left(\frac{\mathbf{M}}{\Delta t} + \theta \mathbf{K}\right) \Delta \tilde{\mathbf{U}} = \frac{\mathbf{M}}{\Delta t}(\mathbf{U}^n - \mathbf{U}_i) + \theta \mathbf{K}(\mathbf{U}^n - \mathbf{U}_i) - \mathbf{K}\mathbf{U}^n$$

$$\left(\frac{\mathbf{M}}{\Delta t} + \theta \mathbf{K}\right) \Delta \tilde{\mathbf{U}} = \frac{\mathbf{M}}{\Delta t}(\mathbf{U}^n - \mathbf{U}_i) - \theta \mathbf{K}\mathbf{U}_i - \mathbf{K}(1-\theta)\mathbf{U}^n$$

Discretising the System: The solver

$$\left(\frac{\mathbf{M}}{\Delta t} + \theta \mathbf{K}\right) \Delta \tilde{\mathbf{U}} = \frac{\mathbf{M}}{\Delta t}(\mathbf{U}^n - \mathbf{U}) - \theta \mathbf{K}\mathbf{U} - \mathbf{K}(1-\theta)\mathbf{U}^n$$

where we drop the "i" subindex to label the current iteration

What if we add something to the LHS to improve convergence?

Preconditioning and pseudo-time step

Discretising the System: The solver

Preconditioning and pseudo-time step:

$$\left(\frac{\mathbf{M}}{\Delta t} + \theta \mathbf{K}\right) \Delta \tilde{\mathbf{U}} = \frac{\mathbf{M}}{\Delta t}(\mathbf{U}^n - \mathbf{U}) - \theta \mathbf{K}\mathbf{U} - \mathbf{K}(1-\theta)\mathbf{U}^n$$

where we drop the "i" subindex to label the current iteration

What if we add something to the LHS to improve convergence?

$$\left(\frac{\mathbf{M}}{\Delta \tau} + \frac{\mathbf{M}}{\Delta t} + \theta \mathbf{K}\right) \Delta \tilde{\mathbf{U}} = \frac{\mathbf{M}}{\Delta t}(\mathbf{U}^n - \mathbf{U}) - \theta \mathbf{K}\mathbf{U} - \mathbf{K}(1-\theta)\mathbf{U}^n$$

$$\mathbf{U}_{i+1} = \mathbf{U} + \Delta \tilde{\mathbf{U}}$$

Discretising the System: The solver

All these schemes are Jacobi schemes or “Fixed Point”

They have no information on the gradients to improve convergence properties

Another possibility are Newton schemes…

if you can afford the gradients’ computation

Discretising the System: The solver

After some algebraic maneuvers, this is the residual for the pseudo-time stepped problem:

$$\left(\frac{\mathbf{M}}{\Delta\tau} + \frac{\mathbf{M}}{\Delta t} + \theta\mathbf{K}\right)\Delta\tilde{\mathbf{U}} = \frac{\mathbf{M}}{\Delta t}(\mathbf{U}^n - \mathbf{U}) - \theta\mathbf{K}\mathbf{U} - \mathbf{K}(1-\theta)\mathbf{U}^n$$

$$\mathbf{M}\frac{\Delta\tilde{\mathbf{U}}}{\Delta\tau} + \mathbf{M}\frac{\Delta\mathbf{U}}{\Delta t} + \mathbf{K}(\mathbf{U})\mathbf{U} = \mathbf{r}$$

$$\theta = 0, \quad \boxed{1,} \quad 0.5$$

Discretising the System: The solver

After some algebraic maneuvers, this is the residual for the pseudo-time stepped problem:

$$\mathbf{M}\frac{\Delta\tilde{\mathbf{U}}}{\Delta\tau} + \mathbf{M}\frac{\Delta\mathbf{U}}{\Delta t} + \mathbf{K}(\mathbf{U})\mathbf{U} = \mathbf{r}$$

Taylor expansion of the resitual at iteration "i":

$$\mathbf{r}_{i+1} \approx \mathbf{r} + \Delta\tilde{\mathbf{U}}\frac{\partial\mathbf{r}}{\partial\mathbf{U}} = 0$$

When 0, it has converged

Discretising the System: The solver

Then, we solve

$$\mathbf{r}_{i+1} \approx \mathbf{r} + \Delta\tilde{\mathbf{U}}\frac{\partial \mathbf{r}}{\partial \mathbf{U}} = 0 \qquad \text{=>} \qquad \Delta\tilde{\mathbf{U}}\frac{\partial \mathbf{r}}{\partial \mathbf{U}} = -\mathbf{r}$$

Where

$$\frac{\partial \mathbf{r}}{\partial \mathbf{U}} = \frac{\partial}{\partial \mathbf{U}}\left[\mathbf{M}\frac{\Delta\tilde{\mathbf{U}}}{\Delta\tau} + \mathbf{M}\frac{\Delta\mathbf{U}}{\Delta t} + \mathbf{K}(\mathbf{U})\mathbf{U}\right]$$

$$\frac{\partial \mathbf{r}}{\partial \mathbf{U}} = \frac{\mathbf{M}}{\Delta\tau} + \frac{\mathbf{M}}{\Delta t} + \mathbf{K} + \mathbf{U}\frac{\partial \mathbf{K}}{\partial \mathbf{U}}$$

considering that residual is evaluated at iteration "i" (the last we have!), therefore

$$\Delta\tilde{\mathbf{U}} = \mathbf{U}_{i+1} - \mathbf{U} \qquad \qquad \Delta\tilde{\mathbf{U}} = \mathbf{U} - \mathbf{U}_{i-1}$$

means

$$\Delta\mathbf{U} = \mathbf{U}_{i+1} - \mathbf{U}^n \qquad \qquad \Delta\mathbf{U} = \mathbf{U} - \mathbf{U}^n$$

Discretising the System: The solver

which leads to

$$\left(\frac{\mathbf{M}}{\Delta\tau}+\frac{\mathbf{M}}{\Delta t}+\mathbf{K}+\mathbf{U}\frac{\partial\mathbf{K}}{\partial\mathbf{U}}\right)\Delta\tilde{\mathbf{U}}=-\mathbf{K}\mathbf{U}-\frac{\mathbf{M}}{\Delta t}(\mathbf{U}-\mathbf{U}^n)-\frac{\mathbf{M}}{\Delta\tau}(\Delta\tilde{\mathbf{U}}_{i-1})$$

Tau-Newton

Compare (suppose Backward Euler):

$$\left(\frac{\mathbf{M}}{\Delta t}+\mathbf{K}+\mathbf{U}\frac{\partial\mathbf{K}}{\partial\mathbf{U}}\right)\Delta\tilde{\mathbf{U}}=-\mathbf{K}\mathbf{U}-\frac{\mathbf{M}}{\Delta t}(\mathbf{U}-\mathbf{U}^n)$$

Newton

Discretising the System: The solver

which leads to

$$\left(\frac{\mathbf{M}}{\Delta\tau} + \frac{\mathbf{M}}{\Delta t} + \mathbf{K} + \mathbf{U}\frac{\partial\mathbf{K}}{\partial\mathbf{U}}\right)\Delta\tilde{\mathbf{U}} = -\mathbf{K}\mathbf{U} - \frac{\mathbf{M}}{\Delta t}(\mathbf{U} - \mathbf{U}^n) - \frac{\mathbf{M}}{\Delta\tau}(\Delta\tilde{\mathbf{U}}_{i-1})$$

Tau-Newton

Compare:

$$\left(\frac{\mathbf{M}}{\Delta t} + \mathbf{K} + \mathbf{U}\frac{\partial\mathbf{K}}{\partial\mathbf{U}}\right)\Delta\tilde{\mathbf{U}} = -\mathbf{K}\mathbf{U} - \frac{\mathbf{M}}{\Delta t}(\mathbf{U} - \mathbf{U}^n)$$

Newton

$$\left(\frac{\mathbf{M}}{\Delta t} + \mathbf{K}\right)\Delta\tilde{\mathbf{U}} = -\mathbf{K}\mathbf{U} - \frac{\mathbf{M}}{\Delta t}(\mathbf{U} - \mathbf{U}^n)$$

Jacobi

Discretising the System: The solver

which leads to

$$\left(\frac{\mathbf{M}}{\Delta\tau}+\frac{\mathbf{M}}{\Delta t}+\mathbf{K}+\mathbf{U}\frac{\partial\mathbf{K}}{\partial\mathbf{U}}\right)\Delta\tilde{\mathbf{U}}=-\mathbf{K}\mathbf{U}-\frac{\mathbf{M}}{\Delta t}(\mathbf{U}-\mathbf{U}^n)-\frac{\mathbf{M}}{\Delta\tau}(\Delta\tilde{\mathbf{U}}_{i-1})$$

Tau-Newton

Compare:

$$\left(\frac{\mathbf{M}}{\Delta t}+\mathbf{K}+\mathbf{U}\frac{\partial\mathbf{K}}{\partial\mathbf{U}}\right)\Delta\tilde{\mathbf{U}}=-\mathbf{K}\mathbf{U}-\frac{\mathbf{M}}{\Delta t}(\mathbf{U}-\mathbf{U}^n)$$

Newton

$$\left(\frac{\mathbf{M}}{\Delta t}+\mathbf{K}\right)\Delta\tilde{\mathbf{U}}=-\mathbf{K}\mathbf{U}-\frac{\mathbf{M}}{\Delta t}(\mathbf{U}-\mathbf{U}^n)$$

Jacobi

$$\left(\frac{\mathbf{M}}{\Delta\tau}+\frac{\mathbf{M}}{\Delta t}+\mathbf{K}\right)\Delta\tilde{\mathbf{U}}=-\mathbf{K}\mathbf{U}-\frac{\mathbf{M}}{\Delta t}(\mathbf{U}-\mathbf{U}^n)$$

Tau-Jacobi

$$\left(\frac{\mathbf{M}}{\Delta\tau}+\frac{\mathbf{M}}{\Delta t}+\mathbf{K}\right)\Delta\tilde{\mathbf{U}}=-\mathbf{K}\mathbf{U}-\frac{\mathbf{M}}{\Delta t}(\mathbf{U}-\mathbf{U}^n)-\frac{\mathbf{M}}{\Delta\tau}(\Delta\tilde{\mathbf{U}}_{i-1})$$

Tau-Jacobi-2

Discretising the System: The solver

which leads to

$$\left(\frac{\mathbf{M}}{\Delta\tau}+\frac{\mathbf{M}}{\Delta t}+\mathbf{K}+\mathbf{U}\frac{\partial\mathbf{K}}{\partial\mathbf{U}}\right)\Delta\tilde{\mathbf{U}}=-\mathbf{K}\mathbf{U}-\frac{\mathbf{M}}{\Delta t}(\mathbf{U}-\mathbf{U}^n)-\frac{\mathbf{M}}{\Delta\tau}(\Delta\tilde{\mathbf{U}}_{i-1})$$

Remarks:

RHS is the same for all: we do not change the original system at all

We have only add terms to the LHS

Not any term, but very specific ones

We are just trying to improve the **condition number** of the system

That is to say, **preconditioning**

Discretising the System: The solver

The system can be solved in either a monolithic or a segregatted way.

Each of them has its own features…

Discretising the System: The solver

Then, if the solution is smooth, we are more or less done.

But if the solution could admit discontinuities or even sharp gradients, a different strategy must be defined.

The main difference is in the starting point: the equation.

Strategy:

Take the Integral Form
Discretise the ***space*** by tessellation
Define compact support functions for each of the cells or elements
Filter the IF by projecting the compact support space
Integrate by parts some terms when needed
Discretise the ***time*** by finite differences
Compute the resulting integrals numerically
Define a stabilization scheme

The IF becomes an algebraic system of equations (linear or linearised)

...

Discretising the System: The solver

Finite Elements - Volumes - Differences

But remeber… in all schemes stabilisation problems persist!!!!

Discretising the System: The solver

Incompressible flows (distraction)

There is an alternative to incompressible flows, that comes from strictly imposing

$$\frac{\partial u_i}{\partial x_i} = 0$$

Then, continuity equation becomes just a constant density condition.

Now, taking the divergence to the momentum equation

$$\frac{\partial U_j}{\partial t} + \frac{\partial}{\partial x_i}(u_i U_j) + \frac{\partial}{\partial x_i}(\delta_{ij} p - \tau_{ij}) + \rho g_j = 0$$

and using the velocity divergence zero, we come up with a Poisson equation for the pressure, leading to the following system…

Discretising the System: The solver

Monolithic scheme

$$\begin{bmatrix} \mathbf{A}_{uu} & \mathbf{A}_{up} \\ \mathbf{A}_{pu} & \mathbf{A}_{pp} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_u \\ \mathbf{b}_p \end{bmatrix}$$

Discretising the System: The solver

$$\begin{bmatrix} \mathbf{A}_{uu} & \mathbf{A}_{up} \\ \mathbf{A}_{pu} & \mathbf{A}_{pp} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_u \\ \mathbf{b}_p \end{bmatrix}$$

Solve momentum
Substitute velocity in continuity

Schur complement system
for the pressure

$$\mathbf{S}\mathbf{p} = \mathbf{b}_S \quad \begin{cases} \mathbf{S} & = \mathbf{A}_{pp} - \mathbf{A}_{pu}\mathbf{A}_{uu}^{-1}\mathbf{A}_{up} \\ \mathbf{b}_S & = \mathbf{b}_p - \mathbf{A}_{pu}\mathbf{A}_{uu}^{-1}\mathbf{b}_u \end{cases}$$

Solve using preconditioned (P)
Richardson iteration

Preconditioned iteration
for the pressure

$$\mathbf{p}^{k+1} = \mathbf{p}^k + (\mathbf{A}_{pp} + \mathbf{P})^{-1}(\mathbf{b}_S - \mathbf{S}\mathbf{p}^k)$$

Final system

$$\begin{cases} \mathbf{A}_{uu}\mathbf{u}^{k+1} & = \mathbf{b}_u - \mathbf{A}_{up}\mathbf{p}^k \\ (\mathbf{A}_{pp} + \mathbf{P})\mathbf{p}^{k+1} & = \mathbf{b}_p - \mathbf{A}_{pu}\mathbf{u}^{k+1} + \mathbf{P}\mathbf{p}^k \end{cases}$$

Discretising the System

Going beyond:

Local Preconditioning

Discretisation:

Algorithms and Codes

Discretising the System

By discretising the system, we highly reduce the DIMENSIONALITY of the problem.

From continuous to discretised.

Discretising time and space, we transformed the differential equations in a (potentially very large) ALGEBRAIC SYSTEM

Discretising the System

Discretisation issues:

Discretisation lead us to the following matrix system

$$M\underbrace{\frac{\Phi^{n+1}-\Phi^{n}}{\Delta t}}_{\text{Time derivative}} + \underbrace{\left[\underbrace{C(\Phi)}_{\text{Non-sym}} + \underbrace{K(\Phi)}_{\text{Sym}}\right]\Phi^{*}}_{\text{Non-linear matrix}} + \underbrace{s(\Phi^{*})}_{\text{Non-linear ODE term}} = b$$

We must adopt a solution strategy (the **solver**), analyzing these issues

- ✓ Explicit
- ✓ Implicit
- ✓ Solving "by blocks"
- ✓ How and what to parallelize

Solution strategies

Discretisation issues:

We arrive at the following matrix system

$$\underbrace{M\frac{\Phi^{n+1}-\Phi^{n}}{\Delta t}}_{\text{Time derivative}} + \underbrace{\left[\underbrace{C(\Phi)}_{\text{Non-sym}} + \underbrace{K(\Phi)}_{\text{Sym}}\right]\Phi^{*}}_{\text{Non-linear matrix}} + \underbrace{s(\Phi^{*})}_{\text{Non-linear ODE term}} = b$$

Depending on the definition of $*$ the scheme is **Explicit** or **Implicit**

$$* := n+1 \quad \longleftarrow \quad \text{Implicit}$$

$$* := n \quad \longleftarrow \quad \text{Explicit}$$

Solution strategies

Explicit schemes

$$M_d \frac{\Phi^{n+1} - \Phi^n}{\Delta t} = -\left[C(\Phi) + K(\Phi)\right]\Phi^n - s(\Phi^n) + b$$

$$\Phi^{n+1} = \Phi^n - M_d^{-1}\Delta t\left(\left[C(\Phi) + K(\Phi)\right]\Phi^n - s(\Phi^n) + b\right)$$

Features:

RHS can be directly computed, no matrix storage required

Computing the RHS (assembly) is, by far, the most time consuming part

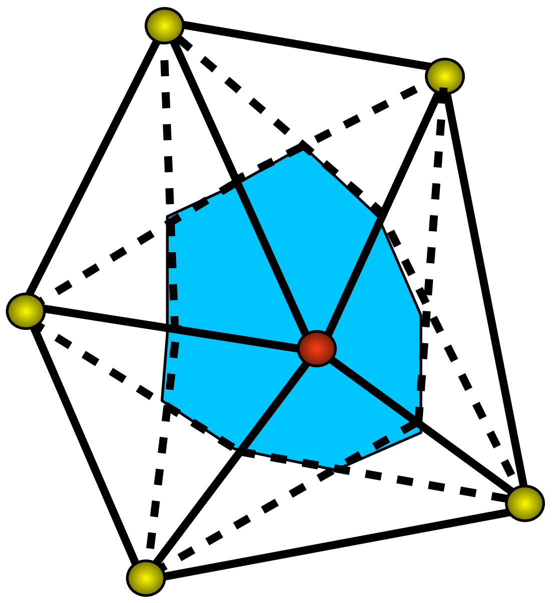
Solution strategies

$$\Phi^{n+1} = \Phi^n - M_d^{-1} \Delta t \left(\left[C(\Phi) + K(\Phi) \right] \Phi^n - s(\Phi^n) + b \right)$$

Features:

Lumped mass matrix represents the "mass" associated to each mesh node

It is a diagonal matrix, trivially inverted



Solution strategies

$$\Phi^{n+1} = \Phi^n - M_d^{-1}\Delta t\left(\left[C(\Phi) + K(\Phi)\right]\Phi^n - s(\Phi^n) + b\right)$$

Features:

Time step computed from stability conditions, such as the CFL condition

What is the time a signal takes to propagate within a given element?

- ✓ Velocity
- ✓ Diffusion
- ✓ Acoustic waves (linear, small perturbations)
- ✓ Shock waves (non linear, strong gradients)

For instance

$$\Delta t = \frac{f^{\mathrm{CFL}}}{\frac{u}{h_1} + \frac{2k}{h_2^2} + \frac{c}{h_3}}$$

Solution strategies

$$\Phi^{n+1} = \Phi^n - M_d^{-1}\Delta t\left(\left[C(\Phi) + K(\Phi)\right]\Phi^n - s(\Phi^n) + b\right)$$

Features:

Time step computed from stability conditions, such as the CFL condition

From theoretical arguments for simple equations, the CFL factor is 1.

However, more complex problems could require smaller figures.

But what is “h”?

$$\Delta t = \frac{f^{\mathrm{CFL}}}{\dfrac{u}{h_1} + \dfrac{2k}{h_2^2} + \dfrac{c}{h_3}}$$

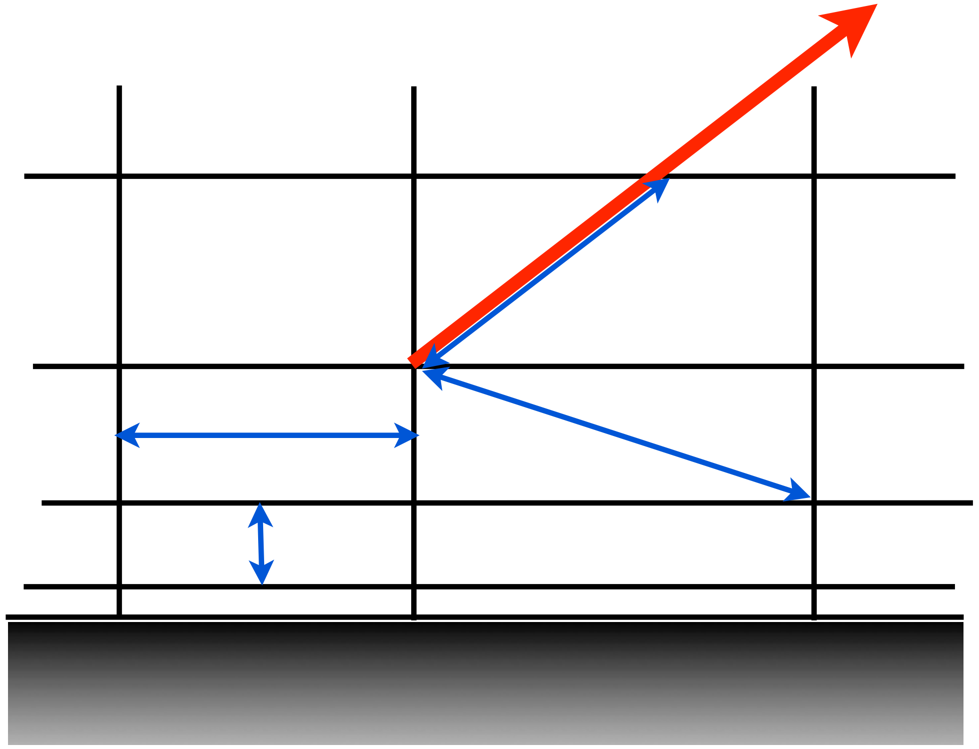
Solution strategies

$$\Phi^{n+1} = \Phi^n - M_d^{-1} \Delta t \left(\left[C(\Phi) + K(\Phi) \right] \Phi^n - s(\Phi^n) + b \right)$$

Features:

Time step computed from stability conditions, such as the CFL condition

But what is “h”?



Solution strategies

$$\Phi^{n+1} = \Phi^n - M_d^{-1} \Delta t \left(\left[C(\Phi) + K(\Phi) \right] \Phi^n - s(\Phi^n) + b \right)$$

Features:

Time step computed from stability conditions, such as the CFL condition

For transient problems, the mesh minimum value is taken

For transient coupled problems, the per-equation minimum value is taken

For stationary problems, local time steps can be used (but eye...!!)

Both can be combined with “pseudo-time step” formulations (Jameson papers)

Solution strategies

$$\Phi^{n+1} = \Phi^n - M_d^{-1}\Delta t \left(\left[C(\Phi) + K(\Phi) \right] \Phi^n - s(\Phi^n) + b \right)$$

Features:

Time step computed from stability conditions, such as the CFL condition

The Characteristic Condition Number is the ratio of the largest to the smallest

characteristic speeds:

$$u, \frac{2k}{h}, c, \ldots$$

When one of them goes to zero, the system becomes extremely "stiff"

Preconditioning is required!

☑ Local preconditioners for Explicit schemes: Turkel, Weiss, Van Leer, ...

☑ Global preconditioners for Implicit iterative schemes: Diagonal, ILU, ...

Solution strategies

$$\Phi^{n+1} = \Phi^n - M_d^{-1} \Delta t \left(\left[C(\Phi) + K(\Phi) \right] \Phi^n - s(\Phi^n) + b \right)$$

Features:

Dirichlet boundary conditions can be imposed after a time advance step

This gives enough flexibility for non-linear conditions: Navier-Stokes / Euler

Neuman boundary conditions enters in the RHS for FEM

Neuman boundary conditions are transformed in Dirichlet ones for FD

Solution strategies

$$\Phi^{n+1} = \Phi^n - M_d^{-1}\Delta t \left(\left[C(\Phi) + K(\Phi) \right] \Phi^n - s(\Phi^n) + b \right)$$

Features:

The downside is the time step limitation, which in some very "stiff" problems can be extremely limiting... such as incompressible flows or solids

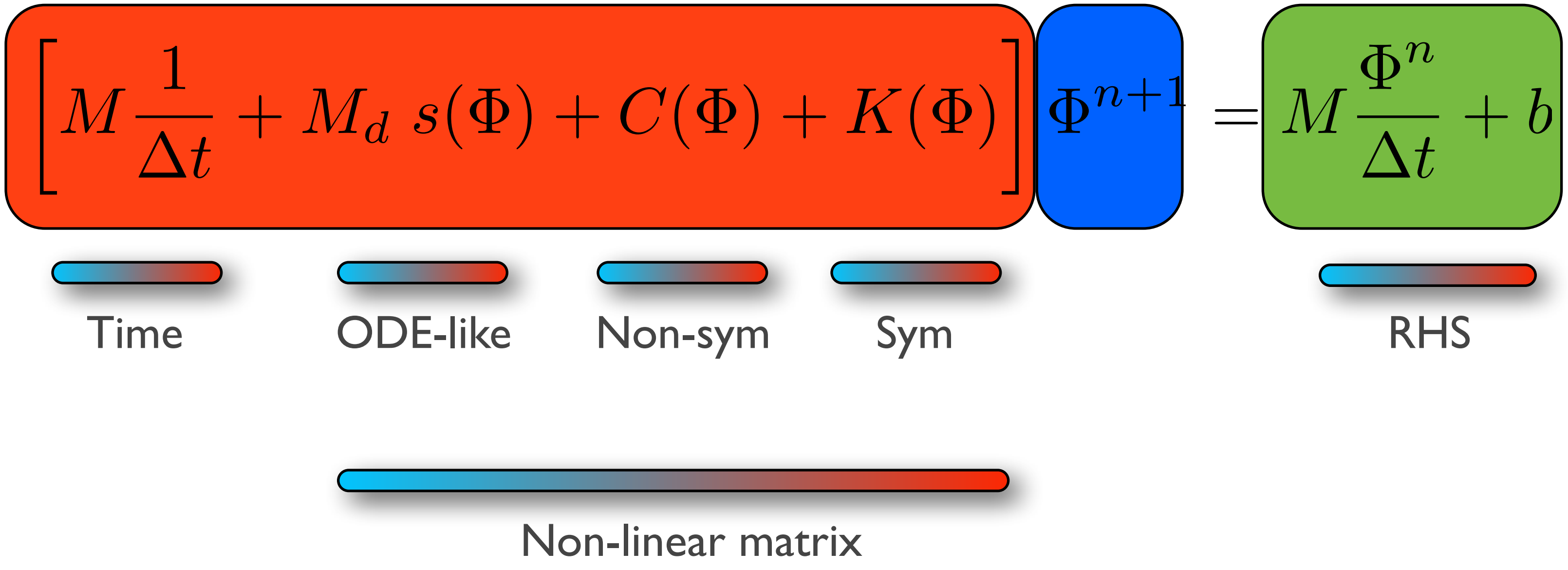
Then... **Implicit** methods are the best option.

Are they indeed?

Solution strategies

$$M\frac{\Phi^{n+1}-\Phi^{n}}{\Delta t}+\left[C(\Phi)+K(\Phi)\right]\Phi^{*}+s(\Phi^{n+1})=b$$

$$\left[M\frac{1}{\Delta t}+M_d\,s(\Phi)+C(\Phi)+K(\Phi)\right]\Phi^{n+1}=M\frac{\Phi^{n}}{\Delta t}+b$$



Solution strategies

$$\left[M\frac{1}{\Delta t} + M_d\, s(\Phi) + C(\Phi) + K(\Phi)\right]\Phi^{n+1} = M\frac{\Phi^n}{\Delta t} + b$$

Time — ODE-like — Non-sym — Sym — RHS

Sub-matrices:

Time matrix: consistent mass matrix, that couples first neighbors (for first order)

ODE-like matrix: diagonal mass matrix to preserve locality. Let us suppose this term non-linear

Non-symmetric matrix: comes from first space derivative matrices, such as convection ones

Symmetric matrix: comes from second space derivative matrices, such as Laplacians (diffusion) or stress (large strain solid mechanics)

Solution strategies

$$\left[M\frac{1}{\Delta t} + M_d\, s(\Phi) + C(\Phi) + K(\Phi)\right]\Phi^{n+1} = M\frac{\Phi^n}{\Delta t} + b$$

Features:

Consider the matrix divided in blocks according to certain grouping of the unknowns. Then, not all the blocks prefer the same solution scheme...

Typically, compact support problems can have a relatively low bandwidth, so sparse algebra is better-suited.

On the other hand, other problems can produce filled matrices (i.e. Fourier, Bessels, plane waves...)

Renumbering can be decisive for a better data distribution

Solution strategies

$$\left[M \frac{1}{\Delta t} + M_d \, s(\Phi) + C(\Phi) + K(\Phi) \right] \Phi^{n+1} = M \frac{\Phi^n}{\Delta t} + b$$

Features:

The problem is now clearly divided in two parts

1. The RHS and matrices assembly
2. The solver

Which is the most time consuming part in an implicit scheme?

Typically the second one, say from 40-60 up to 10-90

Let us analyze it the algorithmic of the implicit form...

Solution strategies

```
Input data
Compute geometrical stuff (only once)
Do time steps
    Do coupling iterations
        Do linear iterations
            Compute Matrix and RHS:
            Do elements (or faces or nodes...) iterations
                Compute elementary Matrix and RHS
                Assemble
            Enddo
            Solver (iterative or direct)
        Enddo
        Update coupling
        Enddo
    Enddo
    Update time step
Enddo
Output data
```

Solution strategies

```
Input data
Compute geometrical stuff (only once)
Do time steps
    Do coupling iterations
        Do linear iterations
            Compute Matrix and RHS:
            Do elements (or faces or nodes...) iterations
                Compute elementary Matrix and RHS
                Assemble
            Enddo
            Solver (iterative or direct)
        Enddo
        Update coupling
        Enddo
    Enddo
    Update time step
Enddo
Output data
```

Mesh
Boundary conditions
Scalar data

Binary formats for large data (Endian vs Big Endian)

Standard formats such as CGNS or marked for scalar

Checkpoint - restart

Solution strategies

```
Input data
Compute geometrical stuff (only once)
Do time steps
    Do coupling iterations
        Do linear iterations
            Compute Matrix and RHS:
            Do elements (or faces or nodes...) iterations
                Compute elementary Matrix and RHS
                Assemble
            Enddo
            Solver (iterative or direct)
        Enddo
        Update coupling
        Enddo
    Enddo
    Update time step
Enddo
Output data
```

Stencils and mass matrix

Analyze matrix system to see what can be pre-computed

Trade off between storage and computing time

Compute normals and boundary values

Solution strategies

```
Input data
Compute geometrical stuff (only once)
Do time steps
    Do coupling iterations
        Do linear iterations
            Compute Matrix and RHS:
            Do elements (or faces or nodes...) iterations
                Compute elementary Matrix and RHS
                Assemble
            Enddo
            Solver (iterative or direct)
        Enddo
        Update coupling
        Enddo
    Enddo
    Update time step
Enddo
Output data
```

Solution strategies

Input data

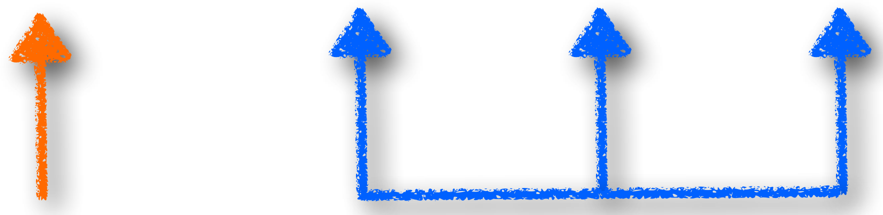
Compute geometrical stuff (only once)

Do time steps

Do coupling iterations

Do linear iterations

$$\left[M\frac{1}{\Delta t} + M_d\; s(\Phi) + C(\Phi) + K(\Phi) \right] \Phi^{n+1} = M\frac{\Phi^n}{\Delta t} + b$$



Coupling:
 related to the solver
 related to the Physics
 related to both

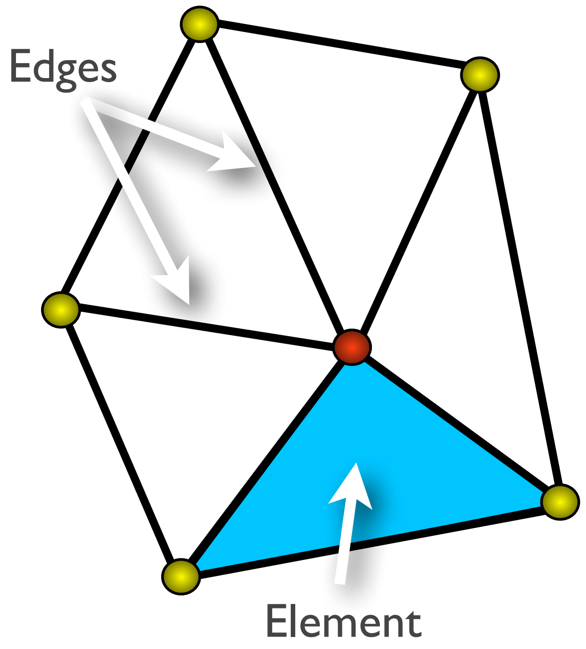
Solution strategies

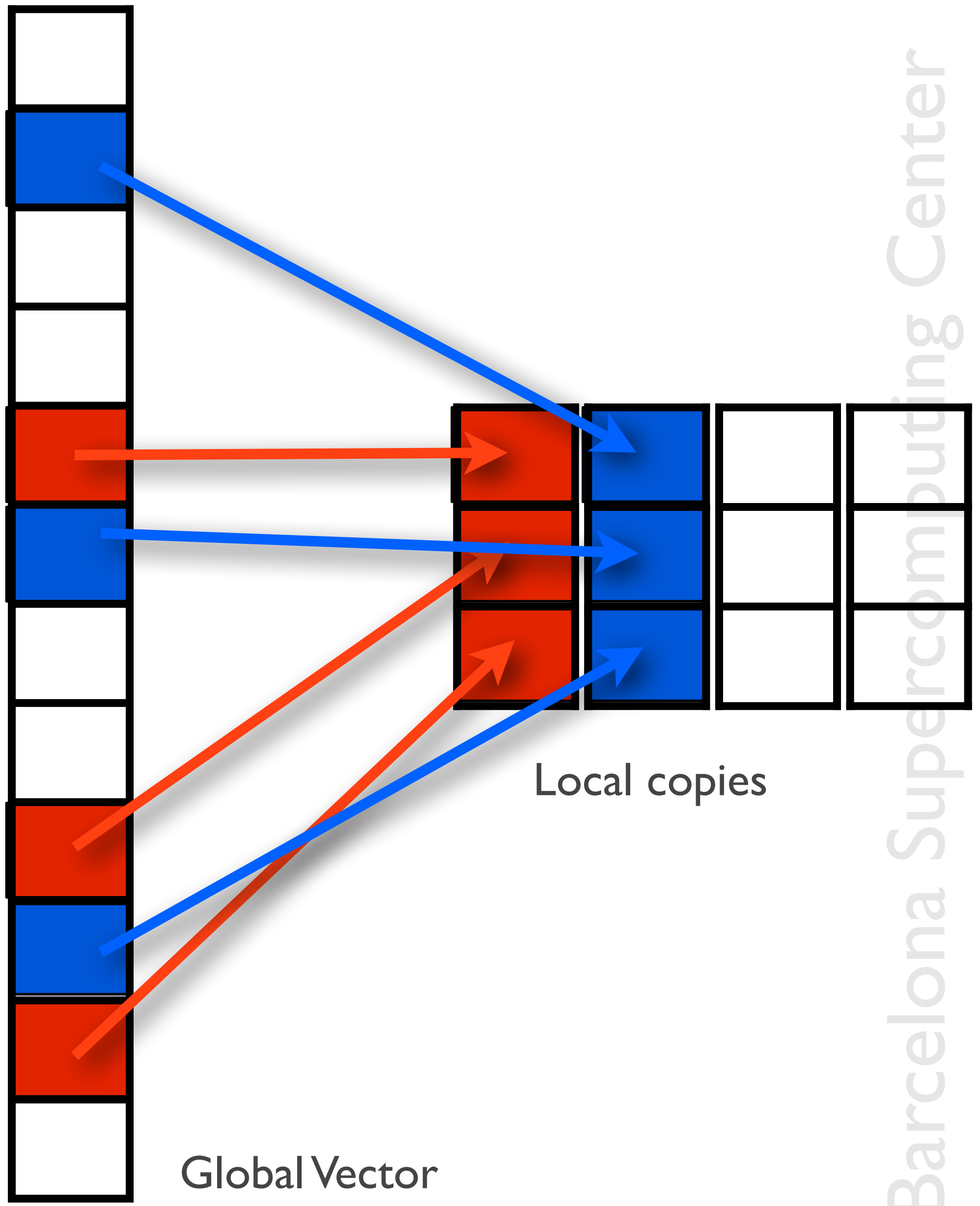
```
Input data
Compute geometrical stuff (only once)
Do time steps
    Do coupling iterations
        Do linear iterations
            Compute Matrix and RHS:
            Do elements (or faces or nodes...) iterations
                Compute elementary Matrix and RHS
                Assemble
            Enddo
            Solver (iterative or direct)
        Enddo
        Update coupling
        Enddo
    Enddo
    Update time step
Enddo
Output data
```

Loop over element, nodes, faces, edges, to compute local contributions to RHS and Matrices

Locality of data (gather - scatter)

Solution strategies





Make a local copy of the working vectors
The more FLOPs per DOF, the better is to do a local copy
Local copies can be done every N elements

Solution strategies

```
Input data
Compute geometrical stuff (only once)
Do time steps
    Do coupling iterations
        Do linear iterations
            Compute Matrix and RHS:
            Do elements (or faces or nodes...) iterations
                Compute elementary Matrix and RHS
                Assemble
            Enddo
            Solver (iterative or direct)
        Enddo
        Update coupling
        Enddo
    Enddo
    Update time step
Enddo
Output data
```

For implicit schemes, this is the bottle neck

Solution strategies

```
Input data
Compute geometrical stuff (only once)
Do time steps
    Do coupling iterations
        Do linear iterations
            Compute Matrix and RHS:
            Do elements (or faces or nodes...) iterations
                Compute elementary Matrix and RHS
                Assemble
            Enddo
            Solver Iterative (GMRES/BCGstab/CG...)
        Enddo
        Update coupling
        Enddo
    Enddo
    Update time step
Enddo
Output data
```

Newton Krylov methods:

Newton for the linear iterations
Iterative for the solver

Solvers

Direct solvers or iterative solvers?

Avoid direct solvers when possible

- Good for small - medium problems
- Bad for large ones, too much memory required
- Very bad scalability
- However, very stiff problems could require a direct solver...

The election of the iterative solver is strongly biased by the form of the matrix:

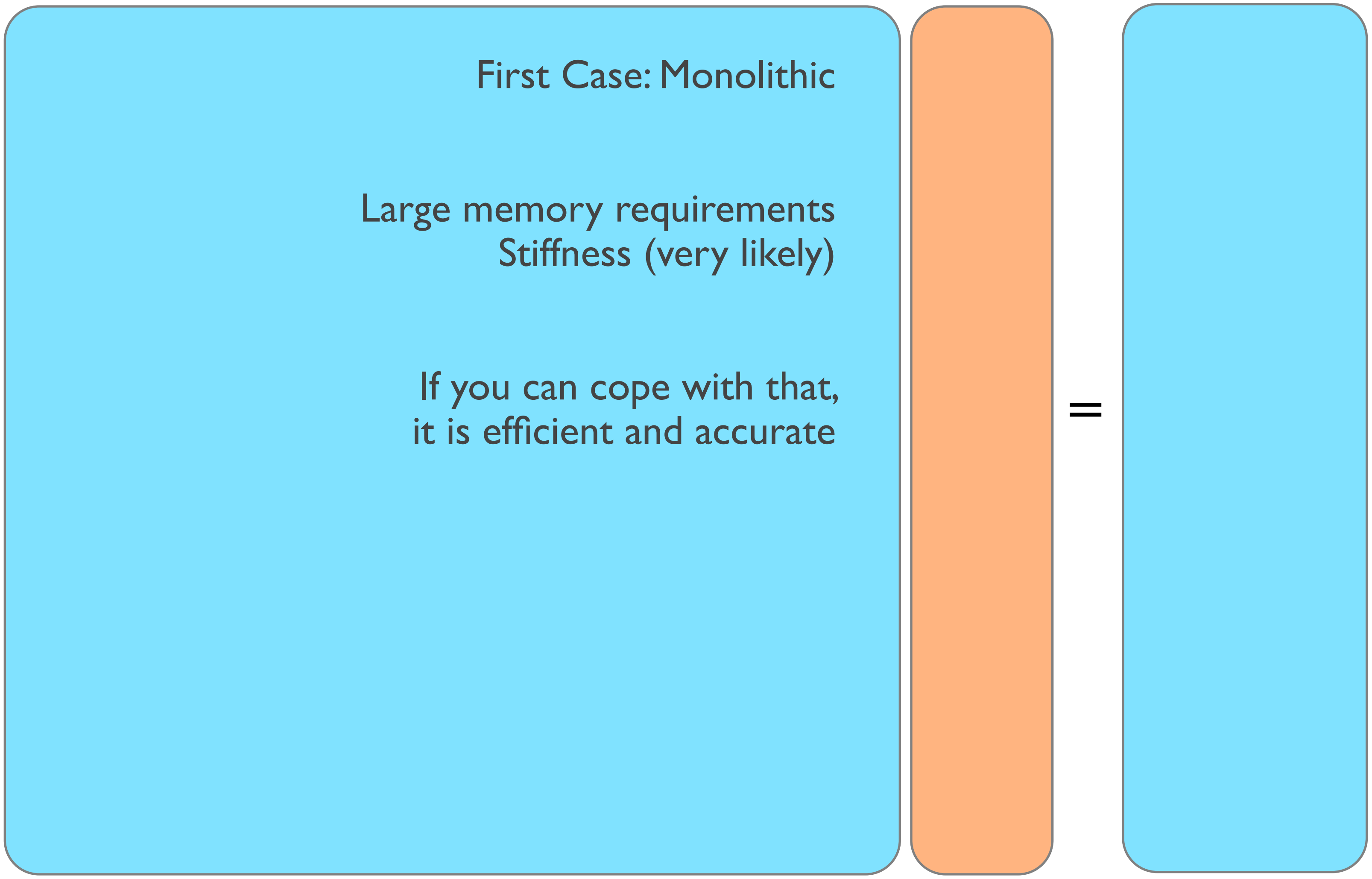
Symmetric, then Conjugate Gradient family
Otherwise, Krylov subspace family

However... some Krylov family can be very good for symmetric problems

Issues: Solution strategy

$$\left[M\frac{1}{\Delta t} + M_d\, s(\Phi) + C(\Phi) + K(\Phi) \right] \Phi^{n+1} = M\frac{\Phi^n}{\Delta t} + b$$

Issues: Solution strategy



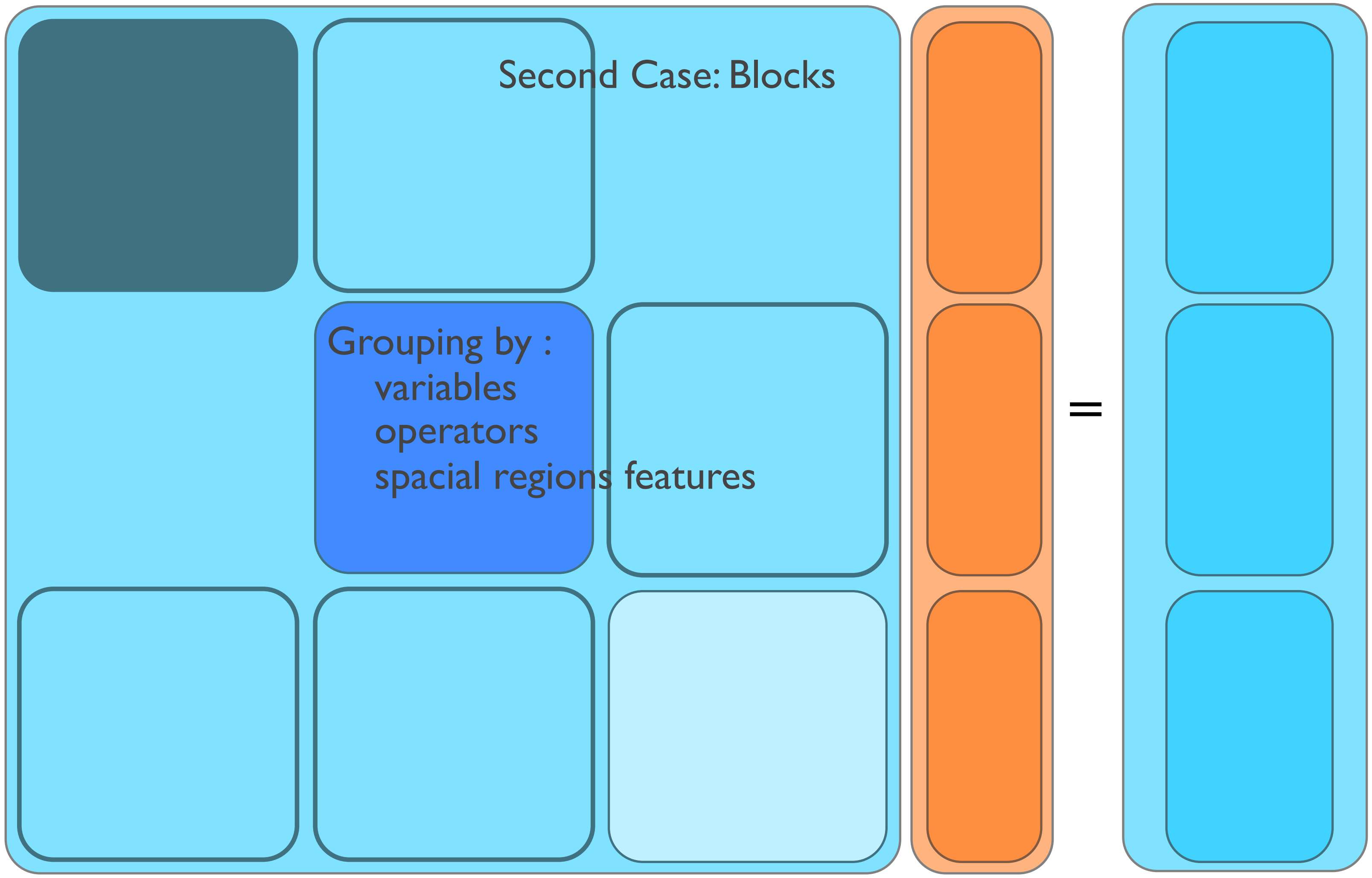
First Case: Monolithic

Large memory requirements
Stiffness (very likely)

If you can cope with that,
it is efficient and accurate

Solvers

Issues: Solution strategy

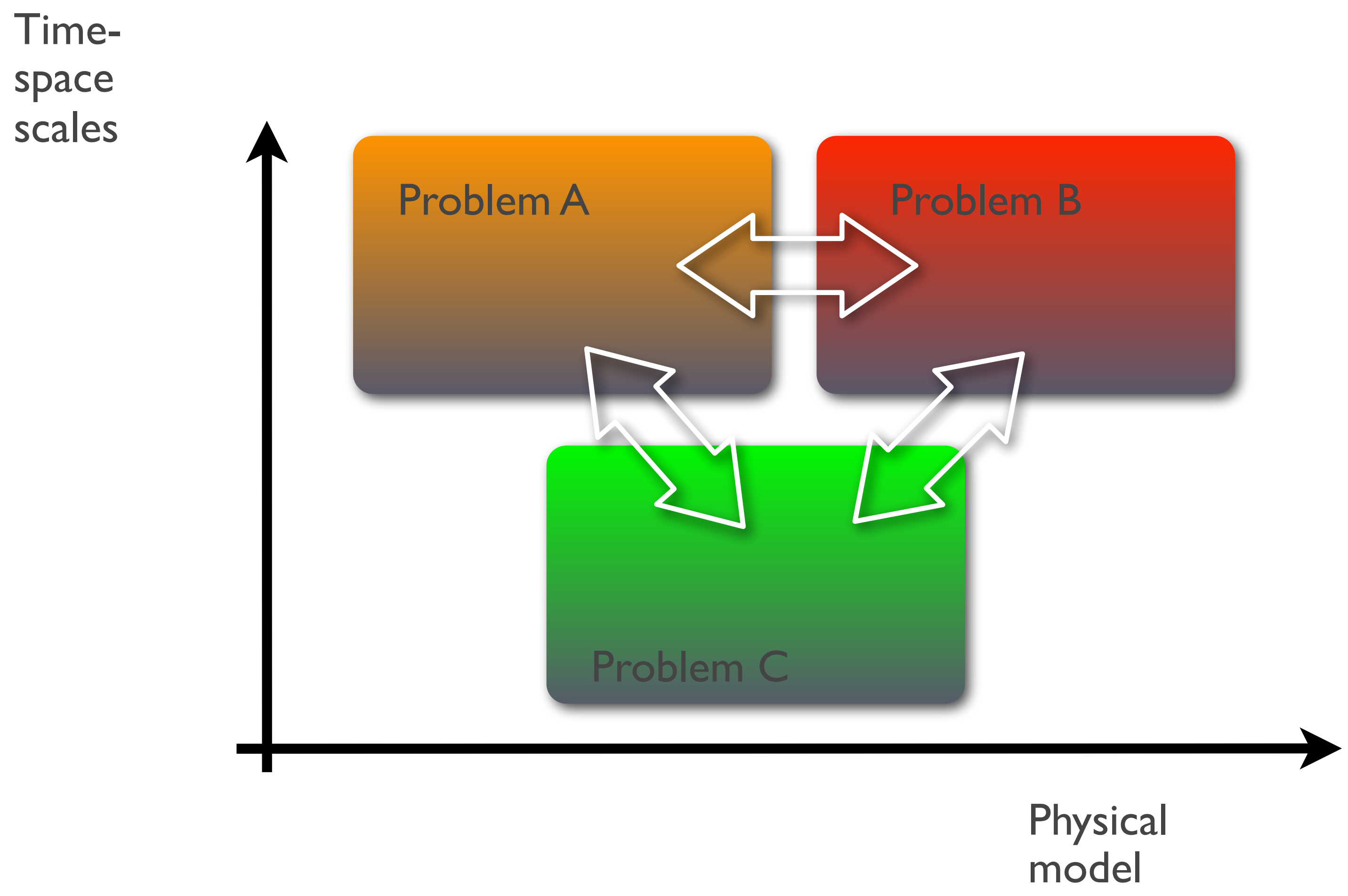


Second Case: Blocks

Grouping by :
- variables
- operators
- spacial regions features

Multiphysics and multiscale revisited

Multiphysics and multiscale revisited



Multiphysics and multiscale revisited

Examples:

Multi-scale, but same physics: Turbulent flows (RANS, LES):

One equation solved for the mean flow

One set of equations solved or modeled for the highly fluctuating turbulence perturbations (turbulent energy and dissipation, length scale, Reynolds tensor...)

Two-way coupling everywhere: turbulent shear stress and mean flow variables

Multiphysics and multiscale revisited

Examples:

Multi-scale, but same physics: Turbulent flows (RANS, LES):

One equation solved for the mean flow

One set of equations solved or modeled for the highly fluctuating turbulence perturbations (turbulent energy and dissipation, length scale, Reynolds tensor...)

Two-way coupling everywhere: turbulent shear stress and mean flow variables

Same scale, but **multi-physics**: Fluid-Structure interaction:

One equation solved for the fluid

One equation solved for the solid

Two-way coupling localized through the interface: wall position and wall force

Multiphysics and multiscale revisited

How do you define a physical system?

Just the governing equations?

Then what is multiphysics coupling?

Multiphysics and multiscale revisited

How do you define a physical system?

Equations + space/time domain + boundary/initial conditions

Multiphysics and multiscale revisited

How do you define a physical system?

Equations + space/time domain + boundary/initial conditions

After applying a numerical method,

Equations + space/time domain + boundary/initial conditions + discretization

Multiphysics and multiscale revisited

How do you define a physical system?

Equations + space/time domain + boundary/initial conditions

After applying a numerical method,

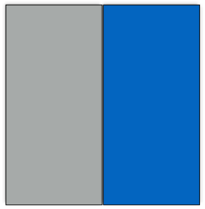
Equations + space/time domain + boundary/initial conditions **+ discretization**

This widens up the concept of "multi-physics coupling":

Two or more coupled problems, where at least one of the terms above varies.

Multiphysics and multiscale revisited

Very generally speaking and to fix ideas…



Contact domains:

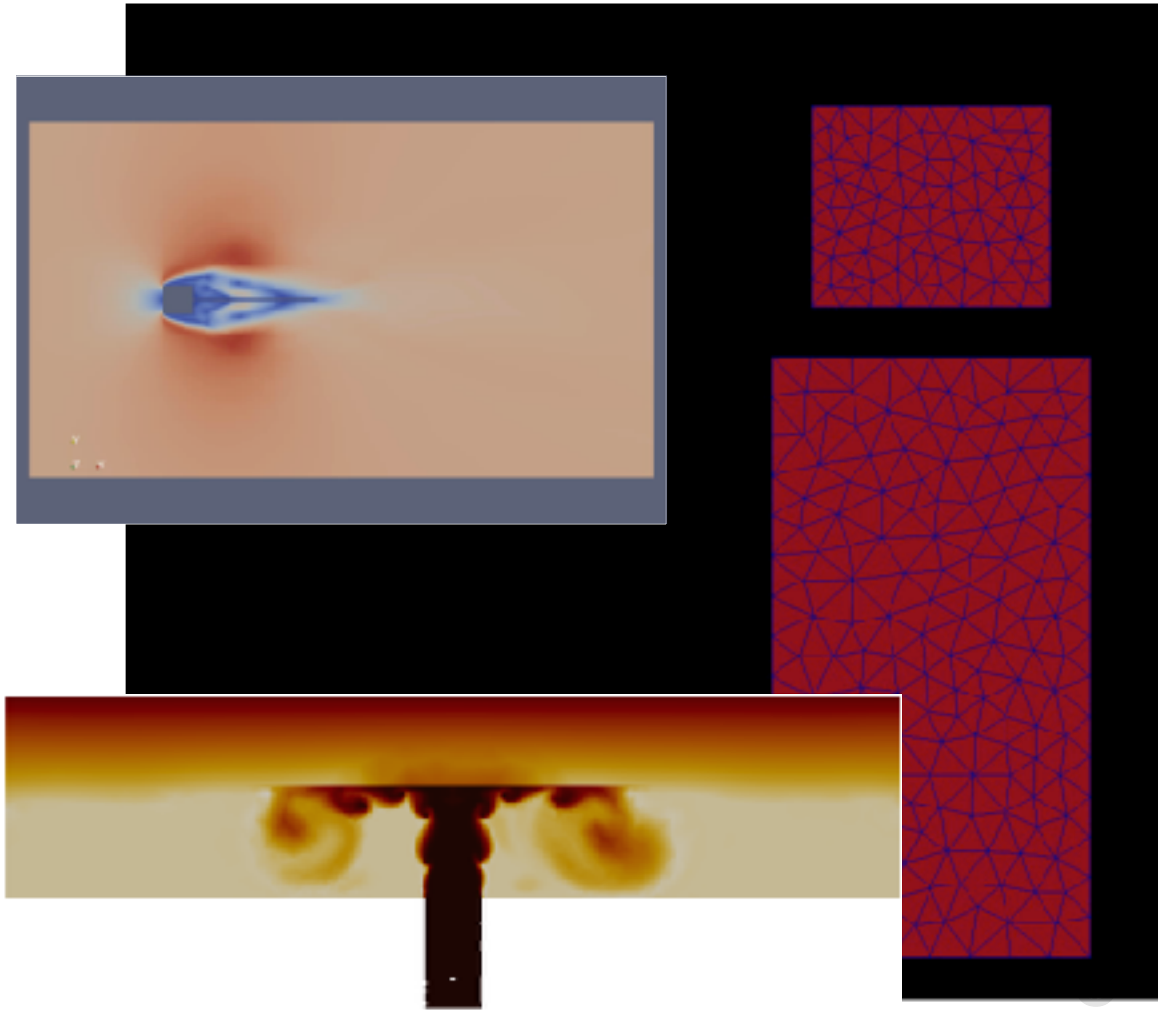
Fluid-structure interaction

Contact and impact problems

N-bodies collisions

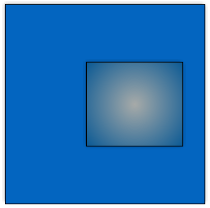
Heat transfer

Meshes can/cannot coincide



Multiphysics and multiscale revisited

Very generally speaking and to fix ideas…



Overlapping domains:

Overset meshes and Chimera

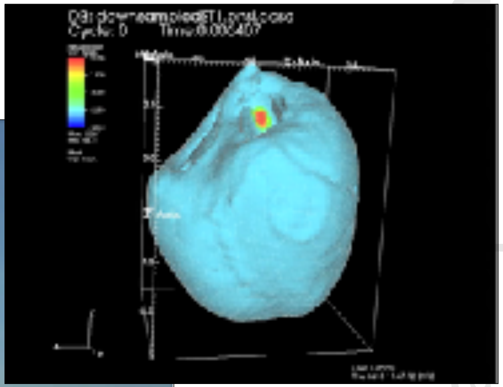
Electromechanical cardiac model

RANS modelled turbulence

Multi-scale problems

Particles and immersed bodies

Meshes can/cannot coincide



Multiphysics and multiscale revisited

Issues

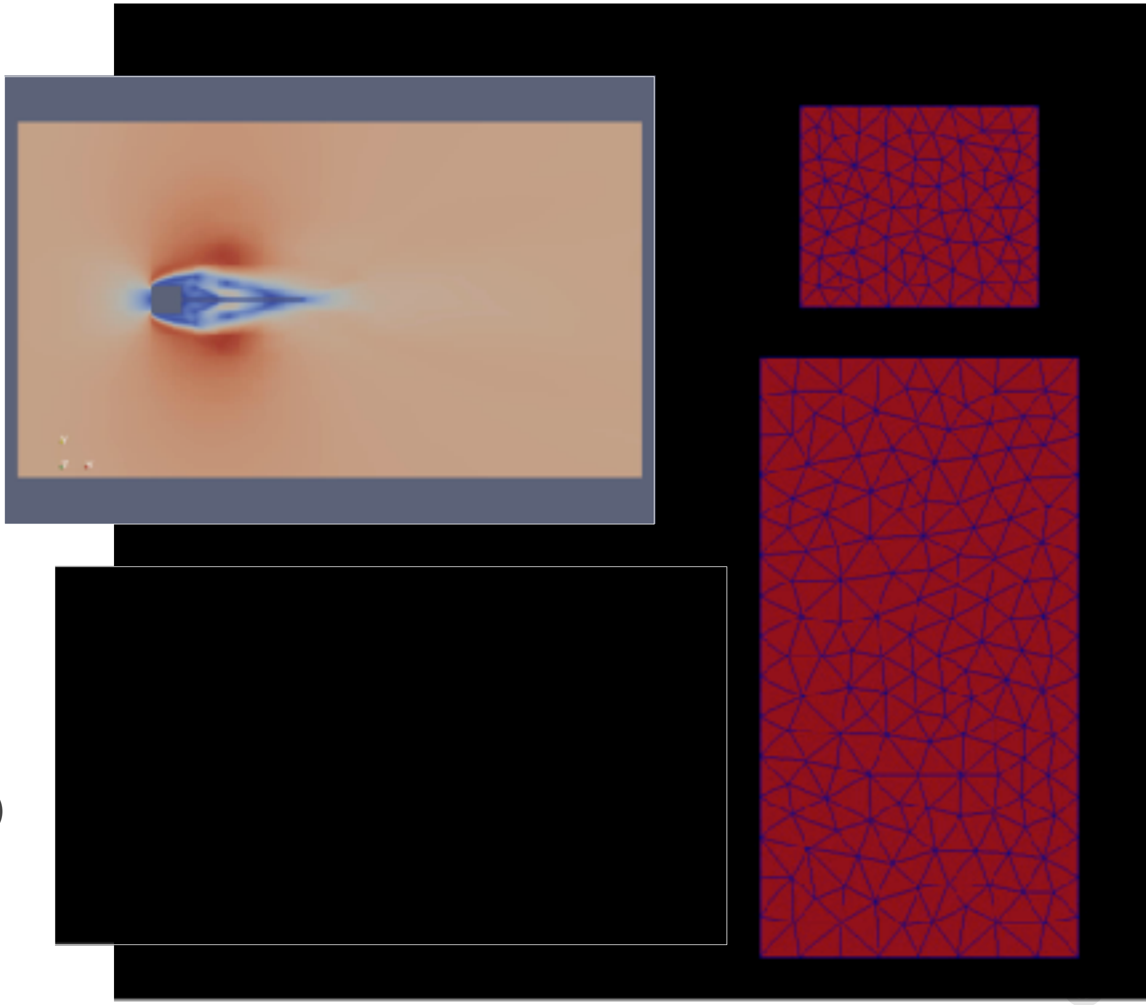
Coupling connectivity among MPI tasks

Numerically stable coupling algorithms

Preconditioners for the coupled scheme

Time-scale disparity

Synchronous/Asynchronous schemes
Coupling different codes (multi-codes)



Multiphysics and multiscale revisited

Coupling strategy: Fluid - Structure Interaction (FSI)

Fluid coupled with a solid

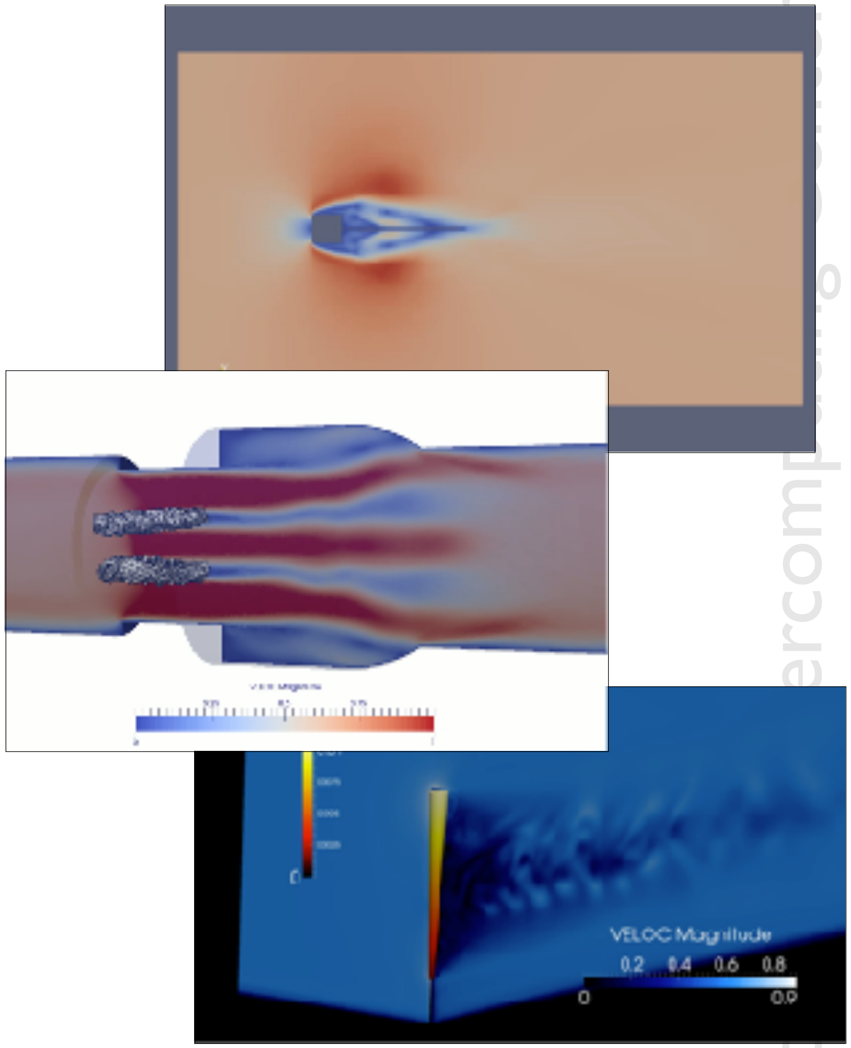
Solid deforming / moving upon the forces exerted by the fluid

Fluid domain changing as the solid is deformed

Mainly two strategies:

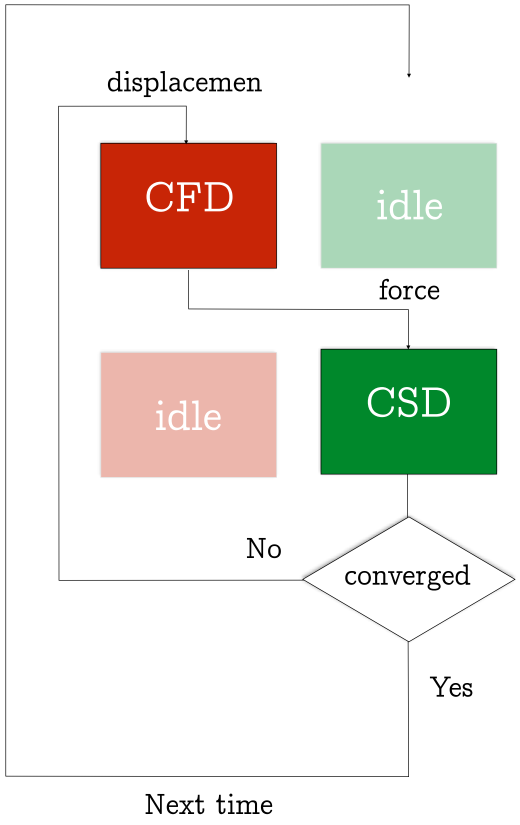
Immersed boundary method (IBM)

Arbitrary Lagrangian - Eulerian (ALE)

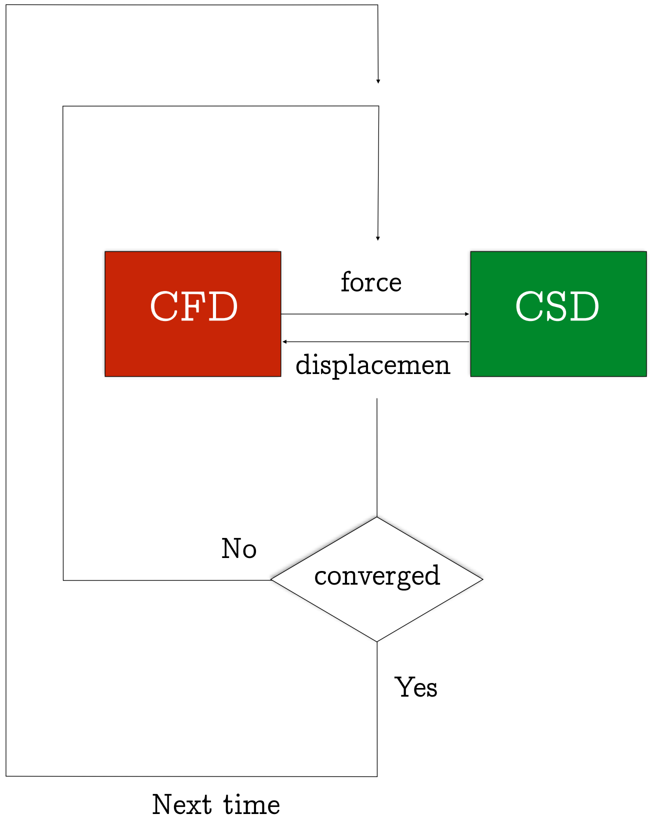


Multiphysics and multiscale revisited

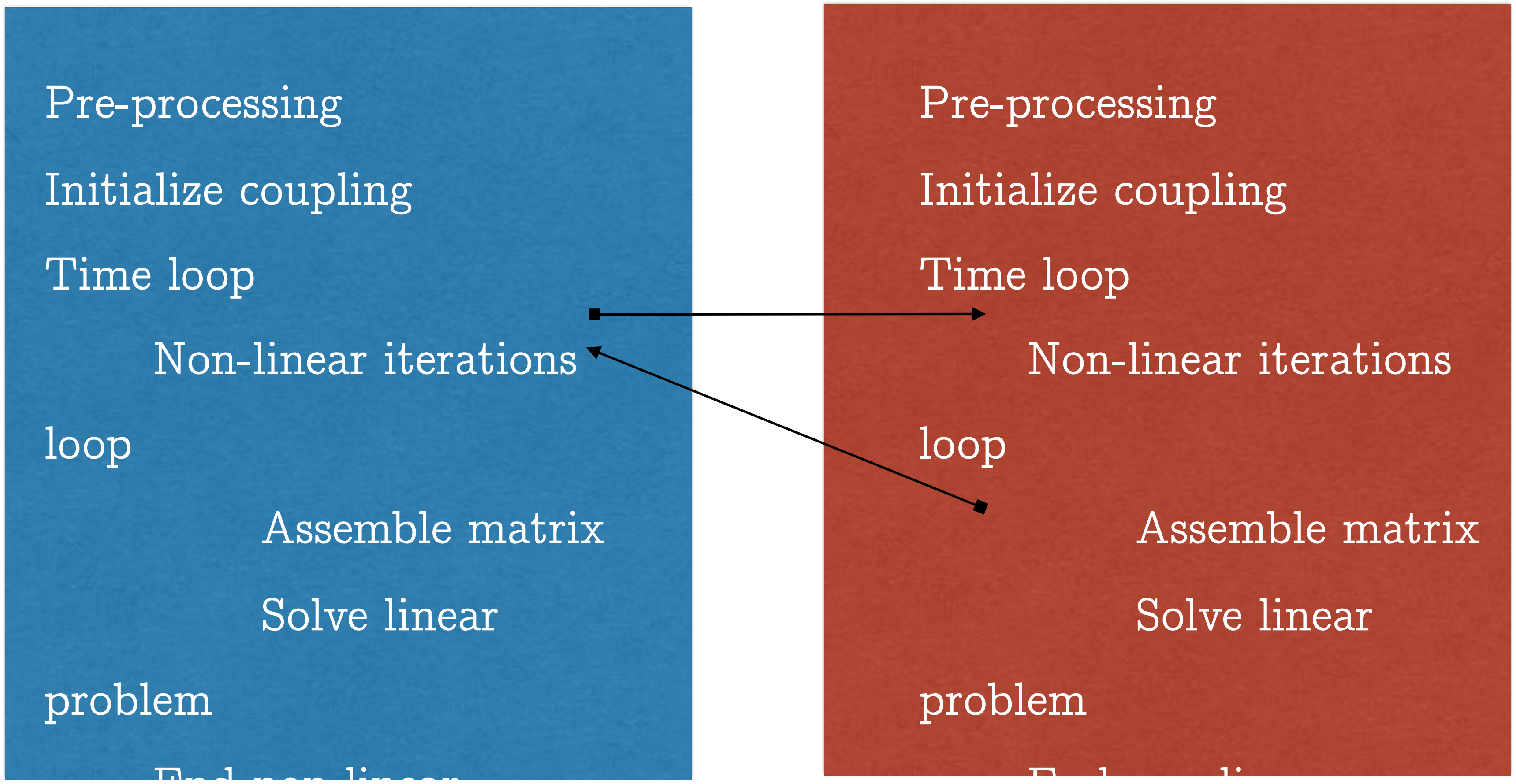
Gauss-Seidel approach



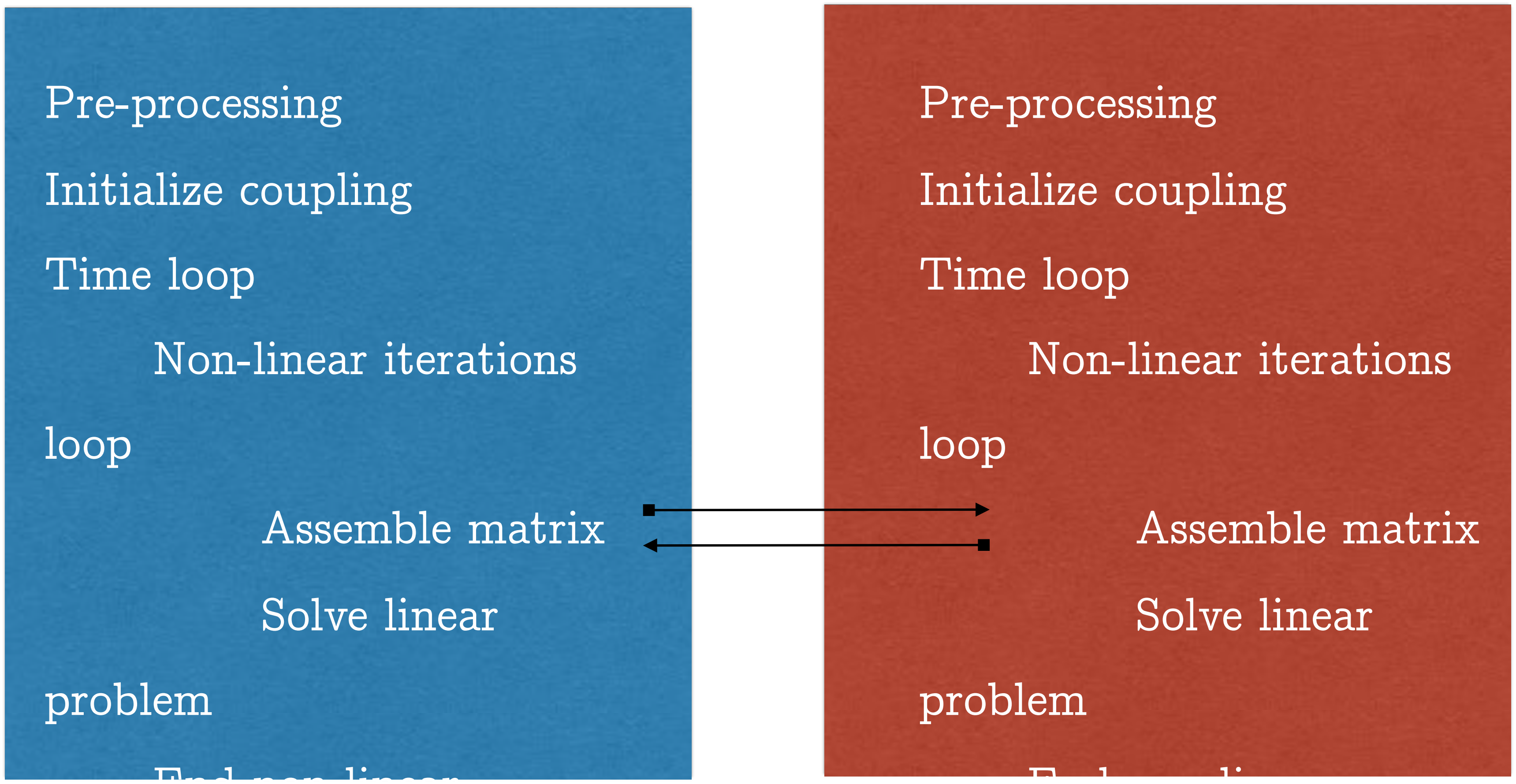
Jacobi approach



Gauss-Seidel approach



Jacobi approach

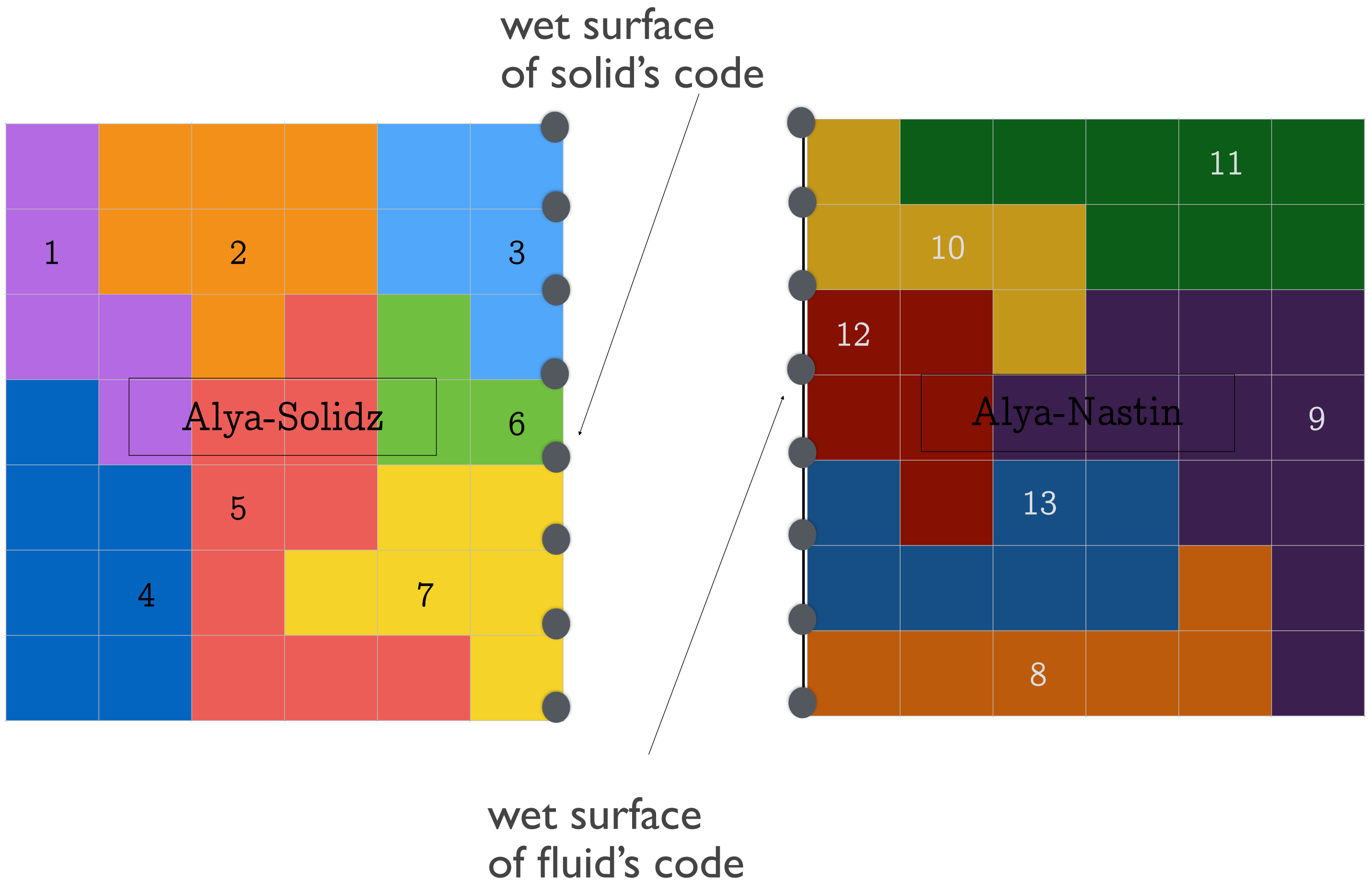


```
Pre-processing
Initialize coupling
Time loop
        Non-linear iterations
loop
                Assemble matrix
                Solve linear
problem
        End non-linear
```

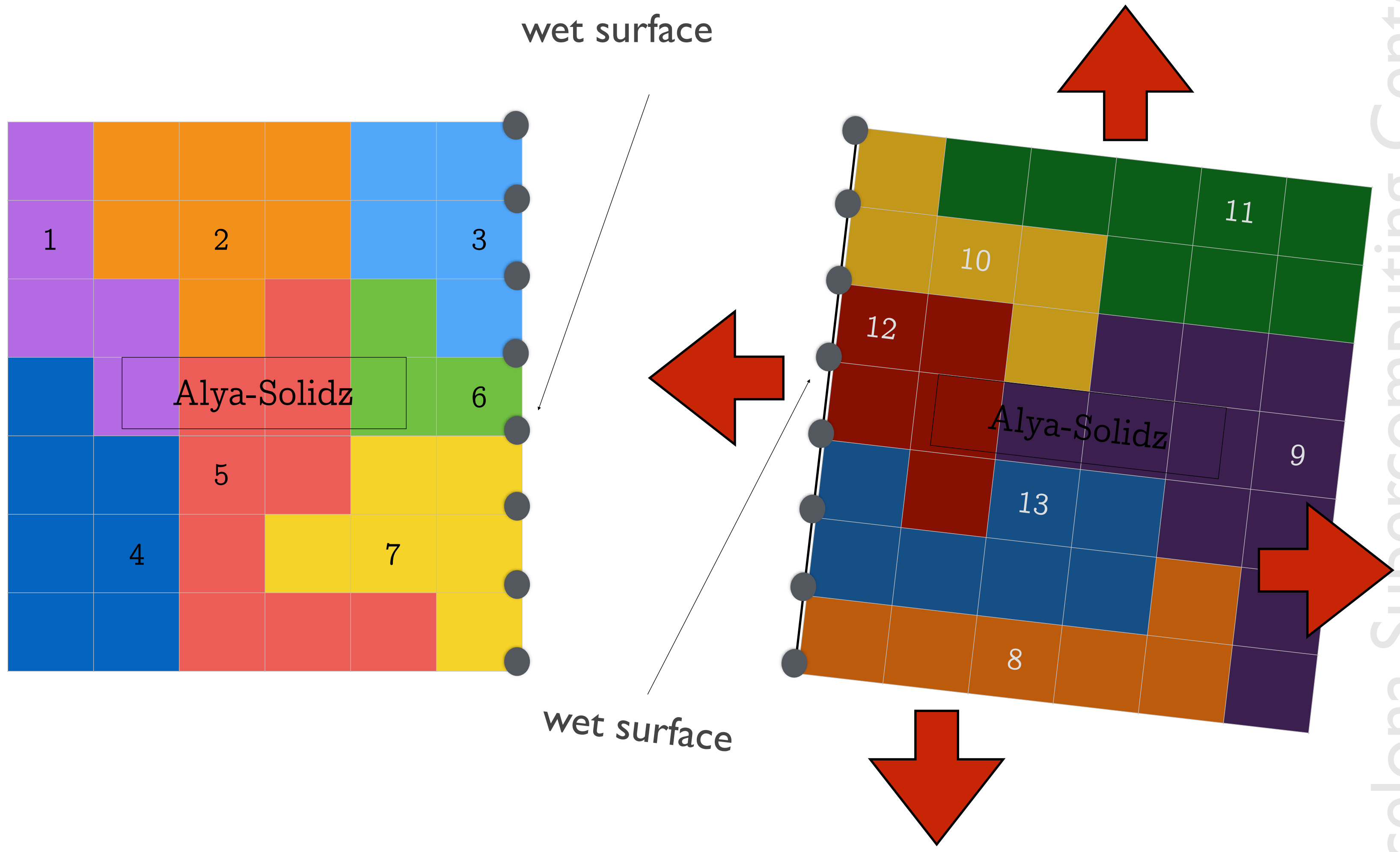
```
Pre-processing
Initialize coupling
Time loop
        Non-linear iterations
loop
                Assemble matrix
                Solve linear
problem
        End non-li
```

Multiphysics and multiscale revisited

Coupling strategy: FSI



Coupling strategy: Contact



HPCCamp 2017 - ECAR 2017
Introduction to computational mechanics in multiphysics

Mariano Vázquez

Barcelona Supercomputing Center

September 2017
FCEN - UBA
Buenos Aires
Argentina