Computational Mechanics

GOLAY Frédéric

Institut de Mathématique de Toulon (IMATH)
Responsable du parcours «Modélisation et Calculs Fluides-Structures» (MOCA),
de l’école d’ingénieur de l’université deToulon (SEATECH)

frederic.golay@univ-tln.fr


Computational Mechanics

Evolution of computeurs

- Computational Mechanics
- Numerical simulation process
- Finite Difference Method
- Variational approach
- Finite Element Method
- Finite Volume Method
- Galerkin Discontinuous Finite Element Method
**Computational Mechanics**

**Evolution of computational mechanics**

**Aim:**
- Reproduce to explain the inaccessible
- Plan before you realize

Replace Theory-Experiment by Theory-computation-Experiment

**Evolution:**
- Multiphysics
- Multiscale
- High Performance Computing

---

**Sources:** engineering.swan
ONERA & Lecture NF04 UTC

---

**Conservation of mass**

\[
\frac{d\rho}{dt} + \rho \frac{\partial \bar{\varepsilon} \cdot \bar{v}}{\partial t} = 0
\]

or

\[
\frac{\partial \rho}{\partial t} + \frac{\partial (\rho \bar{v})}{\partial x} = 0
\]

**Balance of linear momentum**

\[
\begin{aligned}
\nabla \cdot \bar{f} &= \rho \frac{\partial \bar{v}}{\partial t} \\
\bar{f} &= F \\
\bar{\n} &= \bar{c}
\end{aligned}
\]

**Balance of angular momentum**

\[
\begin{aligned}
\bar{\tau} &= \bar{\omega} \\
\bar{\omega} &= \bar{\omega}
\end{aligned}
\]

**Conservation of energy**

\[
\begin{aligned}
\rho \frac{d}{dt} (\bar{\varepsilon} - \bar{\sigma} : \bar{D} = r - \text{div} \bar{q})
\end{aligned}
\]

**Material derivative**

\[
\frac{d}{dt} \bar{\varepsilon} + \bar{\varepsilon} \cdot \bar{\nabla} \bar{\varepsilon} = \frac{\partial \rho}{\partial t} + \frac{\partial (\rho \bar{v})}{\partial x}
\]

**Terminology**

- \(\bar{\varepsilon}\): actual position
- \(\bar{\varepsilon}\): Eulerian variables
- \(\rho\): density
- \(\bar{v}\): velocity
- \(\bar{f}\): body force per unit volume acting in \(\Omega\)
- \(\bar{F}\): contact force per unit surface acting on \(\partial \Omega\)
- \(\bar{\n}\): outward normal vector of \(\partial \Omega\)
- \(\bar{\omega}\): acceleration
- \(\bar{\omega}\): Second order Cauchy stress tensor
- \(\bar{\n}\): specific internal energy
- \(\bar{\n}\): strain-rate (or stretching) tensor
- \(\bar{\n}\): heat flux
- \(\bar{\n}\): heat supply density

---

**Sources:** engineering.swan
ONERA & Lecture NF04 UTC
Computational Mechanics

Example: Wave breaking

Computational Mechanics

Example: Wave breaking, full Navier-Stokes Air-Water simulation

P. Lubin, S. Glockner I2M Bordeaux, France, software Thétis
80 Millions cells on 1024 cores
800 Millions cells on 4096 cores
Computational Mechanics

Example: Wave propagation, wave breaking ..... experiment

Olivier Kimmoun, ECM, Marseille France, workshop B’Waves 2014 Bordeaux
**Computational Mechanics**

*Example: Wave propagation, wave breaking .... Air-Water Euler model (Finite Volume)*

![Wave propagation diagram](attachment:image.png)

**Confrontation of several CFD codes**

- Bi-fluid Euler, VF
- Bi-fluid NSI, VF
- Bi-fluid pseudo-NSI, VF-VOF
- Shallow water, DF
- Irrotational fluid, BIEM
- Bi-fluid NSI, VF-LS
- Bi-fluid pseudo-NS, GD-LS

**Conférence de 2009**

*Nianga 2009 (Fluidbox)*

**Graphs**

- Jauge P4
- Water height in m
- Time in second

---

**References**

Computational Mechanics

Example, metal forming using finite elements (Forge3D – Cemef)

Computational Mechanics

Example of computation on GPU (DEM+Cuda)
Computational Mechanics

Realistic image rendering example applied to fluid mechanics

![Experiment](image1.png) ![Numerical simulation](image2.png)

Elie Hachem, Thierry Coupez, Cemef, code CimLib, anisotropic adaptive mesh refinement technic

Computational Mechanics

Overview of numerical approximation of partial differential equations

- Finite Difference Method (FDM)
- Finite Element Method (FEM)
- Finite Volume Method (FVM)
- Spectral Method
- Boundary Element Method (BEM)
- Lattice Boltzmann (LB)
- Smooth Particle Hydrodynamics (SPH)
- Discrete Element Method (DEM)
- Many others...

Finite Difference Method (FDM)
The oldest method, used for many physics
- Easy to implement
- Easy to make higher order
- Only applicable on structured grids
- Does not fit the geometry

Finite Element Method (FEM)
Usually used in structure mechanics
- Any order of accuracy
- Based on variational methods
- Applicable on unstructured/complex grids
- Naturally implicit, well-adapted to non-linear behaviour
- Excellent for diffusion dominated problem
- Naturally implicit, need sparse linear solver (can be expensive for large systems)
- Not conservative
- More complex

Finite Volume Method (FVM)
Usually used in fluid mechanics
- Applicable on unstructured/complex grids
- Naturally conservative, capture discontinuities
- Easy explicit formulation (parallel computation)
- Difficult to devise stable higher order scheme
- Not very accurate for diffusion dominated problem
- More complex
Numerical simulation process

What does computing engineer mean?

System analysis, modelization, ... Fluid flow

Mechanical / Mathematical Modelization

Model 1: Shallow water
\[
\begin{align*}
\frac{\partial h}{\partial t} + \text{div}(h\mathbf{u}) &= 0 \\
\frac{\partial (h\mathbf{u})}{\partial t} + \text{div}(h\mathbf{u} \otimes \mathbf{u}) + \frac{b^2}{2} &= -phVZ + \ldots
\end{align*}
\]

Model 2: Navier Stokes
\[
\begin{align*}
\text{div}(\mathbf{v}) &= 0 \\
\frac{\partial \mathbf{v}}{\partial t} + \nabla \mathbf{v} \cdot \mathbf{v} &= -\frac{1}{\rho} \nabla p + \mathbf{v} \Delta \mathbf{v}
\end{align*}
\]

Model 3: Euler two-phase
\[
\begin{align*}
\frac{\partial \mathbf{v}}{\partial t} + \text{div}(\rho \mathbf{v}) &= 0 \\
\frac{\partial \rho}{\partial t} + \text{div}(\rho \mathbf{v} \otimes \mathbf{v}) &= \rho \mathbf{g} \\
\frac{\partial \mathbf{v}}{\partial t} + \underline{\mathbf{v}} \cdot \nabla \mathbf{v} &= 0
\end{align*}
\]

Model 4: ...

Numerical simulation process

F. Golay
IRSTEA 15/11/2019 - 16/47 -
Numerical simulation process

What does computing engineer mean?

System analysis, modelization, ... → Fluid flow

Mechanical / Mathematical Modelization : Bifluid Euler

Numerical formulation, discrete approach

Model 1: FDM

Model 2: FEM

Model 3: FVM

Model 4: SPH

Numerical method, solver

Spatial discretisation: order, gradient, ...
Time discretization: explicit, implicit, ...
Solver: direct, iterative, ...
Domain decomposition, mesh refinement, ...
Numerical improvement, ...

Computational Mechanics
Numerical simulation process
Finite Difference Method
Variational approach
Finite Element Method
Finite Volume Method
Galerkin Discontinuous Finite Element Method

F. Golay
IRSTEA 15/11/2019
- 17/47 -
Numerical simulation process

What does computing engineer mean?

- System analysis, modelization,... -> Fluid flow
- Mechanical / Mathematical Modelization -> Bifluid Euler
- Numerical formulation, discrete approach -> FVM
- Numerical method, solver -> Explicit second order parallel

Software implementation process

- Programamation: C, fortran90, C++, ...
- Parallel: Cuda, MPI, ...
- Computer-aided software engineering, Scientific programming, versioning, ...

Preprocessing: can be very time-consuming
Numerical simulation process

What does computing engineer mean?

System analysis, modelization,... -> Fluid flow

Mechanical / Mathematical Modelization -> Bifluid Euler

Numerical formulation, discrete approach -> FVM

Numerical method, solver -> Explicit second order parallel

Software implementation process

Computation implementation: geometry, mesh, initial conditions, boundary conditions,...

Run !

HPC: High Performance Computing

Very critical analysis of the results
Numerical simulation process

What does computing engineer mean?

CFD = Computational Fluid Dynamics

CFD ≠ Color Fluid Dynamics (G. Allaire ?)

System analysis, modelization,…

Mechanical / Mathematical Modelization

Numerical formulation, discrete approach

Numerical method, solver

Software implementation process

Computation implementation: geometry, mesh, initial conditions, boundary conditions, …

Run!

Very critical analysis of the results
Numerical simulation process

What does computing engineer mean?

System analysis, modelization, ...

Mechanical / Mathematical Modelization

Numerical formulation, discrete approach

Numerical method, solver

Software implementation process

Computation implementation: geometry, mesh, initial conditions, boundary conditions, ...

Run !

Very critical analysis of the results

Software

Bad engineer

• Computational Mechanics
• Numerical simulation process
• Finite Difference Method
• Variational approach
• Finite Element Method
• Finite Volume Method
• Galerkin Discontinuous Finite Element Method

F. Golay
IRSTEA 15/11/2019
- 25/47 -
Numerical simulation process

What does computing engineer mean?

System analysis, modelization, ...

Mechanical / Mathematical Modelization

Numerical formulation, discrete approach

Numerical method, solver

Software implementation process

Computation implementation: geometry, mesh, initial conditions, boundary conditions, ...

Run !

Very critical analysis of the results

Most engineer

Numerical simulation process

What does computing engineer mean?

System analysis, modelization, ...

Mechanical / Mathematical Modelization

Numerical formulation, discrete approach

Numerical method, solver

Software implementation process

Computation implementation: geometry, mesh, initial conditions, boundary conditions, ...

Run !

Very critical analysis of the results

Engineer in computational Mechanics
Finite Difference Method

Taylor’s Theorem

Let \( u \) be an \( n \)-times differentiable function on an open interval containing the points \( x+h \) and \( x \), then:

\[
u(x + h) = u(x) + hu'(x) + \frac{h^2}{2!} u''(x) + \frac{h^3}{3!} u'''(x) + \ldots + \frac{h^n}{(n-1)!} u^{(n-1)}(x) + O(h^n)
\]

Where \( O(h^n) = \frac{h^n}{n!} u^{(n)}(c) \) and \( c \) is between \( x \) and \( x+h \)

Finite difference approximation of the first derivative of \( u \)

Forward first order finite difference approximation of \( u' \)

\[
u'(x) = \frac{u(x + h) - u(x)}{h} + O(h)
\]

Backward first order finite difference approximation of \( u' \)

\[
u'(x) = \frac{u(x) - u(x - h)}{h} + O(h)
\]

Centered first order finite difference approximation of \( u' \)

\[
u'(x) = \frac{u(x + h) - u(x - h)}{2h} + O(h)
\]

Finite difference approximation of the second derivative of \( u \)

\[
u(x + h) = u(x) + hu'(x) + \frac{h^2}{2!} u''(x) + \frac{h^3}{3!} u'''(x) + \ldots + \frac{h^n}{(n-1)!} u^{(n-1)}(x) + O(h^n)
\]

\[
u(x - h) = u(x) - hu'(x) + \frac{h^2}{2!} u''(x) - \frac{h^3}{3!} u'''(x) + O(h^4)
\]

\[
u(x + h) + \nu(x - h) = 2u(x) + h^2 u''(x) + O(h^4)
\]

\[
u''(x) = \frac{\nu(x + h) - 2\nu(x) + \nu(x - h)}{h^2} + O(h^2)
\]
Finite Difference Method  Elliptic equation

1D discretization of the Laplacian operator

\[ \Delta u = 0 \]
\[ \frac{\partial^2 u}{\partial x^2} = 0 \]
\[ \text{Notation } u(x_i) = u_i \]
\[ \forall i = 1, n \quad u_i''(x_i) \approx \frac{u_{i+1} - 2u_i + u_{i-1}}{h^2} = 0 \]

Finite Difference Method  Elliptic equation

2D discretization of the Laplacian operator

\[ \Delta u(x, y) = 0 \]
\[ \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0 \]
\[ u(x + h, y + h) = \sum_{jk} h_j h_k \frac{\partial^2 u(x, y)}{\partial x^2} + \frac{\partial^2 u(x, y)}{\partial y^2} + O(h_j^2) \]
\[ u(x - h, y) = u(x, y) - h \frac{\partial u(x, y)}{\partial x} + \frac{h^2}{2} \frac{\partial^2 u(x, y)}{\partial x^2} + O(h^3) \]
\[ u(x, y + h) = u(x, y) + h \frac{\partial u(x, y)}{\partial y} + \frac{h^2}{2} \frac{\partial^2 u(x, y)}{\partial y^2} + O(h^3) \]
\[ u(x, y - h) = u(x, y) - h \frac{\partial u(x, y)}{\partial y} + \frac{h^2}{2} \frac{\partial^2 u(x, y)}{\partial y^2} + O(h^3) \]
\[ \frac{\partial^2 u(x, y)}{\partial x^2} + \frac{\partial^2 u(x, y)}{\partial y^2} = \frac{u(x + h, y + h) - 2u(x, y) + u(x - h, y)}{h^2} + O(h_j^2) + O(h_k^2) \]

Finite Difference Method  Elliptic equation

- Computational Mechanics
- Numerical simulation process
- Finite Difference Method
- Variational approach
- Finite Element Method
- Finite Volume Method
- Galerkin Discontinuous Finite Element Method

F. Golay
IRSTEA 15/11/2019
- 31/47 -
**Finite Difference Method**

2D five points stencil

Notation $u(x, y) = u_{i,j}$

\[ \forall i, j = 1, n, \Delta u(x, y) = u_{i+1,j} + u_{i-1,j} - 4u_{i,j} + u_{i+1,j} + u_{i,j+1} = 0 \]

**Elliptic equation**

\[ u(x,0) = u(x) \quad \frac{\partial}{\partial t} \Delta u = 0 \]

**Parabolic equation**

1D discretization

\[ \begin{cases} \frac{\partial u}{\partial t}(x,t) - \lambda \Delta u = 0 \\ x(x,0) = u_0(x) \end{cases} \]

Notation $u(x, t^n) = u^n_i$

First order approximation

\[ \frac{\partial u}{\partial t} \approx \frac{u^{n+1} - u^n}{\delta t} \]

**Time discretisation: explicit Euler scheme**

\[ \frac{u^{n+1} - u^n}{\delta t} = \lambda \frac{\partial^2 u}{\partial x^2} (x, t^n) \approx \frac{u^{n+1}_{i+1} - 2u^n_i + u^{n+1}_{i-1}}{h^2} \rightarrow u^{n+1}_i = u^n_i + \frac{\lambda \delta t}{h^2} (u^n_{i+1} - 2u^n_i + u^n_{i-1}) \]

**Time discretisation: implicit Euler scheme**

\[ \frac{u^{n+1} - u^n}{\delta t} + \lambda \frac{\partial^2 u}{\partial x^2} (x, t^n) = \frac{u^{n+1}_{i+1} - 2u^{n+1}_i + u^{n+1}_{i-1}}{h^2} \rightarrow u^{n+1}_i = u^n_i + \frac{h^2}{\lambda \delta t} \left( u^n_{i+1} - 2u^n_i + u^n_{i-1} \right) \]
Finite Difference Method  Parabolic equation

Time discretisation: semi-implicit scheme

\[
\frac{u_i^{n+1} - u_i^n}{\Delta t} = \theta \frac{\partial^2 u}{\partial x^2}(x_i, t^n) + (1 - \theta) \frac{\partial^2 u}{\partial x^2}(x_i, t^{n+1}) \approx \theta \lambda \frac{u_i^{n+1} - 2u_i^n + u_i^{n-1}}{h^2} + (1 - \theta) \lambda \frac{u_i^{n+1} - 2u_i^{n+1} + u_i^{n+2}}{h^2}
\]

\[
A = \begin{pmatrix}
2 & -1 & 0 & \cdots & 0 \\
-1 & 2 & -1 & \cdots & 0 \\
0 & \ddots & \ddots & \ddots & \vdots \\
\vdots & \ddots & \ddots & \ddots & \vdots \\
0 & \cdots & 0 & -1 & 2 \\
\end{pmatrix}
\]

\[
\frac{u_1}{u_0} = \begin{pmatrix}
u_0 \\
u_1 \end{pmatrix}
\]

0=0 explicit scheme: Forward Euler  Conditionally stable
0=1 implicit scheme: Backward Euler  Unconditionally stable
0=0.5 semi implicit scheme: Crank-Nicholson  Unconditionally stable for 0≥0.5

Variational approach

Principle

\[
\int_{\Omega} \left( \frac{\partial w}{\partial t} + \text{div} F(w) \right) dv = S \quad \text{in } \Omega
\]

\[
\int_{\Omega} \phi \left( \frac{\partial w}{\partial t} + \text{div} F(w) \right) dv = \int_{\Omega} \phi S \ dv \quad \forall \phi \quad ...
\]

The computational domain \( \Omega \) is split in several cells or elements

\[
\int_{\Omega} \cdots dv = \sum_{\Omega_i} \int_{\Omega_i} \cdots dv
\]

\[
\sum_{\Omega_i} \int_{\Omega_i} \frac{\partial \phi}{\partial t} dv + \sum_{\Omega_i} \int_{\Omega_i} \phi \text{div} F(w) dv = \sum_{\Omega_i} \int_{\Omega_i} \phi S dv
\]

\[
\sum_{\Omega_i} \int_{\Omega_i} \nabla \phi : F(w) dv + \sum_{\Omega_i} \int_{\Omega_i} \text{div}(F(w)) dv = \sum_{\Omega_i} \int_{\Omega_i} \phi S dv
\]

\[
\text{div}(F(w)) = \nabla \phi : F(w) + \phi \text{div} F(w)
\]

Green's formula

\[
\sum_{\Omega_i} \int_{\Omega_i} \nabla \phi : F(w) dv + \sum_{\Omega_i} \int_{\Omega_i} \phi F_n dv = \sum_{\Omega_i} \int_{\Omega_i} \phi S dv
\]

\[
\sum_{\Omega_i} \int_{\Omega_i} \nabla \phi : F(w) dv + \sum_{\Omega_i} \int_{\Omega_i} \phi F_n ds = \sum_{\Omega_i} \int_{\Omega_i} \phi S dv
\]
### Finite Element Method

#### Principle

\[
\frac{dw}{dt} + \text{div}(F(w)) = \mathbf{S} \quad \text{in } \Omega
\]

\[
\sum_{\gamma \in \partial \Omega} \int_{\gamma} \varphi \frac{dw}{dt} \, ds - \sum_{\gamma \in \partial \Omega} \int_{\gamma} \varphi \text{div}(F) \, ds + \sum_{\gamma \in \partial \Omega} \int_{\gamma} \varphi \mathbf{S} \cdot \mathbf{n} \, ds = \sum_{\gamma \in \partial \Omega} \int_{\gamma} \varphi \mathbf{S} \, ds
\]

The solutions \( w \) are approximated in each Element by a polynomial approximation

\[
\tilde{w}_e(x,y,z) \approx \sum_{i} N_i(x,y,z) \tilde{w}_i
\]

Where \( N_i \) are chosen polynomials of order 1, 2, 3, ... called interpolation functions

- **Remark 1**: \( N_i = 1 \) on node \( i \) and \( : N_i = 0 \) on other nodes
- **Remark 2**: the approximation of \( w \) is continuous!
- **Remark 3**: the approximation of \( \nabla w \) is NOT continuous!

\( \tilde{\varphi} \) is chosen as a continuous test function

\[
\sum_{e} \int_{\Omega_e} \tilde{\varphi} \cdot \tilde{F} \, dv = \sum_{e} \int_{\partial \Omega_e} \tilde{\varphi} \cdot \tilde{N} \mathbf{n} \, ds \quad \text{Neumann condition}
\]

Using the Galerkin's rule the test functions are chosen such that:

\[
\tilde{\varphi}(x,y,z) \approx \sum_{i} N_i(x,y,z) \tilde{\varphi}_i
\]

\[
\sum_{e} \int_{\Omega_e} N_i \tilde{\varphi} \, dv - \sum_{e} \int_{\partial \Omega_e} N_i \tilde{\varphi} \cdot \tilde{N} \mathbf{n} \, ds = \sum_{e} \int_{\partial \Omega_e} N_i \tilde{\varphi} \cdot \mathbf{n} \, ds \quad \text{Neumann condition}
\]

Therefore taking successively each component of the test functions equal to 1 and the others to zero, we obtain a system of equations of size : number of degree of freedom x number of nodes

---

### Finite Element Method

#### Some usual finite element types

**1D**

- Linear: L2, L3

**2D**

- Quadratic: L3, T3, Q4, T6, Q8

**3D**

- Linear: H8, P6, TH5, TH4
- Quadratic: H20, P15
Finite Element Method

Reference Element

\[ x(x, y, z) = \sum_{\text{nodes of } e} \bar{N}(x, y, z) \bar{x}_i \]

\( \bar{N}_i \) denote the shape functions

In the case of isoparametric element, we get \( \bar{N}_i = N_i \)

\[ \int_{\Omega_e} \ldots \, dxdy = \int_{\Omega} \ldots \, \det J d\xi d\eta \]

Where \( J \) denotes the Jacobian of the transformation

Numerical integration

\[ \int_{\eta_i} f(\xi, \eta) \, d\xi d\eta = \sum_{\text{Gauss points}} \omega_{PG} f(\xi_{PG}, \eta_{PG}) \]

According to the Gaussian quadrature rule, \( \omega_{PG} \) denotes the weights of each Gaussian point

Finite Element Method

Many other notions specific to FEM to handle ...

- Mesh
- Conforming mesh
- Assembly
- Lumping
- Post processing analysis
- ...

Many other notions to handle in the framework of FEM

- Boundaries conditions
- Time discretisation
- Linear solver: direct, iterative
- Non-linearities, tangent operator
- Convergence
- A posteriori error
- Domain decomposition
- Adaptive mesh refinement
- HPC
- ...
Finite Volume Method

Principle

\[
\frac{d\bar{w}}{dt} + \text{div} F(\bar{w}) = \bar{S} \quad \text{in } \Omega
\]

The solutions \(\bar{w}\) are approximated in each volume by an average value

\[
\bar{w}_k(x, y, z) = \frac{1}{|\Omega_k|} \int_{\Omega_k} \bar{w} \, dv
\]

\(\bar{w}\) is chosen successively as constant on volume \(\Omega_k\) and null elsewhere

\[
\Rightarrow \nabla \bar{w} = 0
\]

\[
\Rightarrow \int_{\Omega_k} \frac{d\bar{w}_k}{dt} \, dv + \int_{\partial\Omega_k} \bar{F}_n \, ds = \int_{\Omega_k} \bar{S} \, dv
\]

\[
\Rightarrow \frac{d\bar{w}_k}{dt} + \int_{\partial\Omega_k} \bar{F}(\bar{w}_k) \bar{n} \, ds = \bar{S} \quad \bar{F}(\bar{w}_k) \text{ is NOT defined on } \partial \Omega_k!
\]

Finite Volume Method

Fluxes computation

\[
\int_{\partial\Omega_k} \bar{F}_n \, ds \approx \sum_{\text{edge}} f(\bar{w}_s, \bar{w}_i, \bar{n}_{sh})
\]

• Centered
• Roe
• Lax-Friedrichs
• Engquist-Osher
• Lax-Wendroff
• Godunov: exact Riemann solver
• …
Finite Volume Method

Many other notions specific to FVM to handle ...

• Structured Mesh, unstructured mesh
• Second order reconstruction (MUSCL)
• Slope limiters
• Post processing analysis
• ...

Many other notions to handle in the framework of FVM

• Boundaries conditions
• Time discretisation
• Linear solver: direct, iterative, multigrid
• Interface sharpening
• Convergence
• A posteriori error
• Domain decomposition, parallel computing
• Adaptive mesh refinement
• HPC
• ...

Galerkin Discontinuous Finite Element Method

Principle

\[ \frac{d\mathbf{w}}{dt} + \nabla \cdot \mathbf{F}(\mathbf{w}) = \mathbf{S} \text{ in } \Omega \]

\[ \sum_{e} \phi \frac{d\mathbf{w}}{dt} \text{ dt} - \sum_{e} \sum_{\partial \Omega} \mathbf{w} \cdot \mathbf{n} \text{ dv} + \sum_{e} \sum_{\partial \Omega} \phi \mathbf{F} \cdot \mathbf{n} \text{ ds} = \sum_{e} \sum_{\partial \Omega} \phi \mathbf{S} \text{ dv} \]

The solutions \( \mathbf{w} \) are approximated in each Element by a polynomial approximation of order \( p \)

\[ \mathbf{w}_e(x,y,z) = \sum_{i,j,k} a_{ijk} x^i y^j z^k \]

Rmk 1: the approximation of \( \mathbf{w} \) NOT is continuous!

Rmk 2: We present a monomial approach, easier to implement, but harder to physically understand. A nodal approximation is still possible

\[ \phi \] is chosen successively as a DIScontinuous test function on element \( \Omega_e \) and null elsewhere

Using the Galerkin’s rule the test functions are also chosen such that:

\[ \sum_{e} \phi \frac{d\mathbf{w}}{dt} \text{ dt} - \sum_{e} \sum_{\partial \Omega} \mathbf{w} \cdot \mathbf{n} \text{ dv} + \sum_{e} \sum_{\partial \Omega} \phi \mathbf{F} \cdot \mathbf{n} \text{ ds} = \sum_{e} \sum_{\partial \Omega} \phi \mathbf{S} \text{ dv} + \nabla \phi \]

therefore taking successively each component of the test functions equal to 1 and the others to zero, we obtain a system of equations of size : number of degree of freedom x number of coefficient of the polynomial approximation
Galerkin Discontinuous Finite Element Method

Fluxes Computation

\[ \int_{\Omega} \varphi_n \frac{dW}{dt} \, dv - \int_{\Omega} \nabla \varphi_n : \mathbf{F} \, dv + \int_{\partial \Omega} \varphi_n \mathbf{F} \cdot \mathbf{n} \, ds = \int_{\Omega} \varphi_s \mathbf{S} \, dv \quad \forall \varphi_n \]

One more time, the fluxes are not defined at the interfaces between elements. An approximation has to be chosen in order to handle with the jump across the interface.

Reference Element

\[ \tilde{x}(x,y,z) \approx \sum \tilde{N}(x,y,z) \tilde{x}_i \]

\[ \int_{\Omega} \cdots \, dx dy = \int_{\Omega} \cdots \, \det J \, d\tilde{x} d\tilde{y} \]

Here we use a nodal approximation, often of order 1.

Numerical integration

\[ \int_{\Omega} f(\xi,\eta) \, d\xi d\eta = \sum_{\text{Gauss points}} \omega_{PQ} f(\xi_{PQ}, \eta_{PQ}) \]

According to the Gaussian quadrature rule, \( \omega_{PQ} \) denotes the weights of each Gaussian point. The number of Gaussian points is adapted to the order of the polynomial approximation.

Galerkin Discontinuous Finite Element Method

Many other notions specific to GD to handle ...

- Mesh, interface meshing
- Non conforming mesh
- Assembly
- Lumping
- Penalization
- Hp adaptation
- Post processing analysis
- ...

Many other notions to handle in the framework of GD

- Boundaries conditions
- Time discretisation
- Linear solver: direct, iterative
- Non-linearities, tangent operator
- Convergence
- A posteriori error
- Domain decomposition
- Adaptive mesh refinement
- HPC
- ...

...
Unstructured finite volume, isothermal eulerian bifluid flow
Jeuck, Charrier, Bonelli, Golay 2019

unstructured DDFV+ Level Set, Navier-Stokes flow + penalization
Lakhlii, Bonelli, Golay, Galusinski, 2015

structured finite volume+ Level Set, MAC scheme, Stokes flow + penalization

La chouette, Golay, Bonelli 2009

Finite volume (Fluent),
Navier-Stokes
flow+Turbulence+
remeshing
Mercier, Bonelli,
Anselmet, 2013