# Lagrange-flux schemes: reformulating second-order accurate Lagrange+remap schemes for higher operational intensity and improved SIMD treatment

Florian De Vuyst, Thibault Gasc, Renaud Motte,

Mathieu Peybernes, Raphaël Poncet

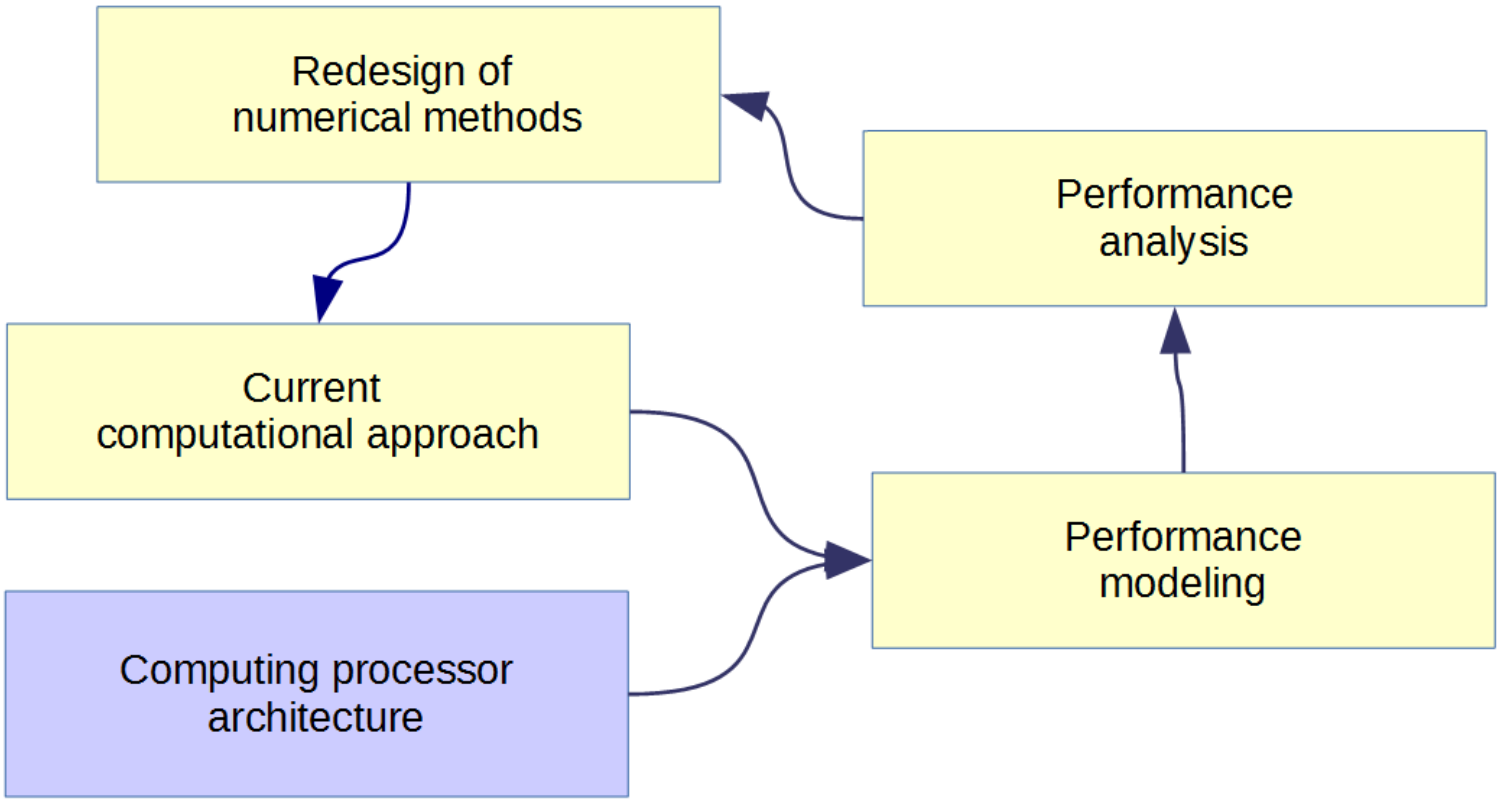
CMLA, ENS Cachan Université Paris-Saclay and CNRS,
Maison de la Simulation, CEA DAM DIF, CEA Saclay and CGG

RS SimRace IFPEN – December 2015, 10th

# Context

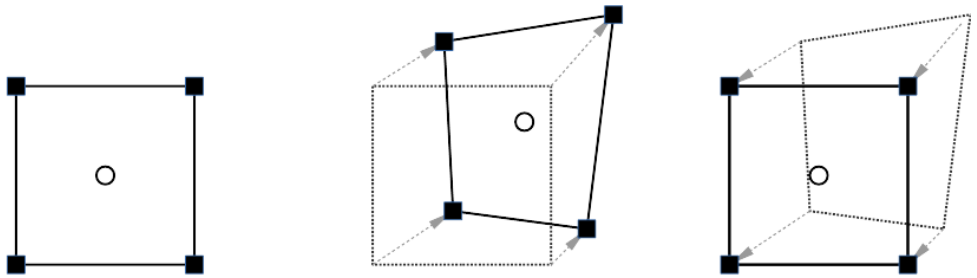
- High Performance Computing / Node-based multi-core performance
- Application: Compressible multiphysics CFD and multimaterial flows
- Miniapp called `Shy` developed in our team

# Our belief: rational process of evolution of codes



# Performance issues of (staggered) Lagrange+remap schemes

(Remember Thibault Gasc's talk on Tuesday)




- Staggered variables induce too much memory transfers
- Geometrical remapping: no SIMD, conditional branching
- Alternating direction (AD) strategies: more memory transfers
- Interface reconstruction (IR): array indirections, no SIMD

# Requirement

Build an alternative solver:

- with collocated (cell-centered) variables
- based on a Lagrangian flow description
- second-order accurate
- involving only geometry from the reference (Eulerian) grid
- suitable for SIMD
- extensible to multimaterial flows
- easily extensible to 3D

# 2D compressible Euler equations

$$\partial_t U_\ell + \nabla \cdot (U_\ell \, \boldsymbol{u}) + \nabla \cdot \boldsymbol{\pi}_\ell = 0, \quad \ell = \rho, \rho u_1, \rho u_2, \rho E.$$

Vector of conservative variables: $U = (\rho, \rho \boldsymbol{u} = (\rho u_i)_i, \rho E)$,

$\boldsymbol{\pi}_\rho = \vec{0}$,
$\boldsymbol{\pi}_{\rho u_1} = (p, 0)$,
$\boldsymbol{\pi}_{\rho u_2} = (0, p)$,
$\boldsymbol{\pi}_{\rho E} = p\boldsymbol{u}$.

Perfect gas EOS: $p = (\gamma - 1)\left(\rho E - \frac{1}{2}\rho |\boldsymbol{u}|^2\right)$.

# Reinterpretation of incremental remapping

- Let $\boldsymbol{v}$ be the velocity vector field used for Lagrangian mesh deformation.

i) Backward "play-back" convection (under Lagrangian description) :

$$\partial_t U + \nabla \cdot (-\boldsymbol{v}\, U) = 0$$

ii) Forward "play" convection (under Eulerian description) :

$$\partial_t U + \nabla \cdot (+\boldsymbol{v}\, U) = 0$$

$\Rightarrow$ Step ii) generates convective fluxes through (Eulerian) cell edges.

# Finite volume writing of Lagrange+remap schemes

$$\underline{(U_\ell)_K} \leftarrow \underline{(U_\ell)_K} - \frac{\Delta t}{|\underline{K}|} \underbrace{\sum_{\underline{A} \subset \partial \underline{K}} |\underline{A}| \boldsymbol{v}_A \cdot \boldsymbol{\nu}_A \, \underline{(U_\ell)_A}}_{\text{Convective Eulerian fluxes}} - \frac{\Delta t}{|\underline{K}|} \underbrace{\sum_{\overline{\underline{A}} \subset \partial \overline{\underline{K}}} |\overline{\underline{A}}| \, \overline{\underline{(\boldsymbol{\pi}_\ell)_A}} \cdot \overline{\underline{\boldsymbol{\nu}_A}}}_{\substack{\text{Pressure-type fluxes} \\ \text{on Lagrangian edges}}}.$$

NB:

1. Writing already better for SIMD treatment
2. But still some (Lagrangian) geometrical elements coming from the Lagrangian solver.

# Method of lines [Schiesser 1991]

By making $\Delta t > 0$ tend to zero, we get the method of lines

$$\frac{d(U_\ell)_K}{dt} = -\frac{1}{|K|}\sum_{A\subset\partial K} |A|\,\boldsymbol{u}_A\cdot\nu_A(U_\ell)_A - \frac{1}{|K|}\sum_{A\subset\partial K} |A|\,(\boldsymbol{\pi}_\ell)_A\cdot\nu_A.$$

NB: from now on, all is defined on the (Eulerian) reference mesh !

$\Rightarrow$ Also ready for time discretization at any order of accuracy (in time).

— We get what we call a Lagrange-flux scheme —

# Summary of Lagrange-flux algorithm

1. MUSCL+limiter slope reconstruction on $\rho$, $\boldsymbol{u}$, $p$
2. Lagrangian approximate Riemann solver: compute $\boldsymbol{u}^\star$ and $p^\star$
3. Compute upwind values $(U_\ell)_A$ at edges $A$
4. Assemble numerical fluxes and update

Using a Runge-Kutta RK2 time integrator, we only need two kernels:

```
PredictionLagrangeFlux()
CorrectionLagrangeFlux()
```

# Performance improvement (scalability analysis)

| **Scheme** | 1 core | 1 core AVX | 16 cores AVX | Scalability |
|---|---|---|---|---|
| Lagrange-flux | 2.6 | 5.8 | 81.0 | 31.1 |
| Staggered Lagrange-remap (reference) | 2.5 | 3.8 | 37.0 | 14.8 |

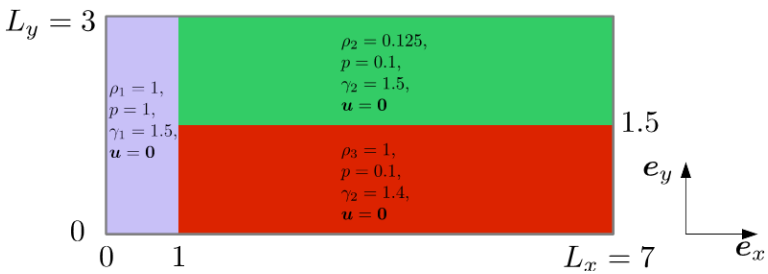
Table: Test on $2 \times 8$ cores Intel Sandy Bridge server E5-2670. Ideal maximum speed-up is 64.

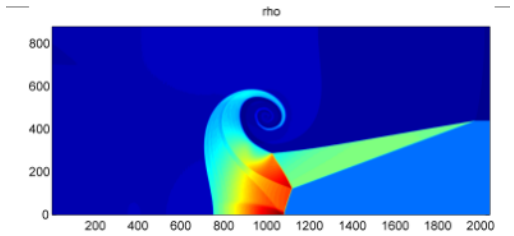
NB Metrics of performance measurement here is millions of updates per second (MUPS).

# Extension to multi-material flows with interface capturing

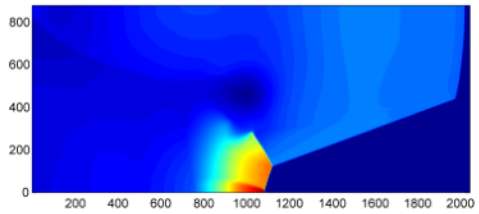
- Interface capturing advection scheme
- Innovative artefact-free shape-preserving approach
- Ready for SIMD

"Triple-point" reference example :

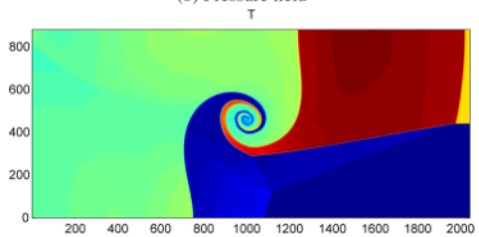


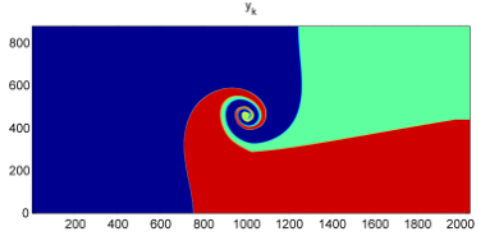

(a) Density field




(b) Pressure field




(c) Temperature field




(d) Colored representation of material indicators