Grid refinement in LBM based on continuous distribution functions

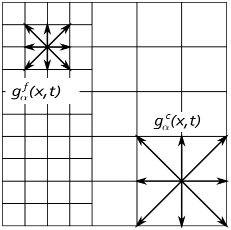
Denis Ricot[1] Simon Marié[1,2] Pierre Sagaut[2]

[1]Renault - Research, Material and Advanced Engineering Department
[2]d'Alembert Institute, Université Paris VI

5th ICMMES - 16-20 june 2008

Grid refinement problem

- For simulating complex engineering flows, mesh refinement technique is necessary
- LB distribution functions g_α are not conserved through a refinement interface [*Filippova & Hanel, 1998*]
- Spatial and time interpolation methods can not be used directly on the distribution functions to evaluate the unknown data



Grid refinement schemes

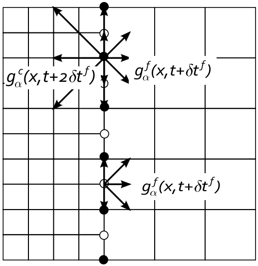
- Vertex centered approach
  - with ([*Yu et al., 2002*][*Dupuis & Chopard, 2003*]) our without rescaling ([*Lin & Lai, 2000*])
  - rescaling and interpolations applied on the post-collision distribution functions ([*Yu et al., 2002*][*Filippova & H anel, 1998*])
  - spatial interpolation performed before time interpolation [*Yu et al., 2002*][*Dupuis & Chopard, 2003*]
- Volumetric approach (cell centered) [*Rohde, 2004*][*Chen et al., 2005 (PowerFLOW)*]
  - Explode ($c \longrightarrow f$) and coalesce ($f \longrightarrow c$) distribution functions : mass conservative scheme
  - No rescaling of distribution functions

Presentation outline

- Continuous distribution functions ?
  - Grid refinement problem
  - DVBE and LBE
  - Numerical verification
- Grid refinement algorithm
  - Rescaling procedure
  - Overlapping nodes
  - Algorithm description
- Validations
  - Grid parameters
  - Pulse propagation in uniform flow
  - Convected vortex

Grid refinement problem

- After one standard LB timestep, some distribution functions are missing
- Distribution functions are *known* to be discontinuous through an interface between meshes with different grid size :
  $g^c_\alpha(\bullet, \circ, t) \neq g^f_\alpha(\bullet, \circ, t)$



Base ideas of Filippova & Hanel (1998)

- Macroscopic properties of the fluid must be conserved (ν, c_s)
  - $\delta x^c / \delta t^c = \delta x^f / \delta t^f$
  - $\tau_g^c = \frac{1}{2}\tau_g^f + \frac{1}{4}$ (the relaxation time is not continuous)
- Macroscopic variables of the flow must be conserved (ρ, $\mathbf{u}$)
  - $g_\alpha^{eq,c} = g_\alpha^{eq,f}$
- The shear stress must be continuous
  - $g_\alpha^{neq,c} = m \frac{\tau_g^c}{\tau_g^f} g_\alpha^{neq,f}$ ($m = \frac{\delta x^c}{\delta x^f} = 2$)

Why the distribution functions of LBM are not continuous ?

- Boltzmann equation

$$\frac{\partial f}{\partial t} + c_i \frac{\partial f}{\partial x_i} = -\frac{f - f^{eq}}{\epsilon\lambda} \qquad (BE)$$

- Quadrature formulae to calculate moments of distribution functions using a discrete velocity set

$$\frac{\partial f_\alpha}{\partial t} + c_{\alpha,i} \frac{\partial f_\alpha}{\partial x_i} = -\frac{1}{\tau}\left(f_\alpha - f_\alpha^{eq}\right) \qquad (DVBE)$$

- By definition the distribution functions of DVBE f_α are continuous in space and time (and also $\nu = \tau\theta_0$, $c_s = \sqrt{\theta_0}$, $\rho = \sum f_\alpha$, $\rho\mathbf{u} = \sum \mathbf{c}_\alpha f_\alpha$)

- Integrate DVBE along the characteristic $\mathbf{c}_\alpha$ for a time interval δt
- The integral of the BGK collision operator is approximated by the trapezium rule :

$$f_\alpha(\mathbf{x}+\mathbf{c}_\alpha\delta t, t+\delta t) - f_\alpha(\mathbf{x},t) = -\frac{\delta t}{2\tau}\{f_\alpha(\mathbf{x}+\mathbf{c}_\alpha\delta t, t+\delta t) - f_\alpha^{eq}(\mathbf{x}+\mathbf{c}_\alpha\delta t, t+\delta t) + f_\alpha(\mathbf{x},t) - f_\alpha^{eq}(\mathbf{x},t)\} + O\left(\delta t^3\right)$$

- Change of variable ([*He et al., 1998*][*Dellar, 2001*])

$$g_\alpha(\mathbf{x},t) = f_\alpha(\mathbf{x},t) + \frac{\delta t}{2\tau}\left(f_\alpha(\mathbf{x},t) - f_\alpha^{eq}(\mathbf{x},t)\right) \quad (1)$$

- Final Lattice Boltzmann Equation

$$g_\alpha(\mathbf{x}+\mathbf{c}_\alpha\delta t, t+\delta t) = \left(1-\frac{\delta t}{\tau_g}\right) g_\alpha(\mathbf{x},t) + \frac{\delta t}{\tau_g} f_\alpha^{eq}(\mathbf{x},t) \quad (2)$$

with $\tau_g = \tau + \delta t/2$.

Numerical verification of the link between f_α and g_α

- LBE and DVBE computations have been performed on the same uniform grid, for the same flow problems and fluid parameters
- LB model : standard D2Q9, BGK collision operator, equilibrium function truncated at second order
- DVBE model : D2Q9 velocity model, BGK collision operator, equilibrium function truncated at second order
  - 6th-order central finite difference scheme for space derivatives
  - 5th-order Runge-Kutta scheme for time integration
- Fluid is air, $\tau/\delta t = 0.93$, same δt and δx for LBE and DVBE
- For two simple flows, comparison of f_α, g_α and $d_\alpha = f_\alpha + \frac{\delta t}{2\tau}\left(f_\alpha - f_\alpha^{eq}\right)$

Reference cases

- Acoustic pressure pulse in an uniform flow

$$\begin{cases} \rho = 1 + a_P \exp\left[-\frac{\ln 2}{b_P}\left(\left(x_1 - x_1^0\right)^2 + \left(x_2 - x_2^0\right)^2\right)\right] \\ u_1 = U_0 \\ u_2 = 0 \end{cases}$$

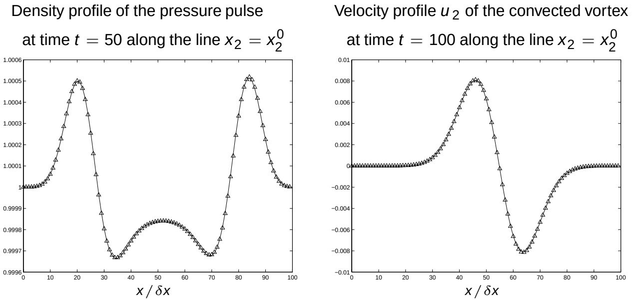
with $a_P = 0.03$ and $b_P = 5$

- Convected vortex

$$\begin{cases} \rho = 1 \\ u_1 = U_0 + a_T\, U_0 \left(x_2 - x_2^0\right) \exp\left[-\frac{\ln 2}{b_T}\left(\left(x_1 - x_1^0\right)^2 + \left(x_2 - x_2^0\right)^2\right)\right] \\ u_2 = -a_T\, U_0 \left(x_1 - x_1^0\right) \exp\left[-\frac{\ln 2}{b_T}\left(\left(x_1 - x_1^0\right)^2 + \left(x_2 - x_2^0\right)^2\right)\right] \end{cases}$$

with $a_T = 0.5$ et $b_T = 25$

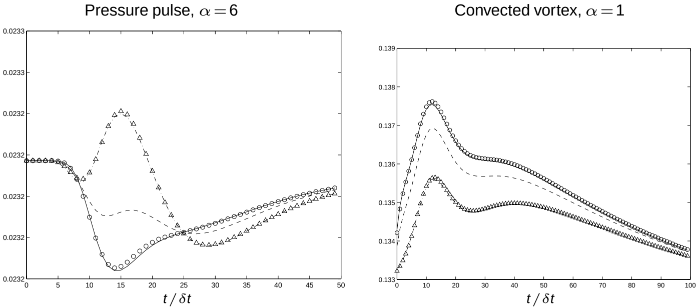
Comparison of the macroscopic variables



—— LBM; $\triangle$ DVBE.

- The simulated results are exactly the same in term of macroscopic variables $\rightarrow g_\alpha^{eq} = f_\alpha^{eq}$

Distribution functions at a given point as a function of time



(———) : g_α
(— — —) : f_α
(○ ○ ○) : $d_\alpha = f_\alpha + \delta t(f_\alpha - f_\alpha^{eq})/2\tau$

(— · — ·) : g_α^{eq}
(△ △ △) : f_α^{eq}

- This numerical test confirms that $g_\alpha = f_\alpha + \frac{\delta t}{2\tau}(f_\alpha - f_\alpha^{eq})$

- Conversion $g_\alpha \leftrightarrow f_\alpha$ for a given $\mathbf{x}$ and t

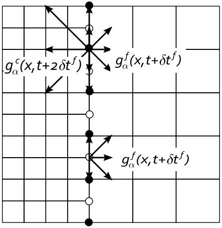
$$g_\alpha = f_\alpha + \frac{\delta t}{2\tau}\left(f_\alpha - f_\alpha^{eq}\right) <=> f_\alpha = \frac{2\tau}{2\tau + \delta t}\left(g_\alpha + \frac{\delta t}{2\tau} f_\alpha^{eq}\right)$$

- By definition f_α, τ, $\rho = \sum f_\alpha$, $\rho\mathbf{u} = \sum \mathbf{c}_\alpha f_\alpha$ ($\rightarrow f_\alpha^{eq}$) are continuous

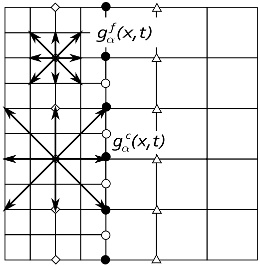
$$\left\{\begin{array}{l} f_\alpha^c = \frac{2\tau}{2\tau+\delta t^c}\left(g_\alpha^c + \frac{\delta t^c}{2\tau} f_\alpha^{eq}\right) \\ f_\alpha^f = \frac{2\tau}{2\tau+\delta t^f}\left(g_\alpha^f + \frac{\delta t^f}{2\tau} f_\alpha^{eq}\right) \\ f_\alpha^f = f_\alpha^c \\ \tau = \tau_g^f - \frac{\delta t^f}{2} = \tau_g^c - \frac{\delta t^c}{2} \\ m = 2 \end{array}\right. ==> \begin{array}{l} g_\alpha^f = \frac{1}{2\tilde{\tau}_g^f+1}\left(\left(2\tilde{\tau}_g^f\right) g_\alpha^c + f_\alpha^{eq}\right) \\ \\ g_\alpha^c = \frac{1}{2\tilde{\tau}_g^f}\left(\left(2\tilde{\tau}_g^f + 1\right) g_\alpha^f - f_\alpha^{eq}\right) \end{array}$$

- Remark: these expressions immediately imply that $g_\alpha^{neq,c} = 2\frac{\tilde{\tau}_g^c}{\tilde{\tau}_g^f} g_\alpha^{neq,f}$

- First approach (implemented with success in our 3D LB code (L-BEAM))
  - Conversion $g_\alpha^c \leftrightarrow f_\alpha$ for needed interface points
  - Spatial and temporal interpolations on f_α
  - Conversion $f_\alpha \leftrightarrow g_\alpha^f$
- Second approach (preferred in this work)
  - Conversion $g_\alpha^c \leftrightarrow g_\alpha^f$ for needed interface points
  - Spatial and temporal interpolations on g_α^f
- In both cases
  - Conversion step needs the equilibrium functions to be known

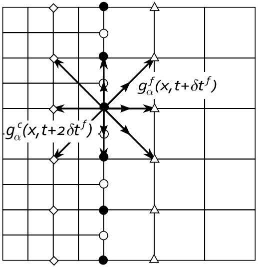


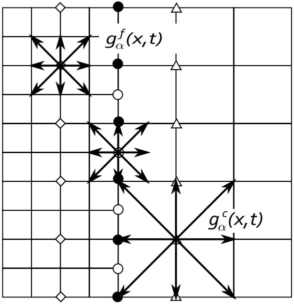
- Fine grid points $\diamond$ must also be calculated as coarse grid points



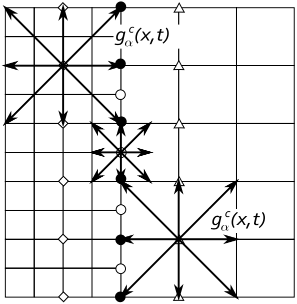
- Conversion $g^f_{\alpha_{out}}(\diamond, t) \longrightarrow g^c_{\alpha_{out}}(\diamond, t)$

- Conversion $g^c_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right) \rightarrow g^f_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$ can be done because macroscopic variables can be now calculated at points $\bullet$

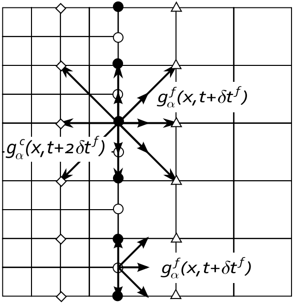




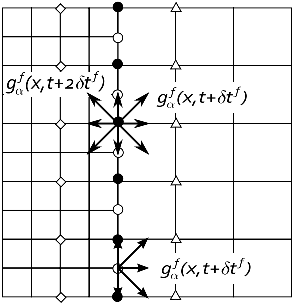
- **Timestep** t : all functions g_{α}^{c} and g_{α}^{f} are known everywhere



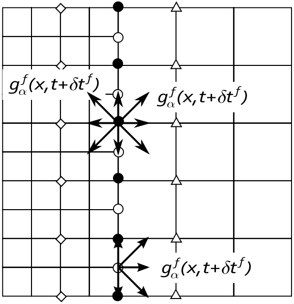
- **Timestep** t : all functions g_α^c and g_α^f are known everywhere
- Stage 1 : convert $g^f_{\alpha_{out}}(\diamond, t) \longrightarrow g^c_{\alpha_{out}}(\diamond, t)$



- **Timestep** t : all functions g^c_α and g^f_α are known everywhere
- Stage 1 : convert $g^f_{\alpha_{out}}(\diamond, t) \longrightarrow g^c_{\alpha_{out}}(\diamond, t)$
- Stage 2 : collide and propagate all points in the fine and coarse regions (including points $\diamond$)

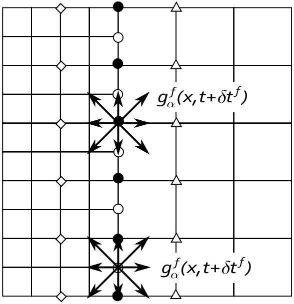


- **Timestep** t : all functions g_α^c and g_α^f are known everywhere
- Stage 1 : convert $g_{\alpha_{out}}^f(\diamond, t) \longrightarrow g_{\alpha_{out}}^c(\diamond, t)$
- Stage 2 : collide and propagate all points in the fine and coarse regions (including points $\diamond$)
- Stage 3 : convert $g_{\alpha_{in}}^c\left(\bullet, t+2\delta t^f\right) \longrightarrow g_{\alpha_{in}}^f\left(\bullet, t+2\delta t^f\right)$

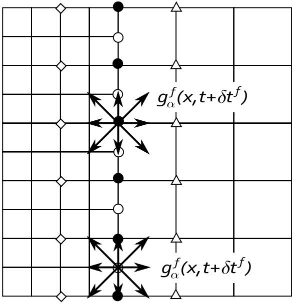


- **Timestep** t : all functions g^c_α and g^f_α are known everywhere
- Stage 1 : convert $g^f_{\alpha_{out}}(\diamond, t) \longrightarrow g^c_{\alpha_{out}}(\diamond, t)$
- Stage 2 : collide and propagate all points in the fine and coarse regions (including points $\diamond$)
- Stage 3 : convert $g^c_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right) \longrightarrow g^f_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$
- Stage 4 : time interpolation of $g^f_{\alpha_{in}}\left(\bullet, t+\delta t^f\right)$

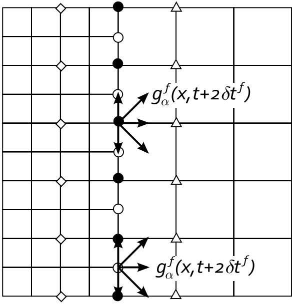
- Injection scheme : $g^f_{\alpha_{in}}\left(\bullet, t+\delta t^f\right) = g^f_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$
- Linear interpolation using $g^f_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$ and $g^f_{\alpha_{in}}(\bullet, t)$ (that must be stored)



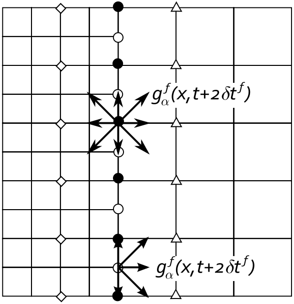
- **Timestep** t : all functions g_α^c and g_α^f are known everywhere
- Stage 1 : convert $g^f_{\alpha_{out}}(\diamond, t) \longrightarrow g^c_{\alpha_{out}}(\diamond, t)$
- Stage 2 : collide and propagate all points in the fine and coarse regions (including points $\diamond$)
- Stage 3 : convert $g^c_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right) \rightarrow g^f_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$
- Stage 4 : time interpolation of $g^f_{\alpha_{in}}\left(\bullet, t+\delta t^f\right)$
  - Injection scheme : $g^f_{\alpha_{in}}\left(\bullet, t+\delta t^f\right) = g^f_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$
  - Linear interpolation using $g^f_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$ and $g^f_{\alpha_{in}}(\bullet, t)$ (that must be stored)
- Stage 5 : spatial interpolation of $g^f_{\alpha_{in}}\left(\circ, t+\delta t^f\right)$ using $g^f_{\alpha_{in}}\left(\bullet - 1, t+\delta t^f\right)$ and $g^f_{\alpha_{in}}\left(\bullet + 1, t+\delta t^f\right)$



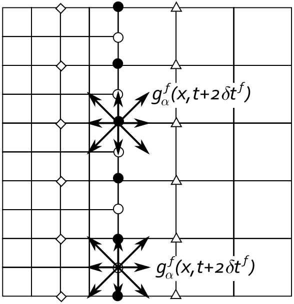
- **Timestep** t : all functions g_α^c and g_α^f are known everywhere
- Stage 1 : convert $g^f_{\alpha_{out}}(\diamond,t) \longrightarrow g^c_{\alpha_{out}}(\diamond,t)$
- Stage 2 : collide and propagate all points in the fine and coarse regions (including points $\diamond$)
- Stage 3 : convert $g^c_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right) \rightarrow g^f_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$
- Stage 4 : time interpolation of $g^f_{\alpha_{in}}\left(\bullet, t+\delta t^f\right)$
  - Injection scheme : $g^f_{\alpha_{in}}\left(\bullet, t+\delta t^f\right) = g^f_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$
  - Linear interpolation using $g^f_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$ and $g^f_{\alpha_{in}}(\bullet, t)$ (that must be stored)
- Stage 5 : spatial interpolation of $g^f_{\alpha_{in}}\left(\circ, t+\delta t^f\right)$ using $g^f_{\alpha_{in}}\left(\bullet - 1, t+\delta t^f\right)$ and $g^f_{\alpha_{in}}\left(\bullet + 1, t+\delta t^f\right)$
- **Timestep** $t+\delta t^f$: all functions g_α^f are known in the fine grid region



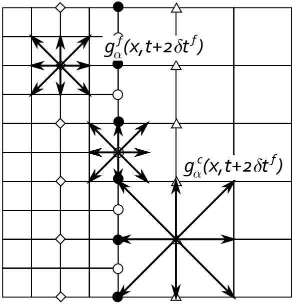
- **Timestep** t : all functions g^c_α and g^f_α are known everywhere
- Stage 1 : convert $g^f_{\alpha_{out}}(\diamond, t) \longrightarrow g^c_{\alpha_{out}}(\diamond, t)$
- Stage 2 : collide and propagate all points in the fine and coarse regions (including points $\diamond$)
- Stage 3 : convert $g^c_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right) \longrightarrow g^f_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$
- Stage 4 : time interpolation of $g^f_{\alpha_{in}}\left(\bullet, t+\delta t^f\right)$
    - Injection scheme : $g^f_{\alpha_{in}}\left(\bullet, t+\delta t^f\right) = g^f_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$
    - Linear interpolation using $g^f_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$ and $g^f_{\alpha_{in}}(\bullet, t)$ (that must be stored)
- Stage 5 : spatial interpolation of $g^f_{\alpha_{in}}\left(\circ, t+\delta t^f\right)$ using $g^f_{\alpha_{in}}\left(\bullet - 1, t+\delta t^f\right)$ and $g^f_{\alpha_{in}}\left(\bullet + 1, t+\delta t^f\right)$
- **Timestep** $t+\delta t^f$: all functions g^f_α are known in the fine grid region
- Stage 6 : collide and propagate all points in the fine region



- **Timestep** t : all functions g^c_α and g^f_α are known everywhere
- Stage 1 : convert $g^f_{\alpha_{out}}(\diamond, t) \rightarrow g^c_{\alpha_{out}}(\diamond, t)$
- Stage 2 : collide and propagate all points in the fine and coarse regions (including points $\diamond$)
- Stage 3 : convert $g^c_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right) \rightarrow g^f_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$
- Stage 4 : time interpolation of $g^f_{\alpha_{in}}\left(\bullet, t+\delta t^f\right)$
  - Injection scheme : $g^f_{\alpha_{in}}\left(\bullet, t+\delta t^f\right) = g^f_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$
  - Linear interpolation using $g^f_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$ and $g^f_{\alpha_{in}}(\bullet, t)$ (that must be stored)
- Stage 5 : spatial interpolation of $g^f_{\alpha_{in}}\left(\circ, t+\delta t^f\right)$ using $g^f_{\alpha_{in}}\left(\bullet - 1, t+\delta t^f\right)$ and $g^f_{\alpha_{in}}\left(\bullet + 1, t+\delta t^f\right)$
- **Timestep** $t+\delta t^f$: all functions g^f_α are known in the fine grid region
- Stage 6 : collide and propagate all points in the fine region
- Stage 7 : recover $g^f_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$ from $g^c_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$



- **Timestep** t : all functions g^c_α and g^f_α are known everywhere
- Stage 1 : convert $g^f_{\alpha_{out}}(\diamond, t) \rightarrow g^c_{\alpha_{out}}(\diamond, t)$
- Stage 2 : collide and propagate all points in the fine and coarse regions (including points $\diamond$)
- Stage 3 : convert $g^c_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right) \rightarrow g^f_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$
- Stage 4 : time interpolation of $g^f_{\alpha_{in}}\left(\bullet, t+\delta t^f\right)$
    - Injection scheme : $g^f_{\alpha_{in}}\left(\bullet, t+\delta t^f\right) = g^f_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$
    - Linear interpolation using $g^f_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$ and $g^f_{\alpha_{in}}(\bullet, t)$ (that must be stored)
- Stage 5 : spatial interpolation of $g^f_{\alpha_{in}}\left(\circ, t+\delta t^f\right)$ using $g^f_{\alpha_{in}}\left(\bullet - 1, t+\delta t^f\right)$ and $g^f_{\alpha_{in}}\left(\bullet + 1, t+\delta t^f\right)$
- **Timestep** $t+\delta t^f$: all functions g^f_α are known in the fine grid region
- Stage 6 : collide and propagate all points in the fine region
- Stage 7 : recover $g^f_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$ from $g^c_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$
- Stage 8 : spatial interpolation of $g^f_{\alpha_{in}}\left(\circ, t+2\delta t^f\right)$

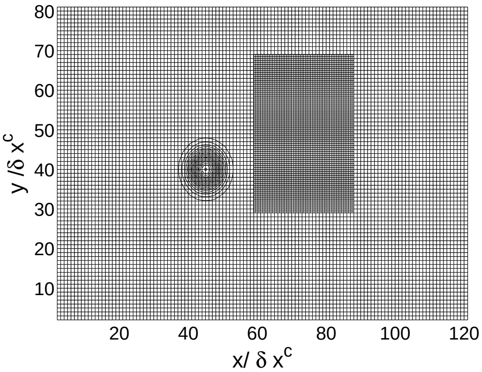


- **Timestep** t : all functions g^c_α and g^f_α are known everywhere
- Stage 1 : convert $g^f_{\alpha_{out}}(\diamond, t) \longrightarrow g^c_{\alpha_{out}}(\diamond, t)$
- Stage 2 : collide and propagate all points in the fine and coarse regions (including points $\diamond$)
- Stage 3 : convert $g^c_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right) \longrightarrow g^f_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$
- Stage 4 : time interpolation of $g^f_{\alpha_{in}}\left(\bullet, t+\delta t^f\right)$
  - Injection scheme : $g^f_{\alpha_{in}}\left(\bullet, t+\delta t^f\right) = g^f_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$
  - Linear interpolation using $g^f_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$ and $g^f_{\alpha_{in}}(\bullet, t)$ (that must be stored)
- Stage 5 : spatial interpolation of $g^f_{\alpha_{in}}\left(\circ, t+\delta t^f\right)$ using $g^f_{\alpha_{in}}\left(\bullet-1, t+\delta t^f\right)$ and $g^f_{\alpha_{in}}\left(\bullet+1, t+\delta t^f\right)$
- **Timestep** $t+\delta t^f$: all functions g^f_α are known in the fine grid region
- Stage 6 : collide and propagate all points in the fine region
- Stage 7 : recover $g^f_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$ from $g^c_{\alpha_{in}}\left(\bullet, t+2\delta t^f\right)$
- Stage 8 : spatial interpolation of $g^f_{\alpha_{in}}\left(\circ, t+2\delta t^f\right)$
- **Timestep** $t+2\delta t^f$: all functions g^c_α and g^f_α are known everywhere

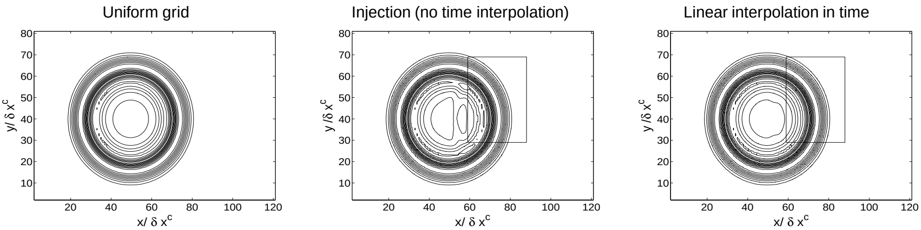
- Coarse grid : 120*x*80, fine grid : 60×80
- Acoustic pulse or vortex is initialized outside or inside the fine grid
- $M = 0.2$, $\nu = 1.5 \times 10^{-5}\, m^2/s$, $c_s = 340\, m/s$, $\delta x^f = 10^{-2}\, m$
- All computations are also done with a uniform coarse grid (reference)

U_0

$\Longrightarrow$

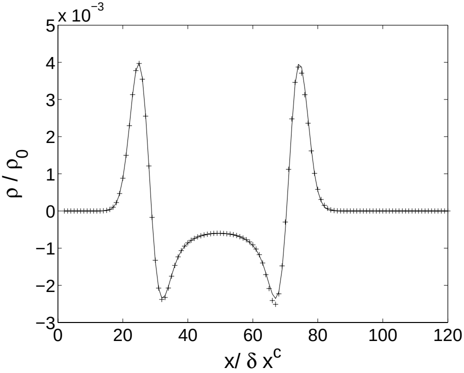


Iso-contours of the density fluctuation



- Small pressure reflexion occurs at the grid interface. The error is reduced when linear time interpolation is used.
- There is an analytical solution for the propagation of the pulse in uniform flow $\longrightarrow$ precise error quantification is possible

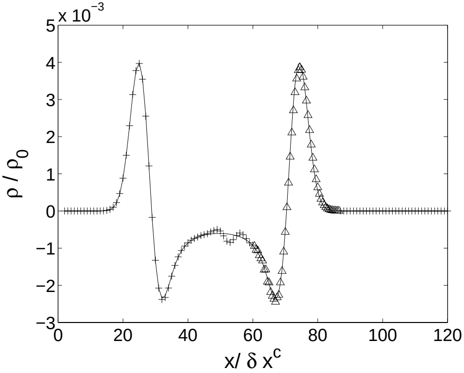
Simulation with the uniform grid



(———) reference analytical solution

(+ + + +) LB simulation

Two-grid simulation with injection (no time interpolation)

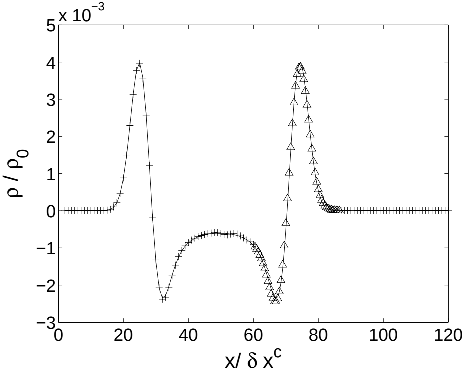


(——) reference analytical solution

(+ + + +) results on coarse points

(△ △ △) results on fine points

Two-grid simulation with linear interpolation in time



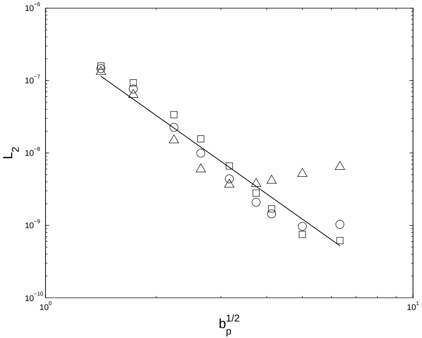
(———) reference analytical solution

(+ + + +) results on coarse points

(△ △ △) results on fine points

Convergence rate

- Parameter b_p gives the initial width of the pressure pulse, i.e. the spatial resolution



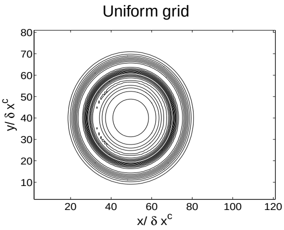
(———) slope -3.5

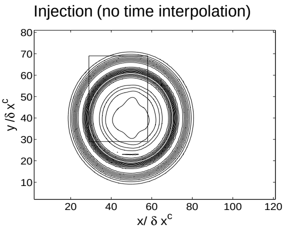
(□ □ □ □) Uniform grid

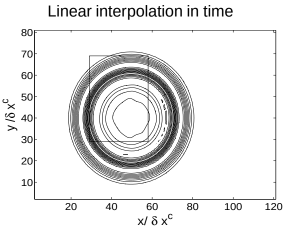
(△ △ △) Two-grid simulation without time interpolation (injection)

(○ ○ ○ ○) Two-grid simulation with linear interpolation in time

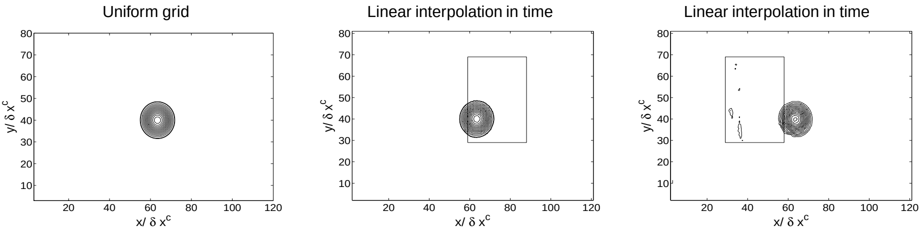
Initial pulse in the fine grid region







Convected vortex (linear interpolation in time)



- Fine to coarse grid convection : spurious vorticity appears inside the fine region due to non-physical reflexion at the grid interface

Conclusion

- New theoretical insight in the link between coarse and fine distribution functions
  - final rescaling expressions are fully equivalent to the previous approaches
  - this new theoretical approach can be useful for other LB models (other collision operator such as MRT models)
- The proposed grid refinement algorithm minimize the unknown functions that must be approximated with interpolations
  - use of overlapping coarse nodes
  - interpolations are done on the distribution functions
  - time interpolation is done before spatial interpolation
- Accuracy of the grid refinement scheme can be improved using higher order interpolation schemes