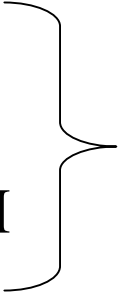


# Particle methods in CFD

G.-H. Cottet  
LMC-IMAG, Université Joseph Fourier

## Outline:

- Basics definitions and methods for inviscid flows
  - Diffusion and implicit subgrid-scale models in PM
  - Regridding and multi-level PM
  - Boundary conditions and interface capturing PM
- 
- Towards  
multi-physics  
multi-level PM

Advection equation in conservation form:

$$Lu = \frac{\partial u}{\partial t} + \operatorname{div}(\mathbf{a}u) + a_0 u$$

Conservation of mass along trajectories for  $Lu=f$ :

$$\frac{d}{dt} \left( \int_{\Omega(t)} u \, dx \right) + \int_{\Omega(t)} a_0 u \, dx = \int_{\Omega(t)} f \, dx$$

where  $\dot{\Omega}(t) = a(\Omega(t), t)$

Idea of particle methods: **concentrate mass on points (particles)**

Integrals become simple values, and above mass balance equation reduces to a **set of differential equations**

Fundamental examples:

Particle solution of  $Lu \equiv 0$  :

$$u(\mathbf{x}, t) = \alpha(t) \delta(\mathbf{x} - \mathbf{x}(t))$$

where

$$\frac{d\alpha}{dt} + a_0(\mathbf{x}(t), t)\alpha = 0$$

$$\frac{d\mathbf{x}_p}{dt} = \mathbf{a}(\mathbf{x}_p, \mathbf{t})$$

Handles also Dirac functions in right hand side:

$$Lu = f$$

where  $f = \beta_0 \delta(\mathbf{x} - \mathbf{x}_0) \otimes \delta(t - t_0)$

then, a particle solution reads

$$u(\mathbf{x}, t) = \beta(t) \delta(\mathbf{x} - \mathbf{X}(t; \mathbf{x}_0, t_0))$$

$$\beta(t) = 0 \quad \text{for} \quad t < t_0 \quad \text{and, for } t \geq t_0$$

$$\frac{d\beta}{dt} + a_0(\mathbf{X}(t; \mathbf{x}_0, t_0), t) \beta(t) = 0 ; \quad \beta(t_0) = \beta_0$$

Here  $\mathbf{X}(t; \mathbf{x}, s)$  denotes the trajectory at time  $t$  originating from  $\mathbf{x}$  at time  $s$

Remarks:

Applies to linear combination of Dirac masses

Applies to vector equations

Mass of particles combine local values of  $u$  and local volumes

What is the sense of “particle solution”?

weak solution of  $LU=f$ , that is satisfies:

$$\int_0^T \langle u(\bullet, t), L^* \varphi(\bullet, t) \rangle dt = \langle u_0, \varphi(\bullet, 0) \rangle + \int_0^T \langle f(\bullet, t), \varphi(\bullet, t) \rangle dt$$

where

$$L^* \phi = -\frac{\partial \phi}{\partial t} - (a \cdot \nabla) \phi + a_0^t \phi$$

Consequence:

- no projection of the original equation involved
- convergence based on approximation of data in appropriate space

Essential feature: transport of Dirac masses directly translates conservative formulation

Three classical + one less classical examples:

- Vlasov-Maxwell equations
- Gas dynamics
- Incompressible Navier-Stokes equations
- Linear elasticity

## Vlasov maxwell equations

Distribution function for ions (or electrons) subject to electric and magnetic fields

$$f = f(\mathbf{x}, \mathbf{v}, \mathbf{t}) \in [0, 1] \quad \mathbf{E} = \mathbf{E}(\mathbf{x}, \mathbf{t}) \quad \mathbf{B} = \mathbf{B}(\mathbf{x}, \mathbf{t})$$

Conservation of charge:

$$\frac{\partial f}{\partial t} + (\mathbf{v} \cdot \nabla_{\mathbf{x}})f + ((\mathbf{E} + \mathbf{v} \times \mathbf{B}) \cdot \nabla_{\mathbf{v}})f = 0$$

$$\mathbf{U} = \begin{bmatrix} \mathbf{v} \\ \mathbf{E} + \mathbf{v} \times \mathbf{B} \end{bmatrix} \text{ satisfies } \operatorname{div}_{\mathbf{x}, \mathbf{v}} \mathbf{U} = 0$$

hence, conservative advection equation for  $f$  in velocity field  $\mathbf{U}$



## Gas dynamics for compressible isentropic flows

$$W = (\rho, \rho u, \rho v, \rho E) ; U = (u, v)$$

Conservation of density, momentum and energy:

$$\frac{\partial W}{\partial t} + \operatorname{div}(U : W) = \Pi + \text{diffusion}$$

$$\Pi = (0, -\nabla p, \operatorname{div}(Up)) ; p = p(\rho e)$$

## Vorticity conservation for incompressible flows

Incompressible Navier-Stokes equations in velocity-pressure formulation

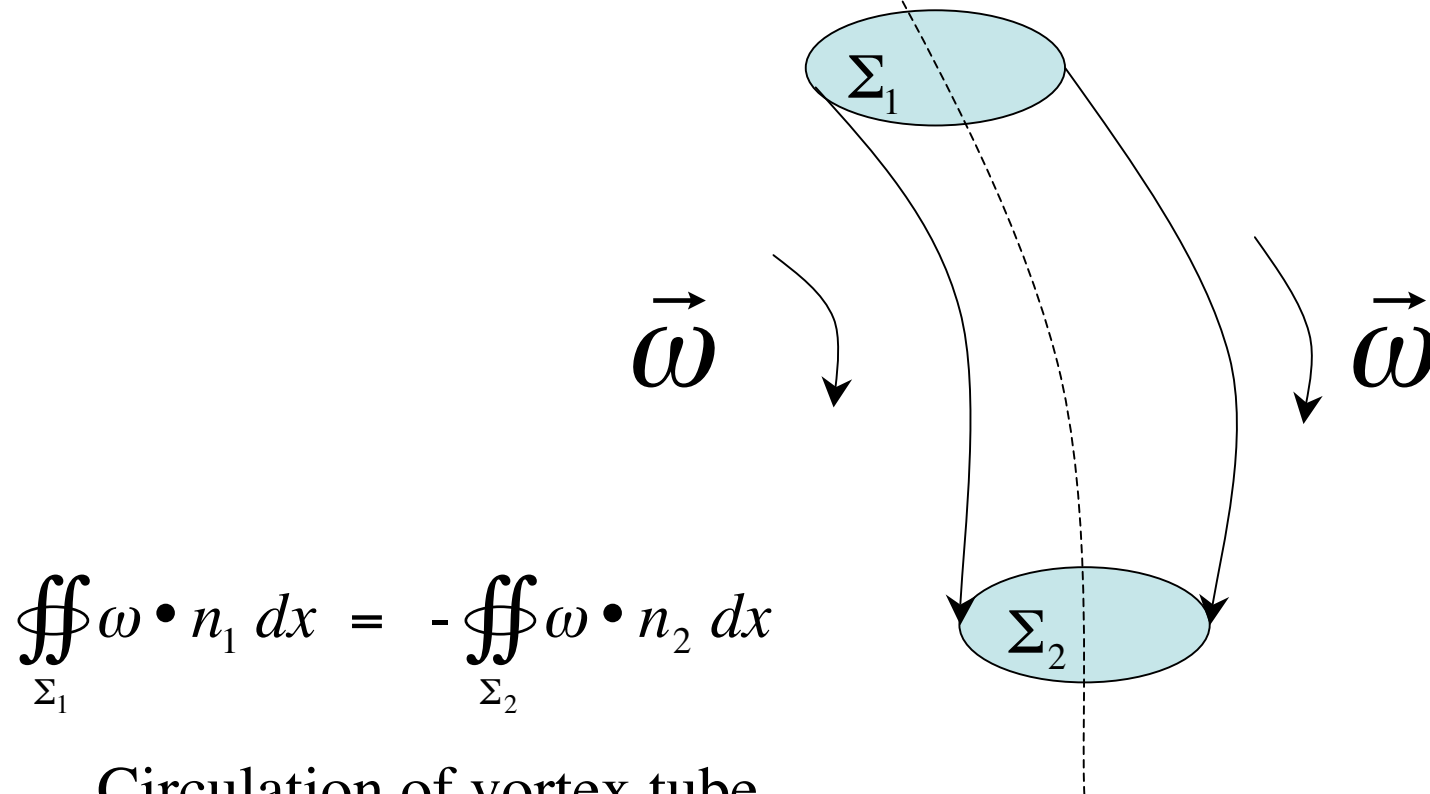
$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla) \mathbf{u} + \nabla p = \nu \Delta \mathbf{u} \quad \text{div} \mathbf{u} = 0$$

$$\boldsymbol{\omega} = \nabla \times \mathbf{u} \quad \frac{\partial \boldsymbol{\omega}}{\partial t} + \text{div}(\mathbf{u} \boldsymbol{\omega}) - [\nabla \mathbf{u}] \boldsymbol{\omega} = \nu \Delta \boldsymbol{\omega}$$

$$\boldsymbol{\omega} = \sum_p \alpha_{\mathbf{p}} \delta(\mathbf{x} - \mathbf{x}_{\mathbf{p}}) \quad \frac{d\mathbf{x}_p}{dt} = \mathbf{u}(\mathbf{x}_p) \quad \frac{d\alpha_{\mathbf{p}}}{dt} = [\nabla \mathbf{u}(\mathbf{x}_p)] \alpha_{\mathbf{p}} + \text{diffusion}$$

## Other types of particles: vorticity contours and filaments

Vorticity filament: a curve that concentrates a vortex tube



Vortex filament: particle with support an oriented curve (centerline of vortex tube) and strength its circulation.

In mathematical words, filament of circulation  $\Gamma$  supported by curve  $F$  parameterized by  $\gamma(\xi)$  :

$$\langle \mu, \varphi \rangle = \alpha \oint_F \varphi(\gamma(\xi)) \cdot \frac{\partial \gamma}{\partial \xi} d\xi$$

Same notions apply for contours of 2D vortex patches  
**(method of contour dynamics):**  $\nabla \times \omega$  satisfies a transport-stretching equation identical to the 3D vorticity equation

In practice, filaments are tracked by a finite number of markers, and method amounts to point-particle method with stretching of particles implicitly accounted for by curve stretching.

So far, considered that velocity field and strain necessary to update particle locations and strain was given.

In fact must be computed in self-consistent way from vorticity:

$$\nabla \times \mathbf{u} = \boldsymbol{\omega} \quad ; \quad \nabla \cdot \mathbf{u} = 0$$

Two classical approaches:

- a completely grid-free approach, based on integral representation formulas
- an approach using grid-based Poisson solvers

First approach uses Biot-Savart law:

$$\mathbf{u} = \mathbf{K} * \boldsymbol{\omega} = \int \mathbf{K}(\mathbf{x} - \mathbf{y}) \times \boldsymbol{\omega}(\mathbf{y}) \, d\mathbf{y} \quad ; \quad \mathbf{K} = \nabla(1/4\pi|\mathbf{x}|)$$

When  $\omega$  replaced by a set of particles, velocity on each particle is expressed as

$$u(x_p) = \sum_q \alpha_q K(x_p - x_q)$$

Two remarks:

- Kernel is singular: need to mollify to avoid large values when particles Approach
- N-body problem: complexity in  $O(N^2)$  if  $N$ =number of particles

Mollification is performed by convolution with cut-off with radial symmetry and core size  $\varepsilon$  -> explicit algebraic formulas

High order formulas obtained by imposing that cut-off shares as many moments as possible with Dirac function

Example of formula at order 4:

$$\mathbf{K}_\varepsilon(x) = -\frac{\mathbf{x}}{4\pi|\mathbf{x}|^3} (1 + e^{-|\mathbf{x}|^3/\varepsilon^3} - 4e^{-2|\mathbf{x}|^3/\varepsilon^3})$$

Complete numerical method can be rephrased as a set of coupled differential equations:

$$\frac{dx_p}{dt} = \sum_q \alpha_q K_\varepsilon(x_p - x_q)$$

Some natural conservation properties result from this formulation

$$\sum_p \alpha_p x_p \quad (= \int x \omega(x) dx, \text{ linear impulse})$$

$$\sum_p \alpha_p |x_p|^2 \quad (= \int |x|^2 \omega(x) dx, \text{ angular impulse})$$

$$\sum_{p,q} \alpha_p \alpha_q G_\varepsilon(x_p - x_q) \quad (= \int \omega(x) \omega(y) G_\varepsilon(x - y) dx dy, \text{ kinetic energy})$$



$O(N^2)$  complexity can be reduced to  $O(N \log N)$  by using Fast Summation Algorithms: idea is to replace kernel by algebraic expansions:

**THEOREM 2.1.** (Multipole expansion). *Suppose that  $m$  charges of strengths  $\{q_i, i = 1, \dots, m\}$  are located at points  $\{z_i, i = 1, \dots, m\}$ , with  $|z_i| < r$ . Then for any  $z \in \mathbb{C}$  with  $|z| > r$ , the potential  $\phi(z)$  is given by*

$$\phi(z) = Q \log(z) + \sum_{k=1}^{\infty} \frac{a_k}{z^k}, \quad (2.2)$$

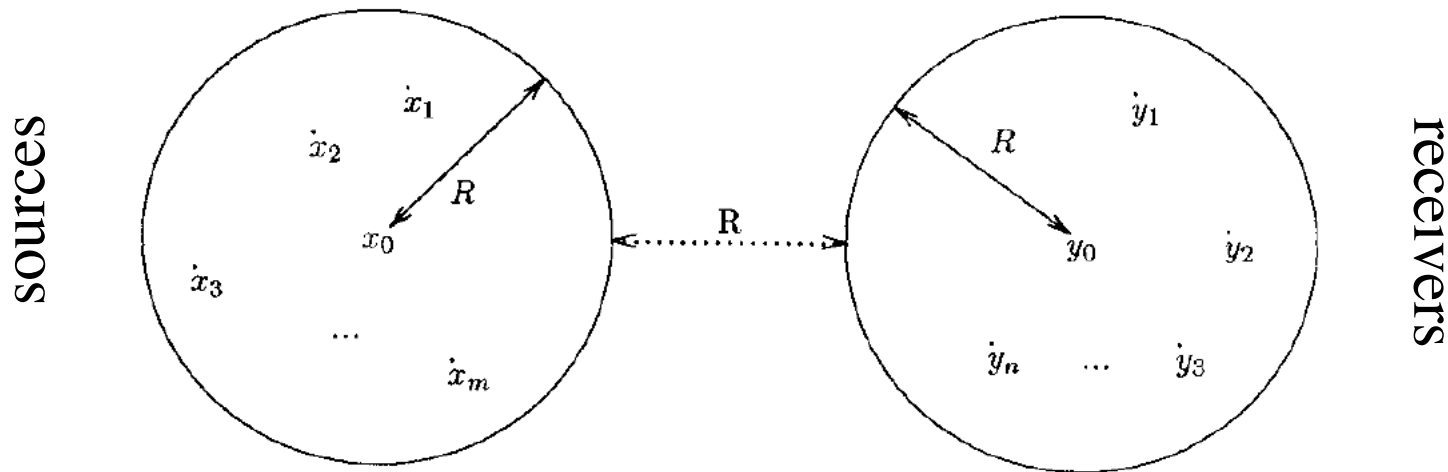
where

$$Q = \sum_{i=1}^m q_i, \quad a_k = \sum_{i=1}^m \frac{-q_i z_i^k}{k}. \quad (2.3)$$

(Greengard-Rocklin, for logarithmic kernel)

with precise estimates

Gain over direct summation can be explained on simple example



Field of  $M$  particles on  $N$  particles:

- direct summation:  $O(MN)$  operations
- Fast summation with  $p$  terms:  $O(Mp + Np)$ 
  - $O(Mp)$  calculations to compute expansion coefficients from sources
  - $O(Np)$  calculations to evaluate expansions on receivers

## Typical tree-code:

Divide recursively into boxes containing about the same number of particles

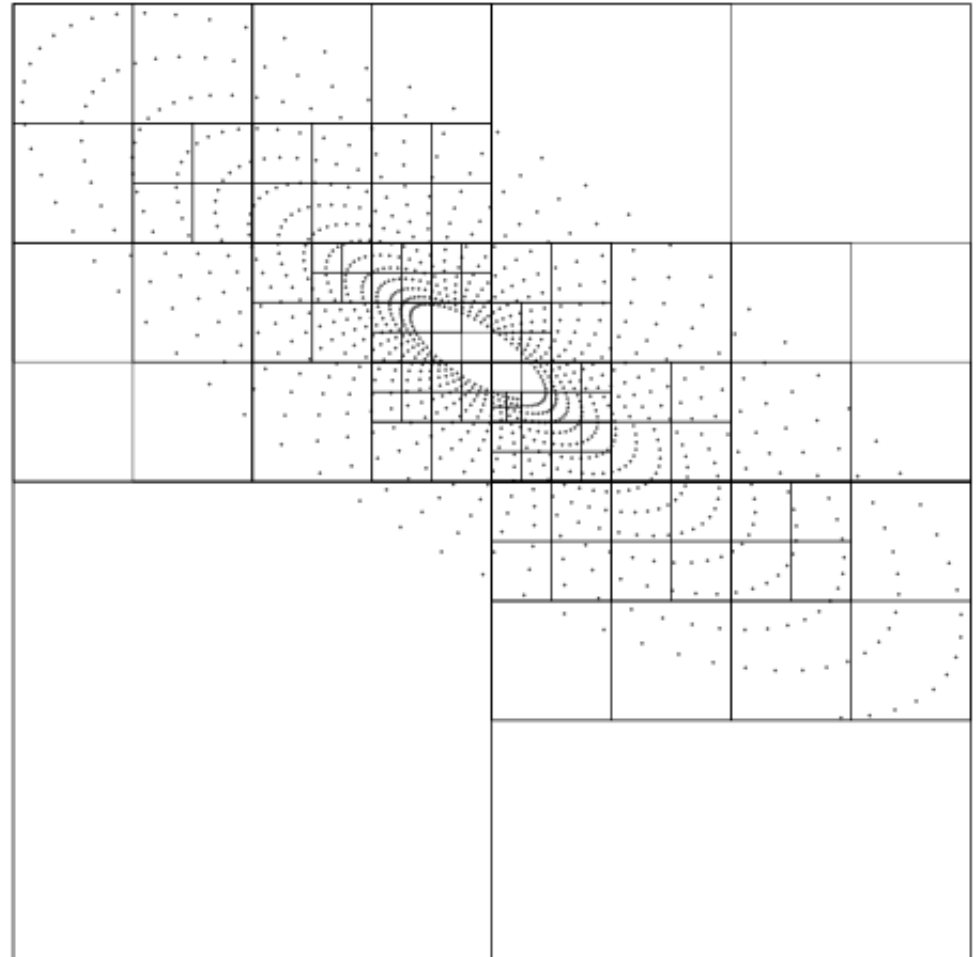
### Upward pass:

form multipole expansions, from finer to coarser level (using shifts of previously computed expansions)

### Downward pass:

accumulate contributions of well-separated boxes, from coarser to finer level

At finest level, complete with direct summation of nearby particles



Still many improvements to come in 3D (parallelism ..)

Other approach for fluid calculations: use an underlying Eulerian grid and grid-based Poisson solvers (Particle-In-Cell/Vortex-In-Cell methods):

- Project particle strength on grid points
- Use a Poisson solver on that grid, and differentiate on the grid to get grid field values
- Interpolate back fields on particles

Drawbacks:

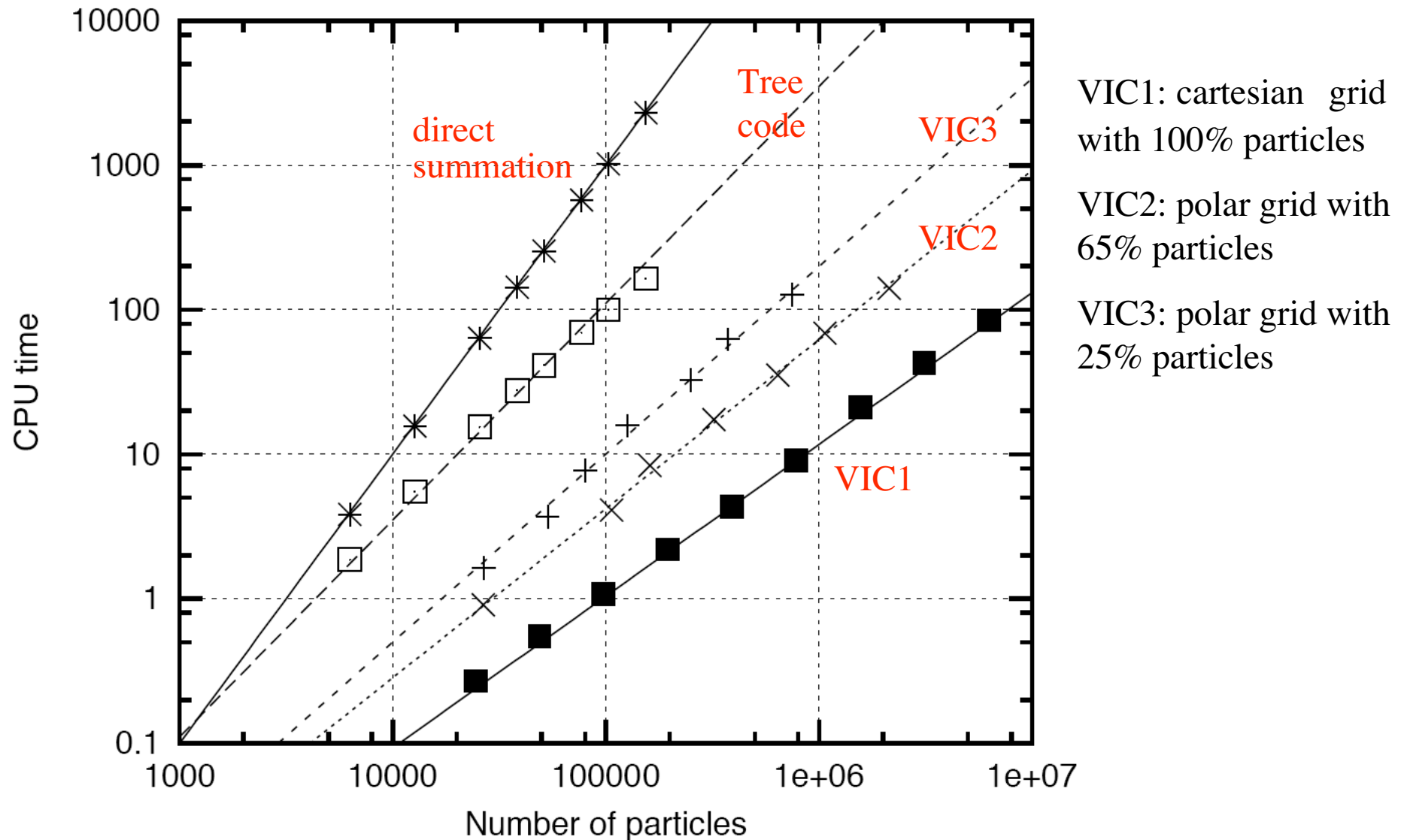
- against Lagrangian features of particles (and possible loss of information in grid-particle interpolations)
- require far-field artificial boundary conditions

Advantage:

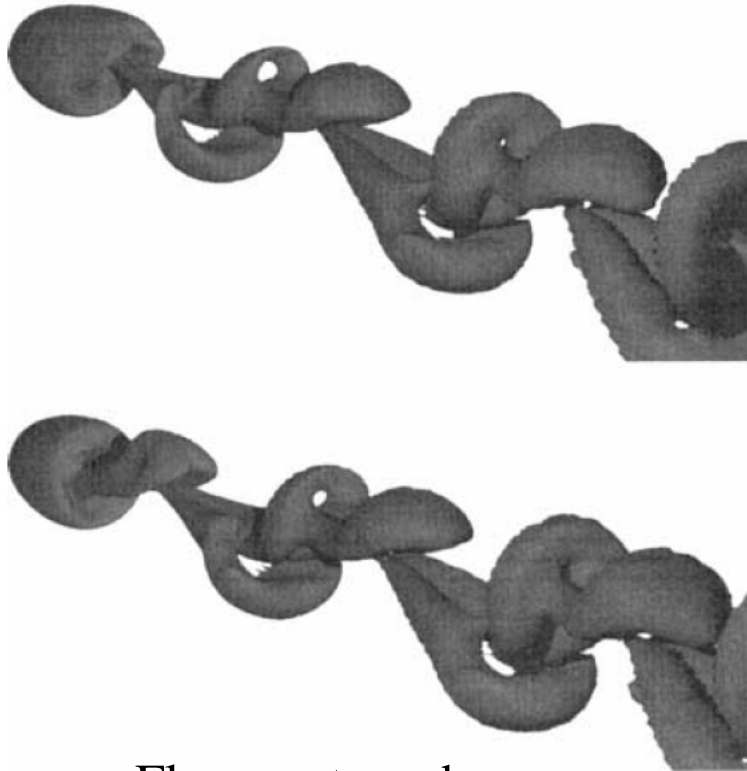
Cheap (for relatively simple geometries)

# Comparison of CPU times for velocity evaluations in 3D

(Krasny tree-code vs VIC with Fishpack and 64 points interpolation formulas)

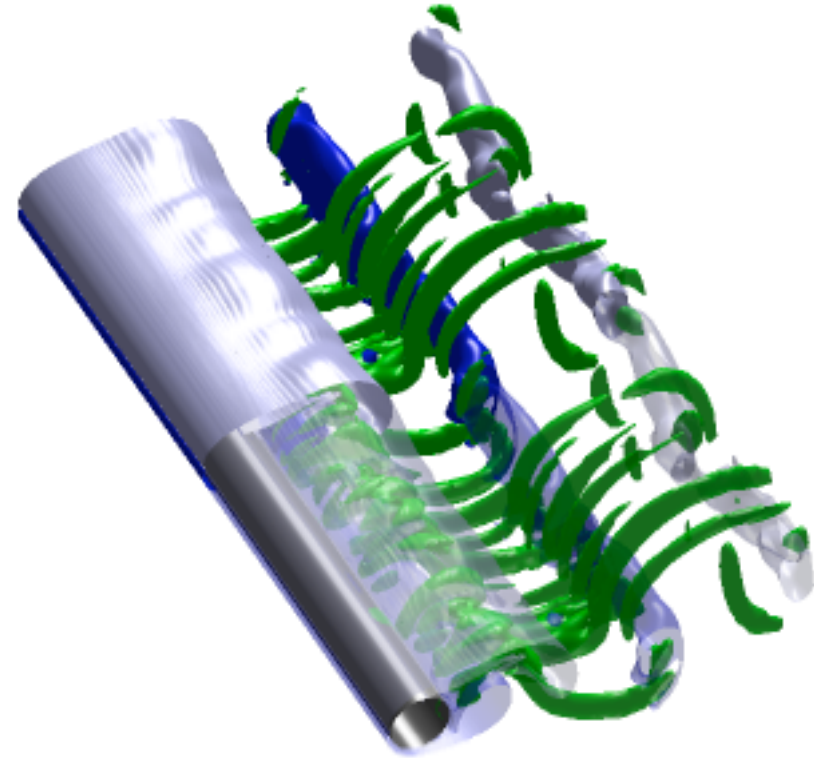


Conclusion: choice of solver depends on how localized vorticity is in the computational box needed:



Flow past a sphere:  
grid-free calculation (Ploumehans et al.)

About 600,000 particles,  
Total cpu is 200 hours on 32  
HP processors



Flow past a cylinder:  
VIC calculation (C.-Poncet)

Cylindrical grid:  $256 \times 128 \times 128$   
in a domain  $4\pi \times 2\pi \times 2\pi$   
filled with 25% particles  
CPU time: 3mn/RK4 iteration on alpha  
single processor, 3hours/shedding cycle

Possible to combine both approaches (grid-free and particle-in-cell) through domain decomposition approaches: far-field by grid-free, “boundary layer” by PIC.

Other issue related to velocity evaluations: time-stepping to push particles and update strengths

In general RK2 or RK4

**Linear stability only requires particle not to cross each other**

(no conventional CFL type condition):

$$\Delta t \leq \frac{C}{\|\nabla \mathbf{u}\|_{\infty}}$$

CPU savings depend on particular flow

## Lecture 2

1. Handling diffusion in particle methods
2. Understanding how particle methods work at sub-grid level
3. Regridding of particles
4. Towards adaptive multi-level particle methods



Focus on heat equation in free space

$$\frac{\partial \omega}{\partial t} = \nu \Delta \omega$$

Explicit exact solution with integral formula

$$\omega(\mathbf{x}, t) = \int \mathcal{G}(\mathbf{x} - \mathbf{y}, \nu t) \omega_0(\mathbf{y}) d\mathbf{y}$$

Where Gaussian kernel given in 2D by

$$\mathcal{G}(\mathbf{x} - \mathbf{y}, s) = \frac{1}{4\pi s} e^{-|\mathbf{x} - \mathbf{y}|^2 / 4s}$$

When field given on particles, obtain superposition of gaussian blobs

First method: random walk method (Chorin 70s)

Sample these blobs by particles with appropriate density

1 particle per blob -> random walk of each particle after advection step

Convergence relies on large number laws

Formula for n steps of random walks for N particles

$$\omega(\mathbf{x}, t_n) \approx \frac{1}{N} \sum_p \omega_0(\mathbf{y}_p) \delta(\mathbf{x} - \mathbf{y}_p - R(u_p^1, \nu \delta t) e^{i\theta_p^1} + R(u_p^2, \nu \delta t) e^{i\theta_p^2} + \dots + R(u_p^n, \nu \delta t) e^{i\theta_p^n})$$

where

$$R(u, t) = \sqrt{-4t \log u}$$

And u's and  $\theta$ 's are independent random variables with uniform density

For advection-diffusion problems: similar formula alternating particles advection and random walks

Can be interpreted as quadrature of exact solution (repeated convolution with Gaussian kernels) on N random points over a space on dimension n  $\rightarrow$  convergence rate  $1/\sqrt{N}$

Physically and mathematically appealing, but not very accurate

## Particle resampling schemes

Idea: evaluate gaussians on particles and use these values to update their weights

Can be interpreted by rewriting diffusion as an integral operator

$$\Delta_\varepsilon \omega(\mathbf{x}) = \varepsilon^{-2} \int (\omega(\mathbf{y}) - \omega(\mathbf{x})) \eta_\varepsilon(\mathbf{y} - \mathbf{x}) d\mathbf{y}$$

Theorem: if kernel  $\eta$  satisfies the following moment properties:

$$\left\{ \begin{array}{lcl} \int x_i x_j \eta(\mathbf{x}) d\mathbf{x} & = & 2\delta_{ij} \text{ for } i, j = 1, 2 \\ \int x_1^{i_1} x_2^{i_2} \eta(\mathbf{x}) d\mathbf{x} & = & 0 \text{ if } i_1 + i_2 = 1 \text{ or } 3 \leq i_1 + i_2 \leq r + 1 \\ \int |\mathbf{x}|^{r+2} |\eta(\mathbf{x})| d\mathbf{x} & < & \infty \end{array} \right.$$

then

$$\|\Delta_\varepsilon \omega - \Delta \omega\|_{0,p} \leq C \varepsilon^r \|\omega\|_{r+2,p}$$

In practice, any positive, compacted supported, radially symmetric function, after proper normalization, will do it

## Resulting particle scheme

$$\omega^h(\mathbf{x}, t) = \sum_p v_p \omega_p^h(t) \delta(\mathbf{x} - \mathbf{x}_p^h)$$

Here, we distinguish local values and volumes that make the particles strengths

$$\frac{d\omega_p^h}{dt} = \nu \varepsilon^{-2} \sum_q (v_q \omega_q^h - v_p \omega_p^h) \eta_\varepsilon(\mathbf{x}_q^h - \mathbf{x}_p^h)$$

Not constrained with time-stepping, can be high order

Slightly more expensive than random walk

This approach extends to anisotropic diffusion

$$\operatorname{div}[B \nabla u](\mathbf{x}) \simeq \varepsilon^{-2} \sum_{i,j=1}^d \int \psi_{ij}^{\varepsilon}(\mathbf{x} - \mathbf{y}) M_{ij}(\mathbf{x}, \mathbf{y}) [u(\mathbf{y}) - u(\mathbf{x})] d\mathbf{y}$$

with (for instance)

$$\psi_{ij}(\mathbf{x}) = x_i x_j \theta(|\mathbf{x}|) \quad M_{ij}(\mathbf{x}, \mathbf{y}) = m_{ij} \left( \frac{\mathbf{x} + \mathbf{y}}{2} \right)$$

$$\int x_i^4 \theta(\mathbf{x}) d\mathbf{x} = d + 2 \quad m = B - \frac{1}{d+2} \operatorname{tr} B$$

# Implicit subgrid-scale models in particle methods

Focus on 3D Euler equations (inviscid flows) in vorticity formulation

$$\frac{\partial \boldsymbol{\omega}}{\partial t} + \operatorname{div}(\mathbf{u}\boldsymbol{\omega}) - [\nabla \mathbf{u}]\boldsymbol{\omega} = 0$$

Particle solution given by

$$\boldsymbol{\omega}(\mathbf{x}, t) = \sum_p \boldsymbol{\Gamma}_p \delta(\mathbf{x} - \mathbf{x}_p)$$

$$\frac{d\mathbf{x}_p}{dt} = \bar{\mathbf{u}}(\mathbf{x}_p, t), \quad \frac{d\boldsymbol{\Gamma}_p}{dt} = \nabla \bar{\mathbf{u}}(\mathbf{x}_p, t) \boldsymbol{\Gamma}_p$$

where  $\bar{u}$  is a mollified velocity field.

weak solution to	$\frac{\partial \omega_i}{\partial t} + \operatorname{div} \bar{\mathbf{u}} \omega_i - \operatorname{div} \boldsymbol{\omega} \bar{u}_i = 0$
------------------	--



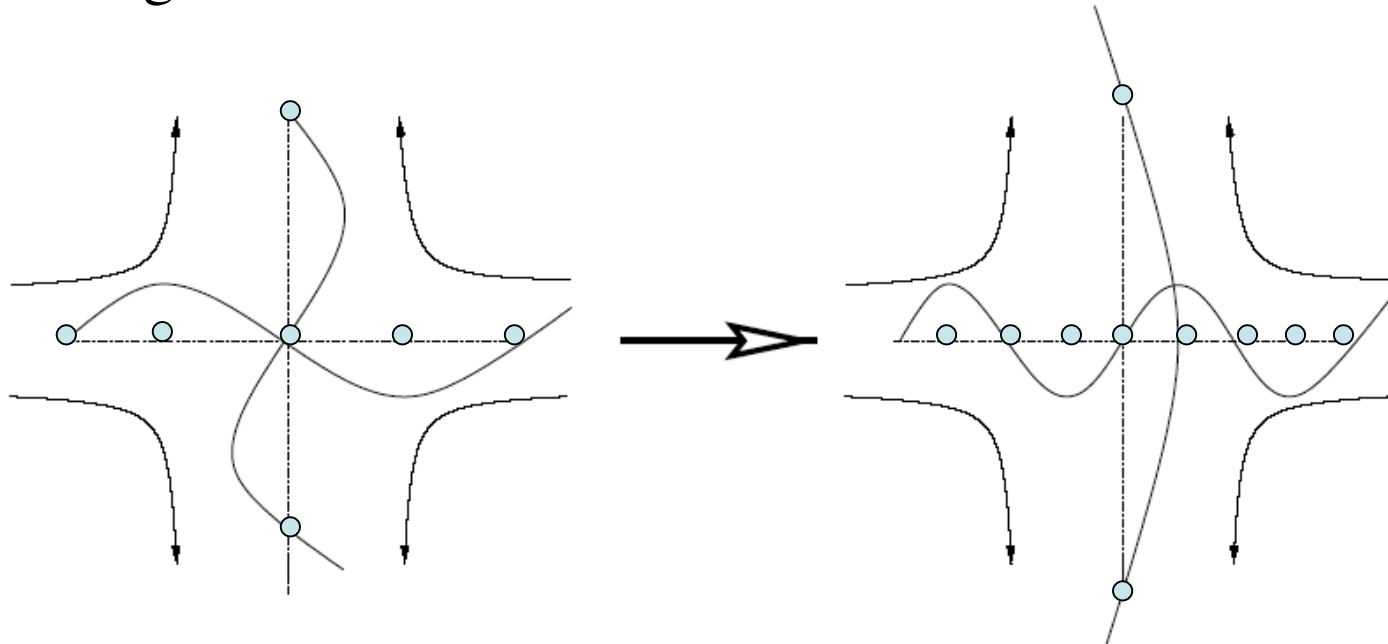
Molified particles (blobs) thus satisfy

$$\frac{\partial \overline{\omega}_i}{\partial t} + \operatorname{div} \overline{\mathbf{u} \omega}_i - \operatorname{div} \overline{\omega \mathbf{u}}_i = 0$$

This is an **averaged** Euler equations

This means that the particle method is achieving some subgrid scale (implicit) model

The transfer from larges scales to samll scales can be illustrated by the following sketch



More precise analysis needed to understand enstrophy transfers  
 For simplicity, consider 2D case and start from

$$\frac{\partial \omega_\varepsilon}{\partial t} + \operatorname{div}(u_\varepsilon \omega_\varepsilon) = E,$$

Where  $\omega_\varepsilon = \overline{\omega} = \omega * \zeta_\varepsilon$

$$E(x) = \operatorname{div}_x \int \omega(y) [u_\varepsilon(x) - u_\varepsilon(y)] \zeta_\varepsilon(x - y) dy$$

$$E = E_1 + E_2$$

$$E_1(x) = \operatorname{div}_x \left( \omega(x) \int [u_\varepsilon(x) - u_\varepsilon(y)] \zeta_\varepsilon(x - y) dy \right)$$

$$E_2(x) = \operatorname{div}_x \int [\omega(y) - \omega(x)] [u_\varepsilon(x) - u_\varepsilon(y)] \zeta_\varepsilon(x - y) dy$$

First term is a drift which does not contribute to enstrophy balance

$$E_1 = \operatorname{div}(\tilde{u}_\varepsilon \omega)$$

Second term can be rewritten after Taylor expansions as

$$E_2(x) = \operatorname{div}_x \sum_{i,j} \int [(y_i - x_i) \partial_i u_\varepsilon(x)] [(x_j - y_j) \partial_j \omega(x)] \zeta_\varepsilon(x - y) dy$$

For symmetry reasons, cross terms disappear and we are left with

$$E_2 = m_2 \varepsilon^2 \operatorname{div}([Du_\varepsilon] \nabla \omega) + O(\varepsilon^4)$$

with 
$$m_2 = \frac{1}{2} \int |x|^2 \zeta(x) dx$$

In other words, « equivalent » equation for large scales (blobs) is  
Euler + diffusion with anisotropic eddy viscosity

Back to the enstrophy balance: compute  $\int \omega E dx$  (drop  $\varepsilon$  for simplicity)

$$\frac{1}{2} \frac{d}{dt} \int \omega^2 dx = \int \int \omega(x) \omega(y) [u(x) - u(y)] \cdot \nabla \zeta(x - y) dx dy.$$

$$\omega(x) = \omega(y) + \omega(x) - \omega(y)$$

$$\begin{aligned} \frac{1}{2} \frac{d}{dt} \int \omega^2 dx &= \int \int \omega^2(y) [u(x) - u(y)] \cdot \nabla \zeta(x - y) dx dy \\ &+ \int \int [\omega(x) - \omega(y)] \omega(y) [u(x) - u(y)] \cdot \nabla \zeta(x - y) dx dy \end{aligned}$$

First term vanishes ( $\text{div } u = 0$ ) and for second term write

$$2\omega(y) = [\omega(x) + \omega(y)] - [\omega(x) - \omega(y)]$$

For symmetry reasons,  $(\omega(x) + \omega(y))$  does not contribute, left with

$$\frac{d}{dt} \int \omega^2 dx = - \int \int [\omega(x) - \omega(y)]^2 [u(x) - u(y)] \cdot \nabla \zeta(x - y) dx dy$$

Positive contributions come from points where

$$[u(x) - u(y)] \cdot \nabla \zeta(x - y) < 0$$

In practice, cut-off is a decreasing function of radius, and condition becomes

$$[u(x) - u(y)] \cdot (x - y) > 0$$

Coherent with the first sketch: backscatter through diverging particles

In practice, codes have always trouble to handle backscatter:  
Need to compensate ,at least partially

Two ways to prevent excess backscatter in a particle code:

- remesh particles or
- use above calculations to tune the proper diffusion model:

$$\frac{d\omega_p}{dt} = \sum_q (\omega_p - \omega_q) v_q \{ [u(x_p) - u(x_q)] \cdot [x_p - x_q] f'_\varepsilon(|x_p - x_q|) |x_p - x_q|^{-1} \} -$$

where  $\varsigma(x) = f(|x|)$

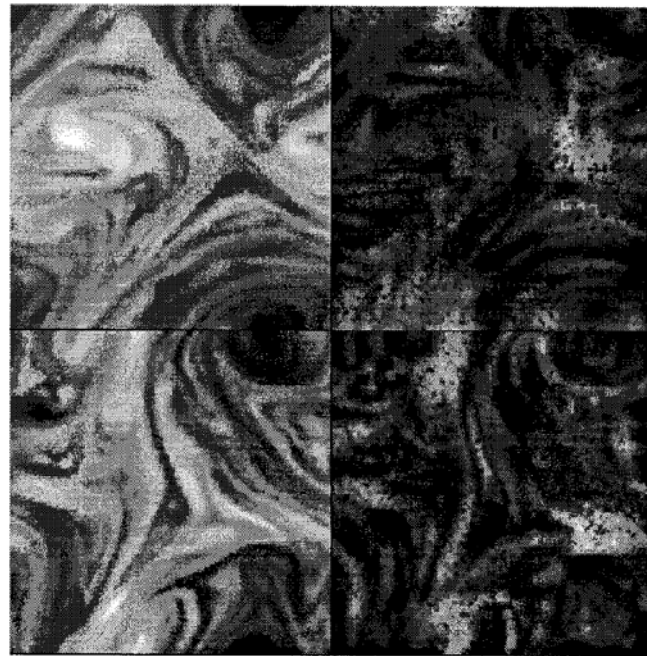
**Non-linear diffusion, acting only on directions of diverging particles**  
(positive eigenvalues of strain tensor)

Illustration: 2D decaying turbulence:  
correlations between eddy viscosity and high strain zones

$$\nu(x) = m_2 \varepsilon^2 \sup_y \frac{[[u(x) - u(y)] \cdot [x - y]]_+}{|x - y|^2}$$

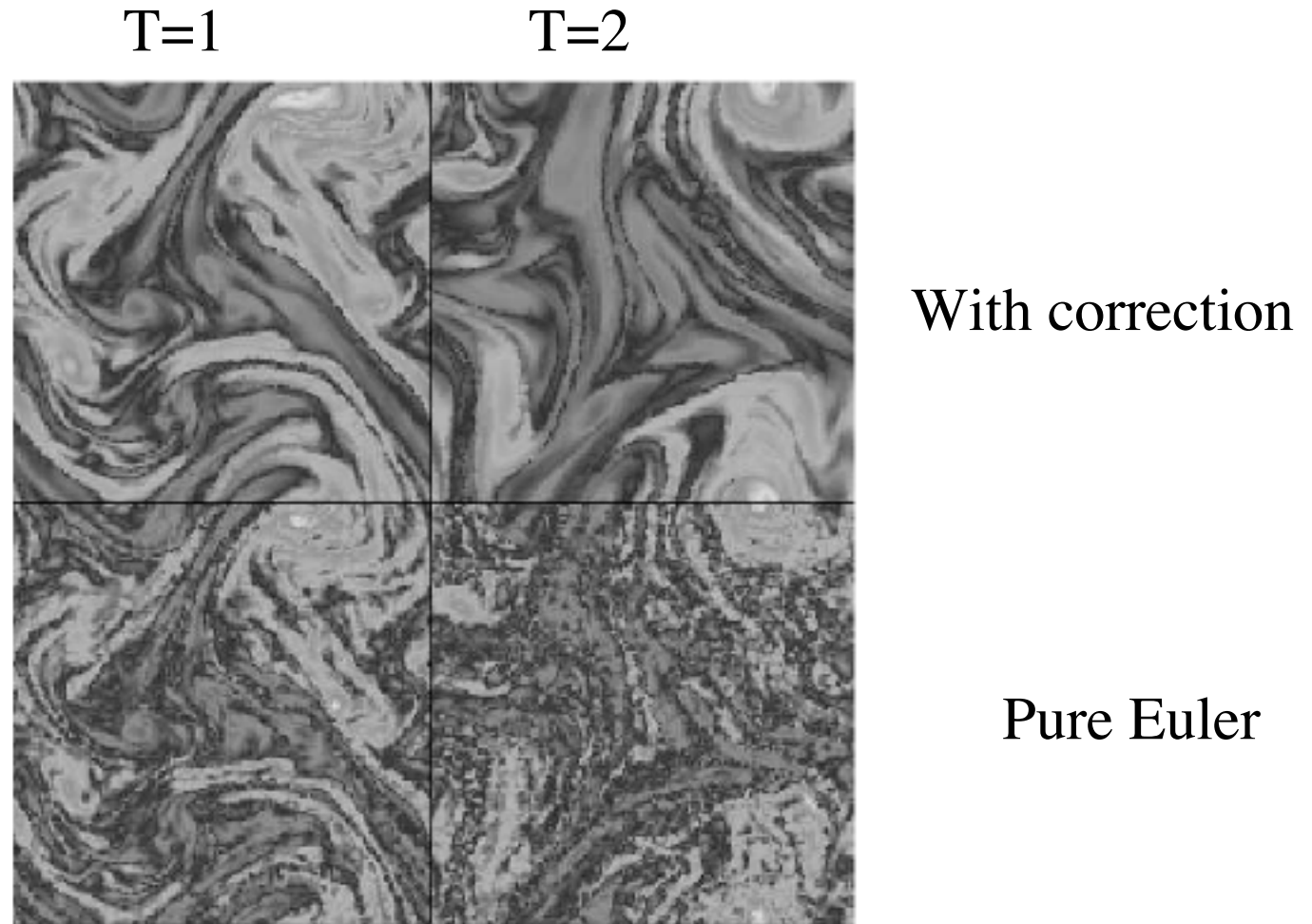
T=2

T=1



vorticity    viscosity

# Comparison of pure Euler simulation (with backscatter) and eddy-viscosity correction





Rk: More traditional way to cope with small scale:  
hairpin removal (Chorin, Leonard)

Conclusion:

- particles give a nice insight into subgrid level
- differences and links with « conventional »subgrid scale modeling in grid-based methods
- numerical analysis allows to control excess backscatter

# Adaptive, multilevel particle methods

First remark: particles methods are naturally adaptive for 2 reasons:

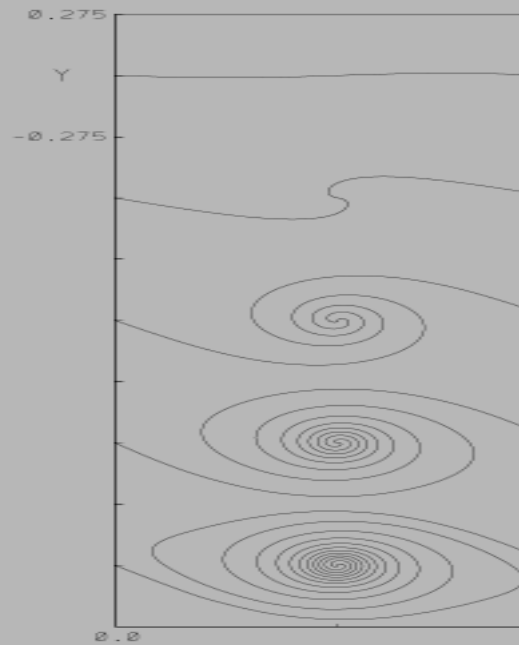
- Particles in general follow zones of interest, which may occupy only a small area

Ex: electron beams in plasma, shear layers in flows

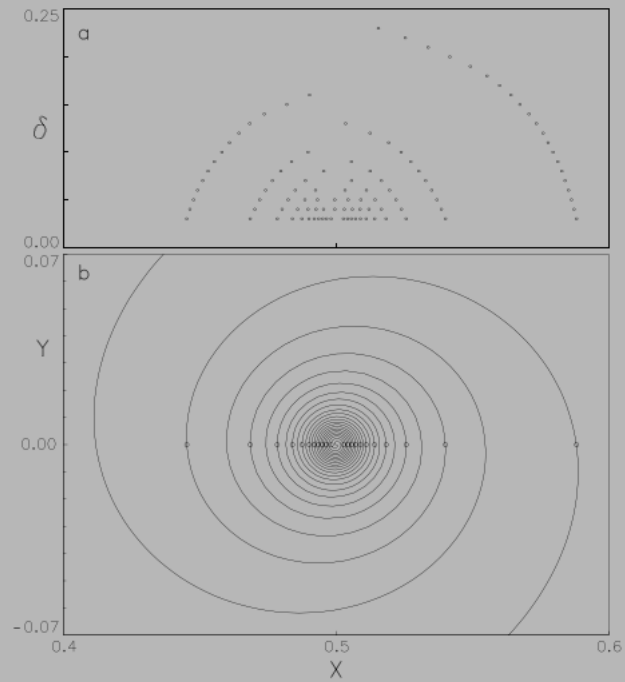
- If connectivity between particle can be tracked, there are easy ways to « refine » locally

Ex: vortex sheet calculations of Krasny in 2d and Meiburg in 3D.

# Krasny's 2D vortex sheet calculations



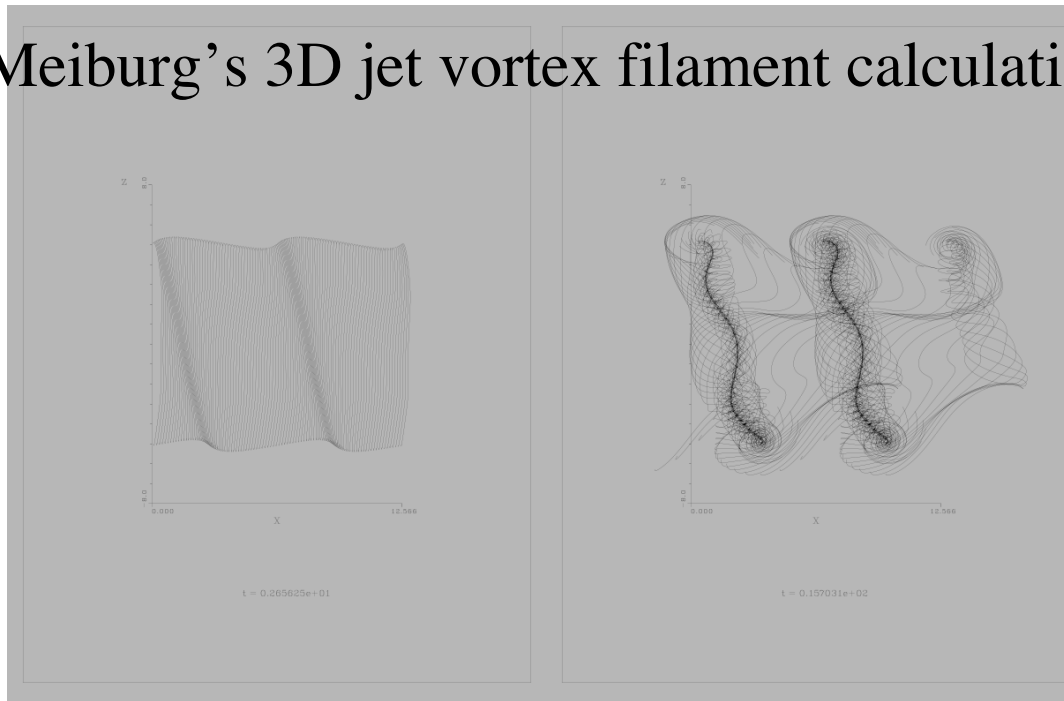
Time evolution of  
the vortex sheet



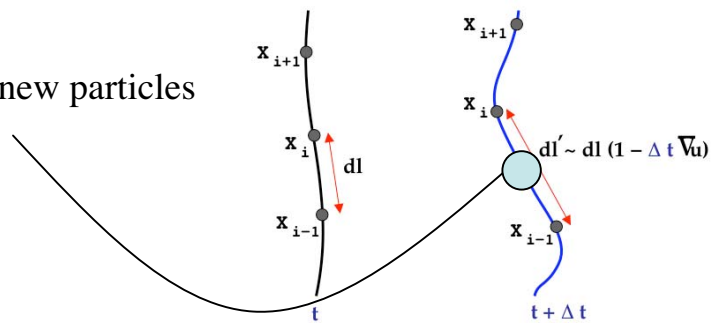
Close view of the spiral

# Meiburg's 3D jet vortex filament calculations

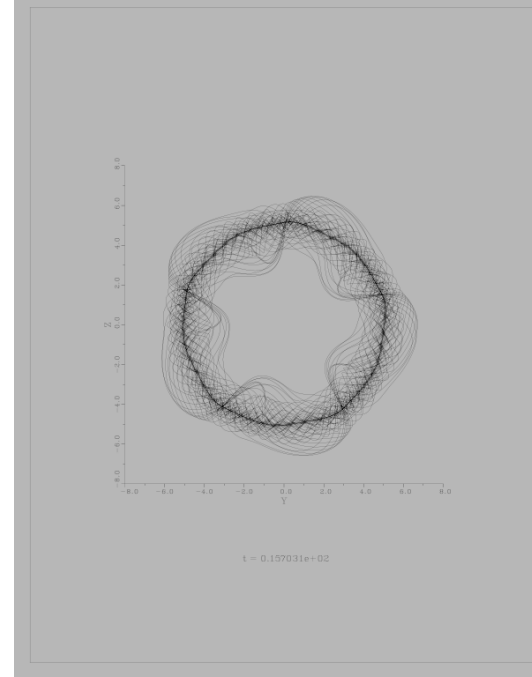
Side views



Inserting new particles



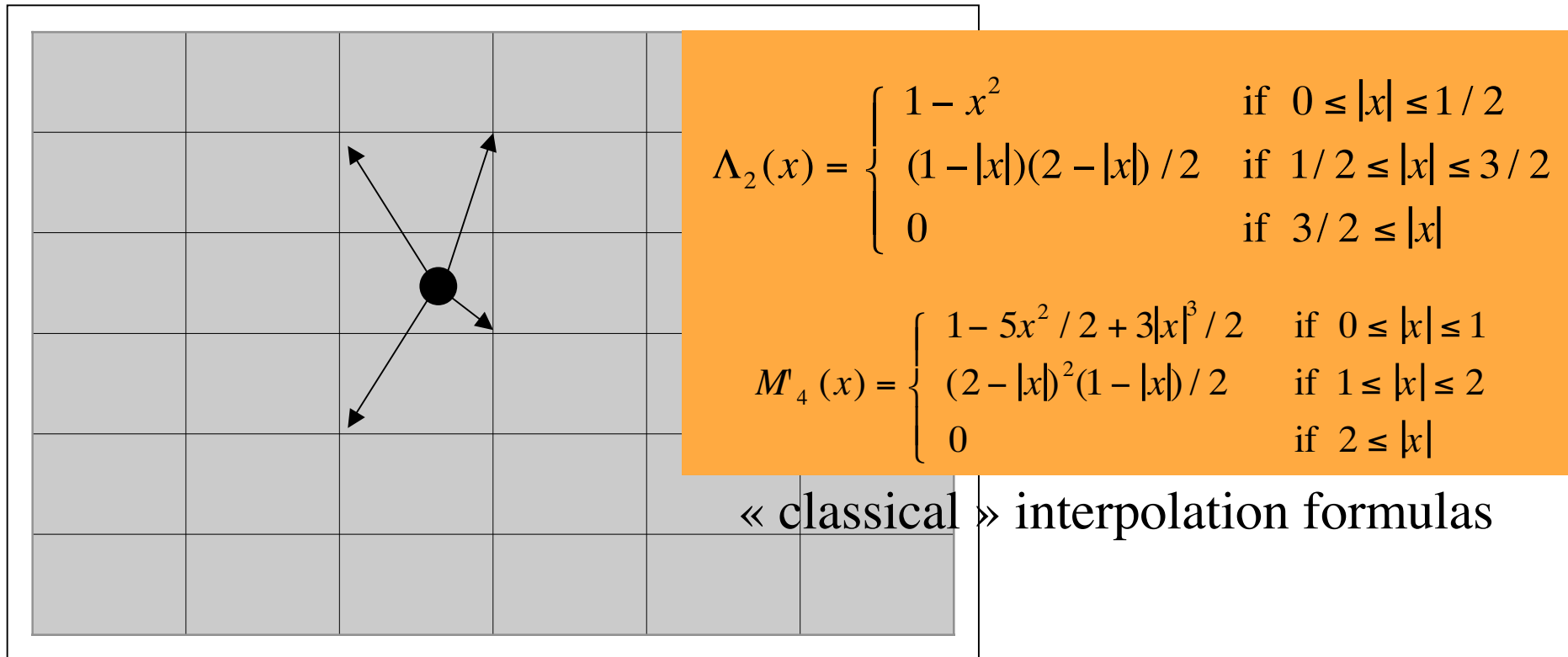
Front view



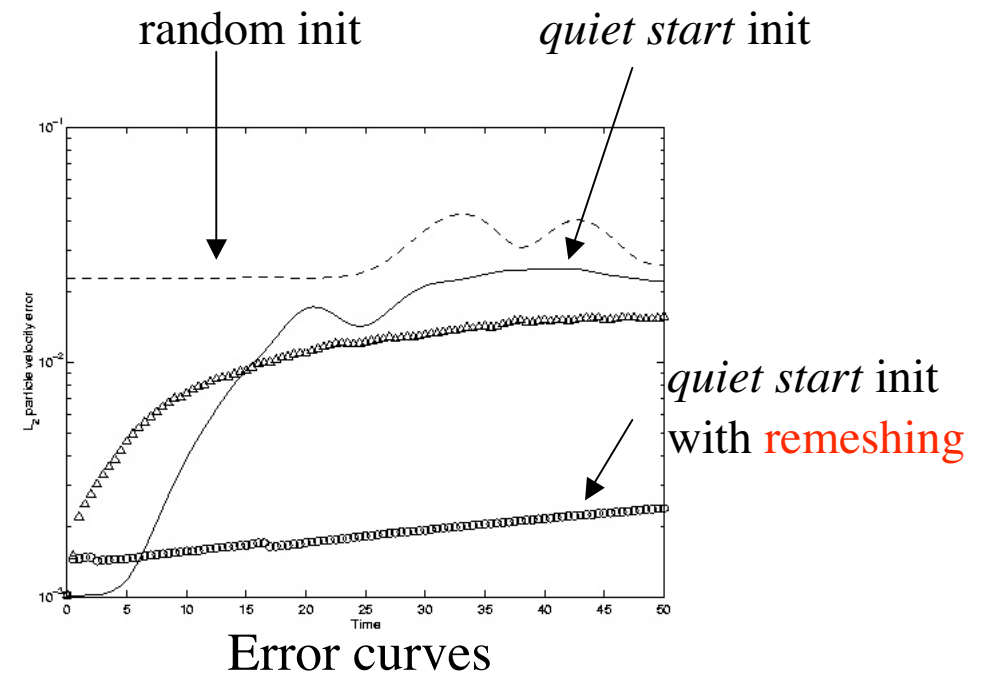
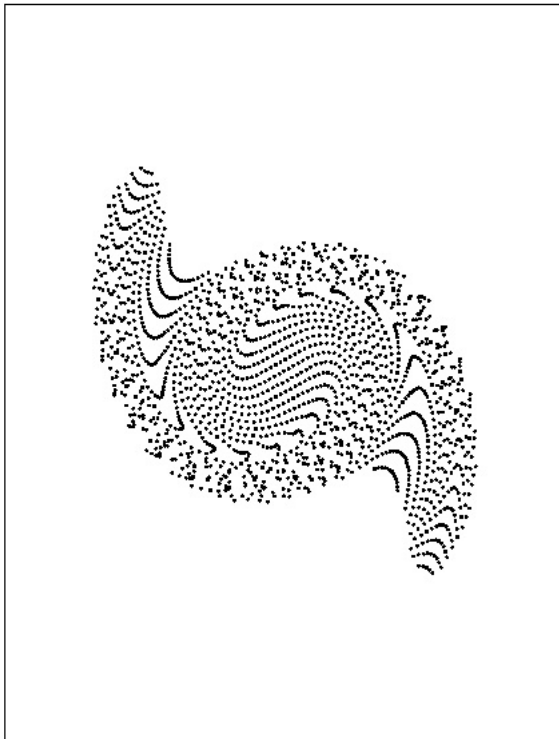
However these techniques do not apply with more complex vorticity generation and/or in presence of diffusion

Accuracy requires frequent regridding  
with high order formulas

conservation of  $\int f dx, \int xf dx, \int x^2 f dx$



Typical example showing importance of regridding: circular patch with high strain



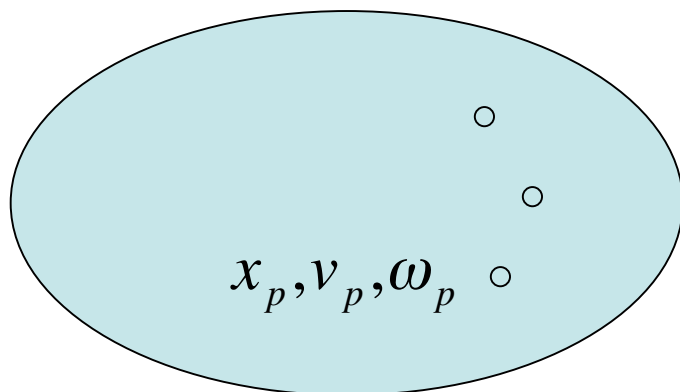
Regridding can also be used as a way to adapt particles to the flow topology -> variable-size particles

three different approaches:

- Regridding into variable-sized particles via global mappings
- Regridding via local mappings
- Regridding onto piecewise uniform particles



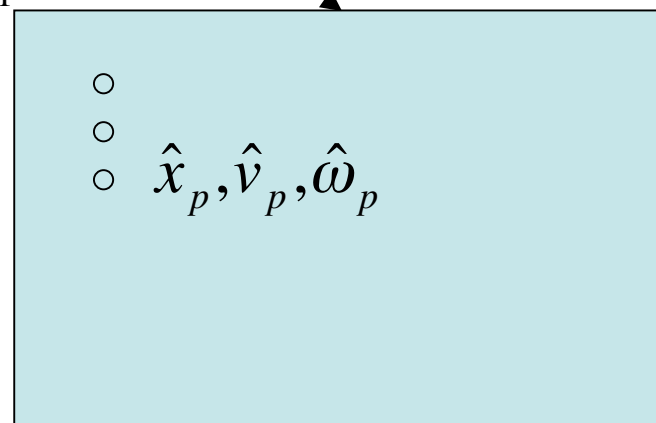
Physical space



$$x = F(\hat{x}); G = F^{-1}$$

$$a_{ij} = \frac{\partial G_i}{\partial x_j} \quad J = \det[a_{ij}]$$

$$v_p J(\mathbf{x}_p) = \hat{v}_p$$



Mapped space

1) Mollified particles (for field evaluation) with blob size

$$\varepsilon_p = \hat{\varepsilon} J^{-1}(x_p)$$

2) Diffusion in mapped coordinates:

$$\Delta_{\mathbf{x}} u = \sum_{i,j,k} a_{ji} \frac{\partial}{\partial \hat{x}_j} \left( a_{ki} \frac{\partial \hat{u}}{\partial \hat{x}_k} \right)$$

Can be written in divergence form, using the fact that

$$\sum_{j=1}^d \frac{\partial}{\partial \hat{x}_j} (a_{ji} J) = 0$$

$$\Delta_{\mathbf{x}}u = J \sum_{j,k} \frac{\partial}{\partial \hat{x}_j} \left( b_{jk} \frac{\partial \hat{u}}{\partial \hat{x}_k} \right) \quad b_{jk} = J^{-1} \sum_i a_{ki} a_{ji}$$

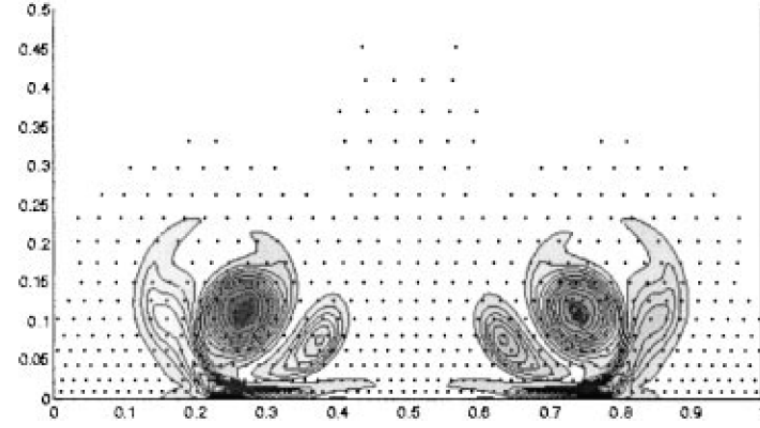
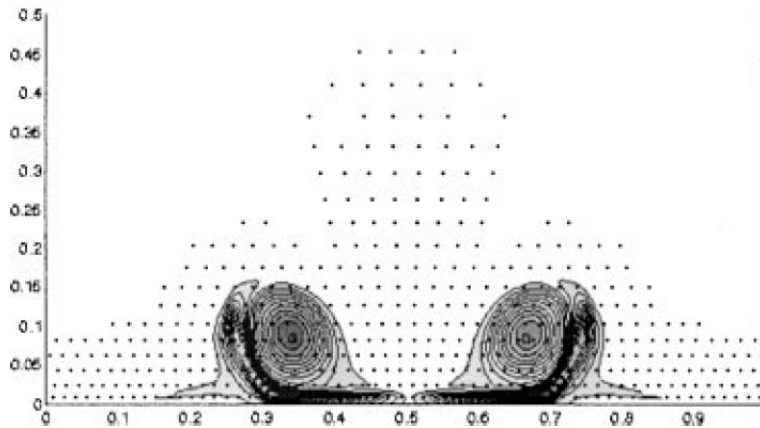
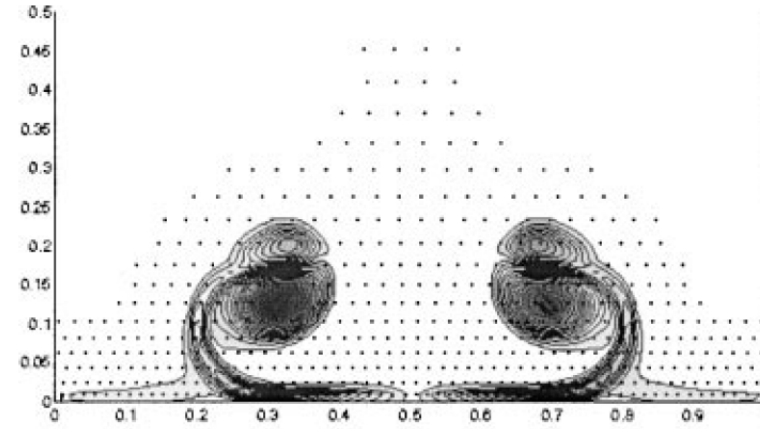
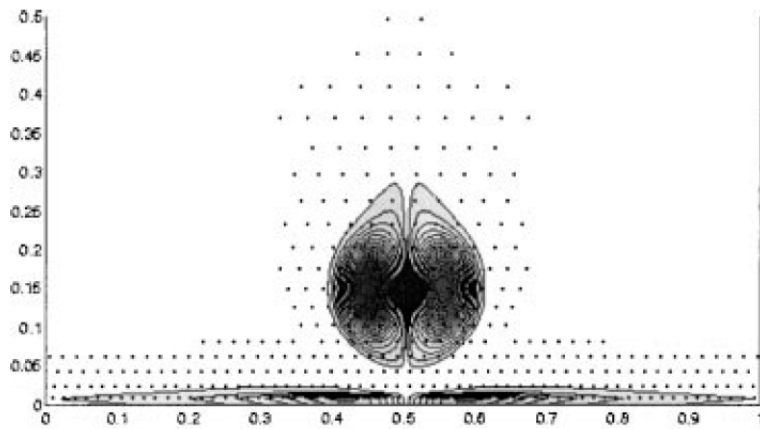
Using particle diffusion formulas for anisotropic diffusion we obtain

$$\operatorname{div}[B \nabla u](\mathbf{x}) \simeq \varepsilon^{-2} \sum_{i,j=1}^d \int \psi_{ij}^{\varepsilon}(\mathbf{x} - \mathbf{y}) M_{ij}(\mathbf{x}, \mathbf{y}) [u(\mathbf{y}) - u(\mathbf{x})] d\mathbf{y}$$

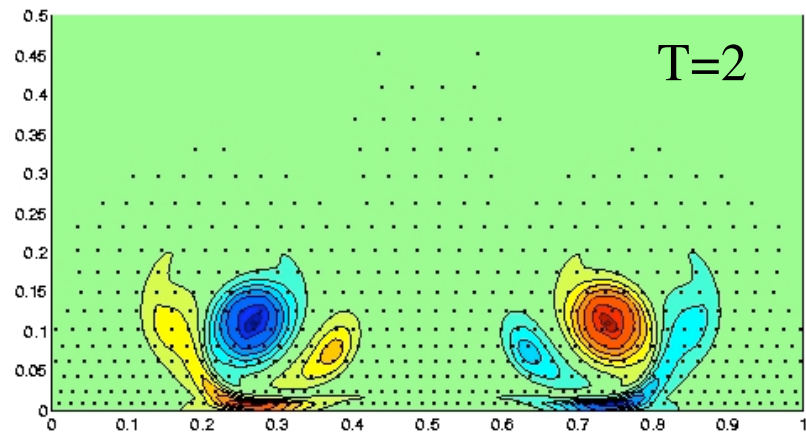
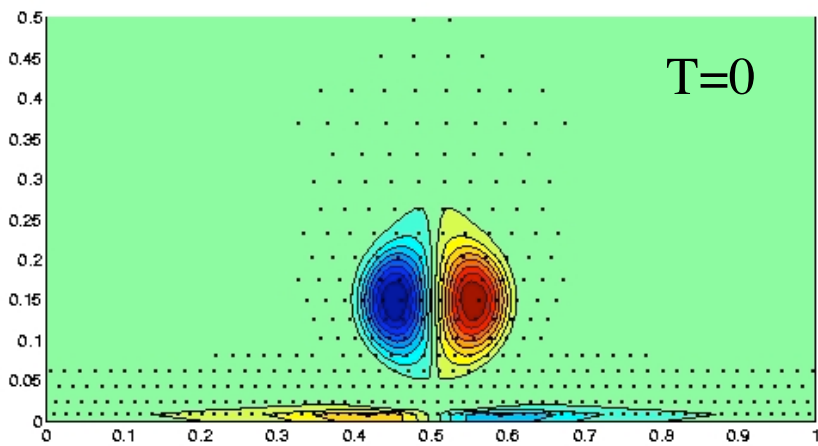
$$\begin{aligned} \frac{d\omega_p}{dt} = & v\varepsilon^{-4} J(\mathbf{x}_p) \sum_{q,i,j} \hat{v}_q (\hat{x}_p^i - \hat{x}_q^i) (\hat{x}_p^j - \hat{x}_q^j) \theta^\varepsilon(\hat{\mathbf{x}}_p - \hat{\mathbf{x}}_q) \\ & \times \left[ b_{ij} - \frac{1}{d+2} \sum_i b_{ii} \delta_{ij} \right] \left( \frac{\hat{\mathbf{x}}_p + \hat{\mathbf{x}}_q}{2} \right) (\omega_q - \omega_p). \end{aligned}$$

Remark: conservative in physical variables

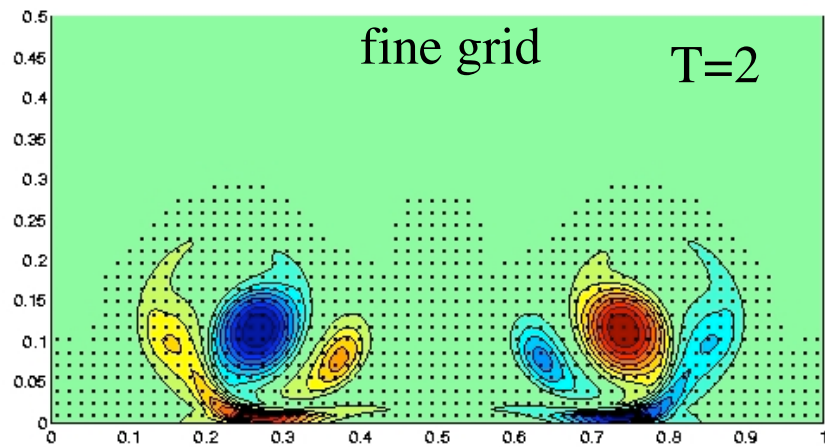
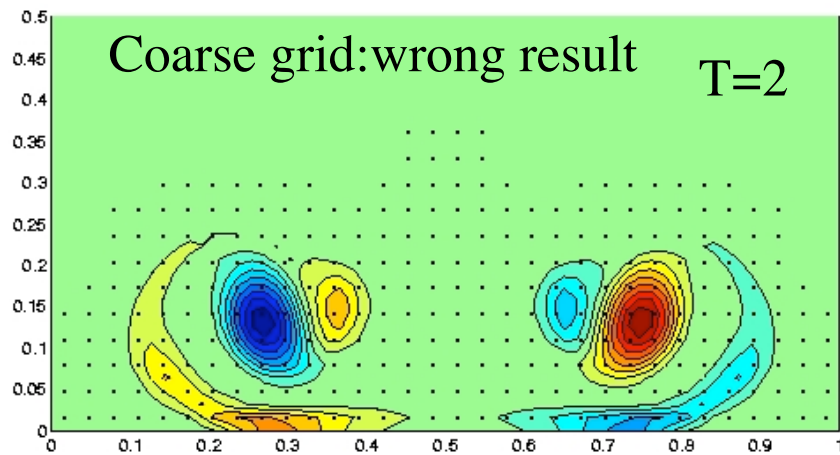
$$v_p J(\mathbf{x}_p) = \hat{v}_p \quad \rightarrow \quad \frac{d}{dt} \left( \sum_p v_p \omega_p \right) = 0$$

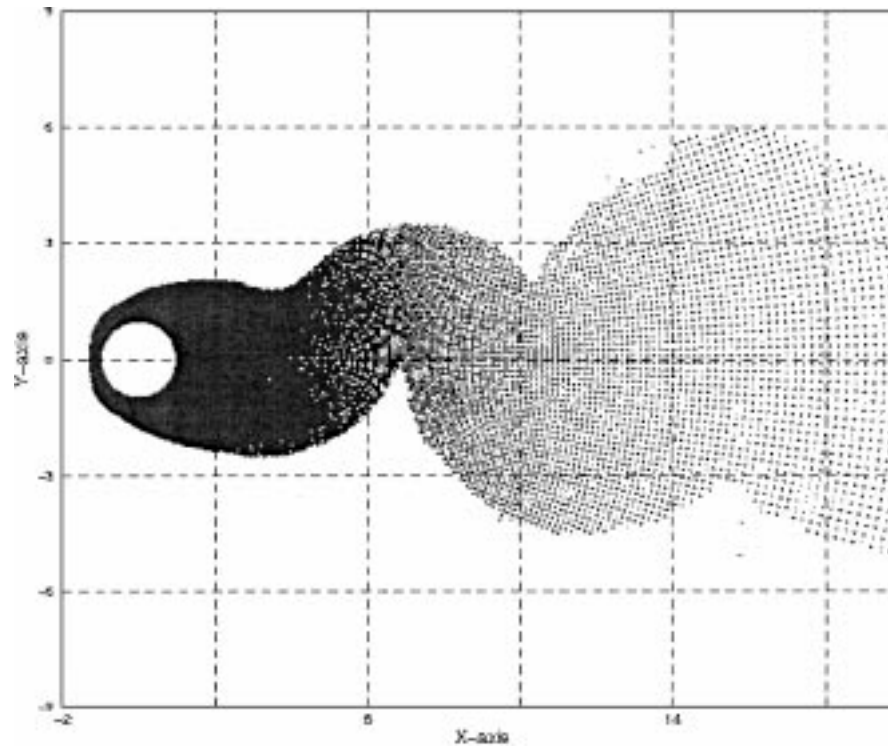


Example: rebound of a dipole with exponentially stretched particle



Coarse to fine grid





Flow around 2D cylinder - Variable-size particles

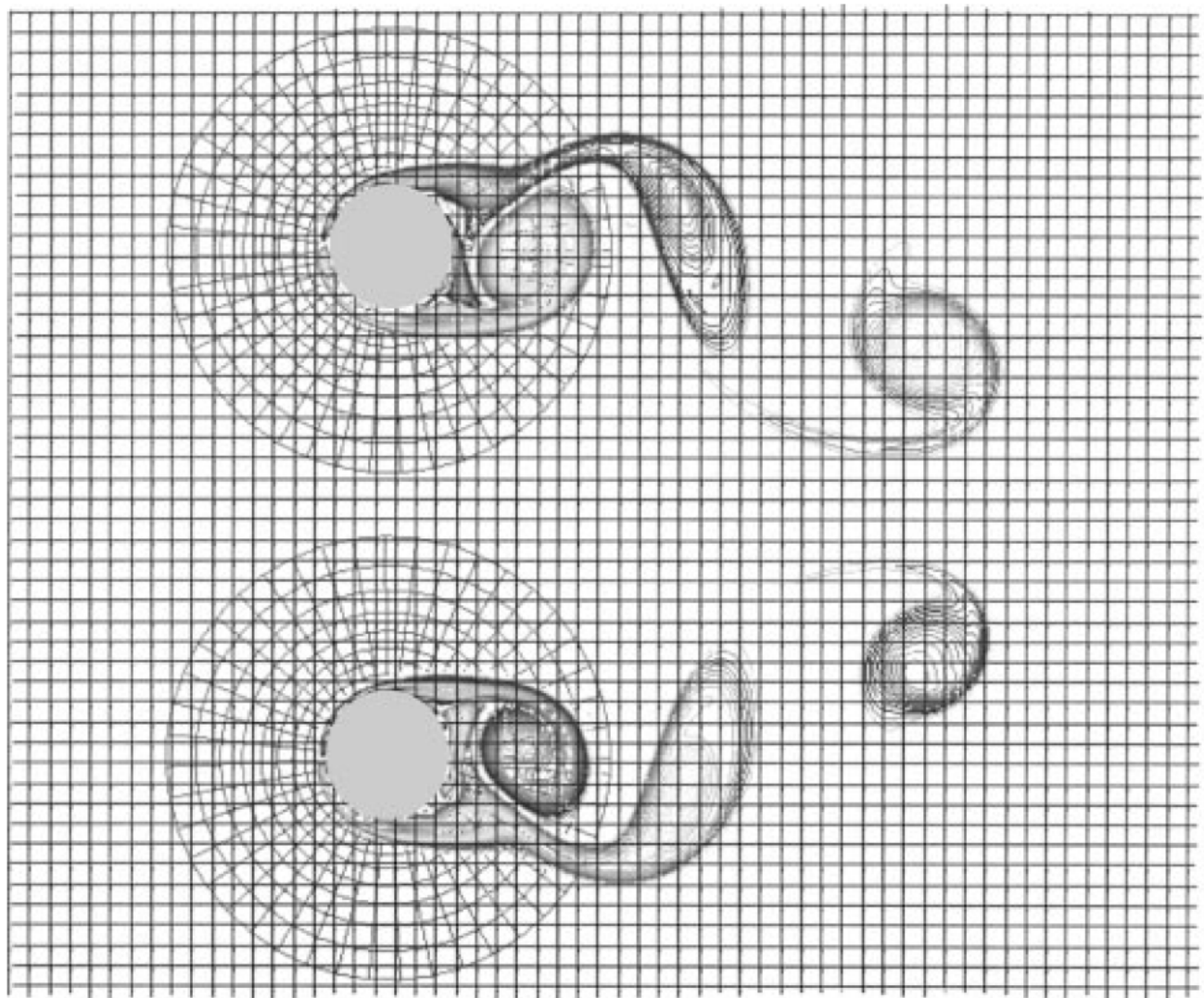
Rather flexible (about 10 lines of codes more than in the « uniform blob » case) but global mapping not always known analytically

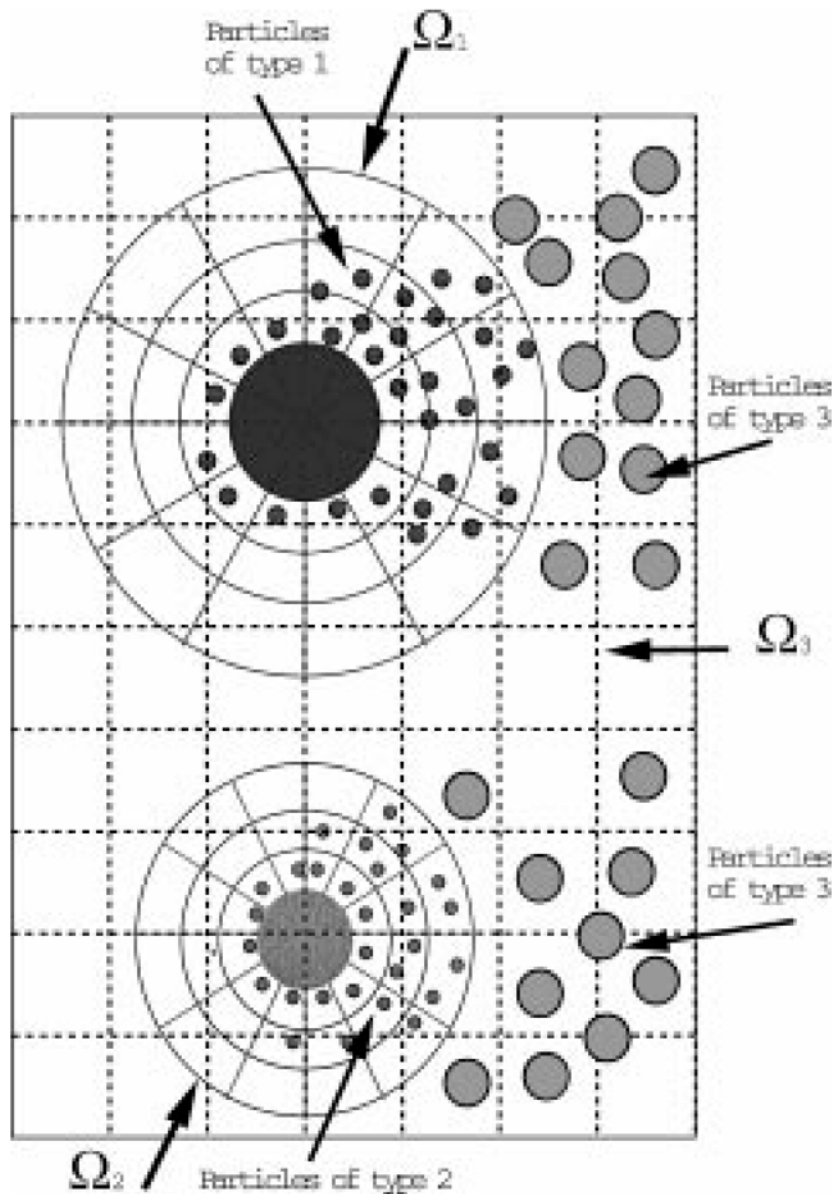
Other option is to combine several local mappings (domain decomposition approach)

Example of 2 obstacles in a flow:

3 mappings: 1 around each obstacle, with refinement in the boundary layer, 1 in between with uniform blobs.







Based on overlapping of domains:

- At beginning of step, in the buffer zone:

- ✓ Particles 1 and 2 are remeshed onto particles of type 3 (with cartesian mapping)

- ✓ Particles of type 3 remeshed onto particles of type 1 and 2 (with polar mapping)

- Then particles are advanced and exchange vorticity in each subdomain

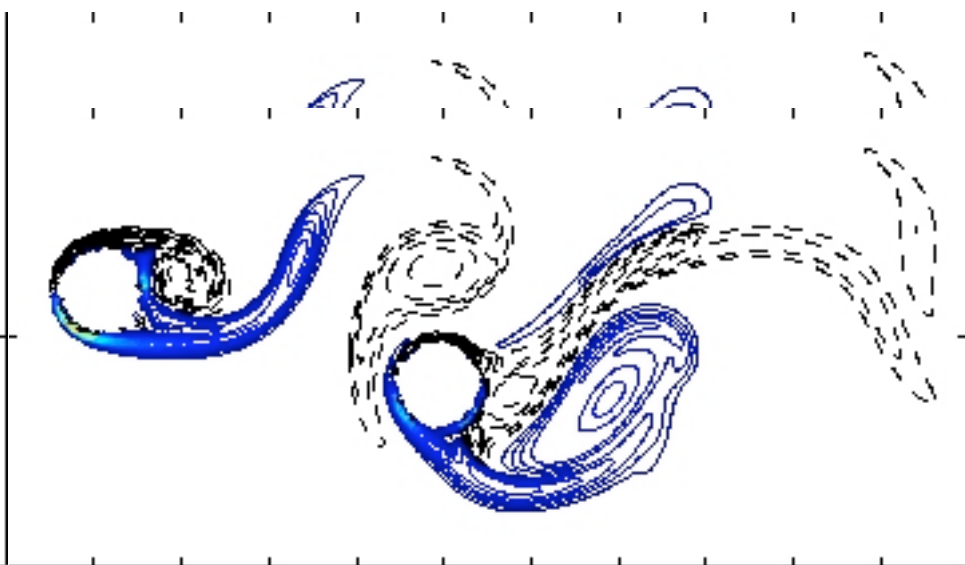
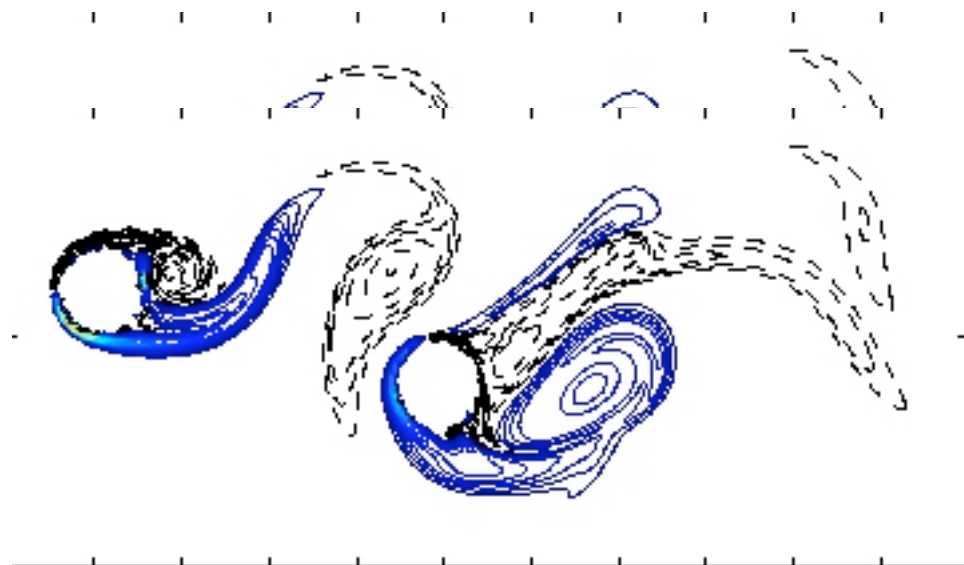
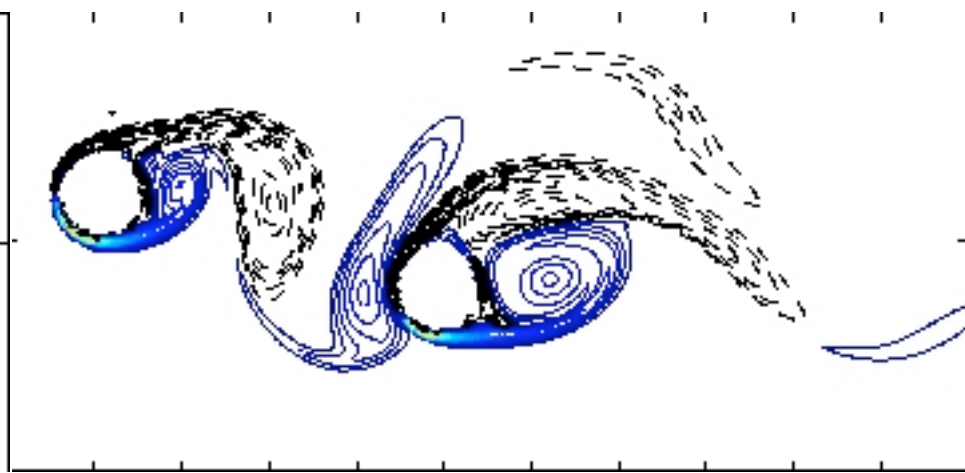
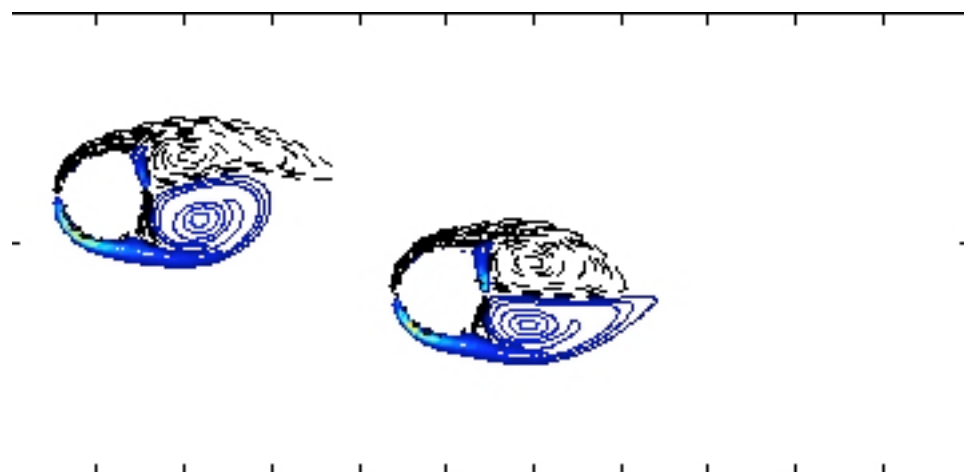
Solution is correct except near the boundaries of overlapping zone

Overlapping zone must be big enough to contain trajectories for one time-step and diffusion stencil (a few grid points)

**Regridding provides interpolation needed to exchange information between non-conforming subdomains**

Method can be extended to Adaptive Mesh Refinement:

- Define zones with hierarchical grid levels, **with overlapping**, based on velocity gradients
- Use regridding in overlapping to exchange information between grids at different levels



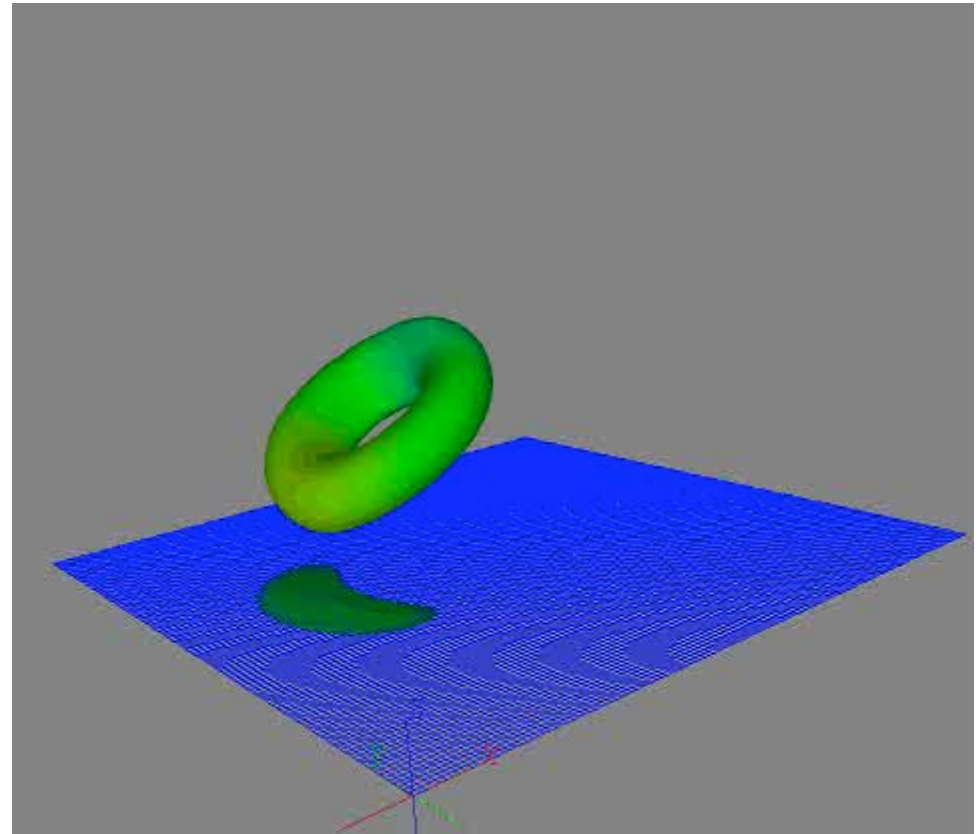
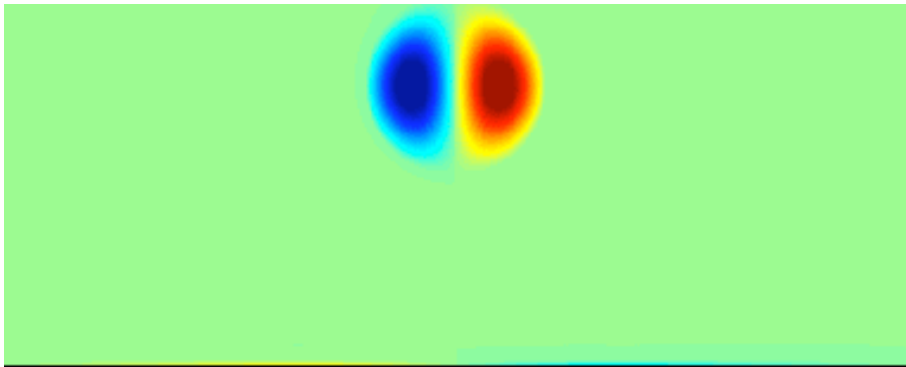
Conclusion: particles have an unusual flexibility to deal with very non-conforming domain decomposition (for advection dominated problems) that allow to adapt locally grid-resolution.

Major achievements of adaptive particle methods are in particular case (sheets, filaments).

Many open pending applications of more general adaptive schemes ..

Intermezzo: dipole ring interaction

- 2D case at moderate Reynolds number ( $Re=3,200$ )
- 3D case: inclined ring impinging on a wall ( $Re=1,400$ )

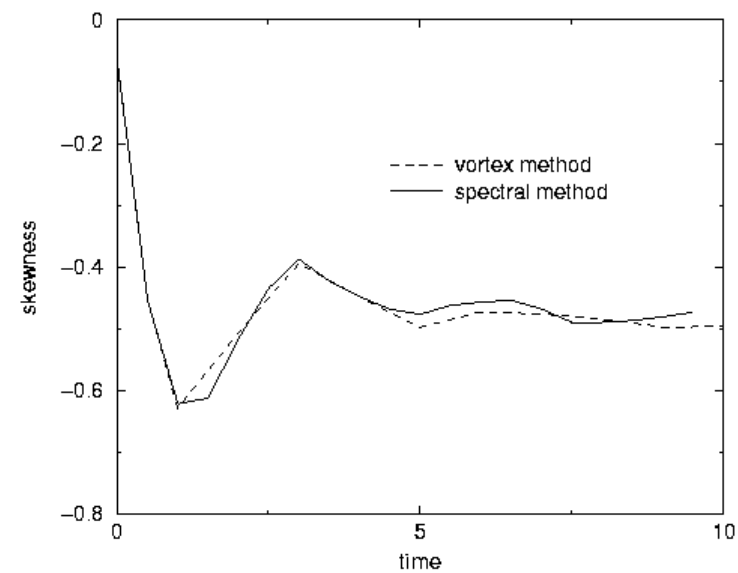
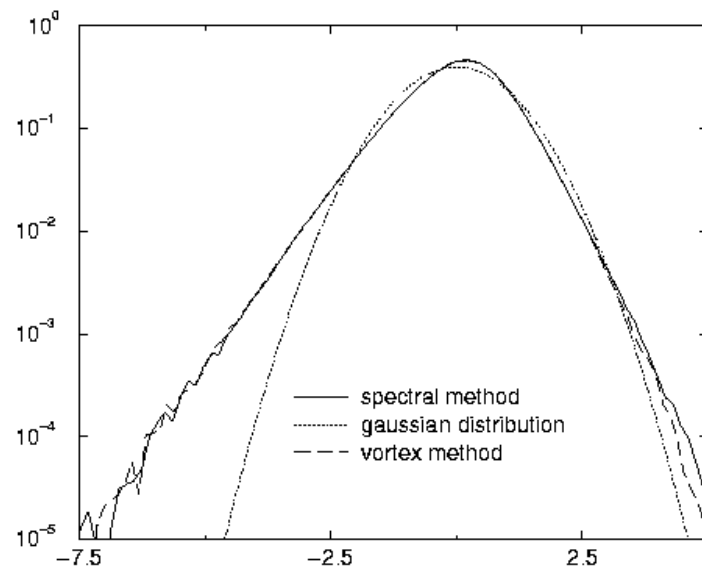
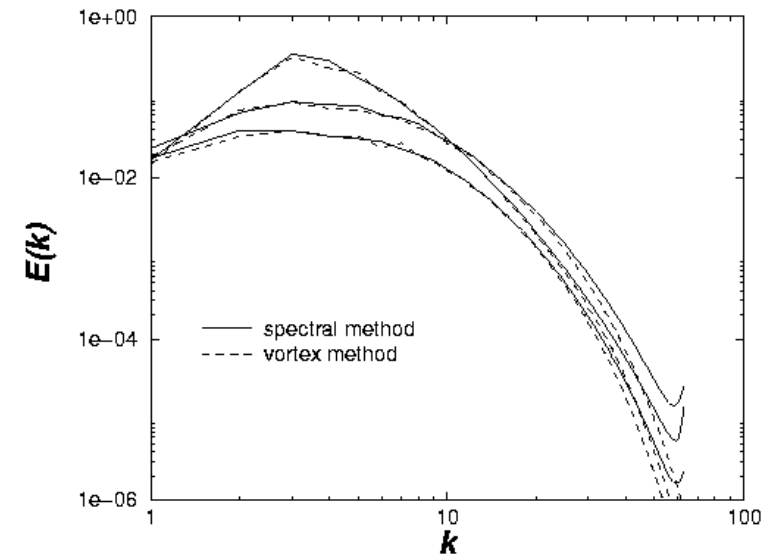
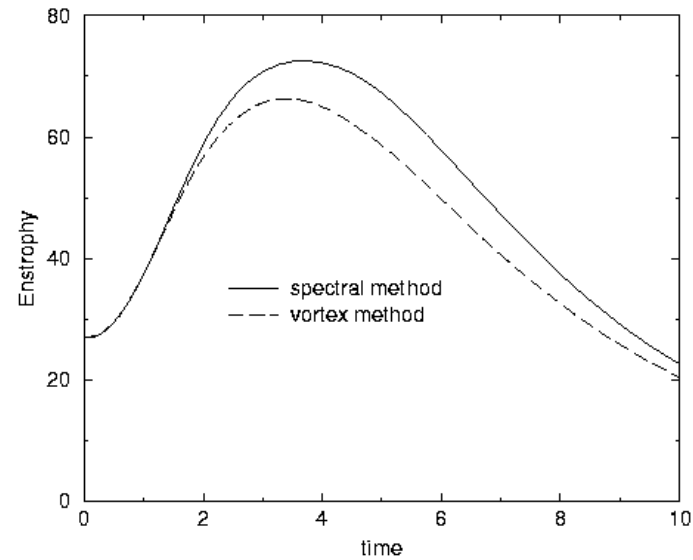


Systematic investigation of truncation error (in particular remeshing)  
In DNS of Isotropic homogeneous turbulence

I. DNS regime ( $128^3$  calculations)

II LES regime (Comte-Bellot experiment,  $32^3$  calculations)

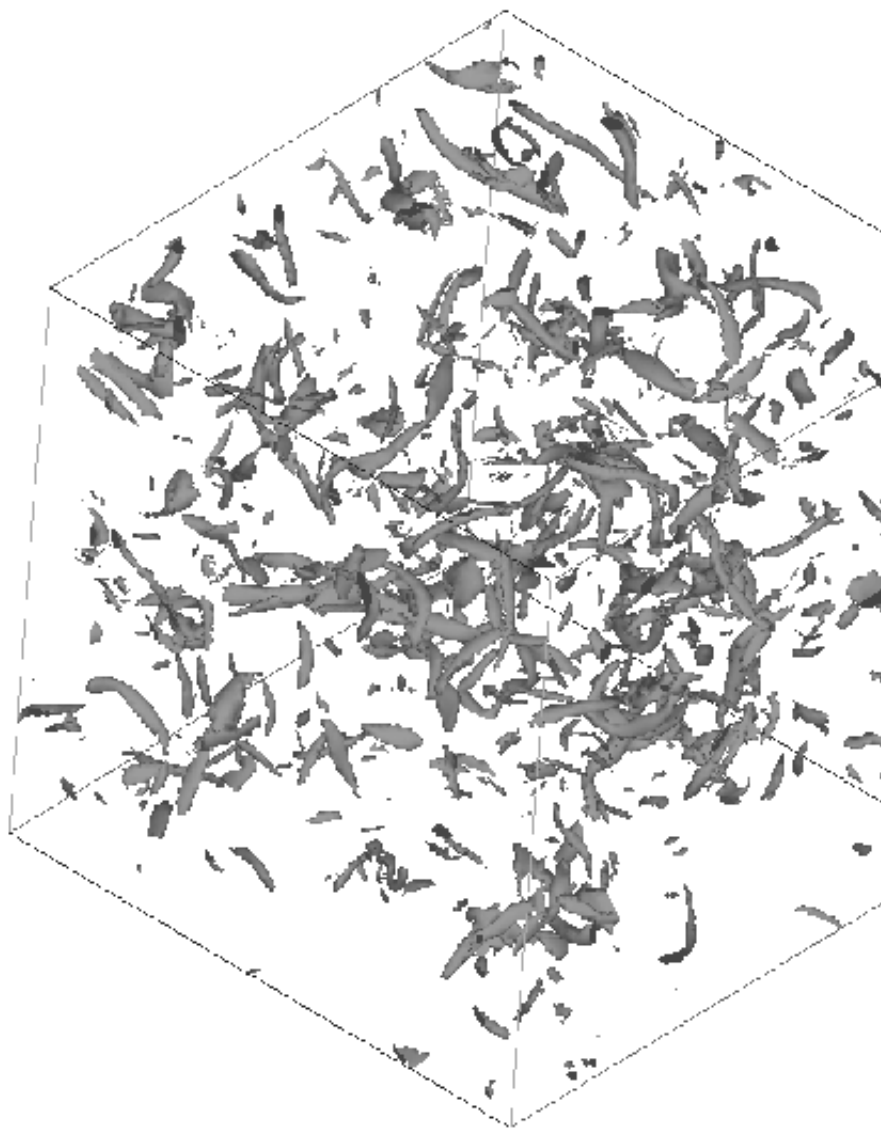
# Statistics: comparisons with spectral reference results



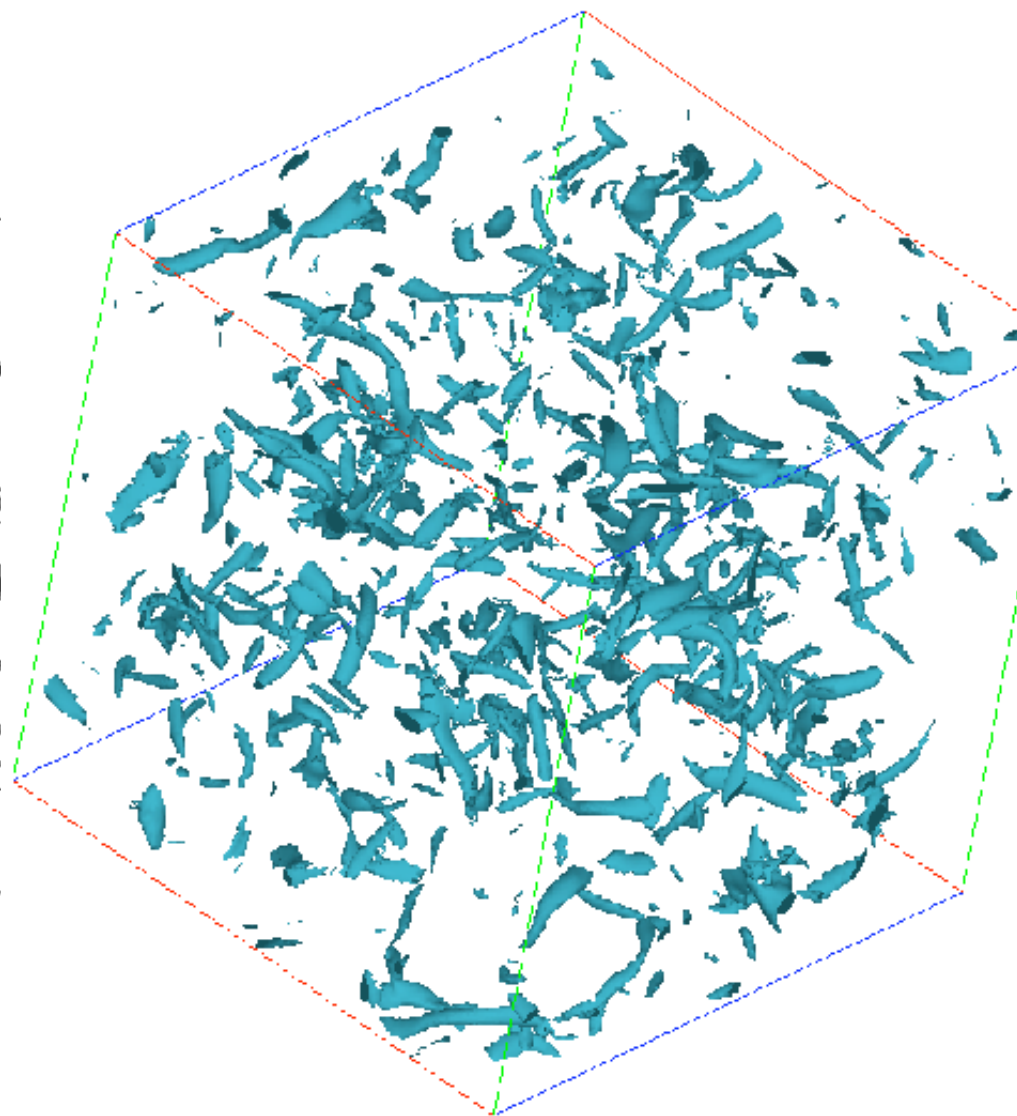


# Coherent structures at $t=5$

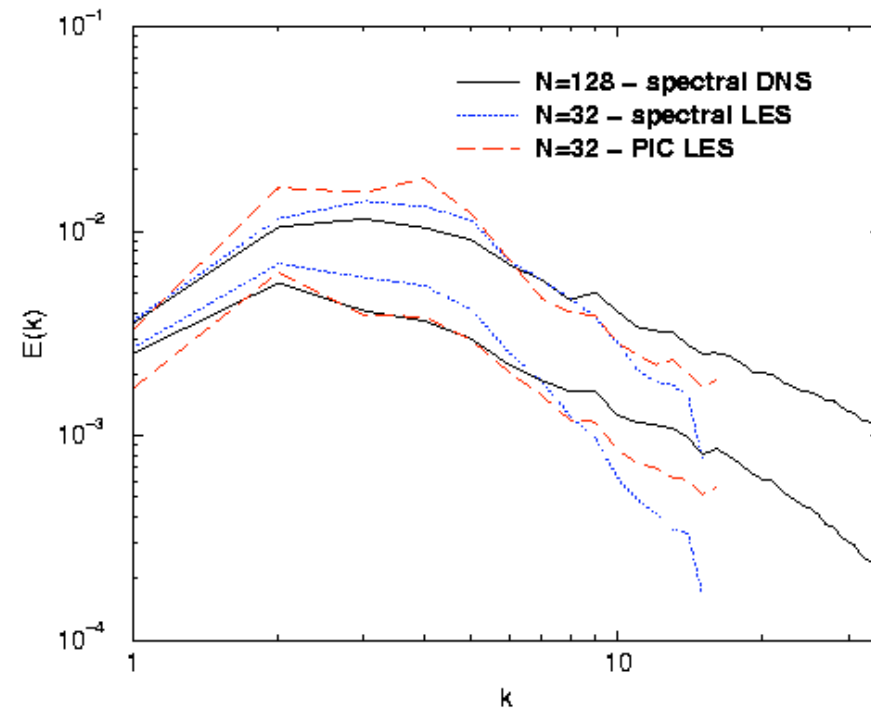
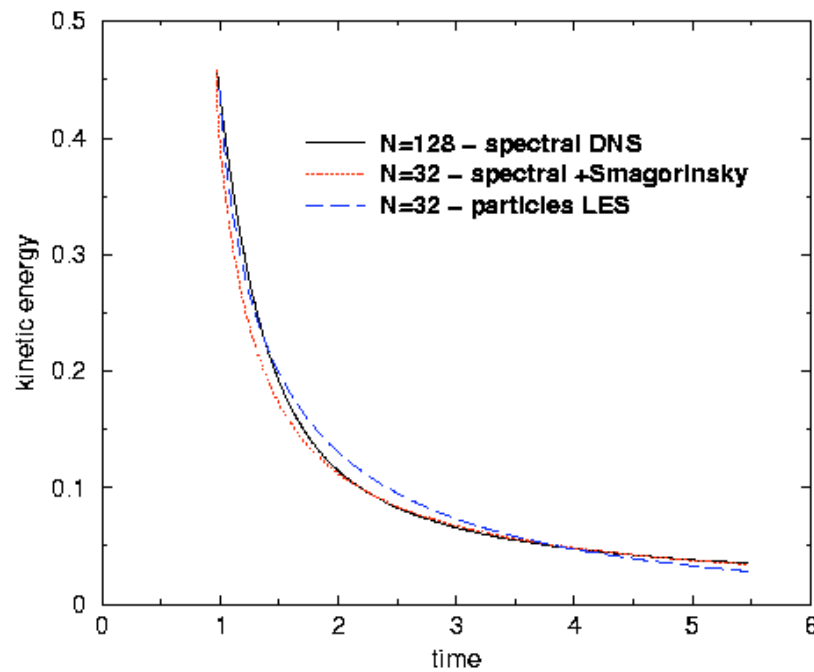
Spectral code



VIC code



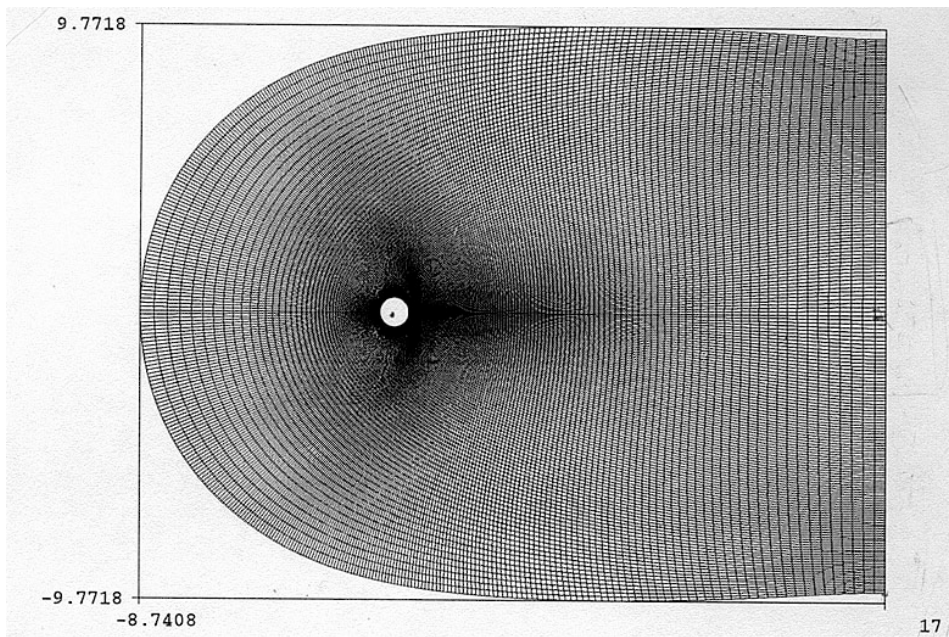
# Decaying turbulence in LES regime (Comte-Bellot experiment): comparison with spectral method with Smagorinsky model



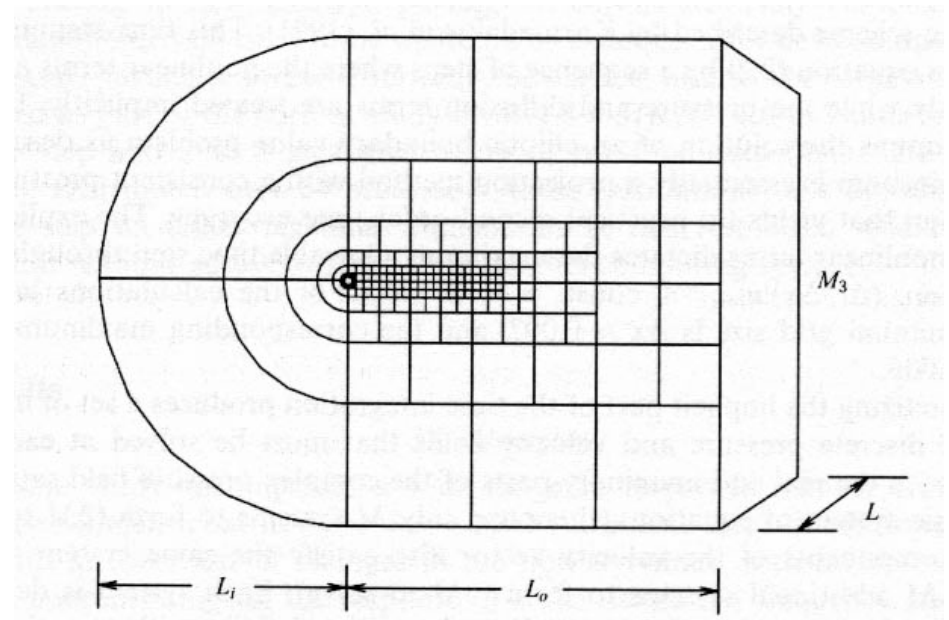
Boundary conditions in vortex methods  
in the spirit of immersed boundary  
techniques

# Why immersed boundaries for flows around bodies?

Even for simple geometries, body-fitted meshes are not straightforward: examples for turbulent cylinder wakes simulations



C-mesh (Mittal, Moin)



Spectral element (Henderson)

For more complex geometries, building the mesh can double the time of a full simulation

## Immersed boundaries for flows around solid boundaries: general idea

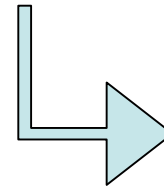
- Deal with boundaries as forcing terms in the dynamics, rather than using a body-fitted grid
- Budget accuracy vs flexibility ?
- Possible with finite-difference methods (Fadlun et al., JCP 2000) but particles have something special, in particular for moving interfaces.

Two levels of boundary conditions:

1. velocity from vorticity
2. vorticity b.c. in the Navier Stokes equation

$$(1) \quad \begin{cases} \nabla \times u = \omega, \nabla \cdot u = 0 & \Omega \\ u \cdot n = 0 & \partial\Omega \end{cases}$$

$$(2) \quad \begin{cases} \frac{\partial \omega}{\partial t} + \dots = 0 & \Omega \\ \partial \omega / \partial n = g & \partial\Omega \end{cases}$$



$g$  s.t.  $u \cdot \tau = 0$  on  $\partial\Omega$

# (1) Velocity calculation in a vortex code

$$\mathbf{u} = \nabla \times \Psi + \nabla \Phi$$

Helmholtz  
decomposition

where

$$\begin{aligned} -\Delta \Psi &= \omega \\ \nabla \cdot \Psi &= 0 \end{aligned}$$

$$\begin{aligned} -\Delta \Phi &= 0 \\ \frac{\partial \Phi}{\partial n} &= -(\nabla \times \Psi) \cdot \mathbf{n} \end{aligned}$$

Remark: decouples b.c. and divergence-free  
constraint on stream-function

## Velocity calculation (2D case)

$$\begin{aligned}\nabla \times u &= \omega \\ \nabla \cdot u &= 0 \\ u \cdot n &= 0\end{aligned}$$



Biot-Savart law:

$$u(x) = u_{\infty} + \int K(x - y)\omega(y)dy + \oint K(x - y)q(y)dy$$

Potential  $q$  determined by an  
integral equation on the boundary

Conclusion: no-through flow b.c. satisfied  
*exactly* even if particles in fluid do not match  
boundary discretization points



Bad news: Elliptic problem with boundary condition on immersed boundary not very well behaved (see fictitious domain methods)

**BUT**

$$\begin{aligned} \nabla \cdot u &= 0 \\ u \cdot \tau_{|\partial\Omega} &= 0 \end{aligned} \Rightarrow u \cdot n = O(h^2) \text{ at grid points near boundary}$$

Solution adopted: 1) tag grid-points near boundary

2) solve the linear system (GMRES type method):

$$\xi_M \rightarrow \Delta\Phi = \xi \rightarrow \left. \frac{\partial\Phi}{\partial n} \right|_M = 0$$

Where M are the grid cells which intersect the boundary

# No-slip boundary condition

Convection-diffusion time-splitting algorithm. At the end of the advection step the residual slip is canceled by an appropriate vorticity flux:

$$\begin{aligned}\frac{\partial \omega}{\partial t} - \nu \Delta \omega &= 0 \\ \nu \frac{\partial \omega}{\partial n} &= -\frac{1}{\Delta t} u \cdot \tau\end{aligned}$$



$$\omega(x, t) = \int \Gamma(x - y, \nu t) \omega_0(y) dy + \oint \Gamma(x - y, \nu t) p(y) dy$$

Potential  $p$  determined by an  
integral equation on the boundary

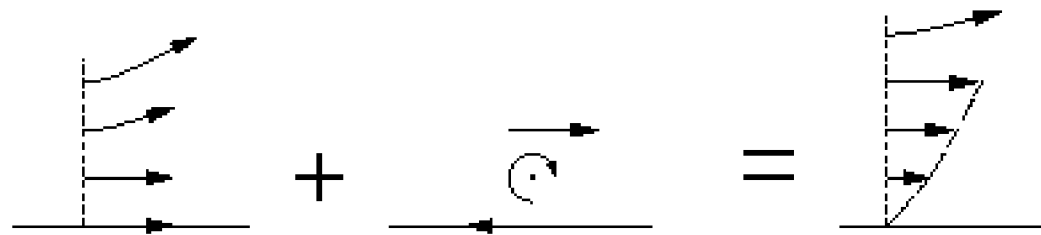
## (2) Vorticity boundary condition

Vorticity creation time-splitting algorithm

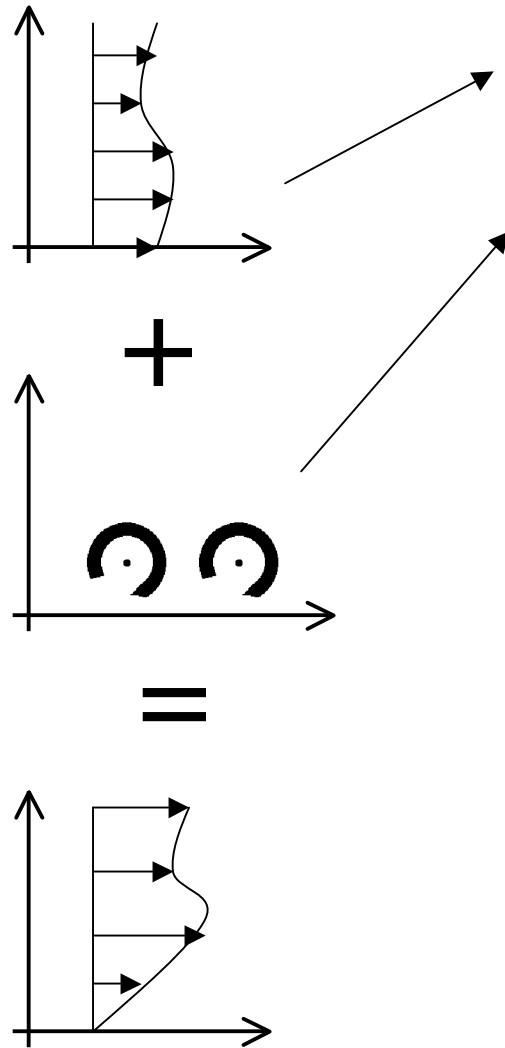
- 1) NS. equation
- 2) Spurious slip computation
- 3) Correction by proper vorticity flux

In 2 dimensions, step 3 reads

$$\begin{aligned} \frac{\partial \omega}{\partial t} - \nu \Delta \omega &= 0 & \Omega \\ \nu \frac{\partial \omega}{\partial n} &= -\frac{1}{\Delta t} u \cdot \tau & \partial \Omega \end{aligned}$$



In 3D, need boundary conditions for 3 vorticity components



After advection step computation of slip

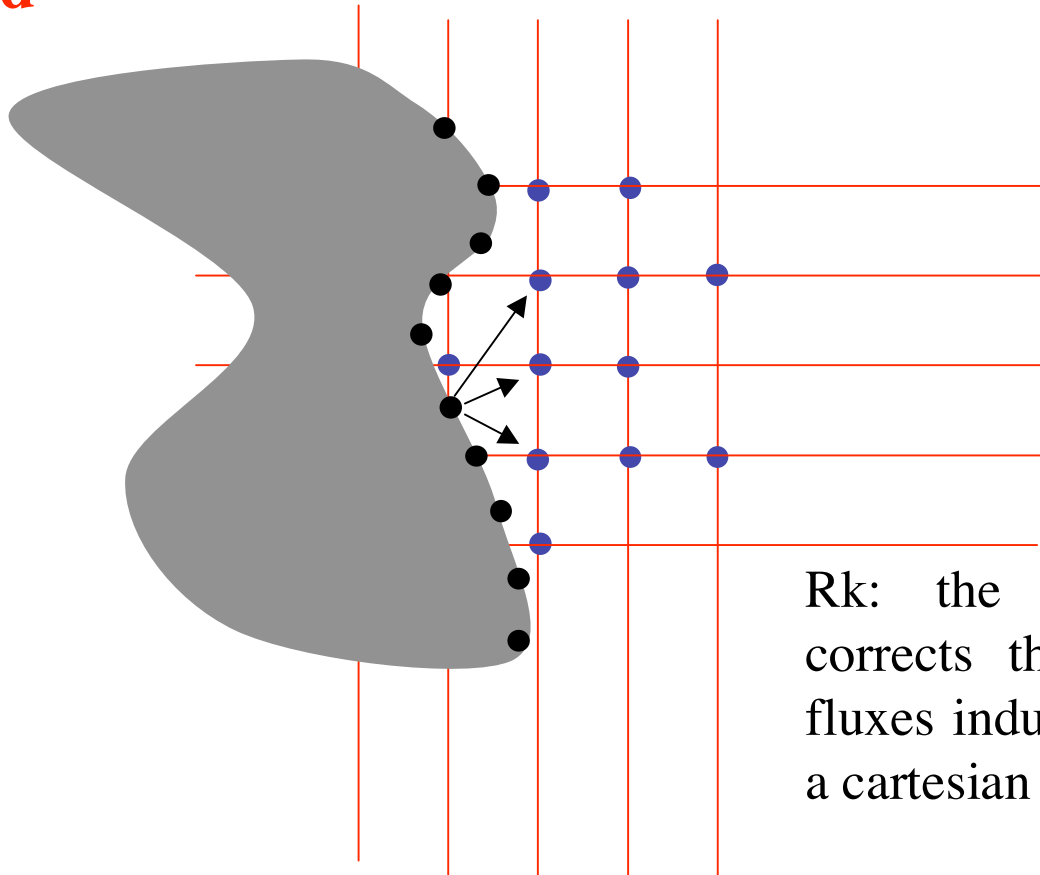
Vorticity flux onto flow particles

$$\left\{ \begin{array}{ll} \frac{\partial \omega}{\partial t} - \nu \Delta \omega = 0 & \text{sur } \Omega \times ]t_0, t_0 + \delta t[ \\ \omega(t_0) = 0 & \text{sur } \Omega \\ \nu \frac{\partial \omega_z}{\partial n} = - \frac{\partial u_\theta}{\partial t} & \text{sur } \partial \Omega \times ]t_0, t_0 + \delta t[ \\ \nu \left( \frac{\partial \omega_\theta}{\partial n} + \kappa \omega_\theta \right) = - \frac{\partial u_z}{\partial t} & \text{sur } \partial \Omega \times ]t_0, t_0 + \delta t[ \\ \omega_r = 0 & \text{sur } \partial \Omega \times ]t_0, t_0 + \delta t[ \end{array} \right.$$

In practice, the Neumann b.c. are implemented by integral formulas:

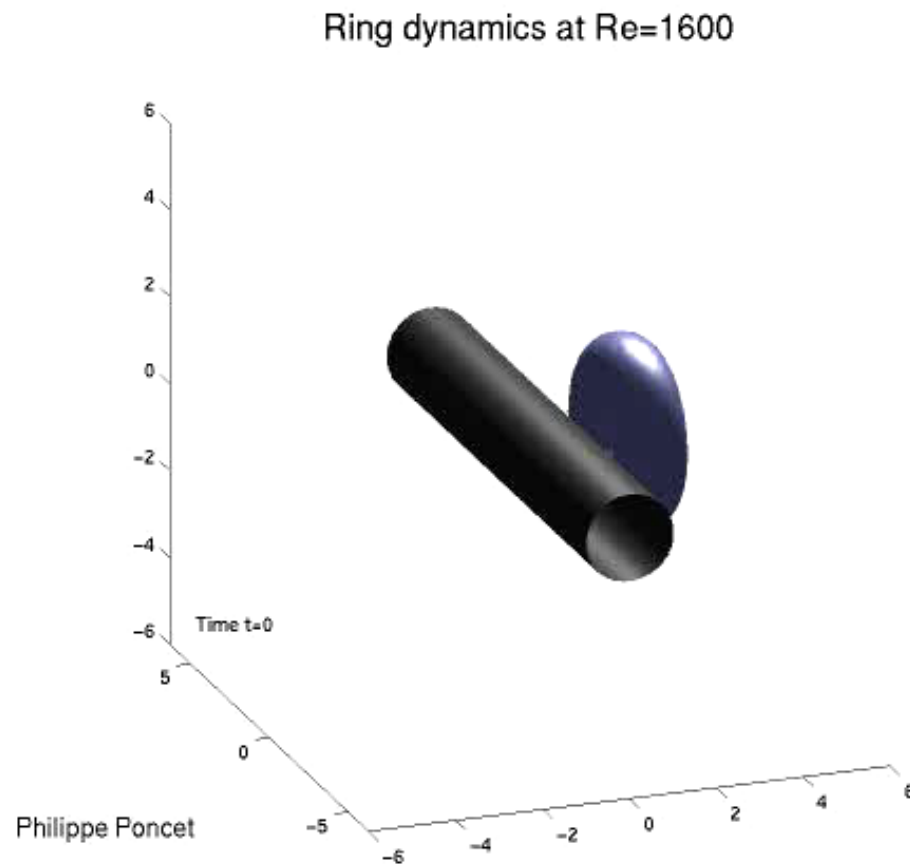
$$\omega' = \omega + \oint_{\partial\Omega} \Gamma(x - y, v\Delta t) g(y) dy$$

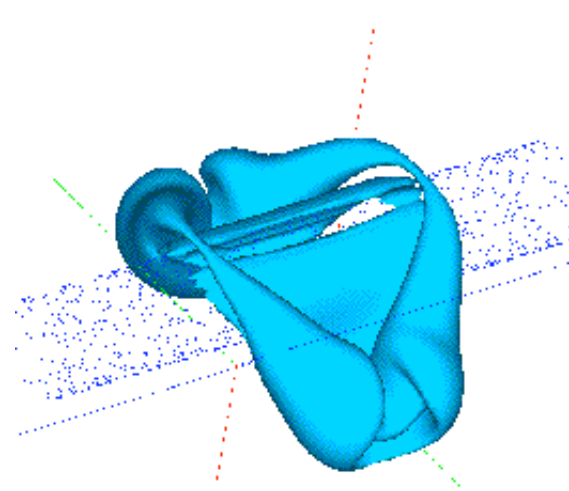
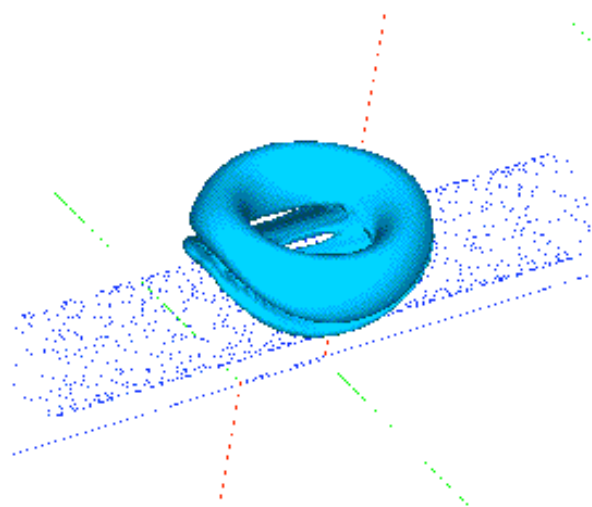
**This algorithm is, by itself, an immersed boundary method**



Rk: the flux formula also corrects the spurious vorticity fluxes induced by regridding on a cartesian grid

# Test case: Collision ring-cylinder on a cartesian mesh





### Ring/Wall interaction

