Visualizing Vorticity with BlobFlow

Lou Rossi

Department of Mathematical Sciences
University of Delaware

Network Delaware Day

The point vortex method: a venerable technique...

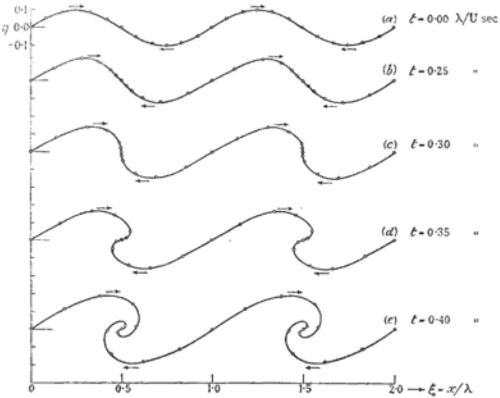


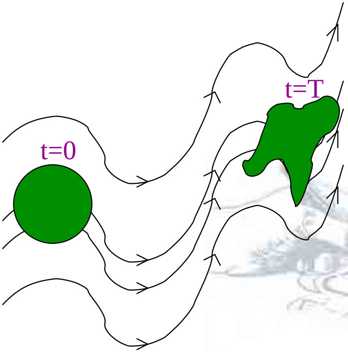
Fig. 4.

L. Rosenhead, “The point vortex approximation of a vortex sheet” *Proc. Roy. Soc.*, 134, 1932.

Particle methods

Computationally, if we know the initial distribution of ρ or ω, we can use Φ_t to follow it forward in time.

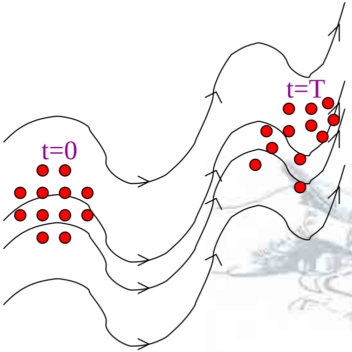
$$\frac{d\vec{x}}{dt} = \vec{u}(\vec{x}, t)$$
$$\vec{x}(0) = \vec{y}$$



Particle methods

Represent ρ as a collection of N particles, and follow them forward in time. Thus, we reduce a PDE to a $d \times N$ dimensional system of ODEs.

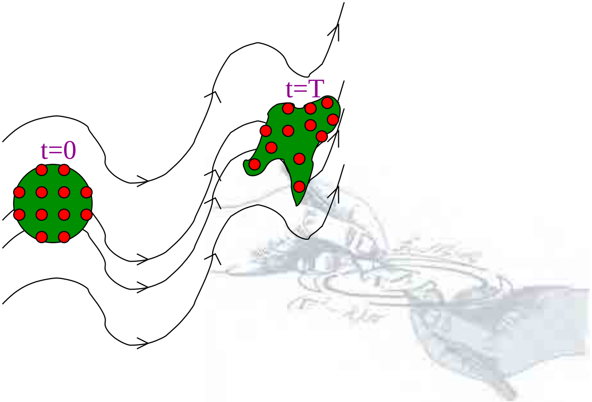
$$\frac{d\vec{x}}{dt} = \vec{u}(\vec{x}, t)$$
$$\vec{x}(0) = \vec{y}$$



Particle methods

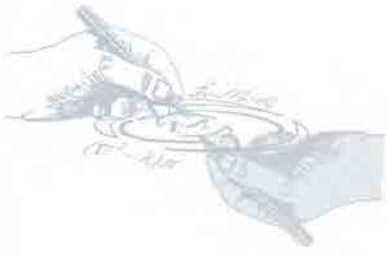
Represent ρ as a collection of N particles, and follow them forward in time. Thus, we reduce a PDE to a $d \times N$ dimensional system of ODEs.

$$\frac{d\vec{x}}{dt} = \vec{u}(\vec{x}, t)$$
$$\vec{x}(0) = \vec{y}$$



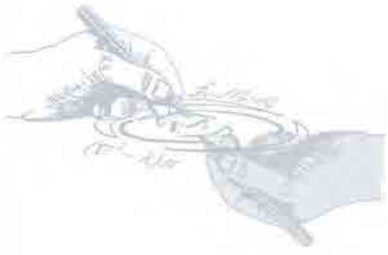
Why does one do this?

- Natural adaptivity.
  Resources are dedicated where ρ or ω are and nowhere else. This yields a strong *general* method for scientific and engineering applications.
- Stability.
- Geometry independent.
- Appeal to modelers.



Why does one do this?

- Natural adaptivity.
- Stability.
  Assuming that the flow field is "well behaved", the initial value problem for the system of ODEs is well-posed and stability is guaranteed.
- Geometry independent.
- Appeal to modelers.



Why does one do this?

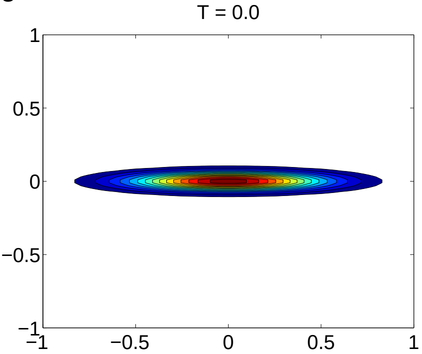
- Natural adaptivity.
- Stability.
- Geometry independent.
  Local solutions to PDEs mean that boundary conditions are applied locally, not globally.
- Appeal to modelers.

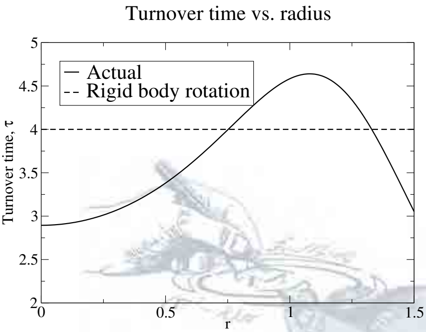
Why does one do this?

- Natural adaptivity.
- Stability.
- Geometry independent.
- Appeal to modelers.
  Such schemes bridge the gap between computations and modeling because each computational element can represent a localized model for the full system.

Example: Mixing in a differentially rotating flow.

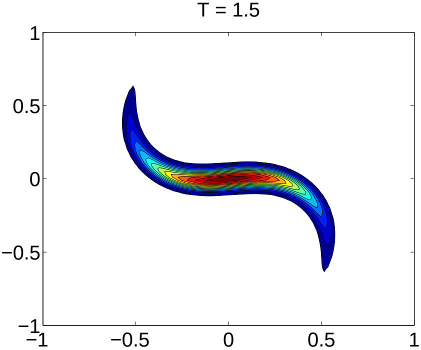
Advantage: Adaptive resolution of fine scales and high gradients. Particle methods can capture diffusion exactly.

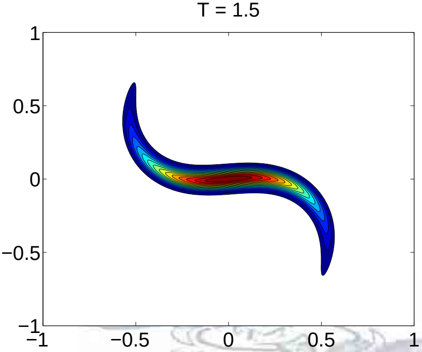




Example: Mixing in a differentially rotating flow.

Advantage: Adaptive resolution of fine scales and high gradients. Particle methods can capture diffusion exactly.

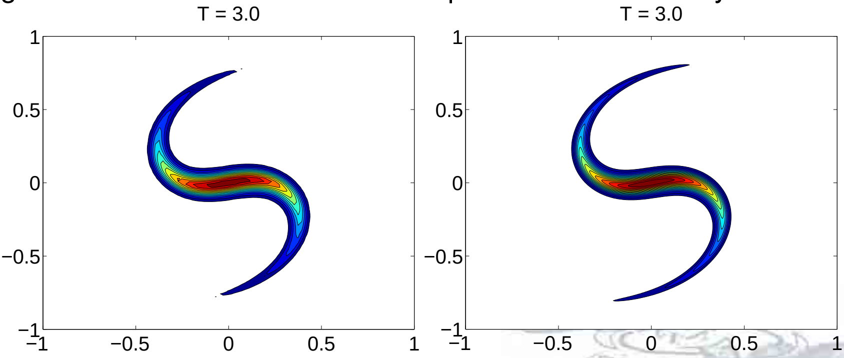




$Pe = 10^4$ $\qquad$ $Pe = \infty$

Example: Mixing in a differentially rotating flow.

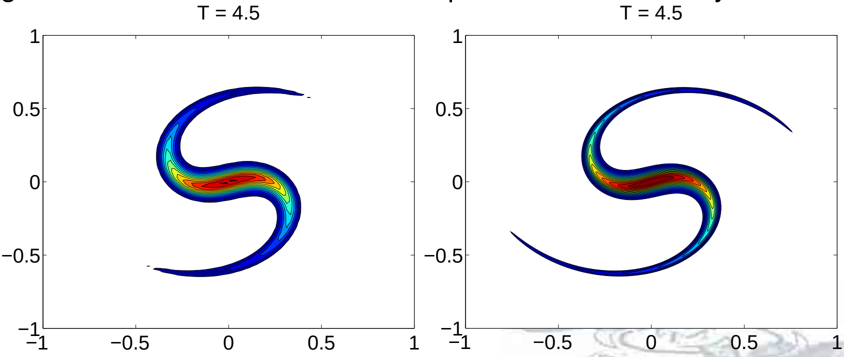
Advantage: Adaptive resolution of fine scales and high gradients. Particle methods can capture diffusion exactly.



$\mathrm{Pe} = 10^4$ $\qquad$ $\mathrm{Pe} = \infty$

Example: Mixing in a differentially rotating flow.

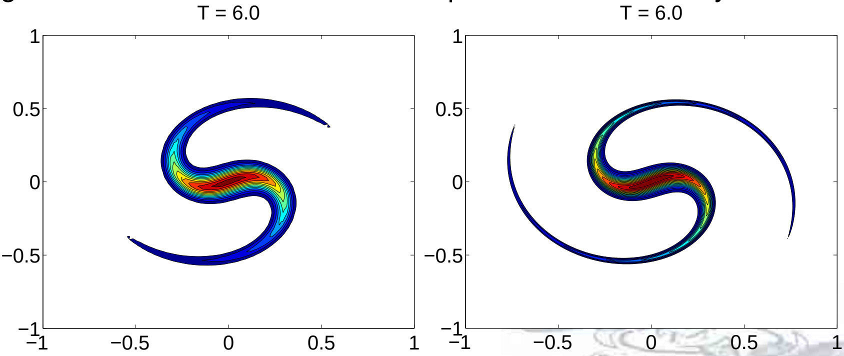
Advantage: Adaptive resolution of fine scales and high gradients. Particle methods can capture diffusion exactly.



$\mathrm{Pe} = 10^4$ $\quad$ $\mathrm{Pe} = \infty$

Example: Mixing in a differentially rotating flow.

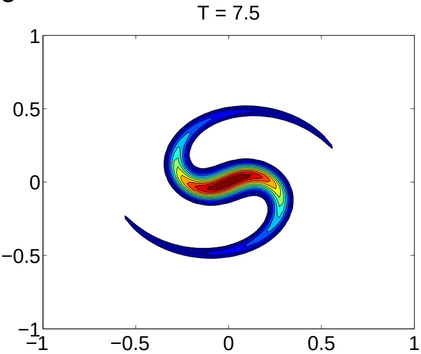
Advantage: Adaptive resolution of fine scales and high gradients. Particle methods can capture diffusion exactly.



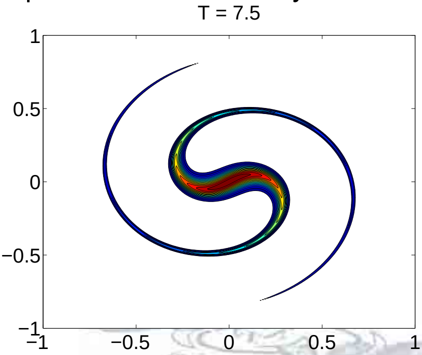
$\mathrm{Pe} = 10^4$ $\qquad$ $\mathrm{Pe} = \infty$

Example: Mixing in a differentially rotating flow.

Advantage: Adaptive resolution of fine scales and high gradients. Particle methods can capture diffusion exactly.



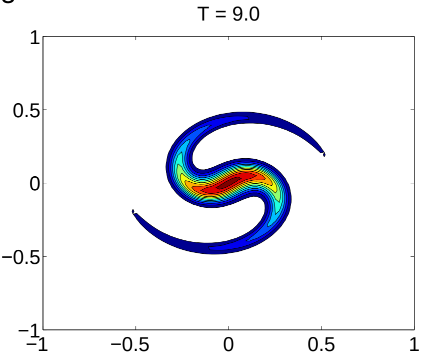
Pe = 10^4

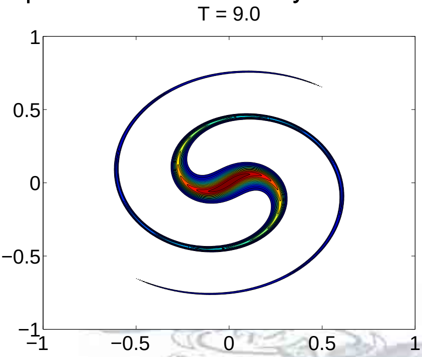


Pe = ∞

Example: Mixing in a differentially rotating flow.

Advantage: Adaptive resolution of fine scales and high gradients. Particle methods can capture diffusion exactly.



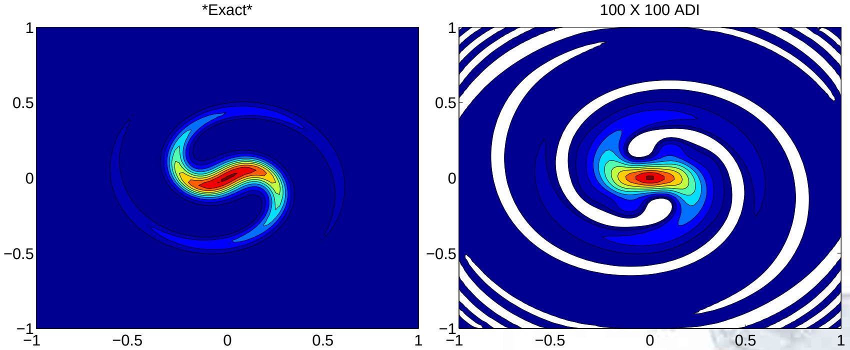


$\mathrm{Pe} = 10^4$

$\mathrm{Pe} = \infty$

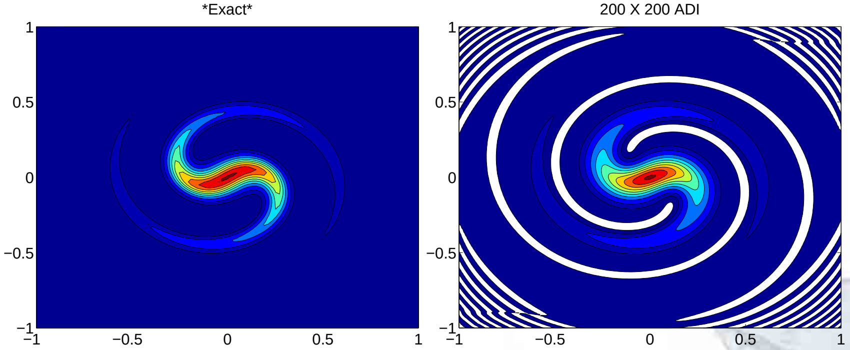
A tough problem

A comparison with a meshed method.



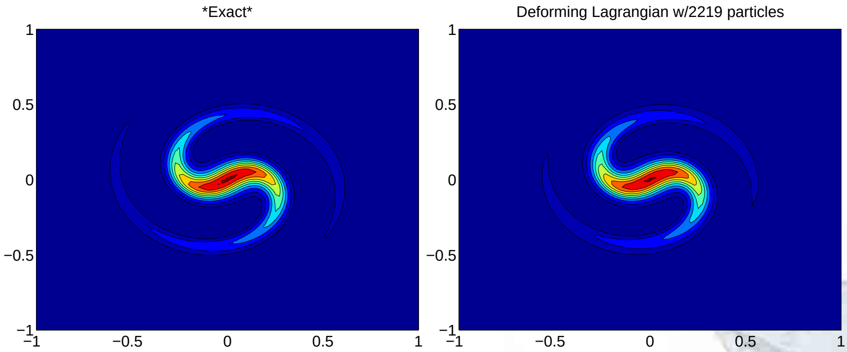
A tough problem

A comparison with a meshed method.



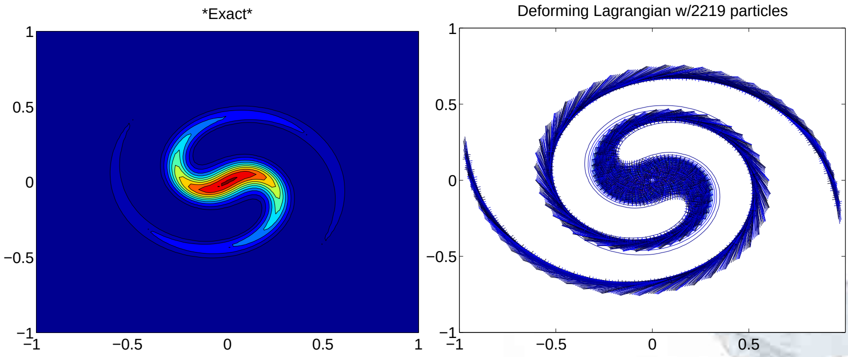
A tough problem

A comparison with a meshed method.



A tough problem

A comparison with a meshed method.

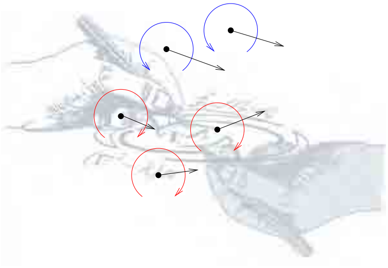


Vortex methods

$$\frac{D\omega}{Dt} = \frac{1}{\mathrm{Re}}\nabla^2\omega$$

Express ω (or $\vec{\omega}$ in 3D) as a linear combination of moving basis functions (blobs):

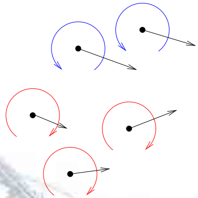
$$\begin{aligned} \widehat{\omega}(\vec{x}) &= \sum_{i=1}^{N} \gamma_i \phi(\vec{x} - \vec{x}_i) \\ \frac{d\vec{x}_i}{dt} &= \vec{u}(\vec{x}_i) \end{aligned}$$



The Biot-Savart integral

$$\widehat{\omega}(\vec{x}) = \sum_{i=1}^{N} \gamma_i \phi(\vec{x} - \vec{x}_i)$$

$$\frac{d\vec{x}_i}{dt} = \vec{u}(\vec{x}_i)$$

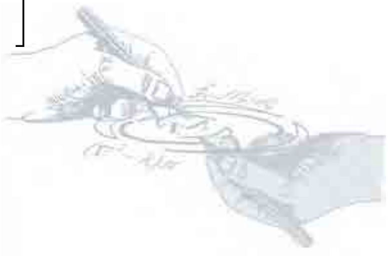


Since ω ($\widehat{\omega}$) is our primitive variable, we must determine $\vec{u}$ from a Biot-Savart integral via a streamfunction ψ.

The Biot-Savart integral

Since ω ($\widehat{\omega}$) is our primitive variable, we must determine $\vec{u}$ from a Biot-Savart integral via a streamfunction ψ.

$$\begin{aligned} \vec{u} &= \begin{bmatrix} -\frac{\partial \psi}{\partial y} \\ \frac{\partial \psi}{\partial x} \end{bmatrix} \\ \nabla^2 \psi &= -\widehat{\omega} \end{aligned}$$



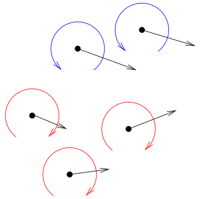
The Biot-Savart integral

Since ω ($\widehat{\omega}$) is our primitive variable, we must determine $\vec{u}$ from a Biot-Savart integral via a streamfunction ψ.

$$\begin{aligned}
\vec{u} &= \begin{bmatrix} -\frac{\partial\psi}{\partial y} \\ \frac{\partial\psi}{\partial x} \end{bmatrix} \\
\nabla^2\psi &= -\widehat{\omega} \\
\psi &= -\frac{1}{4\pi}\iint_{-\infty}^{\infty} \log(|\vec{x}-\vec{s}|^2)\,\widehat{\omega}(\vec{s})\,d\vec{s}
\end{aligned}$$

An example with Gaussian blobs

$$
\begin{aligned}
\phi(\vec{x}) &= \frac{1}{4\pi\sigma^2}\exp\left(-\frac{|\vec{x}|^2}{4\sigma^2}\right)\\
\vec{u}(\vec{x}) &= \sum_{i=1}^{N}\frac{\gamma_i}{2\pi}\frac{\vec{x}-\vec{x}_i}{|\vec{x}-\vec{x}_i|^2}\left[1-\exp\left(-\frac{|\vec{x}-\vec{x}_i|^2}{4\sigma^2}\right)\right]
\end{aligned}
$$



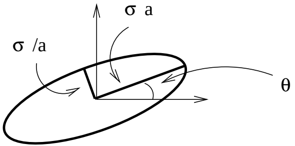
Note: $O(N^2)$ complexity.

Corrected core spreading vortex method (CCVSM)

$$
\begin{aligned}
\phi(\vec{x}) &= \frac{1}{4\pi\sigma_i^2}\exp\left(-\frac{|\vec{x}|^2}{4\sigma_i^2}\right), \\
\vec{u}(\vec{x}) &= \sum_{i=1}^{N} \frac{\gamma_i}{2\pi}\frac{\vec{x}-\vec{x}_i}{|\vec{x}-\vec{x}_i|^2}\left[1-\exp\left(-\frac{|\vec{x}-\vec{x}_i|^2}{4\sigma_i^2}\right)\right], \\
\frac{d\sigma_i}{dt} &= \frac{1}{\mathrm{Re}}.
\end{aligned}
$$

Some comments: Refinement and merging (or remeshing) can maintain spatial accuracy for all time. For high Reynolds number flows and limited resolutions, core size growth may not be important anyway. $\sigma_0 \gg \frac{T}{\mathrm{Re}}$.

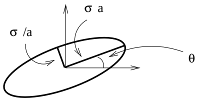
The dynamics of elliptical Gaussians



Find the evolution equations for $\vec{x}_i$, θ_i, σ_i and a_i so that $\phi \equiv \phi(\vec{x}; \vec{x}_i, \sigma_i, a_i, \theta_i)$ satisfies

$$\partial_t \phi + [\vec{u}(\vec{x}_i) + D\vec{u}(\vec{x}_i)(\vec{x} - \vec{x}_i)] \cdot \nabla \phi - \frac{1}{\mathrm{Re}} \nabla^2 \phi = 0.$$

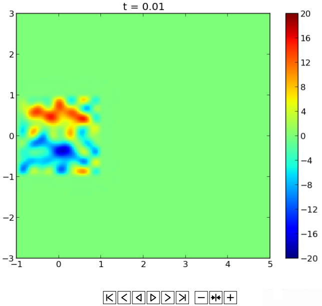
The dynamics of elliptical Gaussians



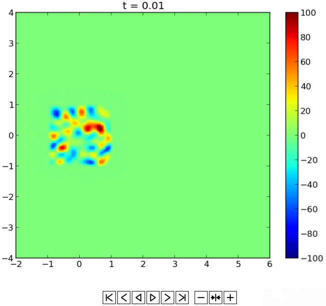
For simplicity $\theta_i = 0$...

$$
\begin{aligned}
\frac{d}{dt}\vec{x}_i &= \begin{bmatrix} u_i \\ v_i \end{bmatrix} \\
\frac{d}{dt}(\sigma_i^2) &= \frac{1}{2\mathrm{Re}}(a_i^2 + a_i^{-2}) \\
\frac{d}{dt}(a_i^2) &= 2d_{11,i}a_i^2 + \frac{1}{2\mathrm{Re}\sigma_i^2}(1 - a_i^4) \\
\frac{d}{dt}\theta_i &= \frac{d_{21,i} - d_{12,i}}{2} - \frac{d_{21,i} + d_{12,i}}{2}\frac{(a_i^{-2} + a_i^2)}{(a_i^{-2} - a_i^2)}
\end{aligned}
$$

Exploring complex flows is time consuming



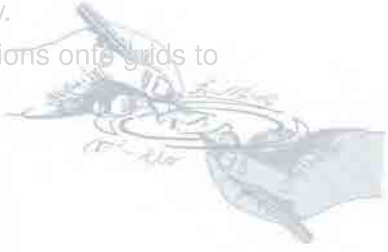
Exploring complex flows is time consuming



Removing the pain from BlobFlow.

Just a thought.
A scientist should not need to ...

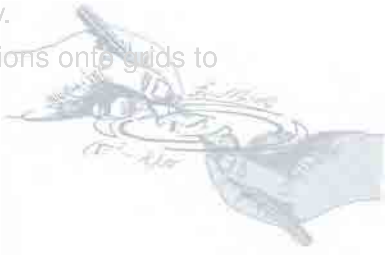
- ... understand my algorithm to explore vorticity.
- ... compile my program to explore vorticity.
- ... own a cluster to explore vorticity.
- ... know how to project basis functions onto grids to explore vorticity.



Removing the pain from BlobFlow.

Just a thought.
A scientist should not need to ...

- ... understand my algorithm to explore vorticity.
- ... compile my program to explore vorticity.
- ... own a cluster to explore vorticity.
- ... know how to project basis functions onto grids to explore vorticity.



Removing the pain from BlobFlow.

Just a thought.
A scientist should not need to ...

- ... understand my algorithm to explore vorticity.
- ... compile my program to explore vorticity.
- ... own a cluster to explore vorticity.
- ... know how to project basis functions onto grids to explore vorticity.

Removing the pain from BlobFlow.

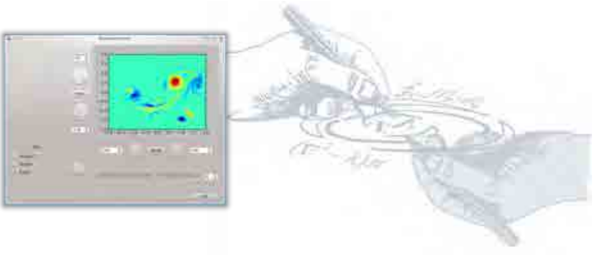
Just a thought.
A scientist should not need to ...

- ... understand my algorithm to explore vorticity.
- ... compile my program to explore vorticity.
- ... own a cluster to explore vorticity.
- ... know how to project basis functions onto grids to explore vorticity.

Removing the pain from BlobFlow.

Goals...

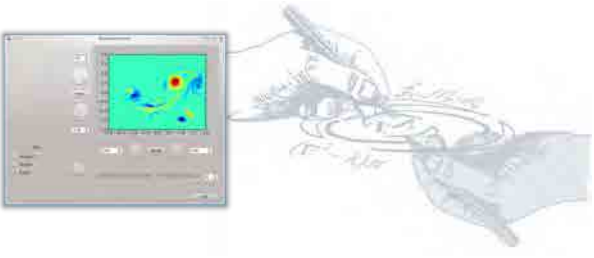
- Useful to any scientist.
- Visual.
- Networked.
- Free, open and cross-platform (Python + Qt4).
- Kinda' like Minecraft.



Removing the pain from BlobFlow.

Goals...

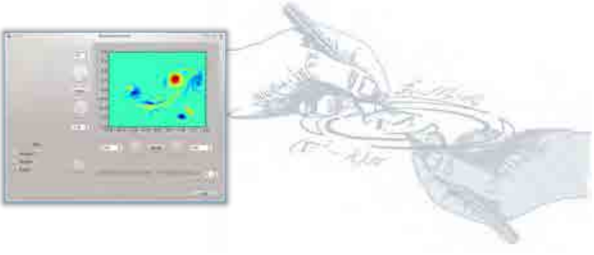
- Useful to any scientist.
- Visual.
- Networked.
- Free, open and cross-platform (Python + Qt4).
- Kinda' like Minecraft.



Removing the pain from BlobFlow.

Goals...

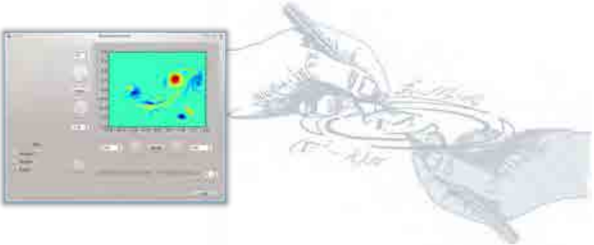
- Useful to any scientist.
- Visual.
- Networked.
- Free, open and cross-platform (Python + Qt4).
- Kinda' like Minecraft.



Removing the pain from BlobFlow.

Goals...

- Useful to any scientist.
- Visual.
- Networked.
- Free, open and cross-platform (Python + Qt4).
- Kinda' like Minecraft.



Removing the pain from BlobFlow.

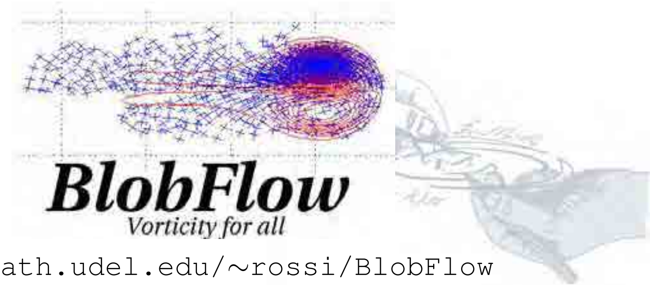
Goals...

- Useful to any scientist.
- Visual.
- Networked.
- Free, open and cross-platform (Python + Qt4).
- Kinda' like Minecraft.

Removing the pain from BlobFlow.

Conclusions

- The high-order vortex method is an accurate, general solver that is competitive with more mainstream methods.
- The next step is to make vorticity exploration more open to scientists.
- Please visit the BlobFlow site and play with vorticity.



www.math.udel.edu/~rossi/BlobFlow