Visualising 2D flows

J. J. Green
LESIA, Observatoire de Paris

Gerris users' meeting, 4th–5th July 2011

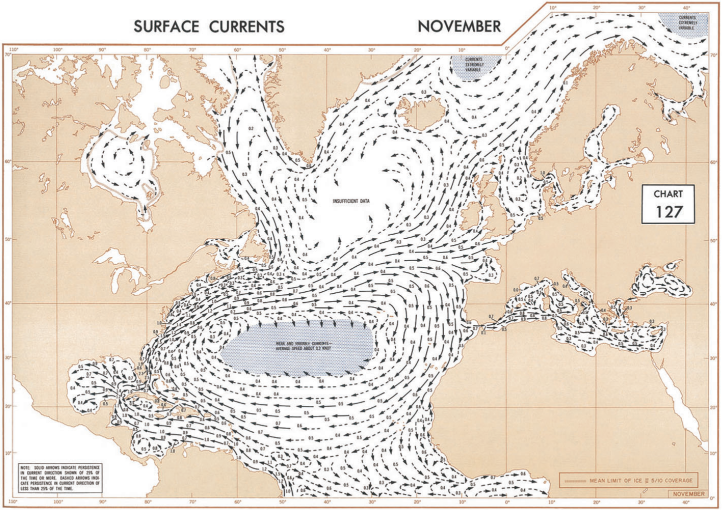
Visualisation of 2D fields with arrows

Most software plots 2D vector fields with small straight arrows on a regular grid.

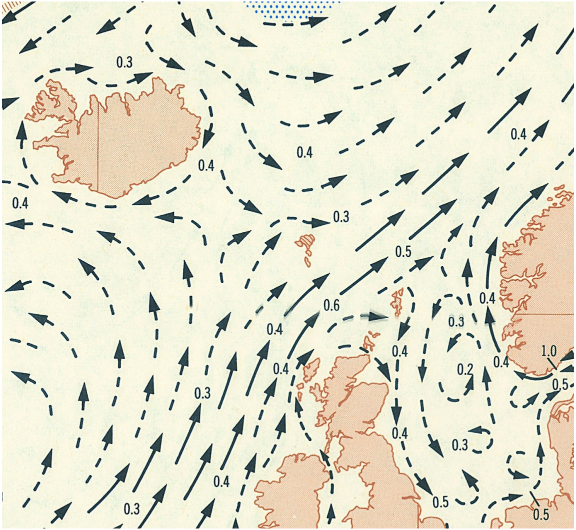
- Arrows on a regular grid must be small if they are not to collide
- Straight arrows need to be small to accurately describe curves
- Small arrows are too small to be perceived individually, in the limit they become a *texture*, so why use arrows?

Is there another way?

Currents in the North Atlantic



Currents in the North Atlantic (detail)



Outline idea

- *Large* arrows which can be perceived individually
- *Curved* to display the curvature of the field
- *Adaptively placed* to avoid collisions.

Circular arrows I

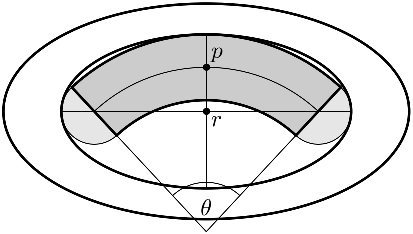
Keep it simple: use a section of an annulus as the arrow shaft. We need to know the (vector) value $\mathbf{v}(\mathbf{p})$ and radius of curvature, given by

$$\kappa = \left| \frac{\partial(u_1, u_2)}{\partial(p_1, p_2)} \mathbf{u} \right|$$

for the direction field $\mathbf{u} = \mathbf{v}/|v|$ at each point.

Circular arrows II

It is easy to define an ellipse which contains such a shaft.



So we have an ellipse at each point in a region (which is a positive definite tensor field).
The plotting problem is to pack these ellipses into the region.

The distance between ellipses

We need an idea of a *distance* between ellipses A and B. Perram and Wertheim[1] have defined a *contact function*

$$f(A,B)\begin{cases} <1 & \text{if } A \text{ and } B \text{ intersect} \\ =1 & \text{if } A \text{ and } B \text{ are tangent} \\ >1 & \text{if } A \text{ and } B \text{ are disjoint.} \end{cases}$$

The function is

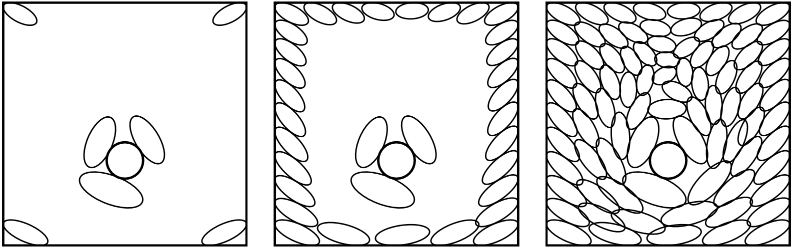
$$f(A,B) = \max_{\lambda\in[0,1]} f_{AB}(\lambda)$$

for a rational function f_{AB} which is convex on $[0,1]$. Thus this maximum can be found rapidly by a Newton iteration.

[1] Statistical mechanics of hard ellipsoids. I. Overlap algorithm and contact function. *J. Comp. Phys.* 58 (1985), 409–416

Dimension climb

Shimada, Yamada, Itoh[2] suggest solving the ellipse packing problem by *dimension climbing*.



Place ellipses at the corners (dimension 0), use these to constrain the packing along the edges (dimension 1) and likewise use these to constrain the packing in the region (dimension 2).

[2] Anisotropic triangulation of parametric surfaces via close packing of ellipsoids. *Int. J. Comp. Geom. Appl. 10*, 4 (2000), 400–424.

Dimension 2 packing

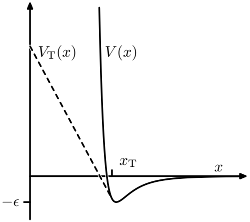
Motivated by the regular packings found in nature at the molecular scale, Shimada *et al.* propose a simulation of a van der Wall dynamics based on an inter-ellipse potential similar to that of Lennard-Jones,

$$V(x) = 4e\left((\sigma/x)^{12} - (\sigma/x)^{6}\right)$$

so $V(\sigma) = 0$ and $-e$ is the minimum of V. Here the distance x between ellipses taken to be their Perram-Wertheim distance and a force proportional to $-V'(x)$ is taken between each ellipse pair.
In fact we truncate the potential when we are a few ellipse-widths apart, so only nearby ellipse-pair forces (and so distances) need be calculated.

How many ellipses to pack?

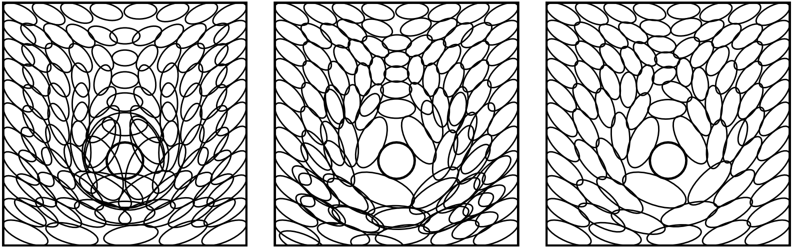
The remaining problem is knowing how many ellipses to use in the simulation. We get a strategy from the following curious fact (discovered by accident).
If the simulation is overpacked and the potential is replaced by one which is linear as we approach zero



then the surplus ellipses are superposed on a reasonably packed subset.

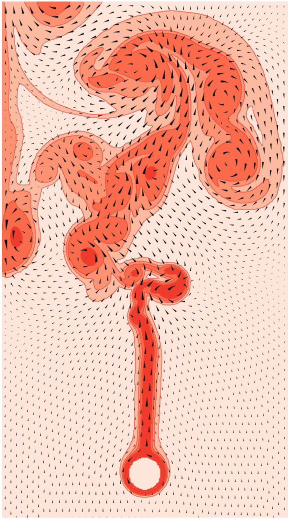
Superposition

Thus a packing strategy is to run the overpacked system with the linearised potential until superposition is stable, identify and remove superposed ellipses, “delinearise” the potential, then continue the simulation to stability.

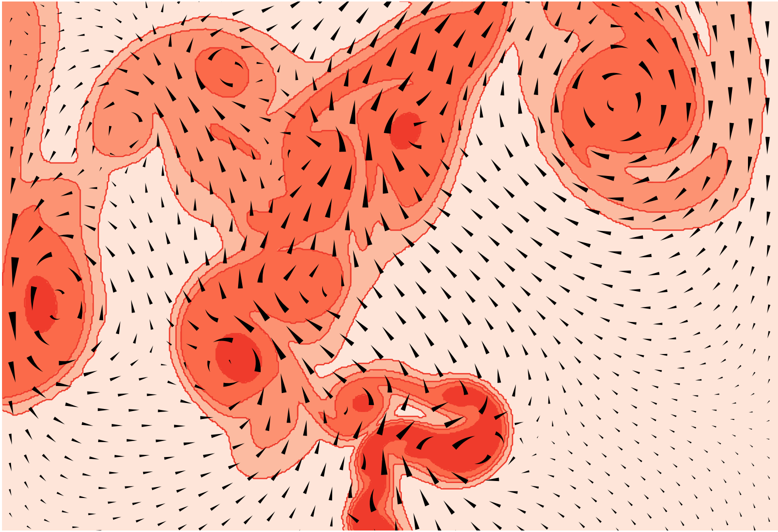


Easiest to see in an animation

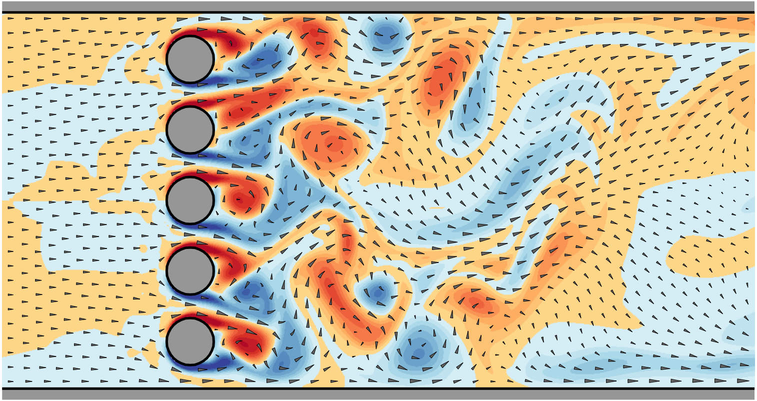
Gerris examples: Boussinesq flow



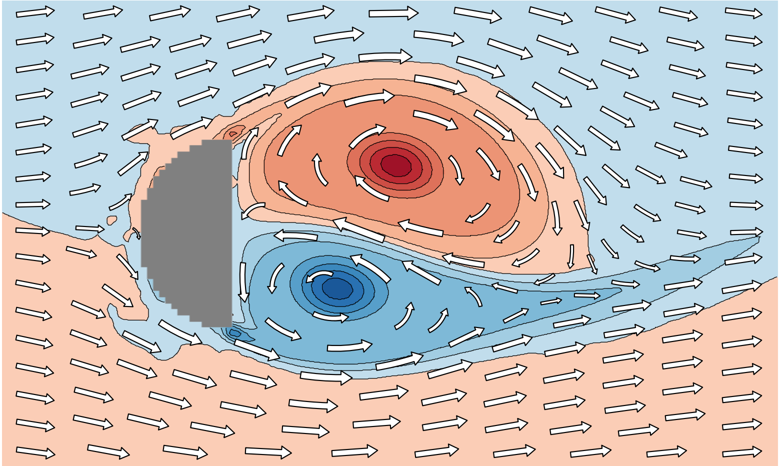
Gerris examples: Boussinesq flow



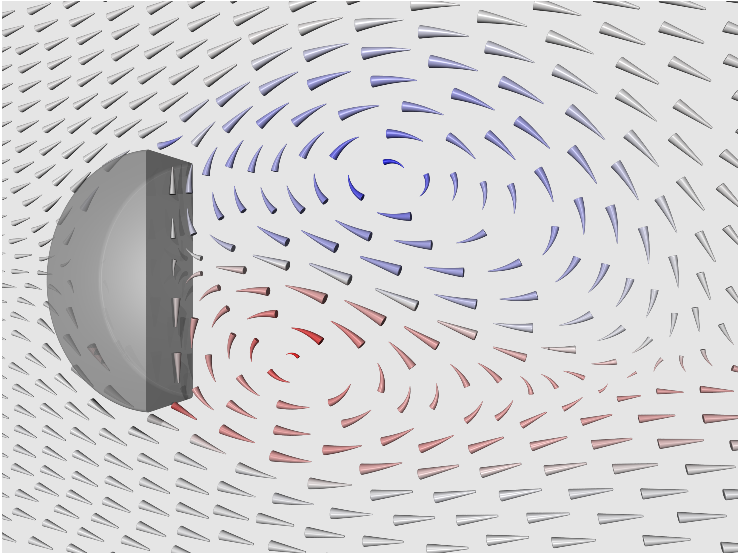
Gerris examples: Flow through a coarse grating



Gerris examples: vortex shedding



Gerris examples: vortex shedding (POV-Ray)



The software: vfplot

Downsides:

- Command-line only, no GUI
- Not the most user friendly, takes some time to learn
- Problems with complex boundaries.

Upsides:

- Command-line only, no GUI
- Unix source free to download an modify (GPLv3)
- Reads Gerris simulation files
- Friendly maintainer