# Exercises: session 1

1) **Accessing subarrays:** make a square subwindow clearer (closer to white) in an image that you have; animation: move the window arround your image.

2) **Memory management:** increasing the size of a 3D array and recording the times it takes. Plot this time as a function of the array size.

3) **Shifting of subarrays:** compute the derivative of a function using shifting. Compare the derivative with the exact derivative, show how the error tends to zero as the number of grid points becomes large.

4) **Auxiliary arrays:** build the differentiation matrix which compute the derivative of a function stored in an array by matrix multiplication; using simple finite differences.

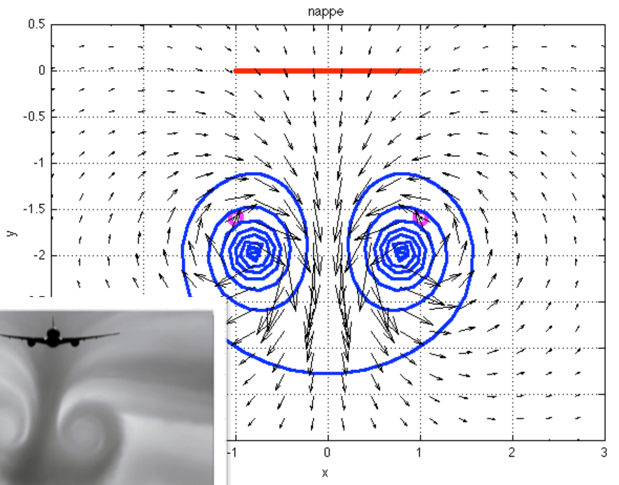
5) **Binary indexing:** build a random gaussian array and count the number of elements larger than 0 and show that when the array becomes large, this number is half the number of elements (with a plot).

6) **Animation:** Make the animation of a function in 2D using drawnow.

# Matlab in Stokholm 2011
## Second session. Tuesday

Vectorization: making many operations with few commands, some graphs, some animations.

Jérôme Hoepffner,
université Pierre et Marie Curie, Paris.
www.ida.upmc.fr/~hoepffner

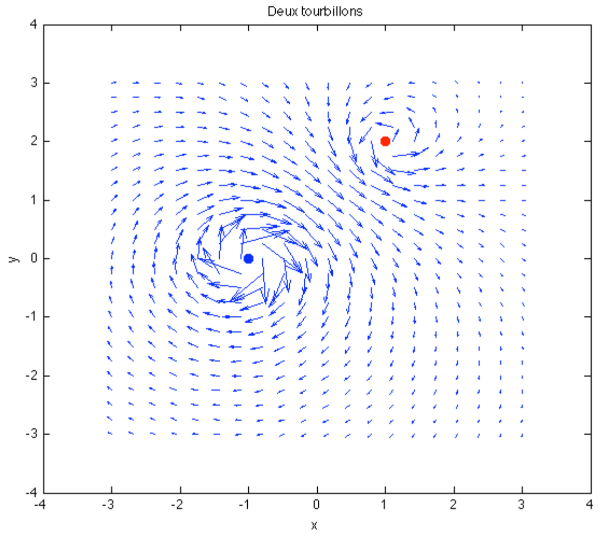


For this exercices, we are interested at the modelisation of inviscid fluid flows by the method of the velocity potential. We will consider the time evolution of a family of vortices. Each vortex is point-like (vorticity infinitely localized at $x_0$, $y_0$) and induces in its surrounding a velocity field with the formula given below. Delta is a small regularisation parameter with value 0.05. The velocity field induced by several vortices is the sum of the velocity fields induced by each vortex, and each vortex moves according to the velocity at its location induced by all the other vortices. The intensity (and sign) of each vortex is described by its circulation Gamma.

Particular cases: two counter-rotating vortices induce each other a translation trajectory, two corotating vortices induce each other a circular trajectory. Three vortices have a pseudo-periodic motion. Many vortices can together have a chaotic behaviour.

Today's operations can be easily coded with nested loops. For each step, code first with as many loops as you like, then use the techniques described in the lecture to skip loops, as many as you can. Each time, try to identify the category of vectorisation you are using to rationalize this process. Use «tic/toc» to compare the computation time of the different formulations.

$$u(x,y)=\Gamma\frac{-(y-y_0)}{2\pi[(x-x_0)^2+(y-y_0)^2+\delta]},\quad v(x,y)=\Gamma\frac{(x-x_0)}{2\pi[(x-x_0)^2+(y-y_0)^2+\delta]},$$



1)
Chose the position and intensity of two vortices, and draw the induced velocity field with the function «quiver» on a cartesian grid. Draw as well the position of the two vortices.
2)
Build a function «tourbi» which for each vortex of a family of n vortices, computes the velocity induced by the other vortices.
3)
Chose the position and intensity for three vortices and simulate their evolution in time, with a very simple marching algorithm: for each vortex and each time instant, the new position is equal to the old position plus the time step times the velocity vector you get from your function «tourbi».
4)
Draw the evolution in time of the velocity field with quiver on a fixed cartesian grid, and of the vortex positions. Make your time marching loop an animation by using the function «drawnow» at each iteration.
5)
Consider a line of many vortices as a model for the vortex sheet in the wake of an airplane, the distribution of the vortex intensity is given by the piece of code below. This corresponds to a continuous distribution of circulation 4x^3. Draw the evolution in time of this vorticity sheet and associated velocity field. Show that this is equivalent of the roll-up of the complete vorticity of two large wing-tip vortices (two counter rotating vortices inducing each other a downward translating trajectory.)

```
n=20;
xloc=linspace(-1,1,n);
yloc=0*xloc;
gamma=8*xloc.^3/n;
```

# Matlab in Stokholm 2011
## Third session. Wednesday

Jérôme Hoepffner,
université Pierre et Marie Curie, Paris.
www.ida.upmc.fr/~hoepffner

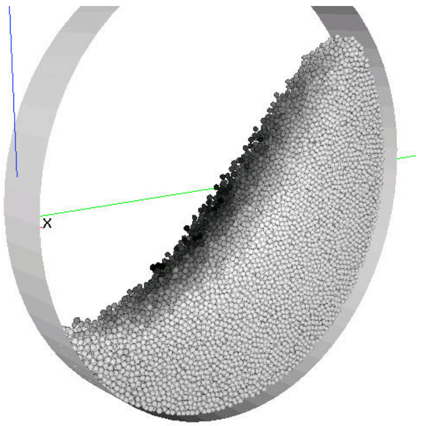
Graphs and animations. The flow of sand.

Underwater

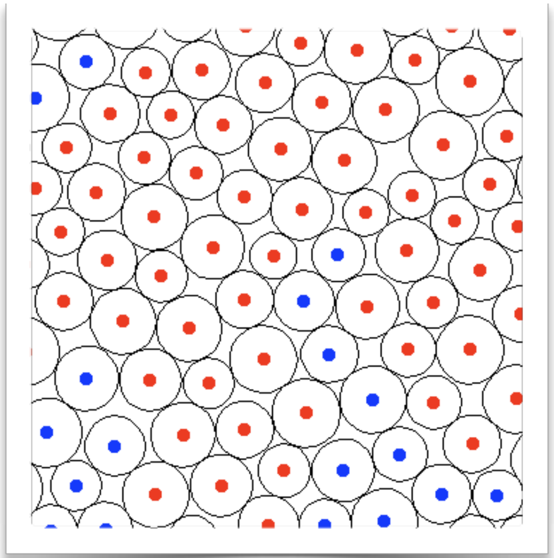
Sand on an inclined plane will flow like a fluid. Sand flow has instabilities: flowing by himself (avalanches) or under the action of wind, water, can become dunes. We have the data from a direct numerical simulation of particles down a 21 degree plane, in the file sandflow.mat (courtesy of Lydie Staron). There are three arrays: x, y, and d: the x position, the y position and the grain diameters. The lines correspond to the different grains, and the columns correspond to the different time instants, following each grain from its initial position at rest. The simulation has periodic boundary conditions in the direction along the plate, so grains come back up periodicaly.

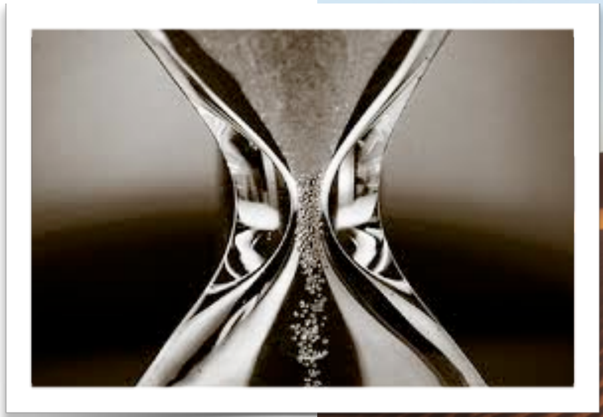
1) Draw the initial condition with circles.
2) Draw a dot at the center of each grain, with a different color depend on wether the grain is going up or is going down.
3) Display an animation of this granular flow.
4) We now will change the zoom during the animation: for the initial condition, focus on a small number of grains close to the bottom, and progressivelly enlarge the view such as to see the full flow at the end of the animation.
5) Use the buttondown property to display the evolution in time of the y position of one grain of the initial condition on which you click.
6) Use the same functionality to draw a the same time the y evolution of one grain and its 6 closest neigbours. This is to check wether grains stay close or depart from each other during their evolution, and check wether this depend on the depth of each grain.
7) Now draw the evolution in x of the selected grains: find the times for which the grain goes through the boundary condition using a logical array operation, and rectify its trajectory.
8) Compute for each time instant the average velocity profile in x. Draw an animation of the evolution in time of this profile.

Avalanche study



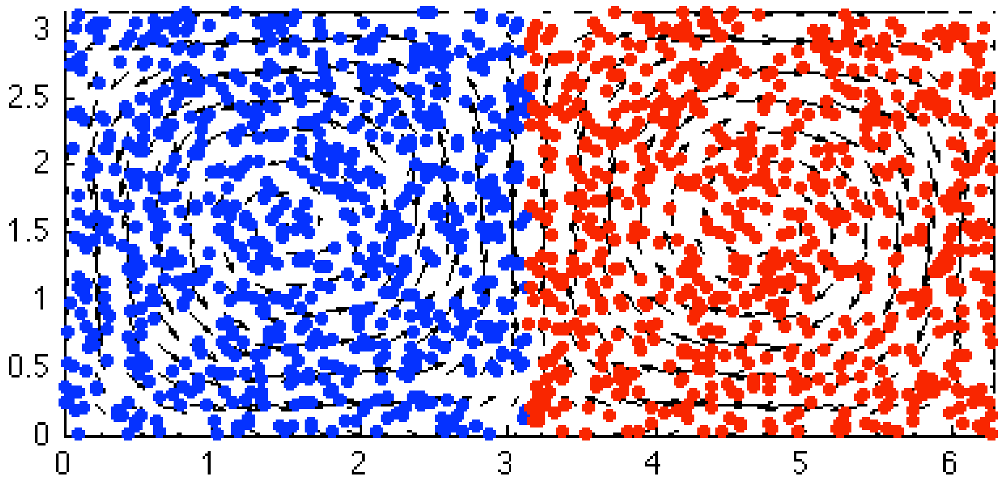
Long and short wavelengthes





# Exercises session 4

1)
Advection of particles by a time dependent velocity field using interpolation. The domain is periodic in all directions.
```
velocity field: u=sin(X-t).*cos(Y); v=-cos(X-t).*sin(Y);

initial condition:
```


2)
Make a scheduler for a parameter study of the fsc.f code, using "eval", "!". Fetch the results and plot in Matlab. Test the effect on the result of the boxheight: plot on top of each other the velocity profiles for many values of the boxheight. Measure the convergence error and plot it.

3)
Write a matlab code which builds a snake of N folders on your disc (a one-branch tree):
rep1/rep2/rep3/rep4/.../repN.
Now write a recursive code which builds a tree of folders.

4)
Imagine and realise a fancy application of interpolation.

# Exercices 5

1)
Try parfor and pmode on your computer.

2)
Do the simulation of the time evolution of the wave equation using the array formalism: initial condition, differentiation matrix, boundary conditions, time marching matrix H.

3) Build the differentiation matrices in 2D using the kron operator: differentiation in x and then in y. Test these differentiations against exact derivatives.

4)
Take the chance to finalize ideas for exercices from previous sessions (compiling the fsc code and scheduling the computations for instance, or vectorization of the vortex advection, or the buttondownfcn for the grains).

5)
Discuss with me your ideas for a project.

Thank you all for kindness and dedication!