Matlab Course
Hands-on workshop

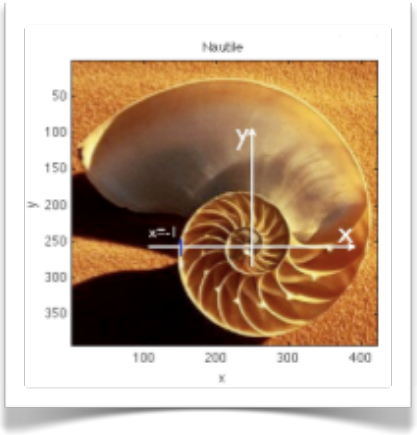
Jérôme Hoepffner/ Université Pierre et Marie Curie, Paris.
http://www.lmm.jussieu.fr/~hoepffner/gallery.php

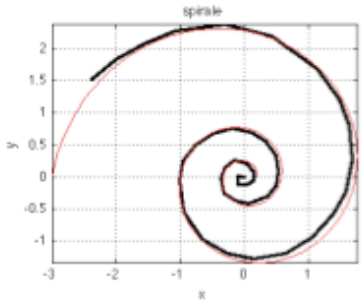
The computer is a tool at **the heart of science**: writing articles, analyzing and visualizing data, but also creating data by simulating accurate models of our physical systems. Creating data is typically done by intensive computations, by solving partial differential equations with a large number of degrees of freedom. This is done using *compiled* languages: the restriction is simulation time, these codes are large, complex, of long development time and hopefully long lifetime. Modern experimental measurement techniques as well produce large fields of data: Particle Image Velocimetry for full velocity fields, the large number of pixels of high speed cameras...

The data obtained this way is heavy and complex, and stored in the memory of computers. It is clear that **special programming skills** are needed to manipulate this data; on one hand to visualize it (data must be *seen*), and on the other hand to extract the relevant information: statistics, automatic identification of special events...

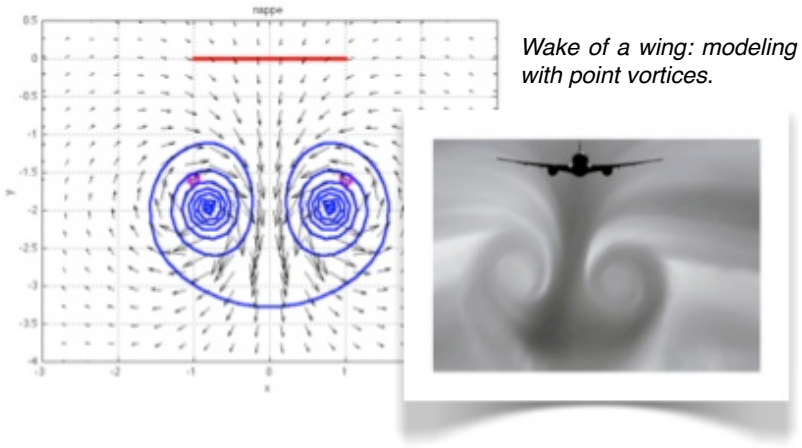
This use of the computer goes beyond post-processing: to understand our physical systems we need to compare them to **simple models**, often ordinary differential equations; light systems of equations which must be themselves solved or simulated.

The goal of this course is to set-up a standard of programing using Matlab to give you the tool you need for all that. These codes are typically simple, of rapid development time and often of short lifetime: the qualities of **interactivity**.



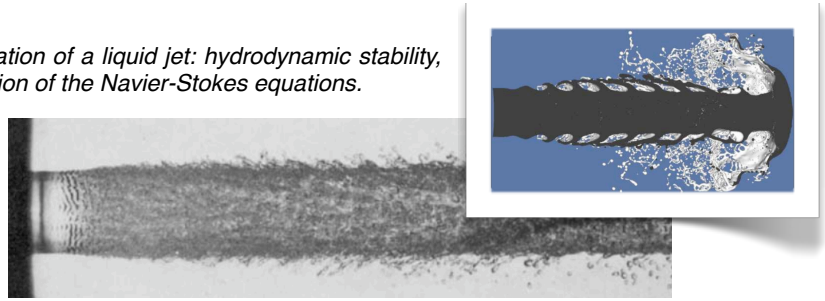


The shell is a shape of logarithmic spiral? This we can check. We can do more in quantifying its properties.



Wake of a wing: modeling with point vortices.

Atomization of a liquid jet: hydrodynamic stability, simulation of the Navier-Stokes equations.



Organisation

Four consecutive mornings, each one structured as: presentation of the general ideas, examples coded "live" and discussed, hands-on where you apply the techniques on various types of physical systems, put your results (graphics) together in a report. We keep the physics in mind at all times.

1) Compiled versus interpreted languages. Basic syntax: loops, tests, arrays, logicals, graphics. Using the documentation. Manipulation of arrays.

2) Graphics: overview of the possibilities. Set/get commands to affect all properties of the graphics through programing. Animations.

3) Making complex operation with short commands: manipulation of arrays, vectorization, use of arrays of logicals. Avoiding the mistakes which make a slow code, learning the basic ideas towards concision.

4) Linking heavy/efficient code to light/quick analyses: inputs and outputs, interfaces. Computations in parallel. Simulating systems and models.