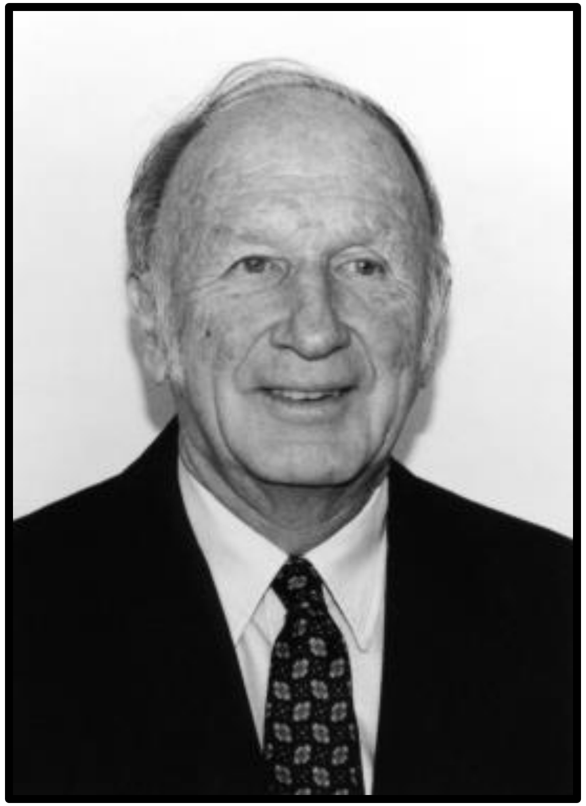
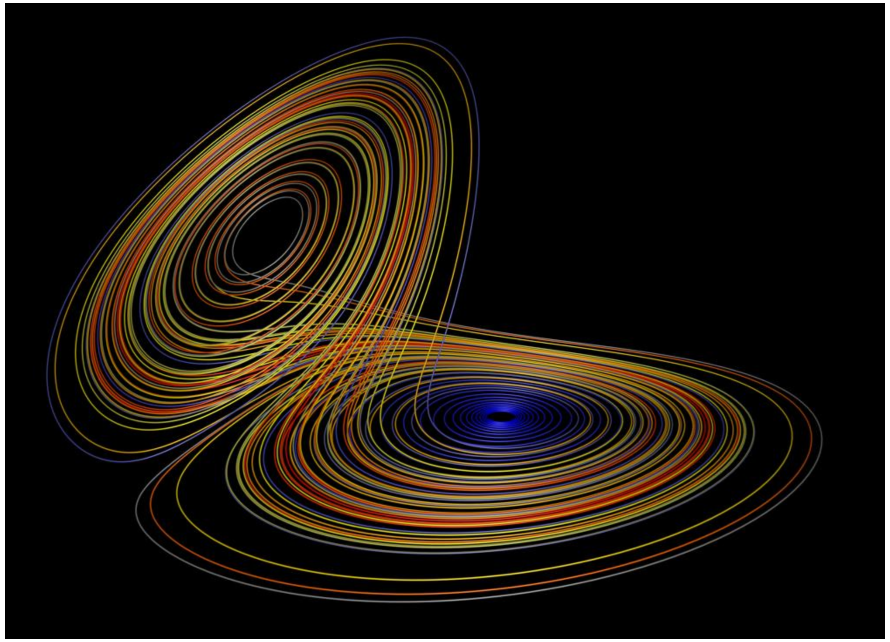
LINMA1731
Stochastic Processes

Project 2025

Lorenz System



Edward Lorenz (1917– 2008)

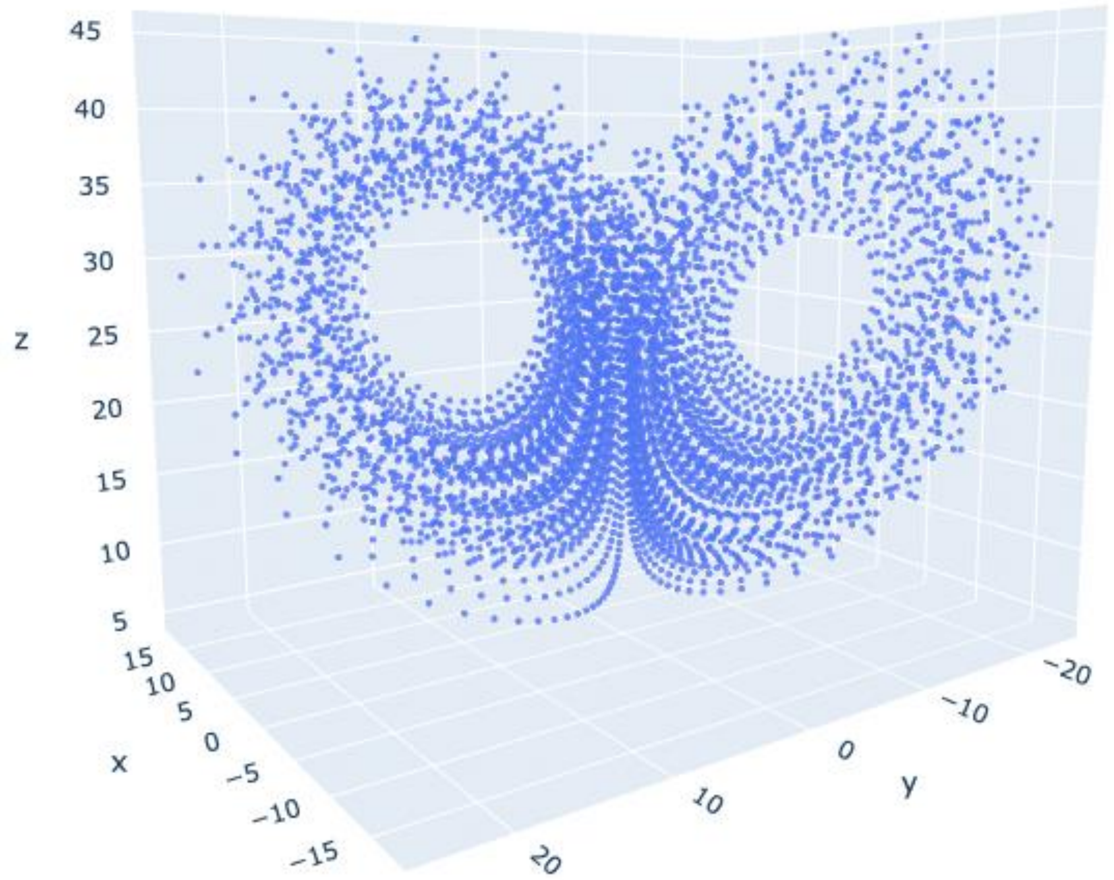


Chaos
Butterfly Effect
…

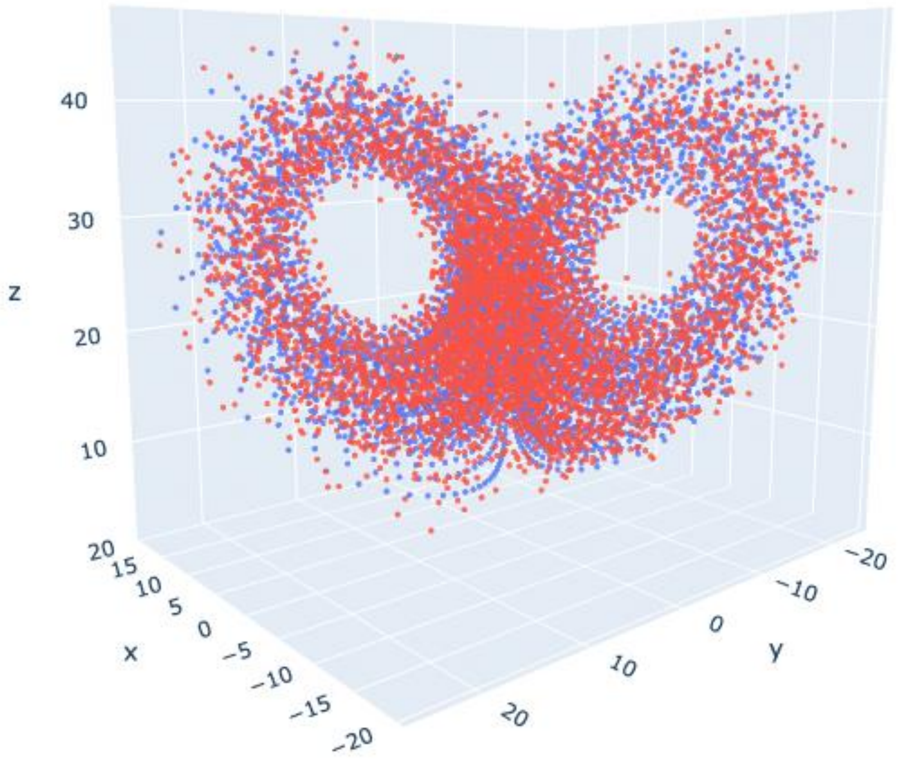
Lorenz System

$$\begin{cases} \dot{x} = \sigma(y - x) \\ \dot{y} = x(\rho - z) - y \\ \dot{z} = xy - \beta z \end{cases}$$

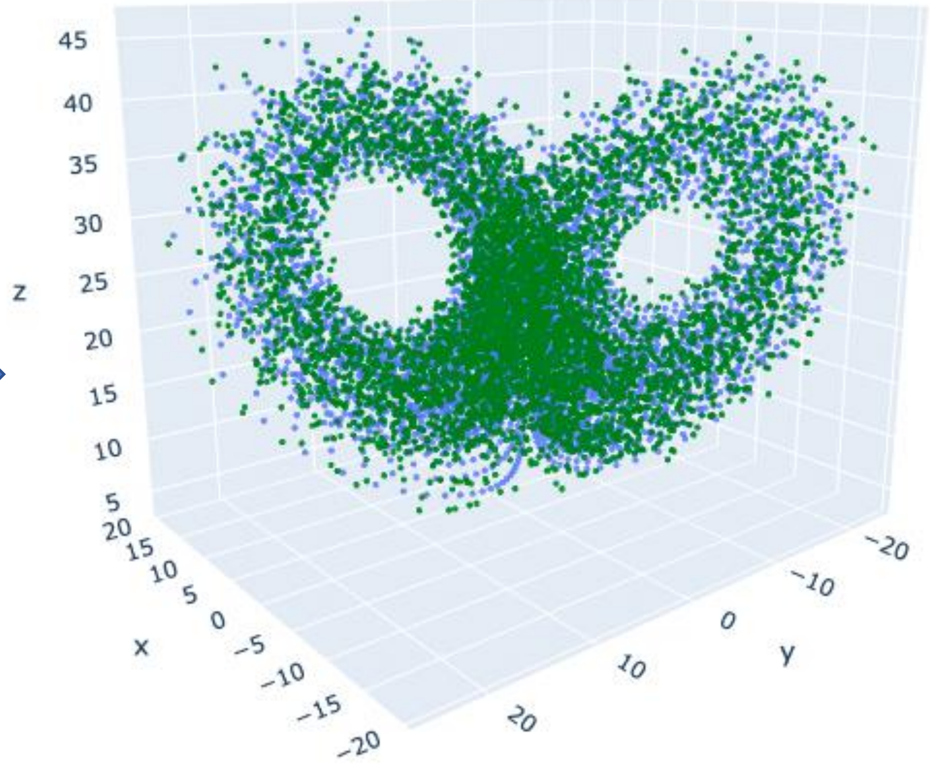
- x, y and z stand for the position coordinates of the point in the system
- σ, ρ and β are positive real parameters



Particle Filter



Noisy observations



Estimated values

Organisation of the project

Part 1 (rather theoretical)

Probability distribution of Lorenz System

Resampling in the particle filter

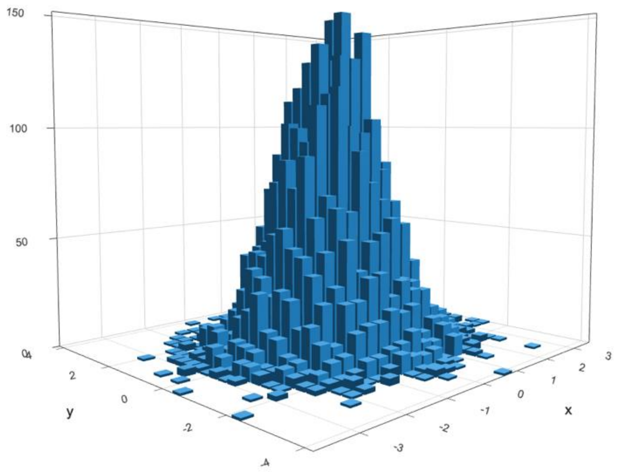
Part 2 (rather practical)

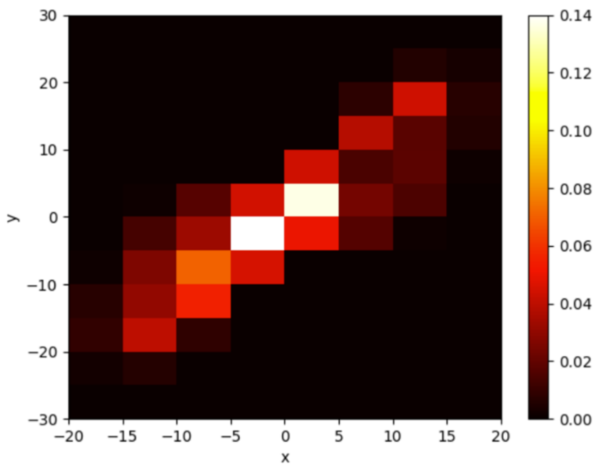
Implementation of the particle filter

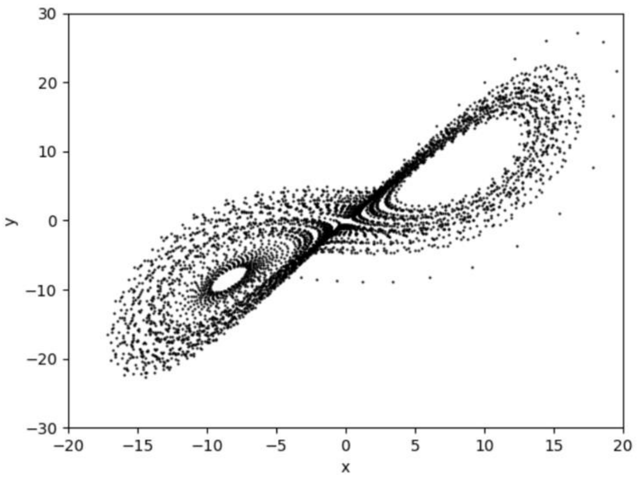
Part 1 : Study of the system and the filter

Probability density function (PDF) of the Lorenz System

Rectangular parallelepiped domain $[-20, 20] \times [-30, 30] \times [0,50]$







https://plotly.com/matlab/3D-histogram/

Part 1 : Study of the system and the filter

Importance of the resampling in the particle filter

Algorithm 40 (Classical SMC) *1. Generate n samples $x_0^i \sim f(x_0)$. Set t=0.*

2. **Prediction**: *Generate the prediction set using:* $\tilde{x}_{t+1}^i \sim f(x_{t+1}|x_t^i)$, $i = 1, 2, \ldots, n$.

3. **Update**: *Compute the weights* $w_{t+1}^i = f(y_{t+1}|\tilde{x}_{t+1})$, *and normalize them using* $\tilde{w}_{t+1}^i = \frac{w_{t+1}^i}{\sum_{j=1}^n w_{t+1}^j}$.

(a) *Estimate* θ_{t+1} *using* $\hat{\theta}_{t+1} = \sum_{i=1}^n g(\tilde{x}_{t+1}^i)\tilde{w}_{t+1}^i$.

(b) *Resample from the set* $\{\tilde{x}_{t+1}^1, \tilde{x}_{t+1}^2, \ldots, \tilde{x}_{t+1}^n\}$ *with probabilities* $\{\tilde{w}_{t+1}^1, \tilde{w}_{t+1}^2, \ldots, \tilde{w}_{t+1}^n\}$, *$n$ times to obtain the samples* x_{t+1}^i, $i = 1, 2, \ldots, n$.

4. Set $t = t + 1$, *and return to Step 2.*

Section 10.4 of
Srivastava, A. *Computational methods in statistics*. 2009.

Troubles to « visualise » what particle filter and resampling do ?

- Explore the scientific litterature
- Use a simplified weight vector ω first with arbitrary probabilities (e.g. for 5 or 10 particles) and do the computation by yourself
- Google and Youtube are your friends !

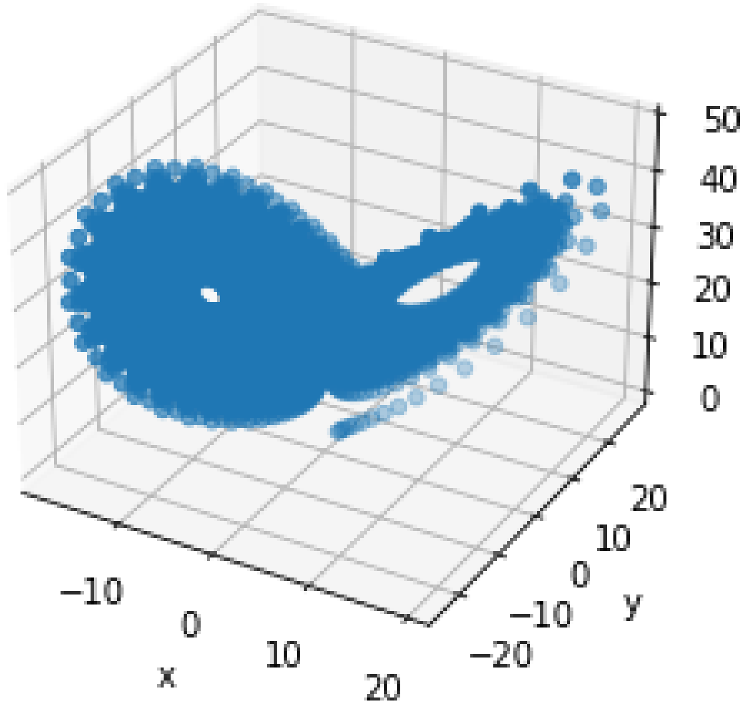
  There are many popularised (fr. « vulgarisé) or scientific references with different ways to explain things (please check carefully your sources before fully trust them…)

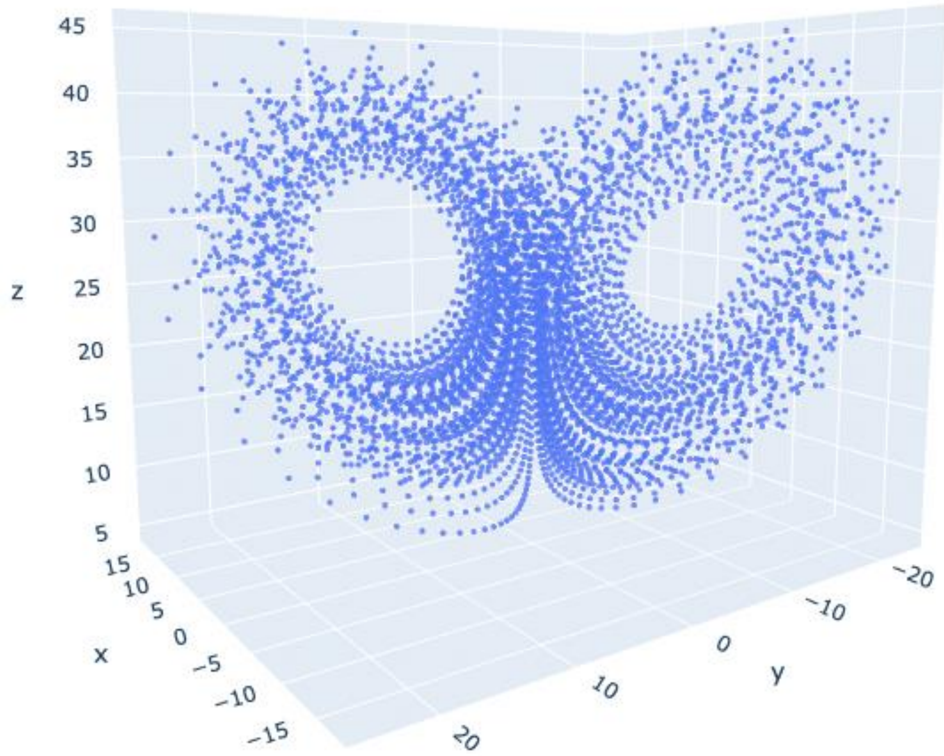
Part 2 : Implementation of the particle filter

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import odeint
from mpl_toolkits.mplot3d import Axes3D
from numpy import pi
import random

sigma = 10.0
rho = 28.0
beta = 8.0/3.0

# Lorenz model

def Lorenz(state,t):
    x, y, z = state  # Unpack the state vector
    return sigma * (y - x), x * (rho - z) - y, x * y - beta * z  # Derivatives

state0 = [1.0, 1.0, 1.0] # initial condition
t = np.arange(0.0, 100.0, 0.02) # time vector

states = odeint(Lorenz, state0, t) # vector containing the (x,y,z) positions for each time step

fig = plt.figure()
ax = fig.gca(projection="3d")
ax.plot(states[:, 0], states[:, 1], states[:, 2])
plt.xlabel('x')
plt.ylabel('y')
plt.legend(['True system'])
plt.draw()
plt.show()
```

Matplotlib VS Plotly

Installation : https://plotly.com/python/getting-started/





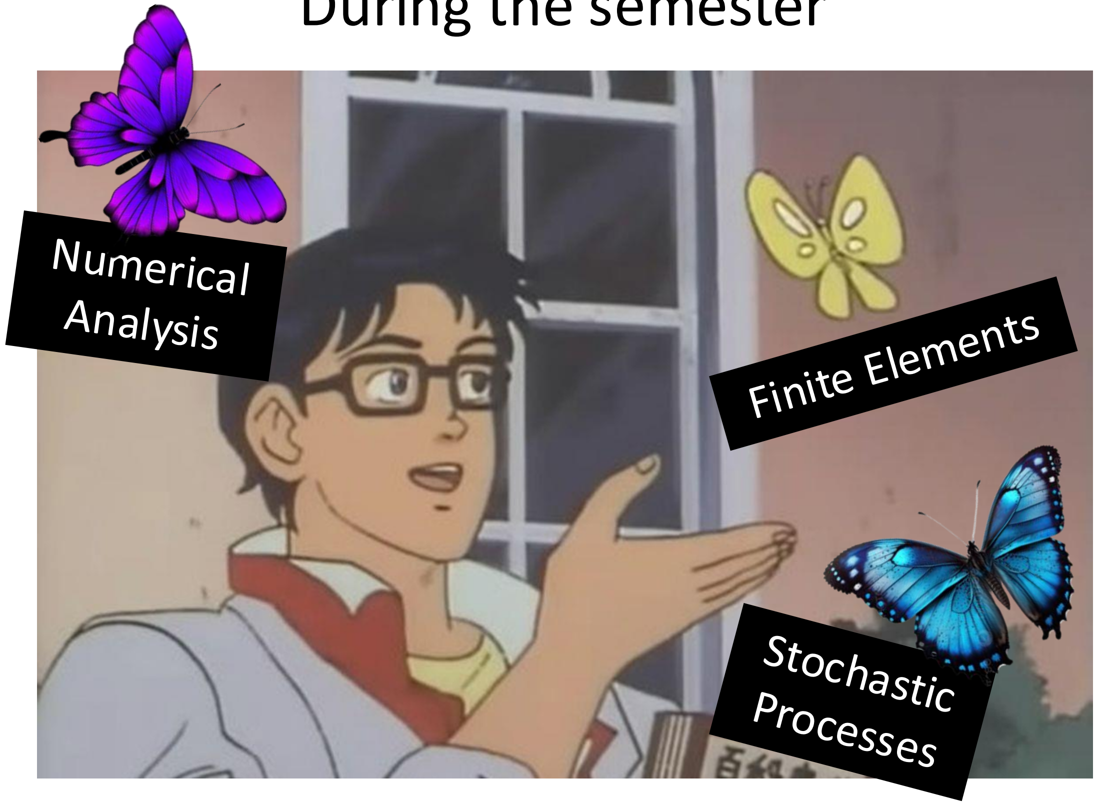
Practical details

- Groups of two students
  - Register on Moodle before Friday 21st March
- Two parts
  - Part 1 : mid of week 9
  - Part 2 : end of week 13
- Permanence
  - From week 7 to week 13
  - Tuesday 3-4 pm @Euler building (room A.007) with Philémon Beghin
  - Thursday 3-4 pm @Euler building (room A.011) with Amir Mehrnoosh
  - Mail to philemon.beghin@uclouvain.be or amir.mehrnoosh@uclouvain.be

Last advice: Don’t wait for the last moment

Random MAP Student

During the semester



At the end of the semester