240AR064 - Scientific Python for Engineers

Coordinating unit: 240 - ETSEIB - Barcelona School of Industrial Engineering
Teaching unit: 707 - ESAII - Department of Automatic Control
Academic year: 2019
Degree: MASTER'S DEGREE IN AUTOMATIC CONTROL AND ROBOTICS (Syllabus 2012). (Teaching unit Optional)
MASTER'S DEGREE IN INDUSTRIAL ENGINEERING (Syllabus 2014). (Teaching unit Optional)
ECTS credits: 3
Teaching languages: English

Teaching staff
Coordinator: Perera Lluna, Alexandre
Others: Perera Lluna, Alexandre
Velasco Garcia, Manuel

Opening hours
Timetable: Fridays 15:00-16:00

Prior skills
Knowledge of a programming language

Degree competences to which the subject contributes

Transversal:
CT3. TEAMWORK: Being able to work in an interdisciplinary team, whether as a member or as a leader, with the aim of contributing to projects pragmatically and responsibly and making commitments in view of the resources that are available.
CT4. EFFECTIVE USE OF INFORMATION RESOURCES: Managing the acquisition, structuring, analysis and display of data and information in the chosen area of specialisation and critically assessing the results obtained.

Teaching methodology
This class will be structured in three main tasks:

Lectures: the teachers will expose theoretical and practical contents, with the active participation of students.
Challenges: Students are exposed to a problem to be solved in a limited time.
Competitive projects: Problem solving projects where students are placed on a simulated scenario. In this scenario students program a simulated bot employing machine learning algorithms in python.

Final project defense includes an oral exposition of the developed work jointly with a discussion on the related methodology.

Learning objectives of the subject

The goal of the class is to learn skills for scientific programming, focused on the application of advanced machine learning tools on robotics. Students will learn to develop structured and problem solving thinking in a competitive environment.
240AR064 - Scientific Python for Engineers

### Study load

<table>
<thead>
<tr>
<th>Total learning time: 75h</th>
<th>Hours large group:</th>
<th>27h</th>
<th>36.00%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self study:</td>
<td></td>
<td>48h</td>
<td>64.00%</td>
</tr>
</tbody>
</table>
### Scientific Python for Engineering

**Learning time:** 30h  
- Theory classes: 15h  
- Laboratory classes: 3h  
- Guided activities: 5h  
- Self study: 7h

**Description:**
Part I
1. Introduction
   a. Why python?
   b. Python History
   c. Installing Python
   d. Python resources
2. Working with Python
   a. Workflow
   b. ipython vs. CLI
   c. Text Editors
   d. IDEs
   e. Notebook
3. Getting started with Python
   a. Introduction
   b. Getting Help
   c. Basic types
   d. Mutable and in-mutable
   e. Assignment operator
   f. Controlling execution flow
   g. Exception handling
4. Functions and Object Oriented Programming
   a. Defining Functions
   b. Decorators
   c. Writing Scripts and New Modules
   d. Input and Output
   e. Standard Library
   f. Object-oriented programming
   g. Magic Functions
5. Iterators and Generators
   a. Iterators
   b. Generators
6. Creating Graphic Interfaces (optional)
7. Debugging code
   a. Avoiding bugs
   b. Debugging workflow
   c. Python’s debugger
   d. Debugging segfaults using gdb
Part II
1. Introduction to NumPy
   a. Overview
   b. Arrays
   c. Operations on arrays
   d. Advanced arrays (ndarrays)
   e. Notes on Performance (%timeit in ipython)
2. Matplotlib
   a. Introduction
   b. Figures and Subplots
   c. Axes and Further Control of Figures
   d. Other Plot Types
   e. Animations
3. Plotting with Mayavi
a. Mlab: the scripting interface
b. Interactive work
4. Advanced Numpy
   a. Life of ndarray
   b. Universal functions
   c. Interoperability features
d. Array siblings: chararray, maskedarray, matrix
e. Summary
f. Contributing to Numpy/Scipy

Part III
1. Scipy
   a. Introduction
   b. Input/Output
   c. Statistics
d. Linear Algebra
e. Fast Fourier Transforms
f. Optimization
g. Interpolation
h. Numerical Integration
i. Signal Processing
j. Image Processing
k. Special Functions
2. Sparse Matrices in SciPy
   a. Introduction
   b. Storage Schemes
c. Linear System Solvers
d. Others
3. Optimizing code
   a. Optimization workflow
   b. Profiling your code
   c. Speeding your code
4. Sympy
   a. First Steps with SymPy
   b. Algebraic manipulations
c. Calculus
d. Equation solving
e. Linear Algebra

Part IV
1. Python scikits
   a. Introduction
   b. scikit-timeseries
c. scikit-audiolab
2. scikit-learn
   a. Datasets
   b. Sample generators
   c. Unsupervised Learning
   i. Clustering
   ii. Gaussian Mixture Models
   iii. Novelty/Outliers Detection
d. Supervised Learning
Qualification system

Class calibration will be obtained as a weighted mean comprising an evaluation of the challenges (50%) and the final project (50%).

Regulations for carrying out activities

Depending on the characteristics of the simulation environment and the bot complexity, the students can do the competition individually or in teams. Students will prepare a project report describing mathematical strategy, code structure and performance metrics.

Bibliography

Basic:
