AULA 1

Introdução ao curso. Regras de avaliação. Materiais de estudo. Revisão de conceitos básicos de programação em python.

Laboratório Numérico 2019

Teóricas: Pedro Miranda pmmiranda@fc.ul.pt, (C8, 8.3.38)

Práticas: Fernando Soares, Pedro Mateus, Pedro Miranda

Objectivos

Resolução de problemas em Física Aplicada e Engenharia com utilização de métodos numéricos

Linguagem: **python**

Avaliação

Exame final: 40%, nota mínima 8

Trabalho de grupo: 60% (*Projectos + Apresentações)* obrigatório

(Para quem queira manter a nota prática anterior, será nas regras anteriores 50%+50%)

Calendário

PL21	23
PL22	20
PL23	20
PL24	20
PL25	0
NO	3

PL25: Quinta 14:30-16:30

<	T	Q	Q	S	S	T	Q	Q	S	S	T	Q	Q	S
18-Feb	19-Feb	20-Feb	21-Feb	22-Feb	25-Feb	26-Feb	27-Feb	28-Feb	01-Mar	04-Mar	05-Mar	06-Mar	07-Mar	08-Mar
			T1		P1/P1	T2	P1/P1	T3/P1					T4	P1
11-Mar	12-Mar	13-Mar	14-Mar	15-Mar	18-Mar	19-Mar	20-Mar	21-Mar	22-Mar	25-Mar	26-Mar	27-Mar	28-Mar	29-Mar
P2/P2	T5	P2/P2	T6/P2		P3/P3	T7	P3/P3	T8/P3		P4/P4	T9	P4/P4	T10/P4	
01-Apr	02-Apr	03-Apr	04-Apr	05-Apr	08-Apr	09-Apr	10-Apr	11-Apr	12-Apr	15-Apr	16-Apr	17-Apr	18-Apr	19-Apr
P5/P5	T11	P5/P5	T12/P5		P6/P6	T13	P6/P6	T14/P6		P2	T15			
22-Apr	23-Apr	24-Apr	25-Apr	26-Apr	29-Apr	30-Apr	01-May	02-May	03-May	06-May	07-May	08-May	09-May	10-May
		P7/P7			P7/P7	T16		T18/P7		P8/P8	T17	P8/P8	T18/P8	
13-May	14-May	15-May	16-May	17-May	20-May	21-May	22-May	23-May	24-May	27-May	28-May	29-May	30-May	31-May
P9/P9	T19	P9/P9	T20/P9		P10/P10	T21	P10/P10	T22/P10		P11/P11		P11/P11	P11	

22 teóricas

11 práticas: 1 (introdução) 4 (Projeto 1) 1 (Apresentação 1) 4 (Projeto 2) 1 (Apresentação 2)

Projetos

Projeto 1

- Competências de programação: variáveis, estruturas de controlo, funções
- Input/output básico
- Gráficos 1D
- Métodos numéricos: regressão linear, equações não lineares, interpolação, sistemas de equações

Projecto 2

- Gestão de dados em multidimensões
- Integração
- Equações diferenciais
- Séries temporais
- Gráficos 2D e cartografia

Trabalho de grupo

Grupos de 2 membros

2 projetos por grupo

Os projetos deverão ser concluídos nas aulas P5 e P10 e enviados até 48h depois (py file comentada)

O trabalho prático (projectos) é obrigatório

Nas apresentações deverão falar os dois membros do grupo em igual tempo. A ordem será alternada entre o Projeto 1 e o Projeto 2. Na apresentação será entregue um ppt.

Importante

O trabalho é em grupo mas a avaliação é individual

Grupos de ≤ 2

Há marcação de presenças nas aulas práticas: máximo de 3 faltas nas aulas P2-P11

Em caso de força maior as apresentações poderão ser apresentadas no horário de outra turma. O mesmo se aplica a presenças.

Estudo

Notas do curso e ppts das aulas (progressivamente…)

fenix

Help: google “python anything”

python

Existem 2 versões (python 2 e python 3) não totalmente compatíveis… Quando se utiliza código já escrito pode ser necessário ter os 2.

Vamos usar a versão 3.6, na distribuição **anaconda**, recorrendo à interface gráfica **spyder**

A versão 3.6 é mais moderna e consistente, mas existem applicações que só correm na 2.7.

spyder

Interface gráfica para o python: inclui editor, linha de commandos (ipython), visualizador de variáveis (debug)

Pode ser necessário limpar a memória do spyder, de vez em quando:

```
reset  #apaga tudo
plt.close('all') #apaga todas as figuras
```

Nota: os scripts *.py são ficheiros de texto e podem ser corridos na linha de comandos (sem acumular lixo na memória):

```
python script.py
```

Objetos

O python processa **objetos**

Cada objeto é identificado por um nome:

X, x, xis, a22, b_2

Notar que x≠X

Cada objeto, consoante a **classe** a que pertença, pode conter conjuntos de números, textos e pedaços de Código

Os objetos mais simples são **escalares**: contêm um número. Esse número é de um dado **tipo**

Tipos numéricos

Plain integer (32 bit)

Long integer (n bit)

Floating point (64 bit)

Complex (2 x 64bit)

Boolean (0/1)

Listas

As listas são objetos constituídos por outros objetos

Exemplos:

```
x=[0,1]
Y=[0,10.,'texto']
z=[x,Y]
```

Um elemento de uma lista é acedido na forma:

```
Y0=Y[0]  #(==0)
Y1=Y[1]  # (==10.)
```

Arrays

O modulo **numpy** permite utilização de objetos multidimensionais que podem ser utilizados em operações algébricas de forma eficiente. Por exemplo:

```
X=numpy.array([0.,10.,20.,30.,40.])
```

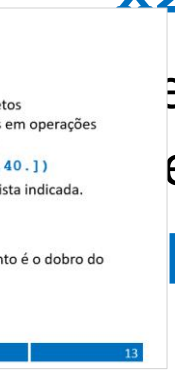
define um **vetor** X do tipo **float**, a partir da lista indicada.

A instrução:

```
X2=X*2
```

efine um novo vetor X2, onde cada elemento é o dobro do
emento correspondente de x, i.e.

```
[k]=x[k]*2, k=[0,1,2,3,4]
```



Módulos

```
from sympy import Symbol
import numpy as np
import math
Y=Symbol('y')
X=np.array([1,2,3])
Pi=math.pi
```

A instrução import dá acesso a bibliotecas de código já disponíveis na instalação. Por vezes pode ser necessário fazer o download de uma nova biblioteca:

Na linha de comandos (admin) na pasta Scripts: conda install biblioteca