1 Laborator 3

Saptamana 8-12 martie 2021

Continut:

1. Biblioteca ipywidgets
2. Exercitii cu ipywidgets

1.1 Folosire de controale grafice

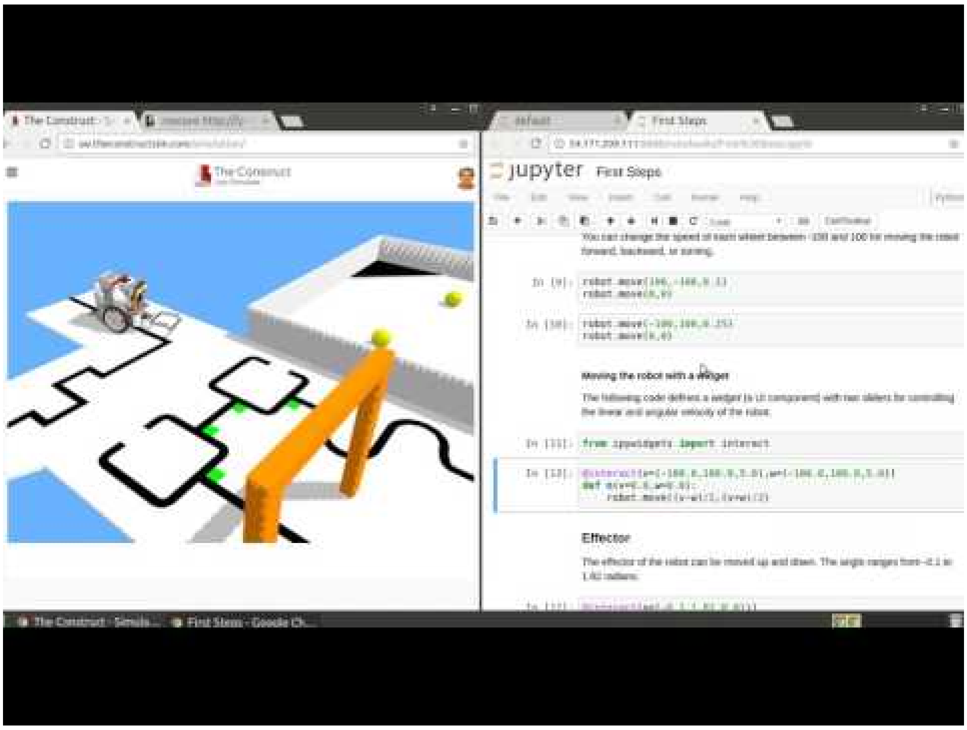
Notebook-urile - indiferent ca se ruleaza in Jupyter lab sau Jupyter notebook - se pot folosi pentru demo-uri interactive. O varianta este modificarea codului in timpul demo-ului si rularea manuala a celulelor afectate - nu intotdeauna rapid de facut. O alta varianta este folosirea de controale grafice care sa permita utilizatorului sa modifice optiuni, valori de parametri etc.

ipywidgets (https://ipywidgets.readthedocs.io/en/stable/) este o biblioteca de controale grafice care permit interactiune cu utilizatorul. Mai jos sunt cateva demo-uri de urmarit.

- Demo 1:

```
import ipywidgets as wg
from IPython.display import display
```

(https://www.youtube.com/watch?v=6SHnmho7zCs)
- Demo 2:



You can change the speed of each wheel between -100 and 100 for moving the robot forward, backward, or turning.

```
In [9]:  robot.move(100,-100,0.1)
         robot.move(0,0)

In [10]: robot.move(-100,100,0.25)
         robot.move(0,0)
```

Moving the robot with a widget

The following code defines a widget (a UI component) with two sliders for controlling the linear and angular velocity of the robot.

```
In [11]: from ipywidgets import interact

In [12]: @interact(v=(-100.0,100.0,5.0),w=(-100.0,100.0,5.0))
         def move(v=0.0,w=0.0):
             robot.move((v-w)/2,(v+w)/2)
```

Effector

The effector of the robot can be moved up and down. The angle ranges from -0.1 to 1.82 radians.

```
In [13]: @interact(a=(-0.1,1.82,0.01))
```

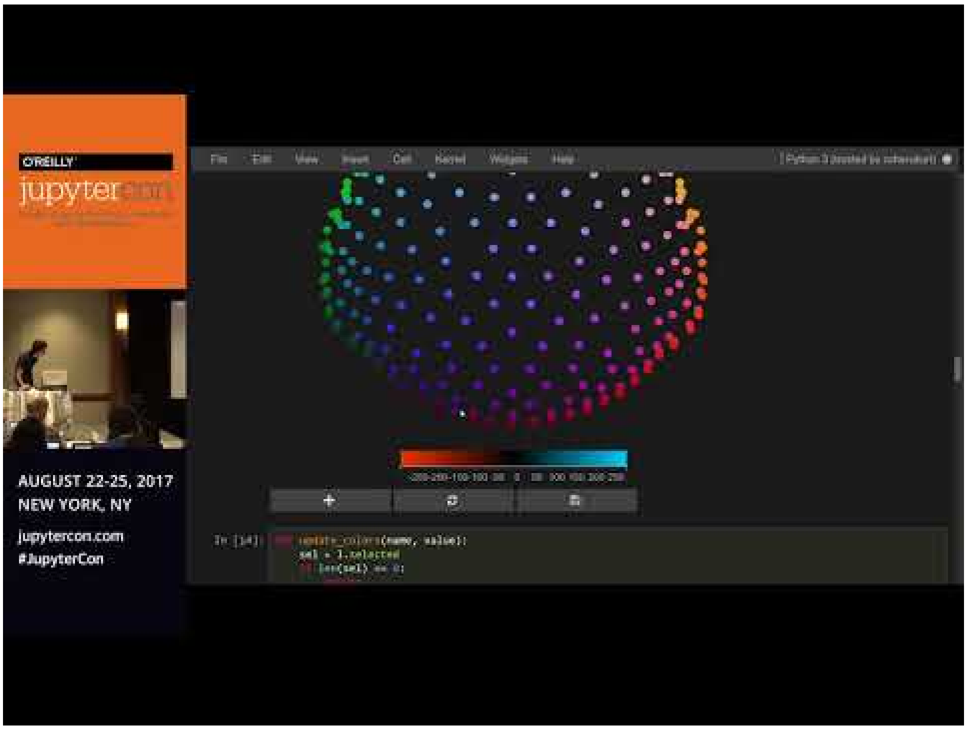
[(https://www.youtube.com/watch?v=BH0rtx0Qq5w)](https://www.youtube.com/watch?v=BH0rtx0Qq5w)

- Demo 3:

SciPy 2020
Scientific Computing with Python
Virtual Conference • July 6-12, 2020
Tutorial: Jupyter Interactive Widget Ecosystem
ENTHOUGHT

[(https://www.youtube.com/watch?v=8IYbdshUd9c)](https://www.youtube.com/watch?v=8IYbdshUd9c)

- Demo 4:



```
In [14]: def update_colors(name, value):
             sel = l.selected
             if len(sel) == 0:
```

[(https://www.youtube.com/watch?v=i40d8-Hu4vM)](https://www.youtube.com/watch?v=i40d8-Hu4vM)

1.2 Exemple de utilizare

Documentatia completa si exemple sunt date [aici (https://ipywidgets.readthedocs.io/en/stable/user_guide.html)](https://ipywidgets.readthedocs.io/en/stable/user_guide.html).

Incarcarea pachetului de `ipywidgets` se face prin:

In [1]:

```
import ipywidgets as widgets
print(f'IPywidgets version: {widgets.__version__}')

import numpy as np
print(f'NumPy version: {np.__version__}')

import matplotlib.pyplot as plt
import matplotlib
print(f'Matplotlib version: {matplotlib.__version__}')
```

executed in 426ms, finished 13:21:23 2021-03-09

```
IPywidgets version: 7.6.3
NumPy version: 1.19.2
Matplotlib version: 3.3.4
```

De regula, e nevoie si de alte pachete, de exemplu:

In [2]:

```
from ipywidgets import interact, interactive, fixed, interact_manual
```

executed in 5ms, finished 13:21:23 2021-03-09

Cel mai simplu control utilizabil este `interact`. El poate prelua ca prim parametru numele unei functii, iar al doilea parametru dicteaza forma controlului: slider, combo box, checkbox etc:

In [3]:

```
def n_factorial(n:int) -> int:
    """Calculeaza n factorial
    :param n: intreg >= 0 pt care se calculeaza factorialul
    :return: valoarea lui n!
    """
    p = 1
    for i in range(1, n+1):
        p *= i
    return str(n) + "!= " + str(p)
```

executed in 12ms, finished 13:21:23 2021-03-09

In [4]:

```
interact(n_factorial, n=100);
```

executed in 61ms, finished 13:21:23 2021-03-09

A Jupyter widget could not be displayed because the widget state could not be found. This could happen if the kernel storing the widget is no longer available, or if the widget state was not saved in the notebook. You may be able to create the widget by running the appropriate cells.

Pentru limitarea domeniului in care n poate sa ia valori se va folosi:

In [5]:

```
interact(n_factorial, n=(0, 100));
```

executed in 34ms, finished 13:21:24 2021-03-09

A Jupyter widget could not be displayed because the widget state could not be found. This could happen if the kernel storing the widget is no longer available, or if the widget state was not saved in the notebook. You may be able to create the widget by running the appropriate cells.

Pentru a evita actualizarea sacadata a valorilor afisate, se prefera inhibarea feedback-ului in timp real, precum in Disabling continuous updates (https://ipywidgets.readthedocs.io/en/stable/examples/Using%20Interact.html#Disabling-continuous-updates).

Pentru alte tipuri de controale folosind interact, se poate folosi:

In [6]:

```
from typing import Any, Tuple
def g(x:Any, y: Any, z: Any, t: Any) -> Tuple[Any, Any, Any, Any]:
    return (x, y, z, t)

interact(g, x=True, y=(1.0, 10.0, 0.5), z='Un text',
         t={'English':'Hello', 'Romanian':'Salut', 'Spanish':'Hola'});
```

executed in 68ms, finished 13:21:24 2021-03-09

A Jupyter widget could not be displayed because the widget state could not be found. This could happen if the kernel storing the widget is no longer available, or if the widget state was not saved in the notebook. You may be able to create the widget by running the appropriate cells.

Exemplu: Sa se deseneze graficul functiei $f : [left, right] \to \mathbb{R}$, $f(x) = a \cdot x^2 + b \cdot x + c$, cu a, b, c coeficienti reali.

Rezolvare:

In [7]:

```
def f_square(a=10, b=20, c=-10, left=-10, right=20) -> None:
    '''Afiseaza graficul unei functii de gradul al doilea de forma:
    f(x)=a*x**2 + b*x + c. Valorile lui x sunt luate din intervalul
    [left, right] prin discretizare.
    :param a: coeficientul lui x**2
    :param b: coeficientul lui x
    :param c: termenul liber
    :param left: capatul din stanga al intervalului peste care se face
    reprezenatrea
    :param right: capatul din dreapta al intervalului peste care se face
    reprezenatrea
    :return: None
    '''
    assert left < right
    range_x = np.linspace(left, right, 100)
    values_f = a * range_x ** 2 + b * range_x + c
    plt.figure(figsize=(10, 8))
    plt.xlabel('x')
    plt.ylabel(str(a) + '$\cdot x^2 + $' + str(b) + '$\cdot x + $' + str(c))
    plt.plot(range_x, values_f, color='red')
    plt.grid(axis='both')
    plt.axhline(y=0, color='k')
    plt.axvline(x=0, color='k')
    plt.show()

interact(f_square, a=(-100, 100.0), b=(-100, 100.0), c=(-100, 100.0), d=(-100, 100.0), e=
```

executed in 617ms, finished 13:21:24 2021-03-09

A Jupyter widget could not be displayed because the widget state could not be found. This could happen if the kernel storing the widget is no longer available, or if the widget state was not saved in the notebook. You may be able to create the widget by running the appropriate cells.

In [8]:

```
def sinusoid(f=10):
    range_x = np.linspace(-5, 5, 100)
    values_f = np.sin(2 * np.pi * f * range_x)
    plt.xlabel('x')
    plt.ylabel(f'$2 \cdot \pi \cdot {f} \cdot x$')
    plt.grid(axis='both')
    plt.axhline(y=0, color='k')
    plt.axvline(x=0, color='k')
    plt.plot(range_x, values_f)

interact(sinusoid, f = (1, 100.0, 0.5));
```

executed in 277ms, finished 13:21:24 2021-03-09

A Jupyter widget could not be displayed because the widget state could not be found. This could happen if the kernel storing the widget is no longer available, or if the widget state was not saved in the notebook. You may be able to create the widget by running the appropriate cells.

In [9]:

```
def f(x):
    """calcul functie intr-un punct"""
    return x ** 2 - 10 * x + 50

def f_values(left=-10, right=10):
    """calcul functie pe interval"""
    x = np.linspace(left, right, 100)
    return x, f(x)

def f_prime(x):
    """Calcul derivata f
    :param x: punctul in care se calculeaza derivata
    :return: f'(x)
    """
    return 2 * x - 10

def graph_f_and_derived(x, left=-30, right=30):
    # calcul valoare functie f
    x_range, fx = f_values(left, right)

    # intervalul pe care se reprezinta tangenta la grafic
    x_segment = np.linspace(x-10, x+10, 100)
    # panta tangentei la grafic este derivata functiei in pctul de tangenta
    slope = f_prime(x)

    #calcul puncte de tangenta
    y_segment = f(x) + slope * (x_segment - x)

    plt.figure(figsize=(20, 10))
    plt.plot(x_range, fx, color='red')
    plt.plot(x_segment, y_segment, color='blue')

# graf_f_and_derived(10, left=-30, right=30)

interact(graph_f_and_derived, x = (-20, 20))
```

executed in 292ms, finished 13:21:25 2021-03-09

A Jupyter widget could not be displayed because the widget state could not be found. This could happen if the kernel storing the widget is no longer available, or if the widget state was not saved in the notebook. You may be able to create the widget by running the appropriate cells.

Out[9]:

```
<function __main__.graph_f_and_derived(x, left=-30, right=30)>
```

2 Exercitii ipywidgets:

Problema 1. (2 puncte) Definiti o functie polinomiala de gradul 3, $f : \mathbb{R} \to \mathbb{R}$, cu coeficienti constanti prestabiliti. Aplicati algoritmul gradient descent pentru a vedea cum evolueaza cautarea minimului. Folositi minim doua controale ipywidgets: unul pentru pozitia initiala a lui x, altul pentru coeficientul $\alpha > 0$ cu care se inmulteste gradientul. Gradientul va fi calculat analitic de voi sau folosind biblioteca autograd (https://github.com/HIPS/autograd). Modificarea facuta prin metoda gradient descent este:

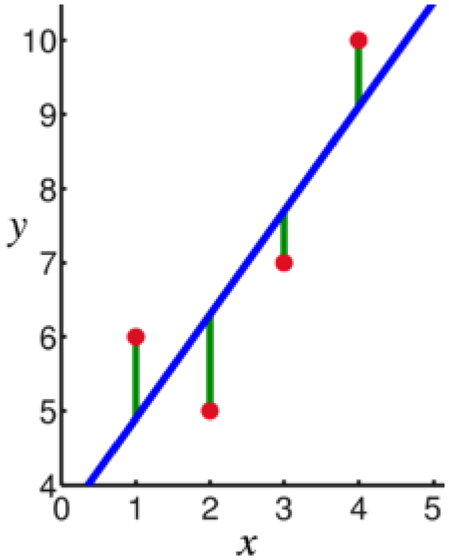
$$x = x - \alpha \cdot f'(x)$$

Se vor efectua minim 10 iteratii (optional: numarul de iteratii poate fi dat printr-un control ipywidgets), se vor marca pe grafic pozitiile succesive, in mod convenabil.

Problema 2. (3 puncte) Generati o lista de $n = 100$ de perechi de valori $\{x_i, y_i\}_{i=0,n-1}$ in intervalul [-20, 10), afisati aceste valori pe un grafic, impreuna cu o dreapta definita de o functie liniara $y = a \cdot x + b$. Intr-un alt plot afisati, ca histograma, distanta dintre punctele de coordonate (x_i, y_i) si punctele de intersectie ale verticalelor duse prin x_i cu dreapta data, $\hat{y}_i$. Dreapta trebuie sa fie controlabila din widgets, prin cei doi coeficienti a si b. Constatati modificarea histogramei in functie de pozitia dreptei si afisati mean squared error:

$$MSE = \frac{1}{n} \cdot \sum_{i=0}^{n-1} (y_i - (a \cdot x_i + b))^2$$

.



Indicatii:

1. Pentru generare de valori distribuite uniform in intervalul [0, 1) puteti folosi functia [numy.random.uniform (https://docs.scipy.org/doc/numpy/reference/generated/numpy.random.uniform.html)](https://docs.scipy.org/doc/numpy/reference/generated/numpy.random.uniform.html) si sa faceti inmultire si adunare in mod convenabil.
2. Puteti opta sa returnati cele n puncte sub forma `vector_x` , `vector_y` .

Problema 3. (2 puncte) Incarcati fisierul `iris.data` din [setul de date iris (https://archive.ics.uci.edu/ml/datasets/iris)](https://archive.ics.uci.edu/ml/datasets/iris). In functie de alegerile exprimate de un utilizator, afisati intr-un grafic 2D coloanele numerice alese (de exemplu, coloana 0 si coloana 2). Numele coloanelor se afla in fisierul iris.names.

Indicatii/optiuni:

- Incarcarea de date se poate face cu numpy, functia [loadtxt (https://docs.scipy.org/doc/numpy/reference/generated/numpy.loadtxt.html)](https://docs.scipy.org/doc/numpy/reference/generated/numpy.loadtxt.html). Specificati faptul ca se sare peste prima linie din fisier (header). Alternativ, puteti folosi [pandas.read_csv (https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.read_csv.html)](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.read_csv.html).
- Pentru cele doua alegeri puteti sa instantiati doua obiecte [Dropdown (https://ipywidgets.readthedocs.io/en/stable/examples/Widget%20List.html#Dropdown)](https://ipywidgets.readthedocs.io/en/stable/examples/Widget%20List.html#Dropdown) sau [Select (https://ipywidgets.readthedocs.io/en/stable/examples/Widget%20List.html#Select)](https://ipywidgets.readthedocs.io/en/stable/examples/Widget%20List.html#Select).

Problema 4 (3 puncte) Generati n perechi de puncte aleatoare, folosind o functie $f : \mathbb{R} \rightarrow \mathbb{R}$ de alease e voi (de exemplu: functie polinomiala + zgomot aleator adaugat). Alegeti 5 metode de interpolare din [scipy.interpolate (https://docs.scipy.org/doc/scipy/reference/interpolate.html)](https://docs.scipy.org/doc/scipy/reference/interpolate.html) si reprezentati grafic valorile interpolate. Folositi controale ipywidgets cel putin pentru alegerea lui n si a metodei de interpolare aleasa.

Predare:

1. Saptamana 21 martie, ora 22, in lucrarea de pe elearning (Tema 3) + repo propriu de pe github. La prezentarea temei coechipierii trebuie sa fie prezenti.
2. Denumirea fisierului predat trebuie sa respecte conventia de la tema 2.