Introduction to Numerical Computing in Python

name1 lastname1, name2 lastname2, name3 lastname3

August 29, 2021

Abstract

In human motion analysis there is a frequently need for processing large amounts of data. Storing data in lists and traversing lists with built-in Python's loops leads to slow code.

Problem 1

Given the list **rnd_seq**. The problem states that you should find the indexes of all occurrences of the item 'blue'.

```
"""
@autor: Eng. Alexander Sierra, Assistant Professor
"""
from __future__ import print_function
import random
import datetime
# seeding the random number generator
random.seed(datetime.datetime.now().year)

spectrum = ['violet', 'blue', 'cyan', 'green', 'yellow', 'orange', 'red']

rnd_seq = [random.choice(spectrum) for _ in range(100000)]

item_idxs = []

print (item_idxs)
```

Script 1: random sequence of string objects

In the script 1, the **seed** method is seeding the underlying random number generator used by Python's random module.

Problem 2

Consider a 1-dimensional list of **True** and **False** objects. The problem states that you should count the number of False-to-True transitions in the sequence.

```python
"""
@autor: Eng. Alexander Sierra, Assistant Professor
"""
from __future__ import print_function
import random
import datetime

# seeding the random number generator
random.seed(datetime.datetime.now().year)

rnd_seq=[random.choice([False, True])
         for _ in range(100000)]

print(len(rnd_seq))
```

Script 2: random sequence of boolean objects

Problem 3

For an array of points $\mathbf{X}$ where each row represents a point in $\mathbb{R}^3$ compute $\|\mathbf{x}\|$ for $\mathbf{x}$ in $\mathbf{X}$.

$$\mathbf{X} = \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ \vdots & \vdots & \vdots \\ x_n & y_n & z_n \end{bmatrix} \tag{1}$$

In the script 3, the $\mathbf{X}$ array is defined.

```python
"""
@autor: Eng. Alexander Sierra, Assistant Professor
"""
from __future__ import print_function
from pprint import pprint
import random
import datetime

# seeding the random number generator
N = 100000
random.seed(datetime.datetime.now().year)

x_array = []
for row in range(N):
    point = []
    for i in range(3):
        point.append(random.random())
    x_array.append(point)

# show the first 10 rows of the X array
pprint(x_array[:10])
```

Script 3: Two-dimensional array

Problem 4

Consider a two-dimensional array containing a 3D-point cloud. Where N is the number of points in the cloud.

$$\mathbf{ptCloud} = \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ \vdots & \vdots & \vdots \\ x_n & y_n & z_n \end{bmatrix} \tag{2}$$

In Figure 1, we can see an example of a point cloud.

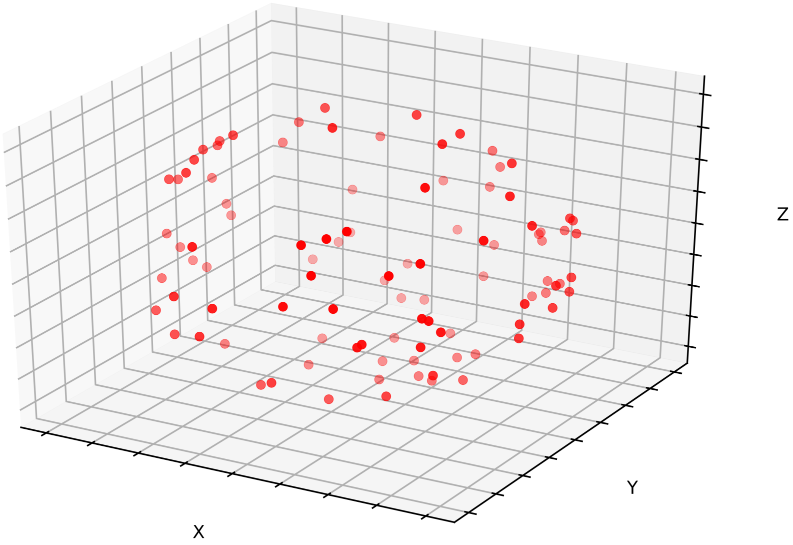

Figure 1: Point Cloud

The problem states that you should compute the radius of the sphere that best fit to all points. Hint: find the centroid of point cloud.

The script 4 generates the point cloud.

```python
"""
@autor: Eng. Alexander Sierra, Assistant Professor
"""
from __future__ import print_function
from pprint import pprint
import random


def point_cloud_generator(n=10):
    """
    Helper function to make an array having shape (n, 3)
    """
    _array = []
    for rows in range(n):
        point = []
        for _ in range(3):
            point.append(random.gauss(0, 1))
        _array.append(point)
    return _array


ptCloud = point_cloud_generator(100000)
print ("point␣cloud␣length:␣{}".format(len(ptCloud)))

# show the first 10 rows of the ptCloud 2d-array
pprint(ptCloud[:10])
```

Script 4: 3D point cloud

Problem 5

The **coordinates.csv** file specify the position of two points in the three-dimensional space at different times. A fragment of this file is shown in the table below.

Table 1: coordinates of each point

rptx	rpty	rptz	lptx	lpty	lptz
$rptx_1$	$rpty_1$	$rptz_1$	$lptx_1$	$lpty_1$	$lptz_1$
$rptx_2$	$rpty_2$	$rptz_2$	$lptx_2$	$lpty_2$	$lptz_2$
$rptx_3$	$rpty_3$	$rptz_3$	$lptx_3$	$lpty_3$	$lptz_3$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$rptx_{1000}$	$rpty_{1000}$	$rptz_{1000}$	$lptx_{1000}$	$lpty_{1000}$	$lptz_{1000}$

The first line of file is a header that contains column labels followed by the rows of data. Each row of data contains the (x, y, z) coordinates of each point.

The problem states that you should compute the mean distance between two points.

```
"""
@autor: Eng. Alexander Sierra, Assistant Professor
"""
from __future__ import print_function

# print(datetime.datetime.now().year)

f = open('coordinates.csv', 'r')



f.close()
```

Script 5: 3D point cloud

Marking criteria

The assignment is marked out of 50. Your work will be marked using the following criteria:

- Problem 1, 5 marks.
- Problem 2, 5 marks.
- Problem 3, 10 marks.
- Problem 4, 10 marks.
- Problem 5, 20 marks.

Hand-out date: 29th Aug. 2021, 22:00
Hand-out Method: Moodle and Microsoft Teams

Hand-in date: 10th Mar. 2021, 18:00
Hand-in Method: Moodle