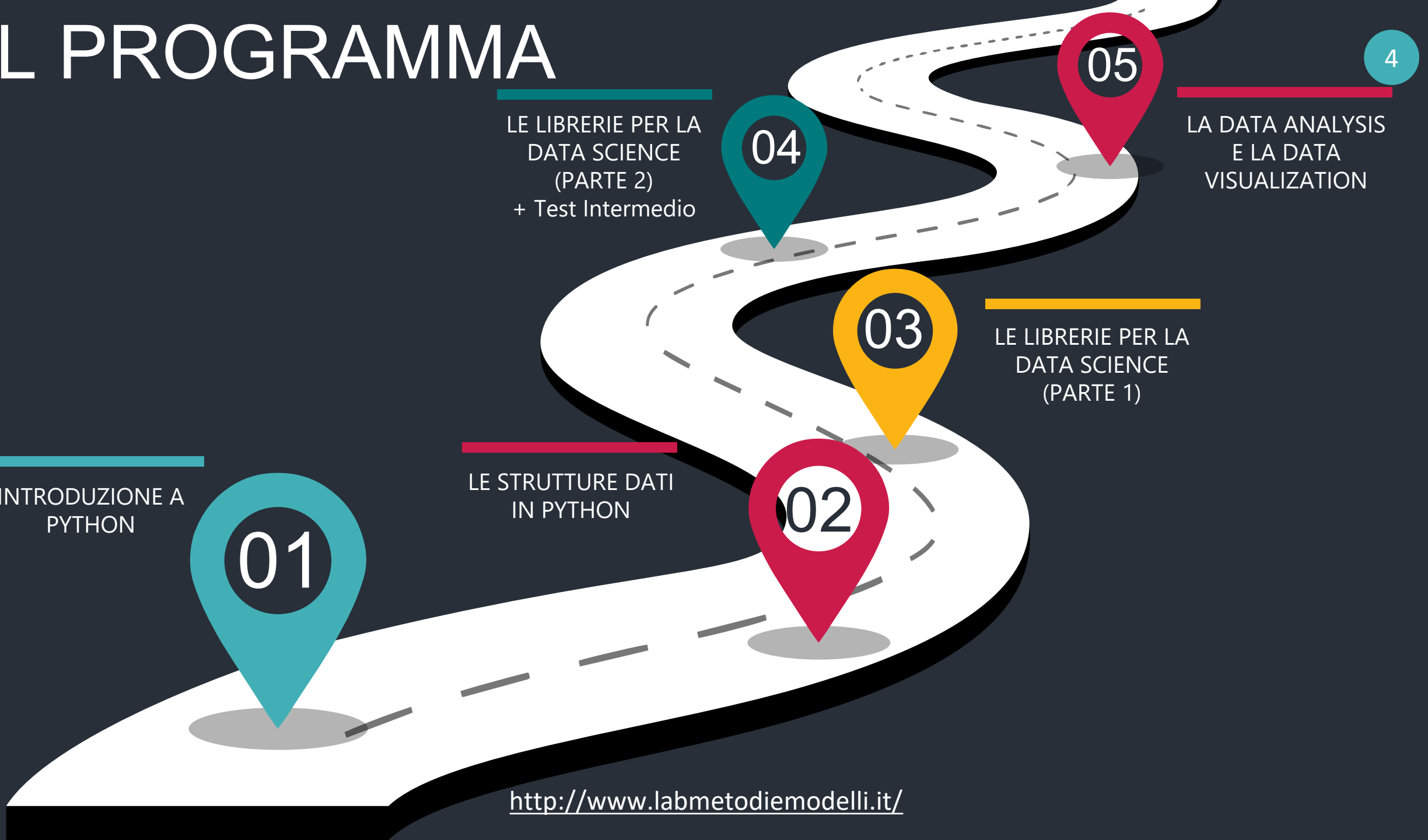
LABORATORIO DI:

METODI E MODELLI MATEMATICI IN PYTHON

A CURA DI: ANTONIO MIRARCHI & GIUSEPPE TROTTA

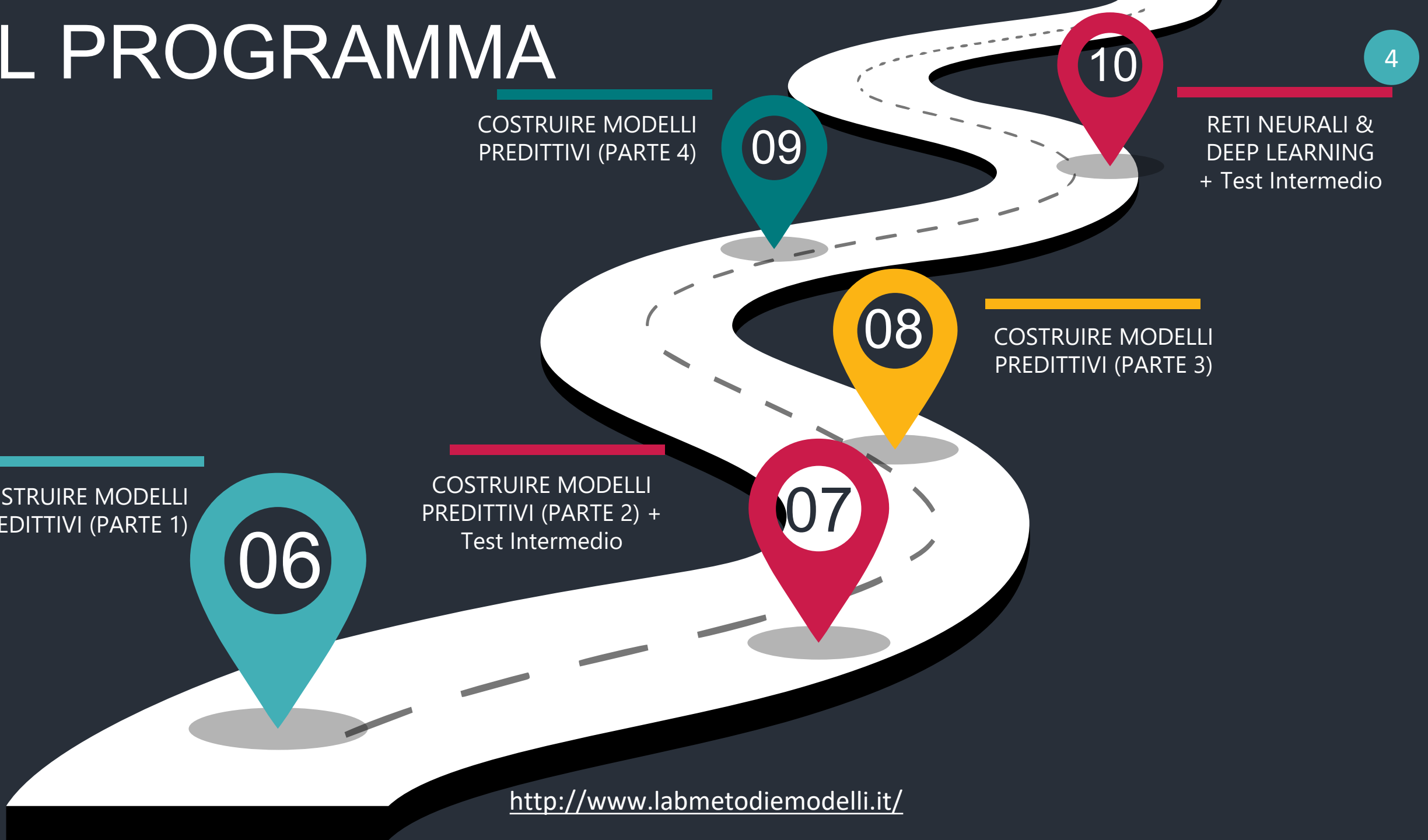
http://www.labmetodiemodelli.it/

IL PROGRAMMA



http://www.labmetodiemodelli.it/

IL PROGRAMMA



http://www.labmetodiemodelli.it/

Let's Code!

http://www.labmetodiemodelli.it/

1° Test intermedio

Manuale di sopravvivenza al laboratorio

4 esercizi

Argomenti lezioni
dalla 1 alla 3

Consegna via mail entro le ore 19:00
antonio.mirarchi@thinkopen.it

http://www.labmetodiemodelli.it/

Dove eravamo rimasti?

Le Librerie per la Data Science

http://www.labmetodiemodelli.it/

Data Science Libreries

A Brave New World!

http://www.labmetodiemodelli.it/

Le librerie scientifiche

01 NumPy

02 SciPy

03 MatplotLib

04 Pandas

https://www.labmetodiemodelli.it/

1. NumPy

Numpy is the core library for scientific computing in Python. It provides a high-performance multidimensional array object, and tools for working with these arrays.

Una delle funzionalità più importanti di NumPy è la sua interfaccia Array. Questa interfaccia può essere utilizzata per esprimere immagini, onde sonore o altri flussi binari grezzi come matrici di numeri reali con dimensione N.

1. NumPy - Array

Numpy is the core library for scientific computing in Python. It provides a high-performance multidimensional array object, and tools for working with these arrays.

```
import numpy as np

a = np.array([1, 2, 3])   # Create a rank 1 array
print(type(a))            # Prints "<class 'numpy.ndarray'>"
print(a.shape)            # Prints "(3,)"
print(a[0], a[1], a[2])   # Prints "1 2 3"
a[0] = 5                  # Change an element of the array
print(a)                  # Prints "[5, 2, 3]"

b = np.array([[1,2,3],[4,5,6]])    # Create a rank 2 array
print(b.shape)                     # Prints "(2, 3)"
print(b[0, 0], b[0, 1], b[1, 0])   # Prints "1 2 4"
```

1. NumPy – Array creation

Numpy also provides many functions to create arrays:

```
import numpy as np

a = np.zeros((2,2)) # Create an array of all zeros
print(a)  # Prints "[[ 0. 0.]
          #          [ 0. 0.]]"
b = np.ones((1,2)) # Create an array of all ones
print(b) # Prints "[[ 1. 1.]]"
c = np.full((2,2), 7) # Create a constant array
print(c)  # Prints "[[ 7. 7.]
          #          [ 7. 7.]]"
d = np.eye(2) # Create a 2x2 identity matrix
print(d)  # Prints "[[ 1. 0.]
          #          [ 0. 1.]]"
e = np.random.random((2,2)) # Create an array filled with random values
```

1. NumPy – Array Slicing

numpy arrays can be sliced. Since arrays may be multidimensional, you must specify a slice for each dimension of the array:

```
import numpy as np
# Create the following rank 2 array with shape (3, 4)
# [[ 1 2 3 4]
# [ 5 6 7 8]
# [ 9 10 11 12]]
a = np.array([[1,2,3,4], [5,6,7,8], [9,10,11,12]])
# Use slicing to pull out the subarray consisting of the first 2 rows
# and columns 1 and 2; b is the following array of shape (2, 2):
# [[2 3] # [6 7]]
b = a[:2, 1:3]
# A slice of an array is a view into the same data, so modifying it
# will modify the original array.
print(a[0, 1]) # Prints "2"
b[0, 0] = 77 # b[0, 0] is the same piece of data as a[0, 1]
print(a[0, 1]) # Prints "77"
```

1. NumPy – Datatypes

Every numpy array is a grid of elements of the same type. Numpy provides a large set of numeric datatypes that you can use to construct arrays. Numpy tries to guess a datatype when you create an array, but functions that construct arrays usually also include an optional argument to explicitly specify the datatype.

```
import numpy as np

x = np.array([1, 2]) # Let numpy choose the datatype
print(x.dtype) # Prints "int64"

x = np.array([1.0, 2.0]) # Let numpy choose the datatype
print(x.dtype) # Prints "float64"

x = np.array([1, 2], dtype=np.int64) # Force a particular datatype
print(x.dtype)
```

1. NumPy – Array Math

Basic mathematical functions operate elementwise on arrays, and are available both as operator overloads and as functions in the numpy module:

```
import numpy as np
x = np.array([[1,2],[3,4]], dtype=np.float64)
y = np.array([[5,6],[7,8]], dtype=np.float64)
```

```
# Elementwise sum; both produce the array # [[ 6.0 8.0]
                                                # [10.0 12.0]]
print(x + y)
print(np.add(x, y))
```

```
# Elementwise difference; both produce the array # [[-4.0 -4.0]
                                                    # [-4.0 -4.0]]
print(x - y)
print(np.subtract(x, y))
```

1. NumPy – Array Math

Basic mathematical functions operate elementwise on arrays, and are available both as operator overloads and as functions in the numpy module:

```
# Elementwise product; both produce the array # [[ 5.0 12.0]
                                                # [21.0 32.0]]
print(x * y)
print(np.multiply(x, y))
```

```
# Elementwise division; both produce the array # [[ 0.2 0.33333333]
                                                 # [ 0.42857143 0.5 ]]
print(x / y)
print(np.divide(x, y))
```

```
# Elementwise square root; produces the array
# [[ 1. 1.41421356]
# [ 1.73205081 2. ]]
print(np.sqrt(x))
```

NumPy for Matlab Users

References

NumPy

NumPy is based on Python, which was designed from the outset to be an excellent general-purpose programming language.

Matlab

MATLAB®'s scripting language was created for doing linear algebra. The syntax for basic matrix operations is nice and clean

https://docs.scipy.org/doc/numpy/user/numpy-for-matlab-users.html

2. SciPy

Numpy provides a high-performance multidimensional array and basic tools to compute with and manipulate these arrays. SciPy builds on this, and provides a large number of functions that operate on numpy arrays and are useful for different types of scientific and engineering application

The best way to get familiar with SciPy is to browse the documentation. We will highlight some parts of SciPy that you might find useful for this class.

https://docs.scipy.org/doc/scipy/reference/index.html

2. SciPy

Integration with Scipy – Numerical Integration

1. When we integrate any function where analytically integrate is not possible, we need to turn for numerical integration
2. SciPy provides functionality to integrate function with numerical integration.
3. scipy.integrate library has single integration, double, triple, multiple, Gaussian quadrate, Romberg, Trapezoidal and Simpson's rules.

https://docs.scipy.org/doc/scipy/reference/index.html

2. SciPy

Integration with Scipy – Numerical Integration

```
from scipy import integrate
# take f(x) function as f

f = lambda x : x**2
#single integration with a = 0 & b = 1
integration = integrate.quad(f, 0 , 1)
print(integration)
```

https://docs.scipy.org/doc/scipy/reference/index.html

1° Test intermedio

Manuale di sopravvivenza al laboratorio

4 esercizi

Argomenti lezioni
dalla 1 alla 3

Consegna via mail entro le ore 19:00
antonio.mirarchi@thinkopen.it

http://www.labmetodiemodelli.it/

Prossimi Appuntamenti

13 NOV	La Data Analysis & Data Visualization
20 NOV	Costruire Modelli Predittivi – P1
27 NOV	Costruire Modelli Predittivi – P2

http://www.labmetodiemodelli.it/