<https://www.unioviedo.es/compnum/compnum_NG/index.php/labs/>

ANACONDA ->Spyder (version 5)

Jupyter notebook

# -\*- coding: utf-8 -\*-

import numpy as np

#Create numpy arrays. Never create arrays in a loop. irst, allocate space and then put the numbers

a = np.array([1, 2, 3, 4])

b = np.array([(1.5, 2, 3), (4, 5, 6)])

#Create lists

a1=[1,2,3,4]

a2=[1.0,2,3,4]

#Create numpy aray of 0s

c = np.zeros((3, 4))

c1 = np.zeros(5)

#Create numpy aray of 1s

d = np.ones((2, 3))

d1 = np.ones(6)

#It starts in 1, ends in 10 (not included) with step 2

e = np.arange(1, 10, 2)

#Same but with floats 1.

f = np.arange(1., 10, 2)

#It starts in 1, ends in 9 (included) with 5 numbers equally distributed

g = np.linspace(1, 9, 5)

#%% Accessing elements and properties

import numpy as np

print('a\n',a,'\n')

print('b\n',b,'\n')

print('c\n',c,'\n')

print('d\n',d,'\n')

print('e\n',e,'\n')

print('f\n',f,'\n')

print('g\n',g)

#Numpy way: b[0,0] (one step). b[0][0] (2 steps)

print('a[0] = ', a[0], '\nb[0,0] = ',b[0,0], '\nb[0][0] = ',b[0][0])

#Access last term: -1

print('e[-1] = ',e[-1])

#shape: rows and columns

print('len a = ', len(a),'; dim b = ', b.ndim, '; shape b = ', b.shape)

#%% Operations

import numpy as np

a = np.array([1, 2, 3, 4])

b = np.array([(1.5, 2, 3, 5)])

a1 = [1, 2, 3, 4]

b1 = [1.5, 2, 3, 5]

#Add numpy arrays: element by element

print('a+b numpy array ')

print(a+b)

#Add lists: concatenatio of the 2 lists

print('\na1+b1 list')

print(a1+b1)

#Operations numpy arrays: element wise

print('a = \n', a)

print('\n3+a = \n', 3+a)

print('\n3\*a = \n', 3\*a)

print('\n\n3/a = \n', 3/a)

print('\na/2 = \n', a/2)

A = np.array( [[1, 1], [0, 1]] )

B = np.array( [[2, 3], [1, 4]] )

print('A')

print(A)

print('\nB')

print(B)

print('\nA\*B')

print(A\*B) #also element wise multiplication

print('\nAB')

print(np.dot(A,B)) #dot product

print(A @ B) #dot product

#%% Indexing

import numpy as np

a = np.arange(10) #same as np.arange(0,10,1) or np.arange(0,10)

print('a\n', a)

#access second element

print('\na[1]\n', a[1])

#access from second element to ninth element (not included)

print('\na[1:8]\n', a[1:8])

#access from second element to ninth element (not included) with step 2

print('\na[1:8:2]\n', a[1:8:2])

#access from second element to the end until the last one (included)

print('\na[1:]\n', a[1:])

#access from first element to the ninth (not included)

print('\na[:8]\n', a[:8])

#access from first element to the last one with step 2

print('\na[::2]\n', a[::2])

#access last element

print('\na[-1]\n', a[-1])

#access from first element to the last one (not included)

print('\na[:-1]\n', a[:-1])

#access from last element to the first (included)

print('\na[::-1]\n', a[::-1])

#access from last element to the first with step 2

print('\na[::-2]\n', a[::-2])

#access from eight element to the second with step 2

print('\na[7:1:-2]\n', a[7:1:-2])

b = np.array([0, 1, 5, -1])

#get elements with the positions that are stored in b

print('\na[b]\n',a[b])

n = 6

s = n\*n

a = np.arange(s)

a = np.reshape(a,(n,n))

print('a = \n', a)

#Acces one element

print('\033[91m \na[1,3] = \n', a[1,3])

#Access sixth column

print('\033[92m \na[:,5] = \n', a[:,5])

#Access fifth row

print('\033[94m \na[4,:] = \n', a[4,:])

#Access submatrix (from second row to fourth (not included) and from first column to the third (not included) )

print('\033[95m \na[1:3, 0:2] = \n', a[1:3, 0:2])

#Access first row from the second column to the sixth (not included)

print('\033[91m \na[0,1:5] = \n', a[0,1:5])

#Access submatrix: fifth row to last (included) and from the fifth column to the last (included)

print('\033[92m \na[4:,4:] = \n', a[4:,4:])

#Access submatrix: third row to last with step 2 and from the first column to the last with step 2

print('\033[94m \na[2::2,::2] = \n', a[2::2,::2])

#%% Copies

import numpy as np

a = np.arange(12)

#It's not a copy is a reference to the same array

b = a

print('a[0] = ', a[0], '\nb[0] = ',b[0])

b[0] = 10

print('a[0] = ', a[0], '\nb[0] = ',b[0])

#To do a copy

b = a.copy()

print('a[0] = ', a[0], '\nb[0] = ',b[0])

b[0] = 0

print('a[0] = ', a[0], '\nb[0] = ',b[0])

#%% Functions

import numpy as np

PI = 3.14

print(np.sin(PI/2))

print(np.exp(-1))

print(np.arctan(np.inf))

print(np.sqrt(4))

a = np.linspace(2,4,5)

print('a =\n', a)

print('\nnp.sqrt(a) =\n', np.sqrt(a))

f1 = lambda x: x \*\* 3

f2 = lambda x,y: x + y

print('f1(2) = ', f1(2))

print('f2(1,1) = ', f2(1,1))

def f3(x):

if x > 2:

return 0

else:

return 1

print('f3(-1) = ', f3(-1))

print('f3(3) = ', f3(3))

#%% Functions

import numpy as np

import matplotlib.pyplot as plt

x = np.linspace(-1,2) # Define the grid x from -1 to 2

f = lambda x : x\*\*3 - 2\*x\*\*2 + 1 # Define the function

OX = 0\*x #To make a line in the drawig at x=0

plt.figure()

plt.plot(x,f(x)) # Plot the function

plt.plot(x,OX,'k-') # Plot X axis

plt.xlabel('x')

plt.ylabel('y')

plt.title('function')

plt.show()

# -\*- coding: utf-8 -\*-

"""

Created on Thu Jan 25 19:15:57 2024

@author: uo294067

"""

#%%Exercise 1

import numpy as np

a = np.array([1, 3, 7])

b = np.array([(2, 4, 3), (0, 1, 6)])

c = np.ones(3)

d = np.zeros(4)

e = np.zeros((3, 2))

f = np.ones((3, 4))

#%%Exercise 2

import numpy as np

np.set\_printoptions(precision=1,suppress=True)

a = np.arange(7, 16, 2)

b = np.arange(10, 5, -1)

c = np.arange(15, -1, -5)

#as floats

a1 = np.linspace(7, 15, 5)

b1 = np.linspace(10, 6, 5)

c1 = np.linspace(15, 0, 4)

d=np.linspace(0, 1, 11)

e=np.linspace(-1, 1, 10)

f=np.arange(1, 2.1, 0.1)

#%%Exercise 3

import numpy as np

v = np.arange(0., 12.2, 1.1)

vi = v[::-1]

v1=v[::2]

v2=v[1::2]

v11=v[::3]

v21=v[1::3]

v31=v[2::3]

v12=v[::4]

v22=v[1::4]

v321=v[2::4]

v4=v[3::4]

#%%Exercise 4

import numpy as np

a = np.array([1, 2, 3])

b1 = np.append(a,0)

b1 =b1[::-1]

b1 = np.append(b1,0)

b1 =b1[::-1]

b2= np.zeros(5)

b2[1:4:1]=b2[1:4:1]+a

c=np.array([0])

b3=np.concatenate((a,c), axis=None)

b3=np.concatenate((c,b3), axis=None)

#%%Exercise 5

import numpy as np

A=np.array([(2,1,3,4),(9,8,5,7),(6,-1,-2,-8),(-5,-7,-9,-6)])

a=A[:,0]

b=A[2,:]

c=A[0:2:1,0:2:1]

d=A[2::,2::]

e=A[1:3:1,1:3:1]

f=A[::,1::]

g=A[1::,1:3:1]

#%%Exercise 6

import numpy as np

def f1(x):

return x\*np.exp(x)

f=f1(2)

def g1(z):

return z/(np.sin(z)\*np.cos(z))

g=g1(np.pi/4)

def h1(x,y):

return (x\*y)/(x\*\*2+y\*\*2)

h=h1(2,4)

#%%Exercise 7

import numpy as np

import matplotlib.pyplot as plt

x = np.linspace(-2\*np.pi,2\*np.pi) # Define the grid x from -6 to 6

f = lambda x : x\*np.sin(3\*x) # Define the function

OX = 0\*x #To make a line in the drawig at x=0

plt.figure()

plt.plot(x,f(x)) # Plot the function

plt.plot(x,OX,'k-') # Plot X axis

plt.xlabel('x')

plt.ylabel('y')

plt.title('x sin(3x)')

plt.show()