Task 1: >> conda info

active environment : base

active env location : /Users/wangwuhao/opt/anaconda3

shell level : 1

user config file : /Users/wangwuhao/.condarc

populated config files : /Users/wangwuhao/.condarc

conda version : 4.10.3

conda-build version : 3.21.5

python version : 3.9.7.final.0

virtual packages : \_\_osx=10.16=0

\_\_unix=0=0

\_\_archspec=1=x86\_64

base environment : /Users/wangwuhao/opt/anaconda3 (writable)

conda av data dir : /Users/wangwuhao/opt/anaconda3/etc/conda

conda av metadata url : None

channel URLs : https://repo.anaconda.com/pkgs/main/osx-64

https://repo.anaconda.com/pkgs/main/noarch

https://repo.anaconda.com/pkgs/r/osx-64

https://repo.anaconda.com/pkgs/r/noarch

package cache : /Users/wangwuhao/opt/anaconda3/pkgs

/Users/wangwuhao/.conda/pkgs

envs directories : /Users/wangwuhao/opt/anaconda3/envs

/Users/wangwuhao/.conda/envs

platform : osx-64

user-agent : conda/4.10.3 requests/2.26.0 CPython/3.9.7 Darwin/21.6.0 OSX/10.16

UID:GID : 502:20

netrc file : /Users/wangwuhao/.netrc

offline mode : False

Task 2:

>>> from numpy import \*

>>> import scipy.linalg

>>> a, b, c, d, x = [identity(5) for i in range(5)]

>>>

>>> v = ones((5))

>>> m, n = 1, 1

>>> print(ndim(a) );

2

>>> print(size(a) );

25

>>> print(shape(a) );

(5, 5)

>>> print(a.shape[n-1] );

5

>>> print(block([[a, b], [c, d]]))

[[1. 0. 0. 0. 0. 1. 0. 0. 0. 0.]

[0. 1. 0. 0. 0. 0. 1. 0. 0. 0.]

[0. 0. 1. 0. 0. 0. 0. 1. 0. 0.]

[0. 0. 0. 1. 0. 0. 0. 0. 1. 0.]

[0. 0. 0. 0. 1. 0. 0. 0. 0. 1.]

[1. 0. 0. 0. 0. 1. 0. 0. 0. 0.]

[0. 1. 0. 0. 0. 0. 1. 0. 0. 0.]

[0. 0. 1. 0. 0. 0. 0. 1. 0. 0.]

[0. 0. 0. 1. 0. 0. 0. 0. 1. 0.]

[0. 0. 0. 0. 1. 0. 0. 0. 0. 1.]]

>>> print(array([[1.,2.,3.], [4.,5.,6.]]) );

[[1. 2. 3.]

[4. 5. 6.]]

>>> print(a[-1] );

[0. 0. 0. 0. 1.]

>>> print(a[1,4] );

0.0

>>> print(a[1] );

[0. 1. 0. 0. 0.]

>>> print(a[0:5] );

[[1. 0. 0. 0. 0.]

[0. 1. 0. 0. 0.]

[0. 0. 1. 0. 0.]

[0. 0. 0. 1. 0.]

[0. 0. 0. 0. 1.]]

>>>

>>> print(a[-5:] );

[[1. 0. 0. 0. 0.]

[0. 1. 0. 0. 0.]

[0. 0. 1. 0. 0.]

[0. 0. 0. 1. 0.]

[0. 0. 0. 0. 1.]]

>>> print(a[0:3][:,4:9] );

[[0.]

[0.]

[0.]]

>>>

>>> print(a[ix\_([1,3,4],[0,2])] );

[[0. 0.]

[0. 0.]

[0. 0.]]

>>>

>>> print(a[::2,:] );

[[1. 0. 0. 0. 0.]

[0. 0. 1. 0. 0.]

[0. 0. 0. 0. 1.]]

>>>

>>> print(a[::-1,:] );

[[0. 0. 0. 0. 1.]

[0. 0. 0. 1. 0.]

[0. 0. 1. 0. 0.]

[0. 1. 0. 0. 0.]

[1. 0. 0. 0. 0.]]

>>>

>>> print(a[r\_[:len(a),0]] );

[[1. 0. 0. 0. 0.]

[0. 1. 0. 0. 0.]

[0. 0. 1. 0. 0.]

[0. 0. 0. 1. 0.]

[0. 0. 0. 0. 1.]

[1. 0. 0. 0. 0.]]

>>>

>>> print(a.transpose() );

[[1. 0. 0. 0. 0.]

[0. 1. 0. 0. 0.]

[0. 0. 1. 0. 0.]

[0. 0. 0. 1. 0.]

[0. 0. 0. 0. 1.]]

>>>

>>> print(a.conj().transpose() );

[[1. 0. 0. 0. 0.]

[0. 1. 0. 0. 0.]

[0. 0. 1. 0. 0.]

[0. 0. 0. 1. 0.]

[0. 0. 0. 0. 1.]]

>>>

>>> print(a.dot(b) );

[[1. 0. 0. 0. 0.]

[0. 1. 0. 0. 0.]

[0. 0. 1. 0. 0.]

[0. 0. 0. 1. 0.]

[0. 0. 0. 0. 1.]]

>>>

>>> print(a \* b );

[[1. 0. 0. 0. 0.]

[0. 1. 0. 0. 0.]

[0. 0. 1. 0. 0.]

[0. 0. 0. 1. 0.]

[0. 0. 0. 0. 1.]]

>>>

>>> print(a/b );

<stdin>:1: RuntimeWarning: invalid value encountered in true\_divide

[[ 1. nan nan nan nan]

[nan 1. nan nan nan]

[nan nan 1. nan nan]

[nan nan nan 1. nan]

[nan nan nan nan 1.]]

>>>

>>> print(a\*\*3 );

[[1. 0. 0. 0. 0.]

[0. 1. 0. 0. 0.]

[0. 0. 1. 0. 0.]

[0. 0. 0. 1. 0.]

[0. 0. 0. 0. 1.]]

>>>

>>> print((a>0.5) );

[[ True False False False False]

[False True False False False]

[False False True False False]

[False False False True False]

[False False False False True]]

>>>

>>> print(nonzero(a>0.5) );

(array([0, 1, 2, 3, 4]), array([0, 1, 2, 3, 4]))

>>>

>>> print(a[:,nonzero(v>0.5)[0]] );

[[1. 0. 0. 0. 0.]

[0. 1. 0. 0. 0.]

[0. 0. 1. 0. 0.]

[0. 0. 0. 1. 0.]

[0. 0. 0. 0. 1.]]

>>>

>>> print(a[:,v.T>0.5])

[[1. 0. 0. 0. 0.]

[0. 1. 0. 0. 0.]

[0. 0. 1. 0. 0.]

[0. 0. 0. 1. 0.]

[0. 0. 0. 0. 1.]]

>>>

>>> a[a<0.5]=0

>>>

>>> print(a \* (a>0.5) );

[[1. 0. 0. 0. 0.]

[0. 1. 0. 0. 0.]

[0. 0. 1. 0. 0.]

[0. 0. 0. 1. 0.]

[0. 0. 0. 0. 1.]]

>>>

>>> a[:] = 3

>>>

>>> y = x.copy()

>>>

>>> y = x[1,:].copy()

>>>

>>> y = x.flatten()

>>>

>>> print(arange(1.,11.) );

[ 1. 2. 3. 4. 5. 6. 7. 8. 9. 10.]

>>>

>>> print(r\_[1.:11.] );

[ 1. 2. 3. 4. 5. 6. 7. 8. 9. 10.]

>>>

>>> print(arange(10.) );

[0. 1. 2. 3. 4. 5. 6. 7. 8. 9.]

>>>

>>> print(r\_[:10.] );

[0. 1. 2. 3. 4. 5. 6. 7. 8. 9.]

>>>

>>> print(r\_[:9:10j] );

[0. 1. 2. 3. 4. 5. 6. 7. 8. 9.]

>>>

>>> print(arange(1.,11.)[:, newaxis] );

[[ 1.]

[ 2.]

[ 3.]

[ 4.]

[ 5.]

[ 6.]

[ 7.]

[ 8.]

[ 9.]

[10.]]

>>>

>>> print(zeros((3,4)) );

[[0. 0. 0. 0.]

[0. 0. 0. 0.]

[0. 0. 0. 0.]]

>>>

>>> print(zeros((3,4,5)) );

[[[0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0.]]

[[0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0.]]

[[0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0.]

[0. 0. 0. 0. 0.]]]

>>>

>>> print(ones((3,4)) );

[[1. 1. 1. 1.]

[1. 1. 1. 1.]

[1. 1. 1. 1.]]

>>>

>>> print(eye(3) );

[[1. 0. 0.]

[0. 1. 0.]

[0. 0. 1.]]

>>>

>>> print(diag(a) );

[3. 3. 3. 3. 3.]

>>>

>>> print(diag(a,0) );

[3. 3. 3. 3. 3.]

>>>

>>> print(random.rand(3,4) );

[[0.97590863 0.03782004 0.79426969 0.3578826 ]

[0.74796395 0.91450931 0.37266242 0.96488347]

[0.08138577 0.04245099 0.29679603 0.36370363]]

>>>

>>> print(linspace(1,3,4) );

[1. 1.66666667 2.33333333 3. ]

>>>

>>> print(mgrid[0:9.,0:6.] );

[[[0. 0. 0. 0. 0. 0.]

[1. 1. 1. 1. 1. 1.]

[2. 2. 2. 2. 2. 2.]

[3. 3. 3. 3. 3. 3.]

[4. 4. 4. 4. 4. 4.]

[5. 5. 5. 5. 5. 5.]

[6. 6. 6. 6. 6. 6.]

[7. 7. 7. 7. 7. 7.]

[8. 8. 8. 8. 8. 8.]]

[[0. 1. 2. 3. 4. 5.]

[0. 1. 2. 3. 4. 5.]

[0. 1. 2. 3. 4. 5.]

[0. 1. 2. 3. 4. 5.]

[0. 1. 2. 3. 4. 5.]

[0. 1. 2. 3. 4. 5.]

[0. 1. 2. 3. 4. 5.]

[0. 1. 2. 3. 4. 5.]

[0. 1. 2. 3. 4. 5.]]]

>>>

>>> print(meshgrid(r\_[0:9.],r\_[0:6.]) );

[array([[0., 1., 2., 3., 4., 5., 6., 7., 8.],

[0., 1., 2., 3., 4., 5., 6., 7., 8.],

[0., 1., 2., 3., 4., 5., 6., 7., 8.],

[0., 1., 2., 3., 4., 5., 6., 7., 8.],

[0., 1., 2., 3., 4., 5., 6., 7., 8.],

[0., 1., 2., 3., 4., 5., 6., 7., 8.]]), array([[0., 0., 0., 0., 0., 0., 0., 0., 0.],

[1., 1., 1., 1., 1., 1., 1., 1., 1.],

[2., 2., 2., 2., 2., 2., 2., 2., 2.],

[3., 3., 3., 3., 3., 3., 3., 3., 3.],

[4., 4., 4., 4., 4., 4., 4., 4., 4.],

[5., 5., 5., 5., 5., 5., 5., 5., 5.]])]

>>>

>>> print(ogrid[0:9.,0:6.] );

[array([[0.],

[1.],

[2.],

[3.],

[4.],

[5.],

[6.],

[7.],

[8.]]), array([[0., 1., 2., 3., 4., 5.]])]

>>>

>>> print(ix\_(r\_[0:9.],r\_[0:6.]) );

(array([[0.],

[1.],

[2.],

[3.],

[4.],

[5.],

[6.],

[7.],

[8.]]), array([[0., 1., 2., 3., 4., 5.]]))

>>>

>>> print(meshgrid([1,2,4],[2,4,5]) );

[array([[1, 2, 4],

[1, 2, 4],

[1, 2, 4]]), array([[2, 2, 2],

[4, 4, 4],

[5, 5, 5]])]

>>>

>>> print(ix\_([1,2,4],[2,4,5]) );

(array([[1],

[2],

[4]]), array([[2, 4, 5]]))

>>>

>>> print(tile(a, (m, n)) );

[[3. 3. 3. 3. 3.]

[3. 3. 3. 3. 3.]

[3. 3. 3. 3. 3.]

[3. 3. 3. 3. 3.]

[3. 3. 3. 3. 3.]]

>>>

>>> print(concatenate((a,b),1) );

[[3. 3. 3. 3. 3. 1. 0. 0. 0. 0.]

[3. 3. 3. 3. 3. 0. 1. 0. 0. 0.]

[3. 3. 3. 3. 3. 0. 0. 1. 0. 0.]

[3. 3. 3. 3. 3. 0. 0. 0. 1. 0.]

[3. 3. 3. 3. 3. 0. 0. 0. 0. 1.]]

>>>

>>> print(concatenate((a,b)) );

[[3. 3. 3. 3. 3.]

[3. 3. 3. 3. 3.]

[3. 3. 3. 3. 3.]

[3. 3. 3. 3. 3.]

[3. 3. 3. 3. 3.]

[1. 0. 0. 0. 0.]

[0. 1. 0. 0. 0.]

[0. 0. 1. 0. 0.]

[0. 0. 0. 1. 0.]

[0. 0. 0. 0. 1.]]

>>>

>>> print(a.max() );

3.0

>>>

>>> print(a.max(0) );

[3. 3. 3. 3. 3.]

>>>

>>> print(a.max(1) );

[3. 3. 3. 3. 3.]

>>>

>>> print(maximum(a, b) );

[[3. 3. 3. 3. 3.]

[3. 3. 3. 3. 3.]

[3. 3. 3. 3. 3.]

[3. 3. 3. 3. 3.]

[3. 3. 3. 3. 3.]]

>>>

>>> print(sqrt(dot(v,v)) );

2.23606797749979

>>>

>>> a, b, c, d, x = [identity(5) for i in range(5)]

>>>

>>> print(logical\_and(a,b) );

[[ True False False False False]

[False True False False False]

[False False True False False]

[False False False True False]

[False False False False True]]

>>>

>>> print(logical\_or(a,b) );

[[ True False False False False]

[False True False False False]

[False False True False False]

[False False False True False]

[False False False False True]]

>>>

>>> print(linalg.inv(a) );

[[1. 0. 0. 0. 0.]

[0. 1. 0. 0. 0.]

[0. 0. 1. 0. 0.]

[0. 0. 0. 1. 0.]

[0. 0. 0. 0. 1.]]

>>>

>>> print(linalg.pinv(a) );

[[1. 0. 0. 0. 0.]

[0. 1. 0. 0. 0.]

[0. 0. 1. 0. 0.]

[0. 0. 0. 1. 0.]

[0. 0. 0. 0. 1.]]

>>>

>>> print(linalg.matrix\_rank(a) );

5

>>>

>>> print(linalg.solve(a,b) );

[[1. 0. 0. 0. 0.]

[0. 1. 0. 0. 0.]

[0. 0. 1. 0. 0.]

[0. 0. 0. 1. 0.]

[0. 0. 0. 0. 1.]]

>>>

>>> print(linalg.cholesky(a).T );

[[1. 0. 0. 0. 0.]

[0. 1. 0. 0. 0.]

[0. 0. 1. 0. 0.]

[0. 0. 0. 1. 0.]

[0. 0. 0. 0. 1.]]

>>>

>>> D,V = linalg.eig(a)

>>>

>>> Q,R = scipy.linalg.qr(a)

>>> P, L, U = scipy.linalg.lu(a)

>>> P, L, U

(array([[0., 0., 0., 0., 1.],

[0., 1., 0., 0., 0.],

[0., 0., 1., 0., 0.],

[0., 0., 0., 1., 0.],

[1., 0., 0., 0., 0.]]), array([[1., 0., 0., 0., 0.],

[0., 1., 0., 0., 0.],

[0., 0., 1., 0., 0.],

[0., 0., 0., 1., 0.],

[0., 0., 0., 0., 1.]]), array([[1., 1., 1., 1., 1.],

[0., 0., 0., 0., 0.],

[0., 0., 0., 0., 0.],

[0., 0., 0., 0., 0.],

[0., 0., 0., 0., 0.]]))

>>> print(sort(a))

[[0. 0. 0. 0. 1.]

[0. 0. 0. 0. 1.]

[0. 0. 0. 0. 1.]

[0. 0. 0. 0. 1.]

[0. 0. 0. 0. 1.]]

>>> np.unique(a)

array([0., 1.])

>>> a.squeeze()

array([[1., 0., 0., 0., 0.],

[0., 1., 0., 0., 0.],

[0., 0., 1., 0., 0.],

[0., 0., 0., 1., 0.],

[0., 0., 0., 0., 1.]])

Task 3:

>>> import matplotlib.pyplot as plt

>>> plt.plot([1,2,3,4], [1,2,7,14])

[<matplotlib.lines.Line2D object at 0x7fee386b15e0>]

>>> plt.axis([0, 6, 0, 20])

(0.0, 6.0, 0.0, 20.0)

>>> plt.show()

图表, 折线图

描述已自动生成

Task 4:

>>> plt.plot(sin(linspace(0, 6)))

>>> plt.show()

图表, 直方图

描述已自动生成

Task 5:

My github account is wuhao .

Task 6: