NumPy_ExamplesAll

September 10, 2026

1 Matrix multiplication in Python

[1]:
```
A = [[1, -2, 4],
     [2, 0, -1]]

B = [[-1, -2],
     [0, 1],
     [2, 3]]
```

[2]:
```
def mat_mul(A, B):
  rowsA, colsA = len(A), len(A[0])
  rowsB, colsB = len(B), len(B[0])

  assert colsA == rowsB

  C = []
  for i in range(rowsA):
    row = []
    for j in range(colsB):
      cij = 0
      for p in range(colsA):
        cij += A[i][p] * B[p][j]
      row.append(cij)
    C.append(row)

  return C

# Fingers crossed ...
C = mat_mul(A, B)
print(C)
```

```
[[7, 8], [-4, -7]]
```

1.1 NumPy version

[3]:
```
import numpy as np

a = np.array(
    [[1, -2, 4],
     [2, 0, -1]])
b = np.array(
    [[-1, -2],
     [0, 1],
     [2, 3]])
a.dot(b)
```

[3]:
```
array([[ 7,  8],
       [-4, -7]])
```

[4]:
```
a @ b
```

[4]:
```
array([[ 7,  8],
       [-4, -7]])
```

[5]:
```
d = np.array(
    [[-2, 1, 0],
     [0, 1, -1]])
print(a, '\n', d)
```

```
[[ 1 -2  4]
 [ 2  0 -1]]
 [[-2  1  0]
 [ 0  1 -1]]
```

[6]:
```
a + d
```

[6]:
```
array([[-1, -1,  4],
       [ 2,  1, -2]])
```

[7]:
```
a @ d
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[7], line 1
----> 1 a @ d

ValueError: matmul: Input operand 1 has a mismatch in its core dimension 0, wit
  ↪gufunc signature (n?,k),(k,m?)->(n?,m?) (size 2 is different from 3)
```

[8]:
```
print(a)
```

```
[[ 1 -2  4]
 [ 2  0 -1]]
```

[9]:
```
print(d)
```

```
[[-2  1  0]
 [ 0  1 -1]]
```

[10]:
```
a * d
```

[10]:
```
array([[-2, -2,  0],
       [ 0,  0,  1]])
```

[11]:
```
l1 = [1 ,2 ,3]
l2 = l1
l1[0] = -1
print(l1)
print(l2)
```

```
[-1, 2, 3]
[-1, 2, 3]
```

---

2 NumPy Tutorial: Examples

[1]:
```
import numpy as np

np.ndarray
```

[1]:
```
numpy.ndarray
```

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

[2]:
```
array([1, 2, 3, 4])
```

[3]:
```
a.shape
```

[3]:
```
(4,)
```

[4]:
```
a.ndim
```

[4]:
```
1
```

[5]:
```
len(a.shape)
```

[5]:
```
1
```

[6]:
```
a = np.array(list(range(4)))
a
```

[6]:
```
array([0, 1, 2, 3])
```

[7]:
```
a + 1
```

[7]:
```
array([1, 2, 3, 4])
```

[8]:
```
a = np.arange(4) + 1
a
```

[8]:
```
array([1, 2, 3, 4])
```

[9]:
```
b = np.arange(8)
b
```

[9]:
```
array([0, 1, 2, 3, 4, 5, 6, 7])
```

[10]:
```
b.shape
```

[10]:
```
(8,)
```

[11]:
```
b.reshape(4, 2)
```

[11]:
```
array([[0, 1],
       [2, 3],
       [4, 5],
       [6, 7]])
```

[12]:
```
b.reshape(4, 2, order = 'F')
```

[12]:
```
array([[0, 4],
       [1, 5],
       [2, 6],
       [3, 7]])
```

[13]:
```
[0 for i in range(5)]
```

[13]:
```
[0, 0, 0, 0, 0]
```

[14]:
```
z = np.zeros(5, dtype = np.float16)
z
```

[14]:
```
array([0., 0., 0., 0., 0.], dtype=float16)
```

[15]:
```
z.dtype
```

[15]:
```
dtype('float16')
```

[16]:
```
np.zeros(5).astype(int)
```

[16]:
```
array([0, 0, 0, 0, 0])
```

[17]:
```
b
```

[17]:
```
array([0, 1, 2, 3, 4, 5, 6, 7])
```

[18]:
```
c = b.astype(float)
c
```

[18]:
```
array([0., 1., 2., 3., 4., 5., 6., 7.])
```

[19]:
```
b
```

[19]:
```
array([0, 1, 2, 3, 4, 5, 6, 7])
```

[20]:
```
bb = b.reshape(4, 2)
bb
```

[20]:
```
array([[0, 1],
       [2, 3],
       [4, 5],
       [6, 7]])
```

[21]:
```
bb[-1,-1] = 10
bb
```

[21]:
```
array([[ 0,  1],
       [ 2,  3],
       [ 4,  5],
       [ 6, 10]])
```

[22]:
```
b
```

[22]:
```
array([ 0,  1,  2,  3,  4,  5,  6, 10])
```

[23]:
```
a
```

[23]:
```
array([1, 2, 3, 4])
```

[24]:
```
bb
```

[24]:
```
array([[ 0,  1],
       [ 2,  3],
       [ 4,  5],
       [ 6, 10]])
```

[25]:
```
np.column_stack((a, bb))
```

[25]:
```
array([[ 1,  0,  1],
       [ 2,  2,  3],
       [ 3,  4,  5],
       [ 4,  6, 10]])
```

[26]:
```
bb
```

[26]:
```
array([[ 0,  1],
       [ 2,  3],
       [ 4,  5],
       [ 6, 10]])
```

[27]:
```
bb.ndim
```

[27]:
```
2
```

[28]:
```
bnew = np.column_stack((np.ones(bb.shape[0]), bb))
```

[29]:
```
bnew
```

[29]:
```
array([[ 1.,  0.,  1.],
       [ 1.,  2.,  3.],
       [ 1.,  4.,  5.],
       [ 1.,  6., 10.]])
```

[30]:
```
data = np.loadtxt('data.txt')
```

[31]:
```
data.shape
```

[31]:
```
(2, 4)
```

[32]:
```
data
```

[32]:
```
array([[ 1. ,  0.5,  2.1, -1.7],
       [-1. ,  0.7,  3.1,  2.7]])
```

[33]:
```
y = data[:, 0]
y
```

[33]:
```
array([ 1., -1.])
```

[34]:
```
X = data[:,1:]
X
```

[34]:
```
array([[ 0.5,  2.1, -1.7],
       [ 0.7,  3.1,  2.7]])
```

[35]:
```
b
```

[35]:
```
array([ 0,  1,  2,  3,  4,  5,  6, 10])
```

[36]:
```
bb
```

[36]:
```
array([[ 0,  1],
       [ 2,  3],
       [ 4,  5],
       [ 6, 10]])
```

[37]:
```
bnew
```

[37]:
```
array([[ 1.,  0.,  1.],
       [ 1.,  2.,  3.],
       [ 1.,  4.,  5.],
       [ 1.,  6., 10.]])
```

[38]:
```
a
```

[38]:
```
array([1, 2, 3, 4])
```

[39]:
```
bb.dot(a)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[39], line 1
----> 1 bb.dot(a)

ValueError: shapes (4,2) and (4,) not aligned: 2 (dim 1) != 4 (dim 0)
```

[40]:
```
a.dot(bb)
```

[40]:
```
array([40, 62])
```

[41]:
```
a.reshape(1, 4)
```

[41]:
```
array([[1, 2, 3, 4]])
```

[42]:
```
a
```

[42]:
```
array([1, 2, 3, 4])
```

[43]:
```
a.reshape(4, 1)
```

[43]:
```
array([[1],
       [2],
       [3],
       [4]])
```

[44]:
```
aa = a.reshape(4, -1)
aa
```

[44]:
```
array([[1],
       [2],
       [3],
       [4]])
```

[45]:
```
bnew
```

[45]:
```
array([[ 1.,  0.,  1.],
       [ 1.,  2.,  3.],
       [ 1.,  4.,  5.],
       [ 1.,  6., 10.]])
```

[46]:
```
aa.shape
```

[46]:
```
(4, 1)
```

[47]:
```
bnew.shape
```

[47]:
```
(4, 3)
```

[48]:
```
bnew.T @ aa
```

[48]:
```
array([[10.],
       [40.],
       [62.]])
```

[49]:
```
aa
```

[49]:
```
array([[1],
       [2],
       [3],
       [4]])
```

[50]:
```
a
```

[50]:
```
array([1, 2, 3, 4])
```

[51]:
```
bnew
```

[51]:
```
array([[ 1.,  0.,  1.],
       [ 1.,  2.,  3.],
       [ 1.,  4.,  5.],
       [ 1.,  6., 10.]])
```

[52]:
```
aa
```

[52]:
```
array([[1],
       [2],
       [3],
       [4]])
```

[53]:
```
aa.T
```

[53]:
```
array([[1, 2, 3, 4]])
```

[54]:
```
a
```

[54]:
```
array([1, 2, 3, 4])
```

[55]:
```
bnew[:,0]
```

[55]:
```
array([1., 1., 1., 1.])
```

[56]:
```
bnew[:,0] * a
```

[56]:
```
array([1., 2., 3., 4.])
```

[57]:
```
bnew[:,0].dot(a)
```

[57]:
```
np.float64(10.0)
```

[58]:
```
bnew * aa
```

[58]:
```
array([[ 1.,  0.,  1.],
       [ 2.,  4.,  6.],
       [ 3., 12., 15.],
       [ 4., 24., 40.]])
```

[59]:
```
b = np.arange(8).reshape(4,-1)
b
```

[59]:
```
array([[0, 1],
       [2, 3],
       [4, 5],
       [6, 7]])
```

[60]:
```
a
```

[60]:
```
array([1, 2, 3, 4])
```

[61]:
```
b.ravel()
```

[61]:
```
array([0, 1, 2, 3, 4, 5, 6, 7])
```

[62]:
```
b.ravel(order = 'F')
```

[62]:
```
array([0, 2, 4, 6, 1, 3, 5, 7])
```

[63]:
```
b.ravel(order = 'C')
```

[63]:
```
array([0, 1, 2, 3, 4, 5, 6, 7])
```

[64]:
```
v = b.ravel(order = 'C')
v
```

[64]:
```
array([0, 1, 2, 3, 4, 5, 6, 7])
```

[65]:
```
v.reshape(4, -1)
```

[65]:
```
array([[0, 1],
       [2, 3],
       [4, 5],
       [6, 7]])
```

[66]:
```
v.reshape(-1, 2)
```

[66]:
```
array([[0, 1],
       [2, 3],
       [4, 5],
       [6, 7]])
```

[160]:
```
d = np.array([2, 2, 2, 2])
d[0] = 1
d[2] = -1
d
```

[160]:
```
array([ 1,  2, -1,  2])
```

[68]:
```
from numpy import newaxis
```

[161]:
```
d[:,newaxis] # Add a new, second dimension
```

[161]:
```
array([[ 1],
       [ 2],
       [-1],
       [ 2]])
```

[162]:
```
d[:,newaxis].shape
```

[162]:
```
(4, 1)
```

[163]:
```
d[newaxis,:].shape # Add a new, first dimension
```

[163]:
```
(1, 4)
```

```
[165]: d[newaxis,:]
```

```
[165]: array([[ 1,  2, -1,  2]])
```

```
[164]: d.reshape(4,-1)
```

```
[164]: array([[ 1],
              [ 2],
              [-1],
              [ 2]])
```

```
[73]: np.eye(10)
```

```
[73]: array([[1., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
              [0., 1., 0., 0., 0., 0., 0., 0., 0., 0.],
              [0., 0., 1., 0., 0., 0., 0., 0., 0., 0.],
              [0., 0., 0., 1., 0., 0., 0., 0., 0., 0.],
              [0., 0., 0., 0., 1., 0., 0., 0., 0., 0.],
              [0., 0., 0., 0., 0., 1., 0., 0., 0., 0.],
              [0., 0., 0., 0., 0., 0., 1., 0., 0., 0.],
              [0., 0., 0., 0., 0., 0., 0., 1., 0., 0.],
              [0., 0., 0., 0., 0., 0., 0., 0., 1., 0.],
              [0., 0., 0., 0., 0., 0., 0., 0., 0., 1.]])
```

```
[74]: np.eye(10) + 10
```

```
[74]: array([[11., 10., 10., 10., 10., 10., 10., 10., 10., 10.],
              [10., 11., 10., 10., 10., 10., 10., 10., 10., 10.],
              [10., 10., 11., 10., 10., 10., 10., 10., 10., 10.],
              [10., 10., 10., 11., 10., 10., 10., 10., 10., 10.],
              [10., 10., 10., 10., 11., 10., 10., 10., 10., 10.],
              [10., 10., 10., 10., 10., 11., 10., 10., 10., 10.],
              [10., 10., 10., 10., 10., 10., 11., 10., 10., 10.],
              [10., 10., 10., 10., 10., 10., 10., 11., 10., 10.],
              [10., 10., 10., 10., 10., 10., 10., 10., 11., 10.],
              [10., 10., 10., 10., 10., 10., 10., 10., 10., 11.]])
```

```
[75]: np.ones((4, 2))
```

```
[75]: array([[1., 1.],
              [1., 1.],
              [1., 1.],
              [1., 1.]])
```

```
[76]: np.ones((4, 2)) * 2
```

```
[76]: array([[2., 2.],
              [2., 2.],
```

```
       [2., 2.],
       [2., 2.]])
```

[77]:
```
np.sum(a)
```

[77]:
```
np.int64(10)
```

[78]:
```
a.sum()
```

[78]:
```
np.int64(10)
```

[79]:
```
a.max()
```

[79]:
```
np.int64(4)
```

[80]:
```
a.min()
```

[80]:
```
np.int64(1)
```

[81]:
```
b.sum()
```

[81]:
```
np.int64(28)
```

[82]:
```
b.sum(axis = 0)
```

[82]:
```
array([12, 16])
```

[83]:
```
b.sum(axis = 1)
```

[83]:
```
array([ 1,  5,  9, 13])
```

[84]:
```
b.max(axis = 1)
```

[84]:
```
array([1, 3, 5, 7])
```

[85]:
```
b.argmax(axis = 1)
```

[85]:
```
array([1, 1, 1, 1])
```

[86]:
```
b.max()
```

[86]:
```
np.int64(7)
```

[87]:
```
b.max(axis = 1)
```

[87]:
```
array([1, 3, 5, 7])
```

[88]:
```
b.argmax(axis = 1)
```

[88]:
```
array([1, 1, 1, 1])
```

[89]:
```
a
```

[89]:
```
array([1, 2, 3, 4])
```

[90]:
```
b
```

[90]:
```
array([[0, 1],
       [2, 3],
       [4, 5],
       [6, 7]])
```

[91]:
```
np.maximum(a,b)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[91], line 1
----> 1 np.maximum(a,b)

ValueError: operands could not be broadcast together with shapes (4,) (4,2)
```

[92]:
```
np.maximum(a[:,newaxis], b)
```

[92]:
```
array([[1, 1],
       [2, 3],
       [4, 5],
       [6, 7]])
```

[93]:
```
a
```

[93]:
```
array([1, 2, 3, 4])
```

[94]:
```
a[:2]
```

[94]:
```
array([1, 2])
```

[95]:
```
b
```

[95]:
```
array([[0, 1],
       [2, 3],
       [4, 5],
       [6, 7]])
```

[96]:
```
b.shape
```

[96]:
```
(4, 2)
```

[97]:
```
a[:2].shape
```

[97]:
```
(2,)
```

[98]:
```
np.maximum(a[:2], b)
```

[98]:
```
array([[1, 2],
       [2, 3],
       [4, 5],
       [6, 7]])
```

[99]:
```
a[:2] * b
```

[99]:
```
array([[ 0,  2],
       [ 2,  6],
       [ 4, 10],
       [ 6, 14]])
```

[100]:
```
a[:2] + b
```

[100]:
```
array([[1, 3],
       [3, 5],
       [5, 7],
       [7, 9]])
```

[101]:
```
b / a[:2]
```

[101]:
```
array([[0. , 0.5],
       [2. , 1.5],
       [4. , 2.5],
       [6. , 3.5]])
```

[102]:
```
b
```

[102]:
```
array([[0, 1],
       [2, 3],
       [4, 5],
       [6, 7]])
```

[103]:
```
b.mean()
```

[103]:
```
np.float64(3.5)
```

[104]:
```
b.sum() / b.size
```

[104]:
```
np.float64(3.5)
```

[105]:
```
b.mean(axis = 0)
```

[105]:
```
array([3., 4.])
```

[106]:
```
b.mean(axis = 1)
```

[106]:
```
array([0.5, 2.5, 4.5, 6.5])
```

[107]:
```
b.std()
```

[107]:
```
np.float64(2.29128784747792)
```

[108]:
```
b.std(axis = 0)
```

[108]:
```
array([2.23606798, 2.23606798])
```

[109]:
```
a
```

[109]:
```
array([1, 2, 3, 4])
```

[110]:
```
np.column_stack((a, a))
```

[110]:
```
array([[1, 1],
       [2, 2],
       [3, 3],
       [4, 4]])
```

[111]:
```
np.vstack((a, a))
```

[111]:
```
array([[1, 2, 3, 4],
       [1, 2, 3, 4]])
```

[112]:
```
np.stack((a, a))
```

[112]:
```
array([[1, 2, 3, 4],
       [1, 2, 3, 4]])
```

[113]:
```
np.stack((a, a), axis = 0)
```

[113]:
```
array([[1, 2, 3, 4],
       [1, 2, 3, 4]])
```

[114]:
```
np.stack((a, a), axis = 1)
```

[114]:
```
array([[1, 1],
       [2, 2],
       [3, 3],
       [4, 4]])
```

[115]:
```
b
```

[115]:
```
array([[0, 1],
       [2, 3],
       [4, 5],
       [6, 7]])
```

[116]:
```
np.split(b, 2)
```

[116]:
```
[array([[0, 1],
        [2, 3]]),
 array([[4, 5],
        [6, 7]])]
```

[117]:
```
np.split(b, 2, axis = 1)
```

[117]:
```
[array([[0],
        [2],
        [4],
        [6]]),
 array([[1],
        [3],
        [5],
        [7]])]
```

[118]:
```
[b1, b2] = np.split(b, 2)
```

[119]:
```
b
```

[119]:
```
array([[0, 1],
       [2, 3],
       [4, 5],
       [6, 7]])
```

[120]:
```
b1
```

[120]:
```
array([[0, 1],
       [2, 3]])
```

[121]:
```
b2
```

[121]:
```
array([[4, 5],
       [6, 7]])
```

[122]:
```
b1[0,0] = -1
```

[123]:
```
b1
```

[123]:
```
array([[-1,  1],
       [ 2,  3]])
```

[124]:
```
b2
```

[124]:
```
array([[4, 5],
       [6, 7]])
```

[125]:
```
b
```

[125]:
```
array([[-1,  1],
       [ 2,  3],
       [ 4,  5],
       [ 6,  7]])
```

2.1 Linear algebra module

[136]:
```
from numpy import linalg as la
```

[137]:
```
a
```

[137]:
```
array([[ 1,  4, -1],
       [-2, -3,  2],
       [ 3, -4,  7]])
```

[138]:
```
la.norm(a)
```

[138]:
```
np.float64(10.44030650891055)
```

[148]:
```
np.sum(a*a)
```

[148]:
```
np.int64(109)
```

[149]:
```
np.sqrt(np.sum(a*a))
```

[149]:
```
np.float64(10.44030650891055)
```

[151]:
```
a.dot(a)
```

[151]:
```
array([[-10,  -4,   0],
       [ 10,  -7,  10],
       [ 32,  -4,  38]])
```

[152]:
```
np.sqrt(a.dot(a))
```

```
/var/folders/xy/z2pc35sn14x0stwqc5jsvytm0000gp/T/ipykernel_22193/2042561907.py:1
: RuntimeWarning: invalid value encountered in sqrt
  np.sqrt(a.dot(a))
```

[152]:
```
array([[       nan,        nan, 0.        ],
       [3.16227766,        nan, 3.16227766],
       [5.65685425,        nan, 6.164414  ]])
```

[153]:
```
a.dot(a)
```

[153]:
```
array([[-10,  -4,   0],
       [ 10,  -7,  10],
       [ 32,  -4,  38]])
```

[154]:
```
b
```

[154]:
```
array([[-1,  1],
       [ 2,  3],
       [ 4,  5],
       [ 6,  7]])
```

[155]:
```
b[0,0] = 0
b
```

[155]:
```
array([[0, 1],
       [2, 3],
       [4, 5],
       [6, 7]])
```

[156]:
```
la.norm(b)
```

[156]:
```
np.float64(11.832159566199232)
```

[157]:
```
la.norm(b, axis = 0)
```

[157]:
```
array([7.48331477, 9.16515139])
```

[158]:
```
la.norm(b, axis = 1)
```

[158]:
```
array([1.        , 3.60555128, 6.40312424, 9.21954446])
```

[159]:
```
a
```

[159]:
```
array([[ 1,  4, -1],
       [-2, -3,  2],
       [ 3, -4,  7]])
```

[166]:
```
a[:, newaxis].shape
```

[166]:
```
(3, 1, 3)
```

[167]:
```
aa = a[:, newaxis]
aa
```

[167]:
```
array([[[ 1,  4, -1]],
```

```
       [[-2, -3,  2]],

       [[ 3, -4,  7]]])
```

[168]:
```
aa @ aa.T
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[168], line 1
----> 1 aa @ aa.T

ValueError: matmul: Input operand 1 has a mismatch in its core dimension 0, wit
  ↪gufunc signature (n?,k),(k,m?)->(n?,m?) (size 1 is different from 3)
```

[169]:
```
np.outer(a, a)
```

[169]:
```
array([[  1,   4,  -1,  -2,  -3,   2,   3,  -4,   7],
       [  4,  16,  -4,  -8, -12,   8,  12, -16,  28],
       [ -1,  -4,   1,   2,   3,  -2,  -3,   4,  -7],
       [ -2,  -8,   2,   4,   6,  -4,  -6,   8, -14],
       [ -3, -12,   3,   6,   9,  -6,  -9,  12, -21],
       [  2,   8,  -2,  -4,  -6,   4,   6,  -8,  14],
       [  3,  12,  -3,  -6,  -9,   6,   9, -12,  21],
       [ -4, -16,   4,   8,  12,  -8, -12,  16, -28],
       [  7,  28,  -7, -14, -21,  14,  21, -28,  49]])
```

[ ]:
```
la.matrix_rank(np.outer(a,a))
```

[170]:
```
c = np.random.randint(0, 10, (4, 4))
c
```

[170]:
```
array([[8, 7, 5, 4],
       [4, 2, 5, 9],
       [8, 5, 7, 5],
       [0, 7, 6, 7]])
```

[171]:
```
la.matrix_rank(c)
```

[171]:
```
np.int64(4)
```

[172]:
```
la.eig(np.eye(3))
```

[172]:
```
EigResult(eigenvalues=array([1.+0.j, 1.+0.j, 1.+0.j]),
eigenvectors=array([[1.+0.j, 0.+0.j, 0.+0.j],
       [0.+0.j, 1.+0.j, 0.+0.j],
       [0.+0.j, 0.+0.j, 1.+0.j]]))
```

[173]:
```
c
```

[173]:
```
array([[8, 7, 5, 4],
       [4, 2, 5, 9],
       [8, 5, 7, 5],
       [0, 7, 6, 7]])
```

[174]:
```
np.trace(c)
```

[174]:
```
np.int64(24)
```

[175]:
```
qr = la.qr(c)
qr
```

[175]:
```
QRResult(Q=array([[-0.66666667,  0.1696731 ,  0.64052768,  0.34130762],
       [-0.33333333, -0.12339862, -0.6645109 ,  0.65733319],
       [-0.66666667, -0.10797379, -0.30827222, -0.66997422],
       [-0.        ,  0.97176411, -0.23047297, -0.05056409]]), R=array([[-12.
,  -8.66666667,  -9.66666667,  -9.        ],
       [  0.        ,   7.20339426,   5.30614053,   5.83058465],
       [  0.        ,   0.        ,  -3.66065954,  -6.57315936],
       [  0.        ,   0.        ,   0.        ,   3.5774095 ]]))
```

[ ]:
```
Q = qr[0]
R = qr[1]
```

[ ]:
```
Q
```

[ ]:
```
R
```

[ ]:
```
Q @ R
```

[ ]:
```
Q.dot(R)
```

[ ]:
```
Q[0]
```

[ ]:
```
Q[0] @ Q[0]
```

[ ]:
```
Q[0] @ Q[1]
```

[ ]:
```
np.round(Q[0] @ Q[1])
```

[ ]:
```
from numpy import linalg as la
```

[ ]:
```
np.linalg.det(c)
```

[ ]:
```
la.solve
```

[ ]:
```
la.inv(c)
```

[176]:
```
# Solve the system of equations x0 + 2 * x1 = 1 and 3 * x0 + 5 * x1 = 2
a = np.array([[1, 2], [3, 5]])
b = np.array([1, 2])
x = np.linalg.solve(a, b)
x
```

[176]:
```
array([-1.,  1.])
```

[177]:
```
a @ x
```

[177]:
```
array([1., 2.])
```

[178]:
```
la.inv(a) @ b
```

[178]:
```
array([-1.,  1.])
```

[12]:
```
import numpy as np
import numpy.linalg as la

A = np.arange(4) + 1
A
```

[12]:
```
array([1, 2, 3, 4])
```

[13]:
```
A = (np.arange(4) + 1).reshape(2, -1)
A
```

[13]:
```
array([[1, 2],
       [3, 4]])
```

[14]:
```
A = np.arange(4).reshape(2, -1) + 1
A
```

[14]:
```
array([[1, 2],
       [3, 4]])
```

[17]:
```
la.inv(A)
```

[17]:
```
array([[-2. ,  1. ],
       [ 1.5, -0.5]])
```

[18]:
```
np.diag
```

[18]:
```
<function diag at 0x107eef6f0>
```

[19]:
```
np.diag(np.ones(3))
```

[19]:
```
array([[1., 0., 0.],
       [0., 1., 0.],
       [0., 0., 1.]])
```

[22]:
```
np.eye(2) @ A
```

[22]:
```
array([[1., 2.],
       [3., 4.]])
```

[ ]:
```

```

3 Examples from September 10, 2026

[129]:
```
import numpy as np

a = np.array([[1, 4, -1], [-2, -3, 2], [3, -4, 7]])
a
```

[129]:
```
array([[ 1,  4, -1],
       [-2, -3,  2],
       [ 3, -4,  7]])
```

[131]:
```
import numpy.linalg as la

la.trace(a)
```

[131]:
```
np.int64(5)
```

[133]:
```
np.eye(*a.shape)
```

[133]:
```
array([[1., 0., 0.],
       [0., 1., 0.],
       [0., 0., 1.]])
```

[134]:
```
a * np.eye(*a.shape)
```

[134]:
```
array([[ 1.,  0., -0.],
       [-0., -3.,  0.],
       [ 0., -0.,  7.]])
```

[135]:
```
np.sum(a * np.eye(*a.shape))
```

[135]:
```
np.float64(5.0)
```

[139]:
```
la.norm(a)
```

[139]:
```
np.float64(10.44030650891055)
```

[140]:
```
a
```

[140]:
```
array([[ 1,  4, -1],
       [-2, -3,  2],
       [ 3, -4,  7]])
```

[141]:
```
a * a
```

[141]:
```
array([[ 1, 16,  1],
       [ 4,  9,  4],
       [ 9, 16, 49]])
```

[144]:
```
np.sum(a * a, axis = 0)
```

[144]:
```
array([14, 41, 54])
```

[145]:
```
np.sum(a * a, axis = 1)
```

[145]:
```
array([18, 17, 74])
```

[147]:
```
np.sqrt(np.sum(a * a))
```

[147]:
```
np.float64(10.44030650891055)
```

[179]:
```
a
```

[179]:
```
array([[1, 2],
       [3, 5]])
```

[184]:
```
a = np.hstack((a, np.ones(2).reshape(2, -1)))
```

[185]:
```
a
```

[185]:
```
array([[1., 2., 1.],
       [3., 5., 1.]])
```

[186]:
```
a[-1, -1] = -1
a
```

[186]:
```
array([[ 1.,  2.,  1.],
       [ 3.,  5., -1.]])
```

[187]:
```
v = np.array([2, 3, 1])
v
```

[187]:
```
array([2, 3])
```

[188]:
```
a + v
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[188], line 1
----> 1 a + v

ValueError: operands could not be broadcast together with shapes (2,3) (2,)
```

[189]:
```
v = np.array([2, 3, 1])
v
```

[189]:
```
array([2, 3, 1])
```

[190]:
```
a + v
```

[190]:
```
array([[3., 5., 2.],
       [5., 8., 0.]])
```

[191]:
```
a
```

[191]:
```
array([[ 1.,  2.,  1.],
       [ 3.,  5., -1.]])
```

[192]:
```
v
```

[192]:
```
array([2, 3, 1])
```

[193]:
```
a + v
```

[193]:
```
array([[3., 5., 2.],
       [5., 8., 0.]])
```

[194]:
```
a.shape
```

[194]:
```
(2, 3)
```

[195]:
```
v.shape
```

[195]:
```
(3,)
```

[ ]: