5ELEN018W - Tutorial 3 2026 Solutions

[7]:
```
import math
import numpy as np
from scipy import linalg, optimize
import matplotlib.pyplot as plt
from spatialmath import *
from spatialmath.base import *
from spatialmath.base import sym
from spatialgeometry import *
from roboticstoolbox import *

import pprint
```

Exercise 1

[16]:
```
def ex1(theta, units):
    if units == 'deg':
        return rot2(math.radians(theta))
    else:
        return rot2(theta)

print(ex1(math.pi, 'rad'))
print(ex1(180, 'deg'))
```

```
[[-1.0000000e+00 -1.2246468e-16]
 [ 1.2246468e-16 -1.0000000e+00]]
[[-1.0000000e+00 -1.2246468e-16]
 [ 1.2246468e-16 -1.0000000e+00]]
```

Exercise 2

[17]:
```
list1 = [[1, 2], [3, 4]]
list2 = [[2, 3], [4, 5]]
np.array(list1) @ np.array(list2)
```

[17]:
```
array([[10, 13],
       [22, 29]])
```

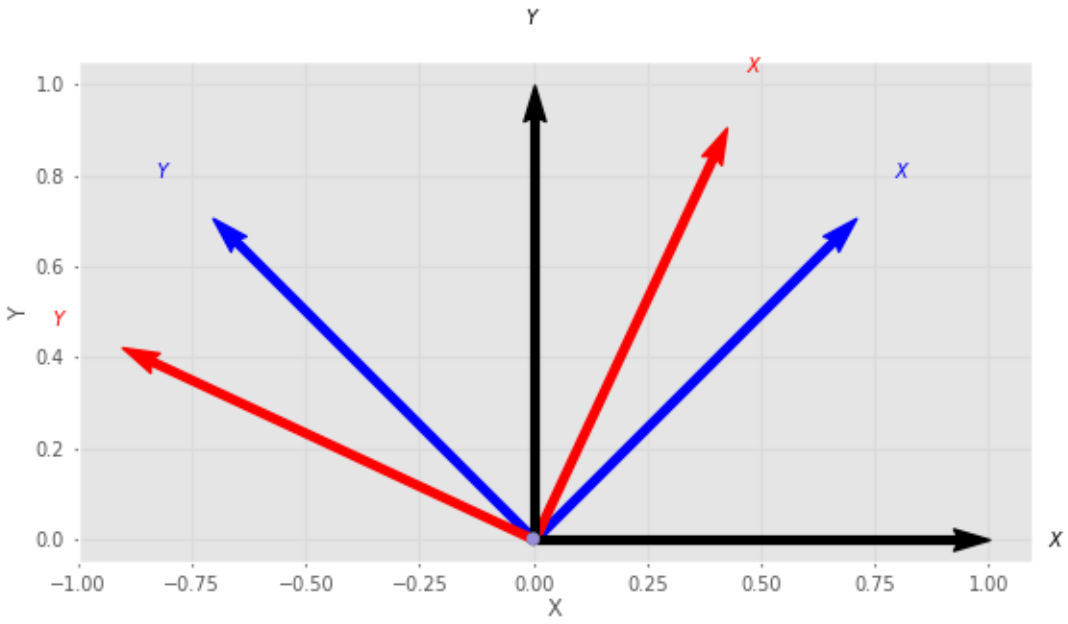
Exercise 3

[28]:
```
def ex3(q1, q2):
    e1 = trot2(math.radians(q1))
    e2 = e1@trot2(math.radians(q2))

    trplot2(e1, color='b')
    trplot2(e2, color='r')

    # reference frame
    e0 = trot2(0)
    trplot2(e0, color='k')

ex3(45, 20)
```



Exercise 4

[40]:
```
from math import *

def ex4(theta, tr_list):
    s = [[cos(theta), -sin(theta), tr_list[0]],
         [sin(theta), cos(theta),  tr_list[1]],
         [0,              0,          1]]

    return np.array(s)
```

```
print(ex4(math.pi, [1, 2]))

# equivalent with the robotics toolbox
print()
print(transl2(1,2) @ trot2(math.pi))
```

```
[[-1.0000000e+00 -1.2246468e-16  1.0000000e+00]
 [ 1.2246468e-16 -1.0000000e+00  2.0000000e+00]
 [ 0.0000000e+00  0.0000000e+00  1.0000000e+00]]

[[-1.0000000e+00 -1.2246468e-16  1.0000000e+00]
 [ 1.2246468e-16 -1.0000000e+00  2.0000000e+00]
 [ 0.0000000e+00  0.0000000e+00  1.0000000e+00]]
```

Exercise 5

[43]:
```
B = transl(20, 10, 0) @ trotz(math.radians(70))
P = [3.1, 8, 2, 1] # augment P Euclidean coordinates with an '1'
B@P
```

[43]:
```
array([13.54272148, 15.64920827,  2.        ,  1.        ])
```