Harold_test

November 23, 2018

In [1]:
```
!pip install harold
```

```
Requirement already satisfied: harold in /opt/conda/lib/python3.6/site-packages (1.0.1)
Requirement already satisfied: scipy in /opt/conda/lib/python3.6/site-packages (from harold) (1.
Requirement already satisfied: numpy in /opt/conda/lib/python3.6/site-packages (from harold) (1.
Requirement already satisfied: tabulate in /opt/conda/lib/python3.6/site-packages (from harold)
Requirement already satisfied: matplotlib in /opt/conda/lib/python3.6/site-packages (from harold
Requirement already satisfied: six>=1.10 in /opt/conda/lib/python3.6/site-packages (from matplot
Requirement already satisfied: python-dateutil in /opt/conda/lib/python3.6/site-packages (from m
Requirement already satisfied: pytz in /opt/conda/lib/python3.6/site-packages (from matplotlib->
Requirement already satisfied: cycler>=0.10 in /opt/conda/lib/python3.6/site-packages (from matp
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=1.5.6 in /opt/conda/lib/python
```

In [2]:
```
import harold as ha
import numpy as np
```

In [3]:
```
# MIMO transfer function matrix
tau1=75.0
tau2=7.5

com_den = [tau1*tau2, tau1+tau2, 1]
g11 = +8.78 *np.array([10,1])
g12 = -8.64 *np.array([12,1])
g21 = +10.82*np.array([12,1])
g22 = -10.96*np.array([10,1])
num = [[g11, g12],
       [g21, g22]]

Gtf = ha.Transfer(num,com_den)

# Conversion to State Space
Gss = ha.transfer_to_state(Gtf)
```

In [4]:
```
# State Space obtained from same transfer function matrix using Python Control
A = np.array([[-3.90570952e-18, 5.95578818e-19, 2.42861287e-17, 1.77777778e-02],
              [ 1.89662142e-18,-4.33422699e-18, 1.77777778e-02, 1.38777878e-17],
              [ 0.00000000e+00,-1.00000000e-01,-1.46666667e-01, 3.77302356e-17],
```

```
                    [-1.00000000e-01, 2.77555756e-17, 6.93889390e-17,-1.46666667e-01]])

        B = np.array([[-0.15608889,  0.1536    ],
                      [ 0.19235556, -0.19484444],
                      [-0.23082667,  0.19484444],
                      [ 0.15608889, -0.18432   ]])

        C = np.array([[  0.00000000e+00,   0.00000000e+00,   0.00000000e+00, 1.00000000e+00],
                      [  0.00000000e+00,   0.00000000e+00,  -1.00000000e+00, 1.11022302e-16]])

        D = np.array([[ 0.,  0.],
                      [ 0.,  0.]])
```

In [5]:
```
        # A Matrix from Harold
        Gss.a
```

Out[5]:
```
array([[ 0.        ,  1.        ,  0.        ,  0.        ],
       [-0.00177778, -0.14666667,  0.        ,  0.        ],
       [ 0.        ,  0.        ,  0.        ,  1.        ],
       [ 0.        ,  0.        , -0.00177778, -0.14666667]])
```

In [6]:
```
        # A Matrix from Python Control
        np.round(A,4)
```

Out[6]:
```
array([[-0.    ,  0.    ,  0.    ,  0.0178],
       [ 0.    , -0.    ,  0.0178,  0.    ],
       [ 0.    , -0.1   , -0.1467,  0.    ],
       [-0.1   ,  0.    ,  0.    , -0.1467]])
```

In [7]:
```
        # Define a function to evaluate State Space
        from numpy.linalg import solve
        def evalfr(s,A,B,C,D):
            A = np.asmatrix(A)
            B = np.asmatrix(B)
            C = np.asmatrix(C)
            D = np.asmatrix(D)
            return np.asarray(C * solve(s * np.eye(len(A)) - A, B) + D)
```

In [8]:
```
        # result from python control State Space Realization
        np.round(evalfr(0,A,B,C,D),4)
```

Out[8]:
```
array([[  8.78,  -8.64],
       [ 10.82, -10.96]])
```

In [9]:
```
        # result Harold transfer function Matrix
        Rtf, __ = ha.frequency_response(Gtf,0)
        Rtf[:,:,0]
```

Out[9]:
```
array([[  8.78+0.j,  -8.64+0.j],
       [ 10.82+0.j, -10.96+0.j]])
```

In [10]:
```
# result from Harold State Space Realization
Rss, __ = ha.frequency_response(Gss,0)
Rss[:,:,0]
```

Out[10]:
```
array([[  87.80-0.j, -103.68+0.j],
       [ 129.84-0.j, -109.60+0.j]])
```

In [11]:
```
from scipy.linalg import eigvals
def tzero(A,B,C,D):

    M = np.block([[A, B], [C, D]])

    Ig = np.block([[np.eye(len(A)), np.zeros(B.shape)],
                   [np.zeros(C.shape), np.zeros(D.shape)]])

    z = eigvals(M, Ig)

    return z[np.isfinite(z)]
```

In [12]:
```
# Transmission zeros from the python control State Space Realization
tzero(A,B,C,D)
```

Out[12]:
```
array([ 0.00785742+0.j, -0.09096890+0.j])
```

In [13]:
```
# Transmission zeros from Harold transfer function matrix
Gtf.zeros
```

Out[13]:
```
array([ -10.99276767,  127.26798633])
```

In [14]:
```
# Transmission zeros from Harold State Space Realization
Gss.zeros
```

Out[14]:
```
array([ -10.99276767,  127.26798633])
```

In [15]:
```
# Confirming result for Harold State Space Realization Transmission zeros
tzero(Gss.a,Gss.b,Gss.c,Gss.d)
```

Out[15]:
```
array([ 127.26798633+0.j,  -10.99276767+0.j])
```

In [16]:
```
# Poles from the python control State Space Realization
np.linalg.eigvals(A)
```

Out[16]:
```
array([-0.01333333, -0.13333333, -0.13333333, -0.01333333])
```

In [17]:
```
# Poles from Harold transfer function matrix
Gtf.poles
```

Out[17]:
```
array([-0.01333333+0.j, -0.13333333+0.j, -0.01333333+0.j, -0.13333333+0.j])
```

In [18]:
```
# Poles from Harold State Space Realization
Gss.poles
```

Out[18]:
```
array([-0.01333333+0.j, -0.13333333+0.j, -0.01333333+0.j, -0.13333333+0.j])
```