Implementing noise in MATLAB and Simulink

```
>> noise

mean1 =

    0.5109

var1 =

    0.0836

mean2 =

    0.0298

var2 =

    0.9785
```

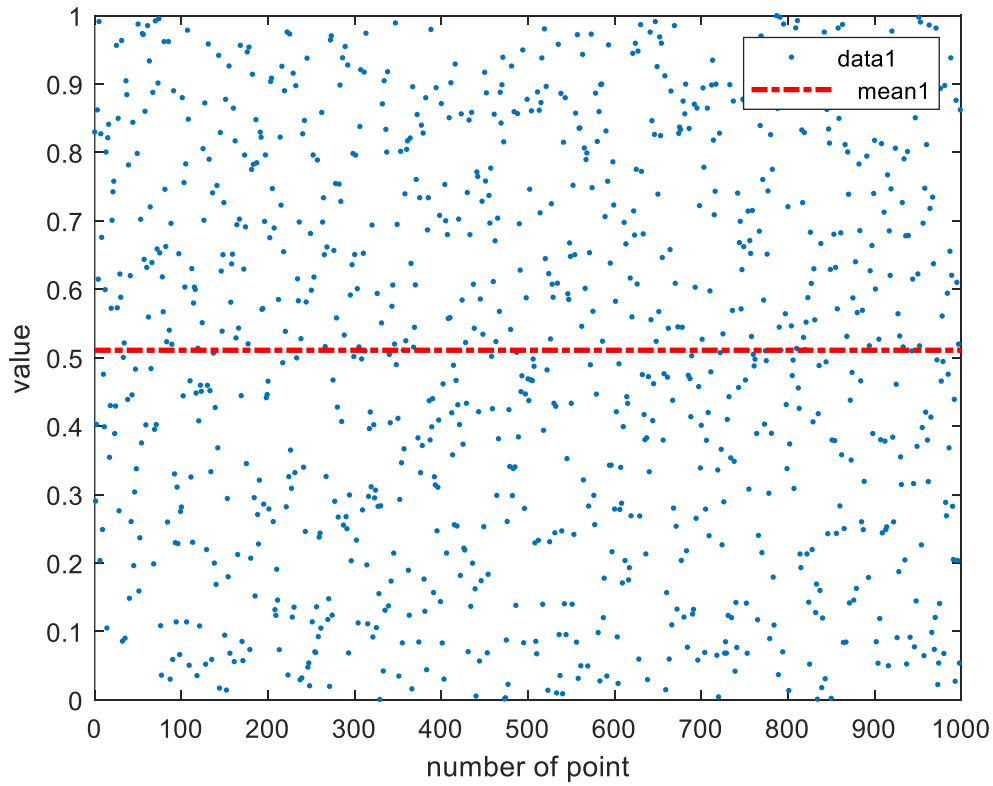
Data Statistics - 1

Statistics for data1

Check to plot statistics on figure:

	X		Y	
min	0	☐	0.0006587	☐
max	999	☐	0.9998	☐
mean	499.5	☐	0.5109	☐
median	499.5	☐	0.5168	☐
mode	0	☐	0.0006587	☐
std	288.8	☐	0.2892	☐
range	999		0.9991	

Save to workspace... Help Close



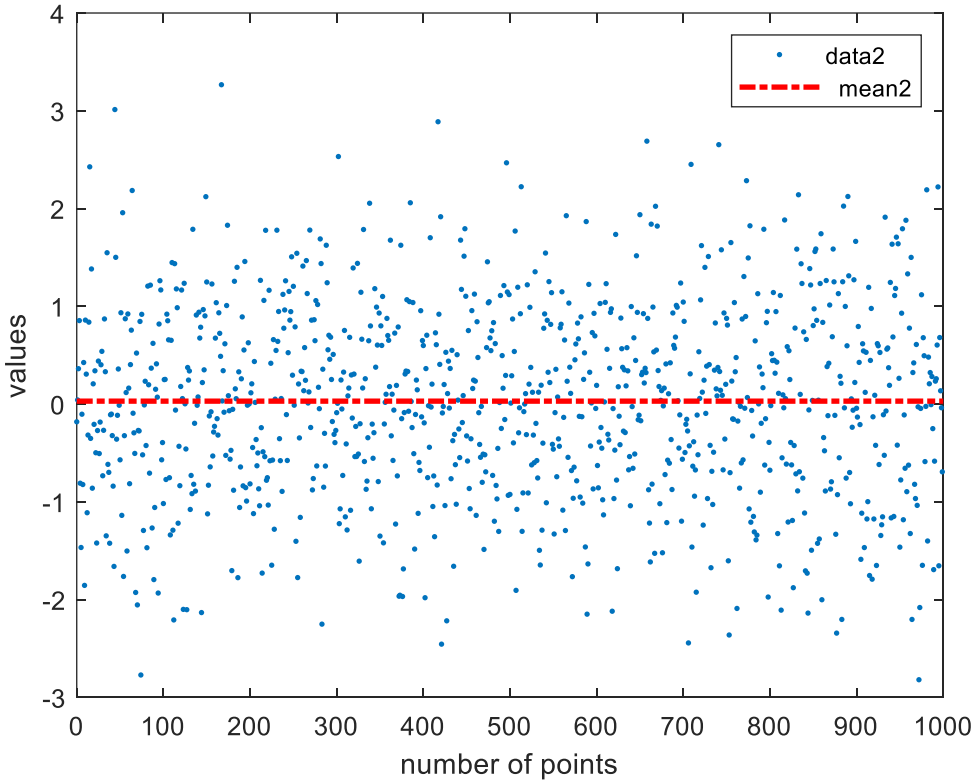
Data Statistics - 2

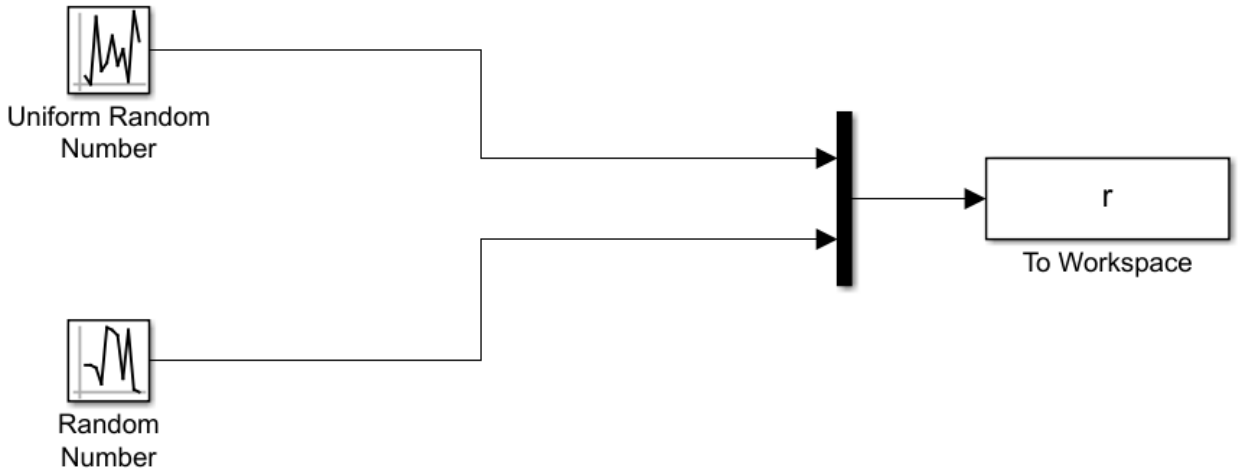
Statistics for data1

Check to plot statistics on figure:

	X		Y	
min	0	☐	-2.82	☐
max	999	☐	3.266	☐
mean	499.5	☐	0.02983	☐
median	499.5	☐	0.04009	☐
mode	0	☐	-2.82	☐
std	288.8	☐	0.9892	☐
range	999		6.086	

Save to workspace... Help Close





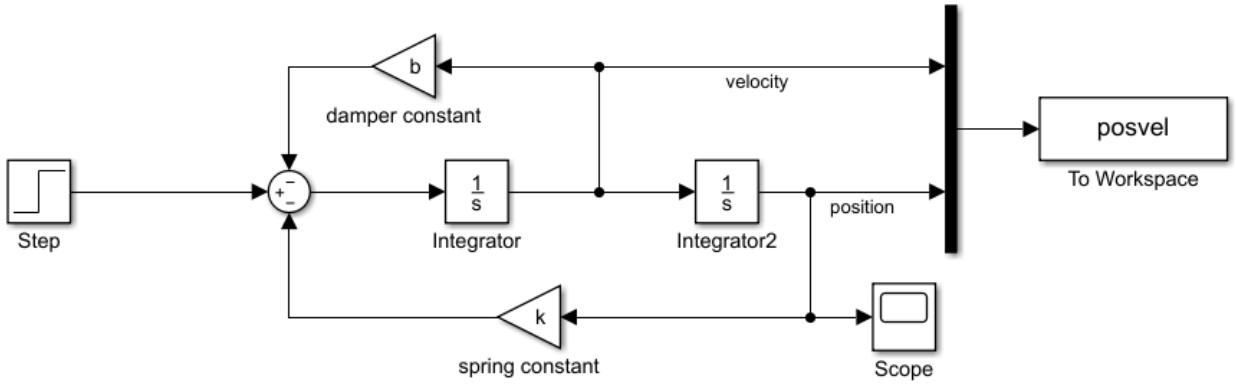

```
>> mean3=mean(r(:,1))

mean3 =

    0.0032

>> var3=var(r(:,1))

var3 =

    0.3158
```

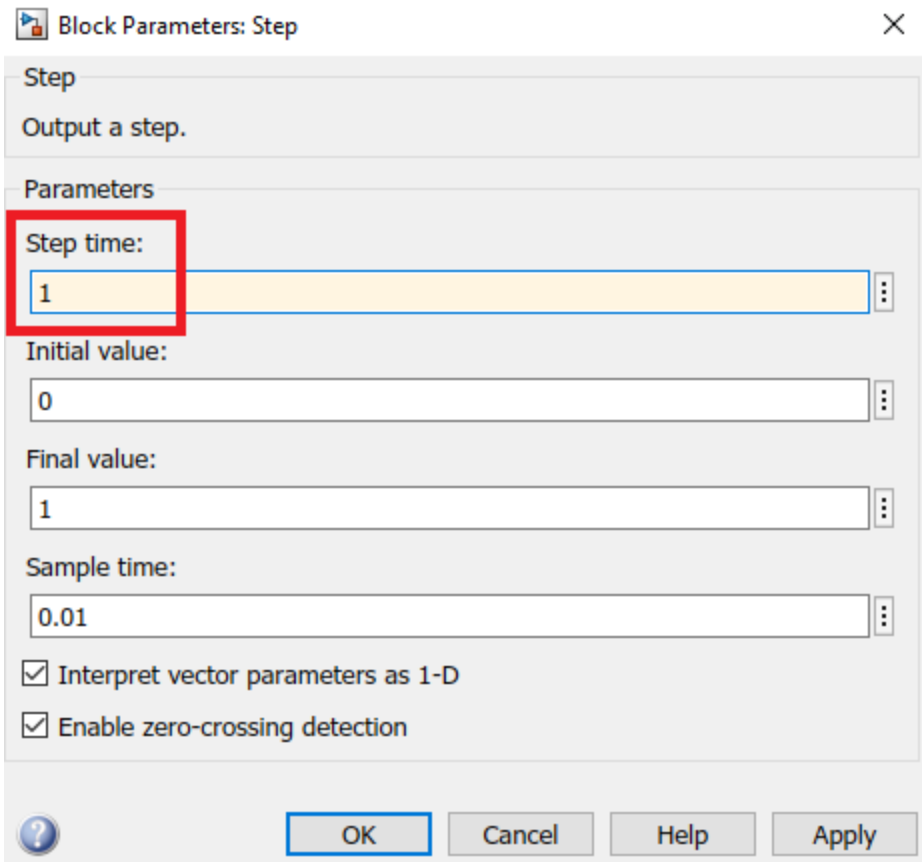
```
>> mean4=mean(r(:,2))

mean4 =

   -0.0139

>> var4=var(r(:,2))

var4 =

    1.0621
```

A basic spring damper system in Simulink

Commands to define fixed parameters:

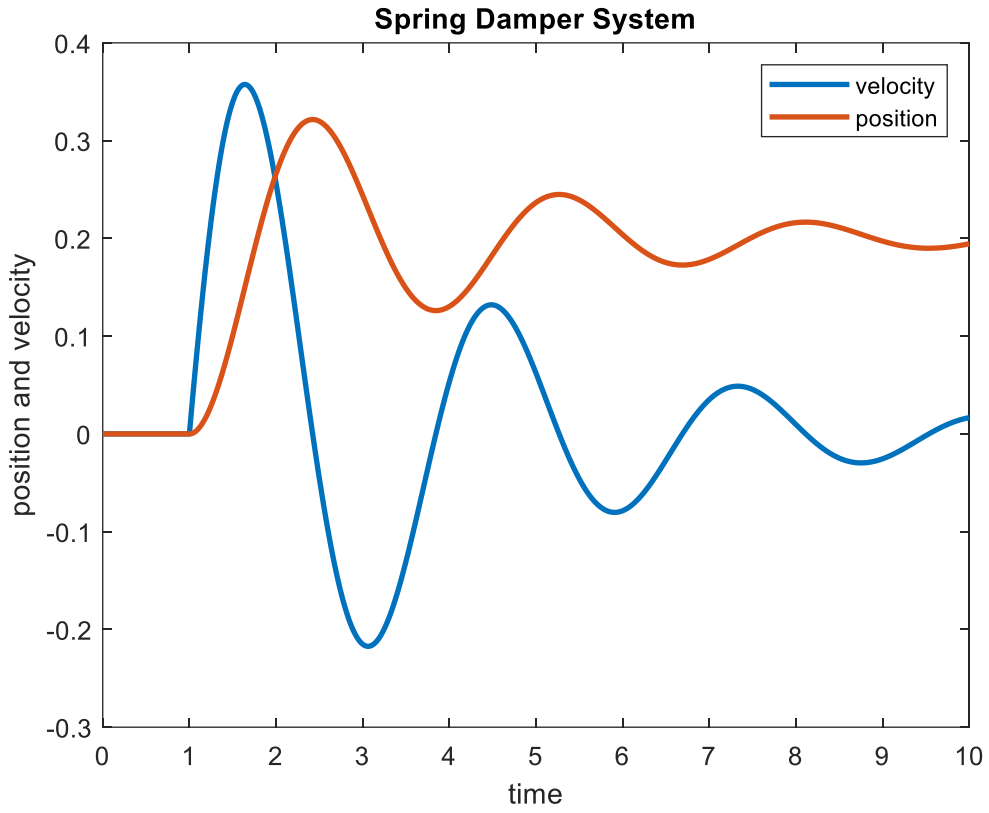
```
b=0.7;
k=5;
t=0:0.01:10;
A=[0 1;-k -b];
B=[0;1];
C=[1 0];
D=0;
```

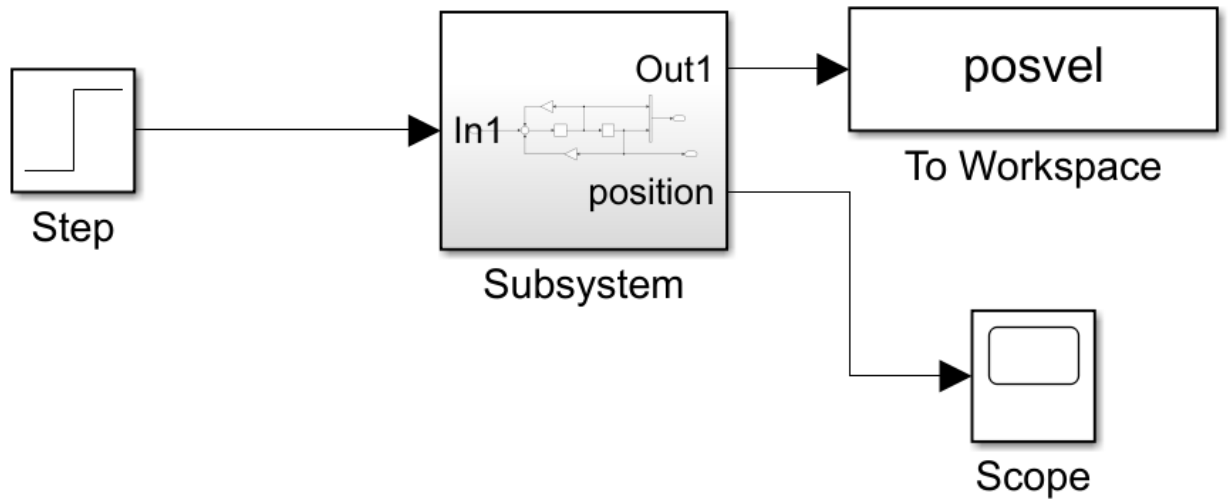




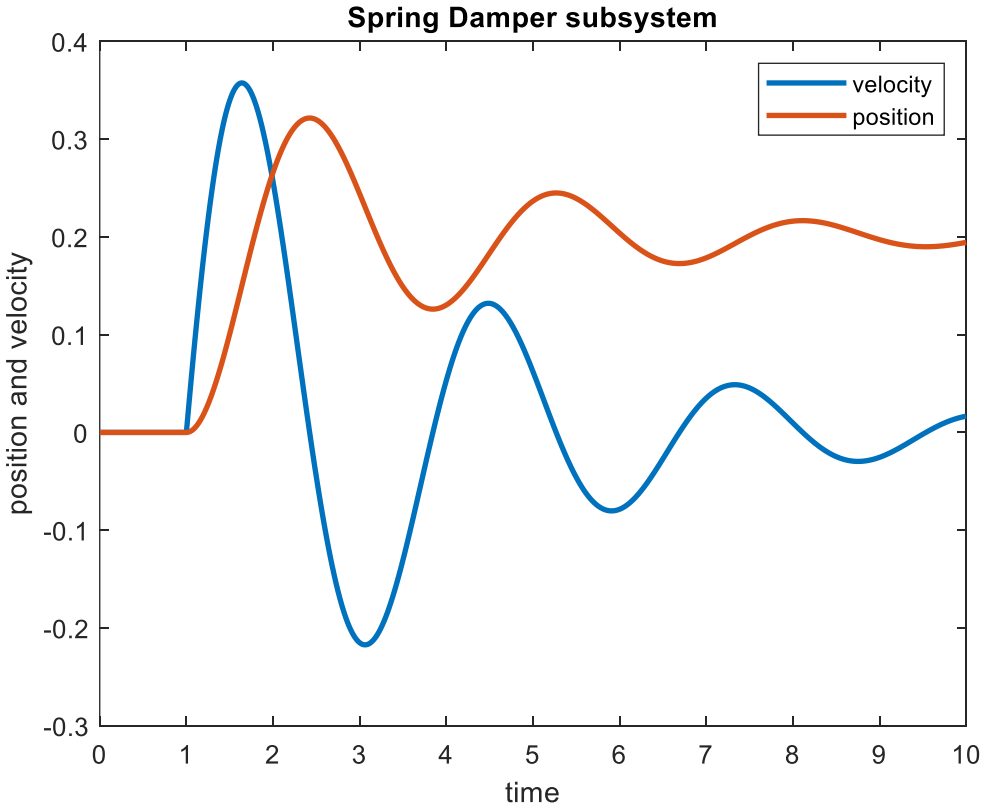
The plots start from 1 second as the step time is set at 1. This can be reset to zero or altered accordingly

Plot command - `plot(t,posvel(:,1),t,posvel(:,2))`

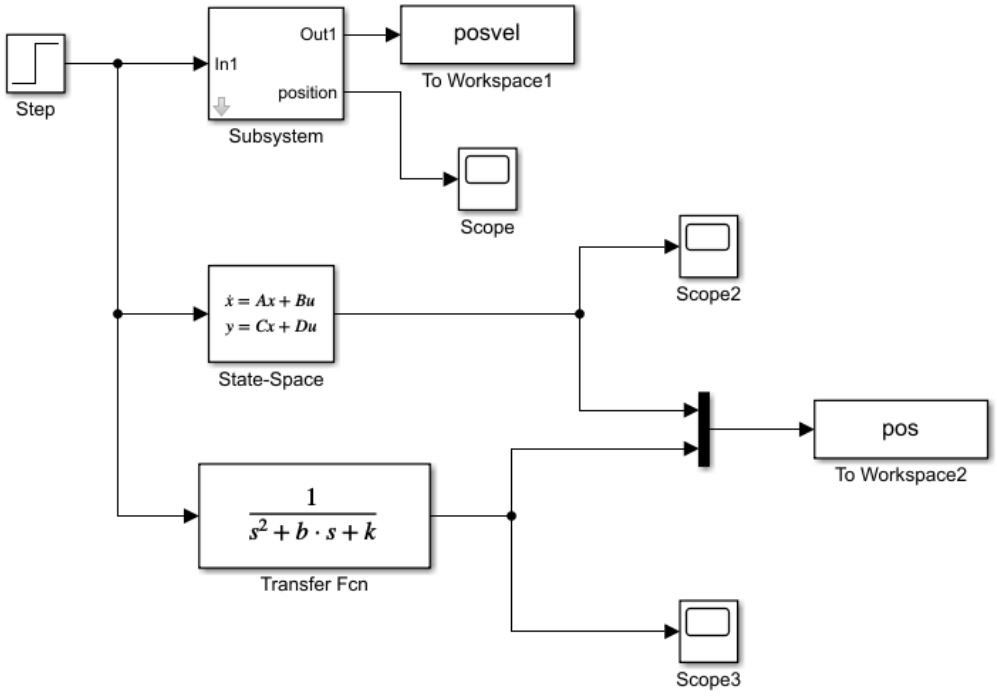




Plot command - `plot(t,posvel(:,1),t,posvel(:,2))`

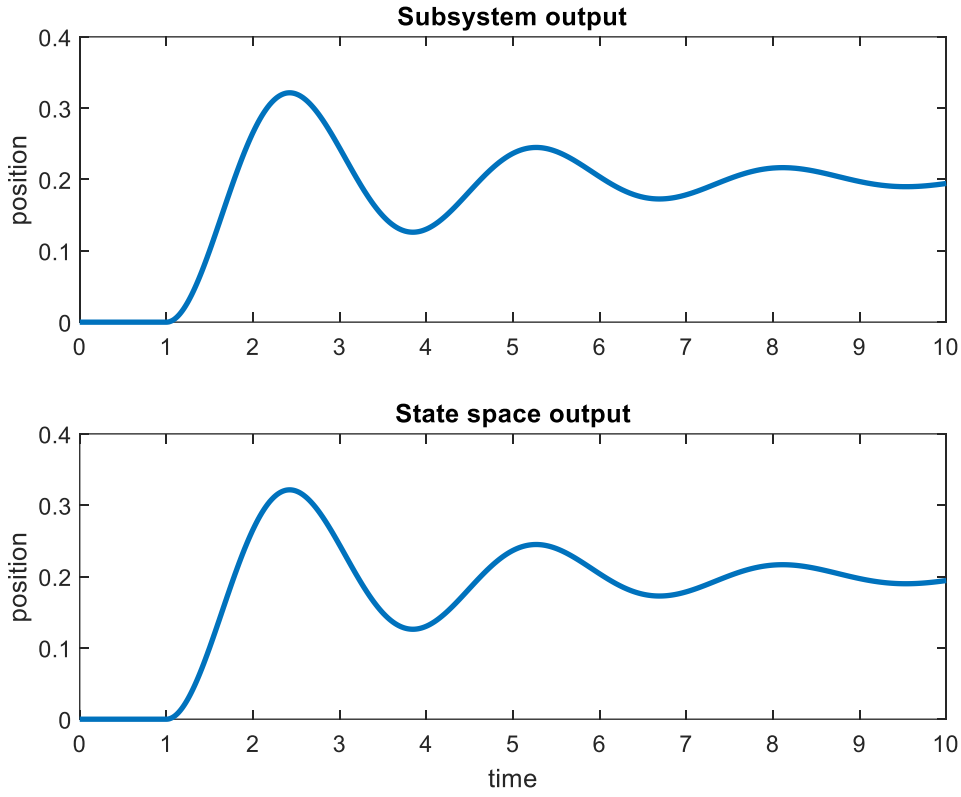

Spring Damper subsystem
velocity
position
position and velocity
time
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
0
1
2
3
4
5
6
7
8
9
10

Masking components and implementing state-space and transfer functions of above system in Simulink



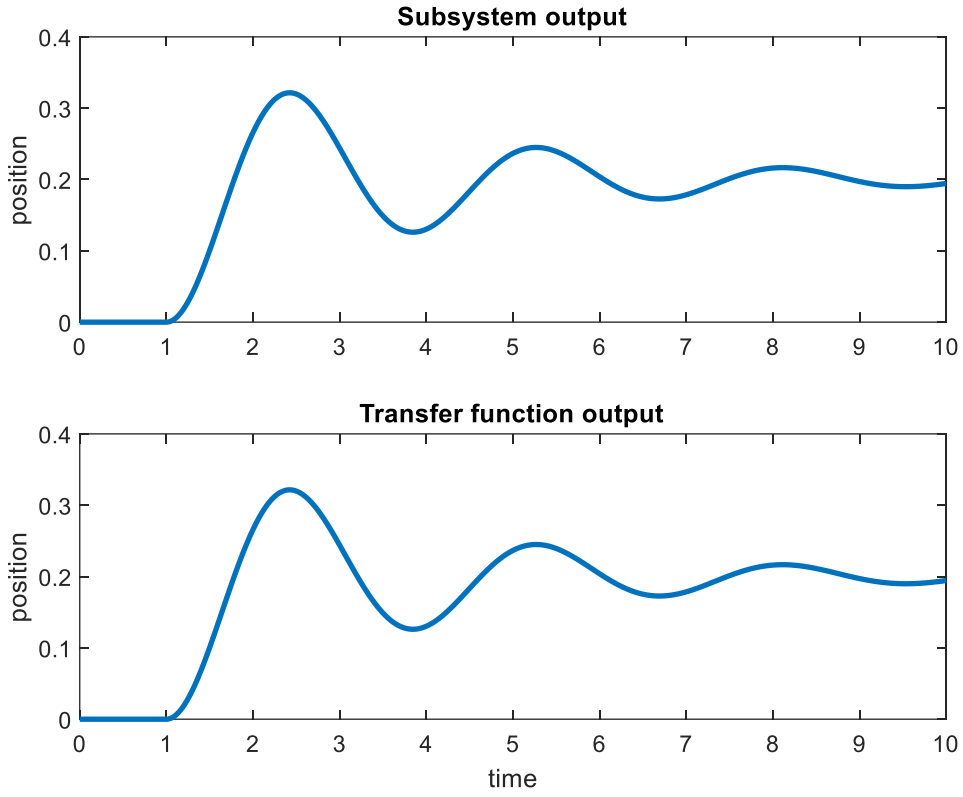
Plot commands

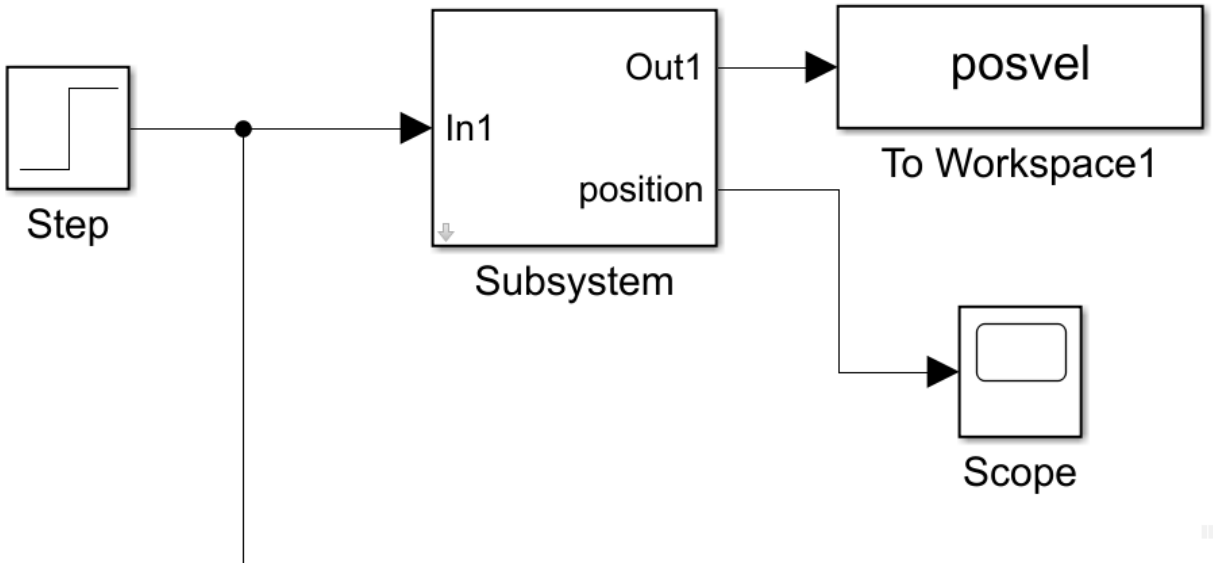
```
figure
subplot(2,1,1);
plot(t,posvel(:,2))
subplot(2,1,2);
plot(t,pos(:,1))
```

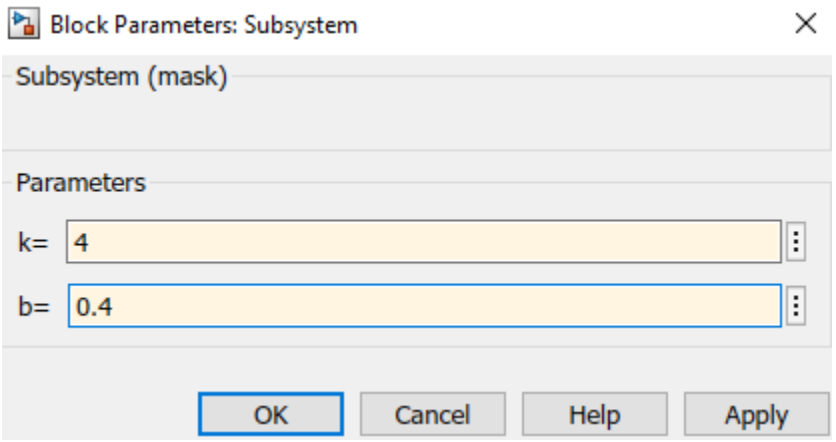


Plot commands

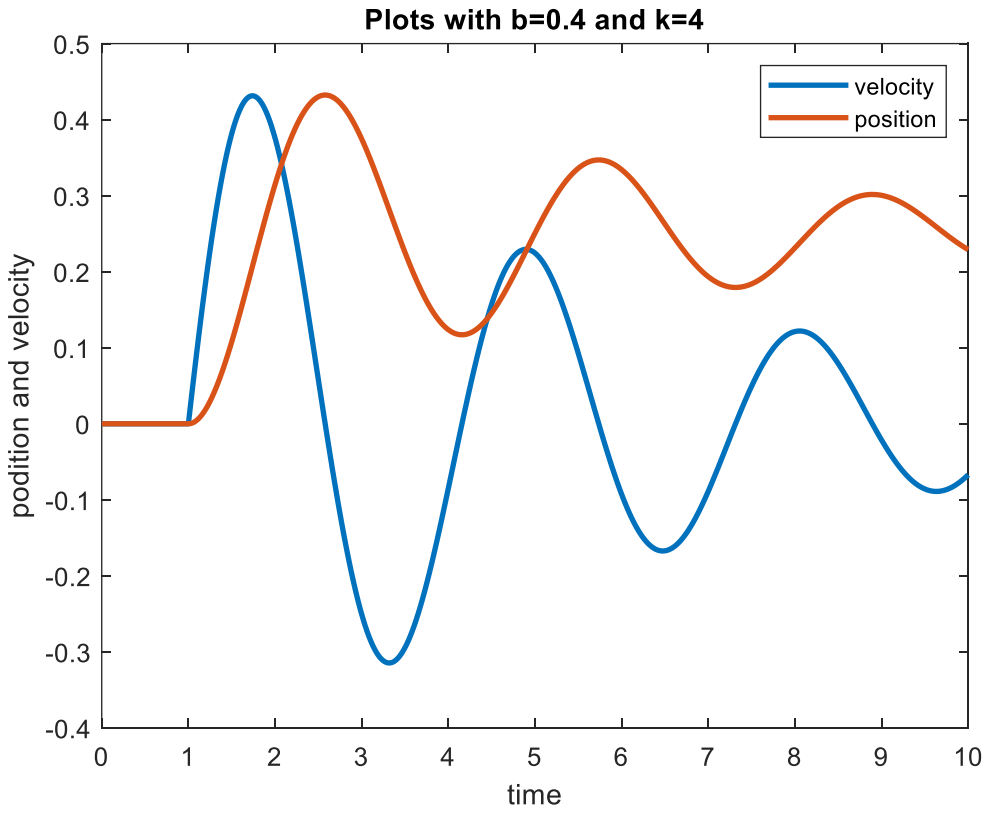
```
figure
subplot(2,1,1);
plot(t,posvel(:,2))
subplot(2,1,2);
plot(t,pos(:,2))
```


Subsystem output
position
0.4
0.3
0.2
0.1
0
0 1 2 3 4 5 6 7 8 9 10
Transfer function output
position
0.4
0.3
0.2
0.1
0
0 1 2 3 4 5 6 7 8 9 10
time


Step
In1
Out1
position
Subsystem
posvel
To Workspace1
Scope



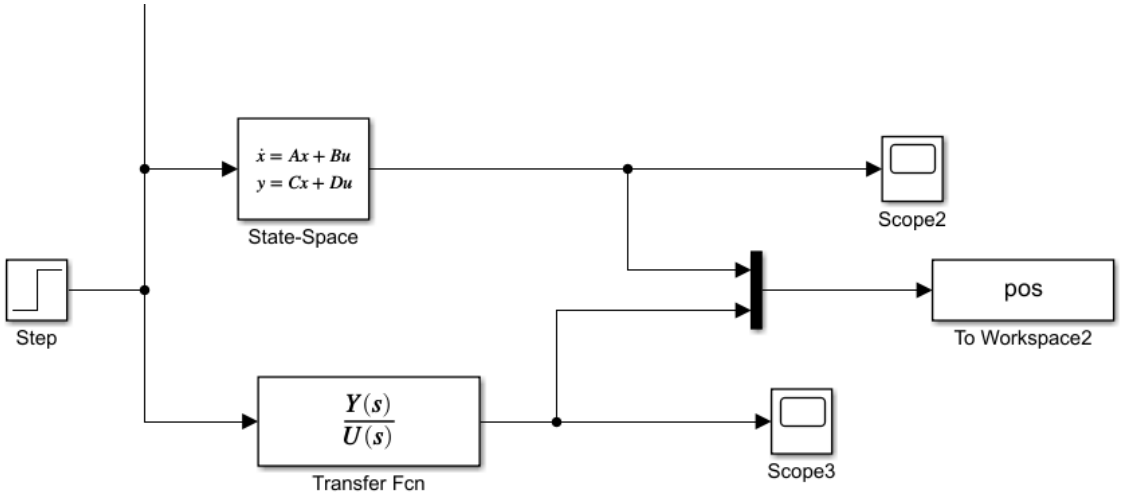
Plot command - `plot(t,posvel(:,1),t,posvel(:,2))`

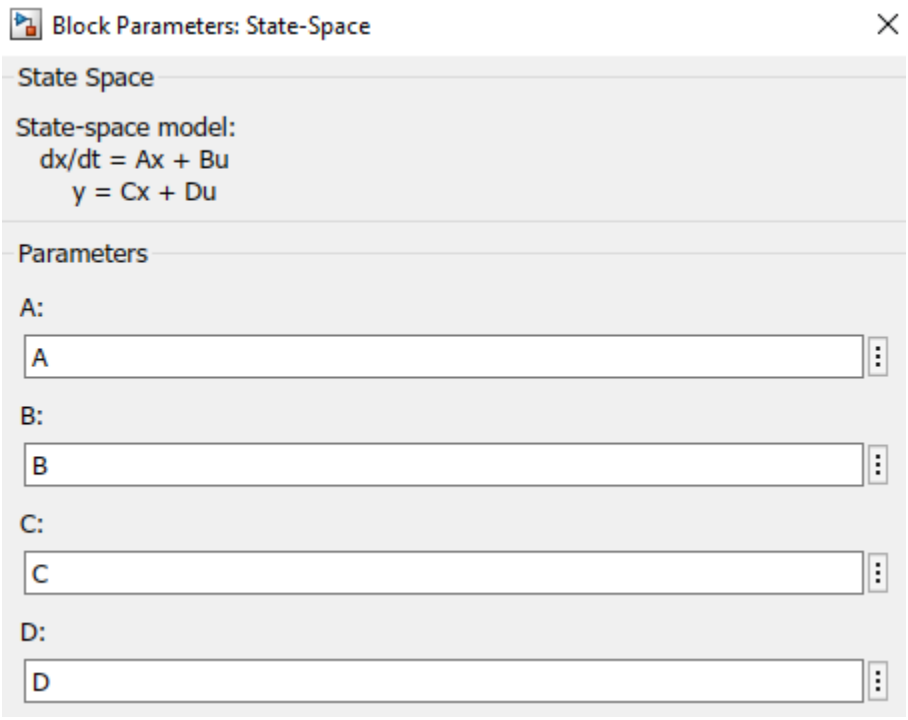


Implementing spring mass system in discrete time and comparing outputs

Discrete Time function commands:

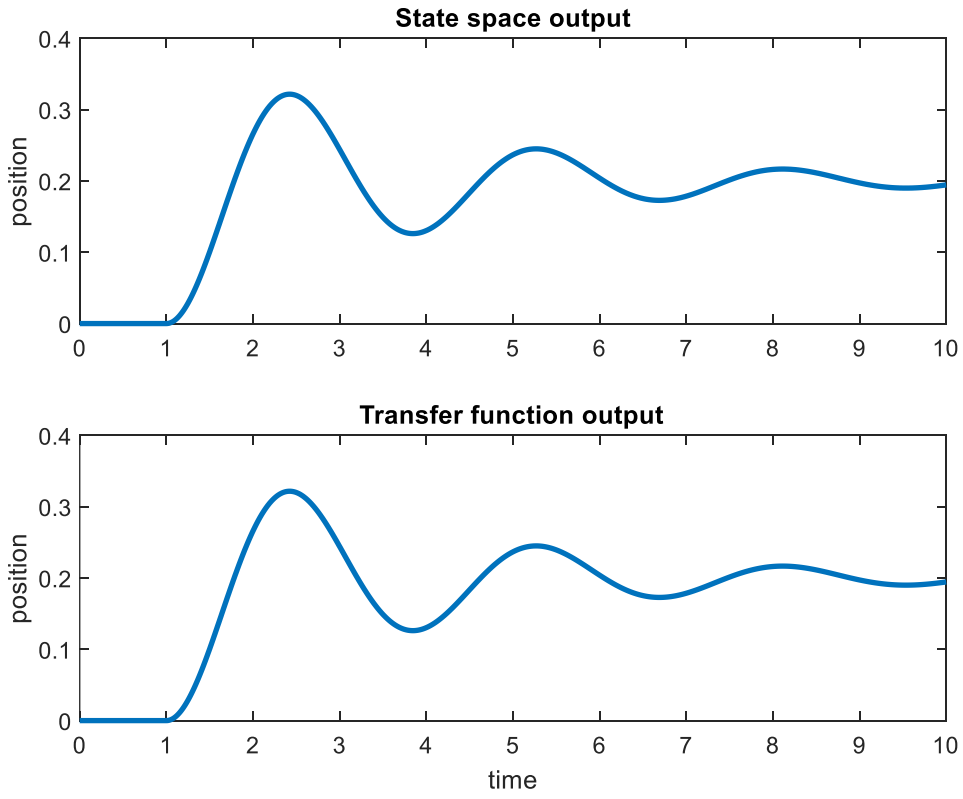
```
[num,den]=ss2tf(A,B,C,D)

num =

     0     0     1

den =

    1.0000    0.7000    5.0000

>> TF=tf(num,den)

TF =

          1

  ---------------

  s^2 + 0.7 s + 5

                      Continuous-time transfer function.

DF=c2d(TF,0.1)

DF =

  0.004865 z + 0.004753

  ----------------------

  z^2 - 1.884 z + 0.9324

Sample time: 0.1 seconds

Discrete-time transfer function.
```





Plot Commands:

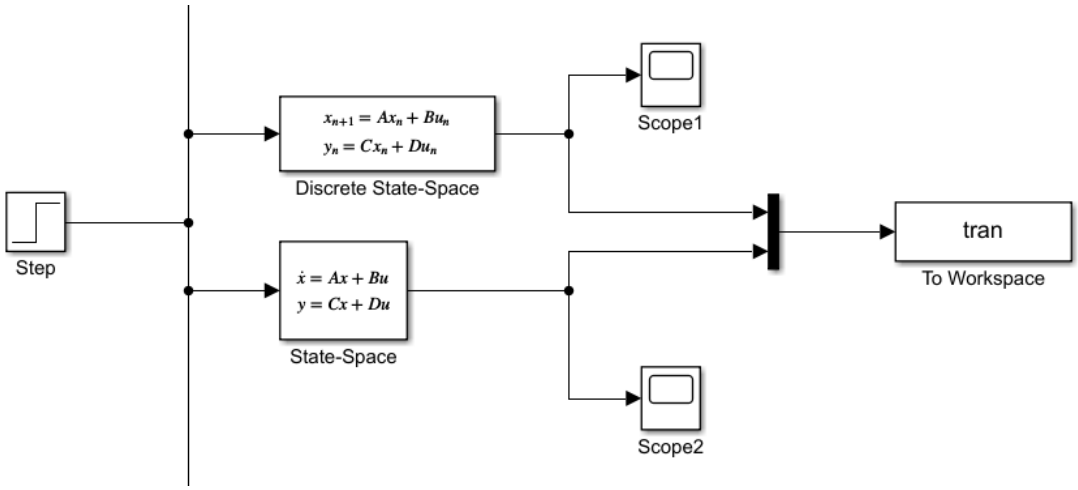
```
figure
subplot(2,1,1);
plot(t,pos(:,1))
subplot(2,1,2);
plot(t,pos(:,2))
```



Discrete time state space commands:

```
SS=ss(A,B,C,D)

SS =

  A =

         x1    x2

   x1     0     1

   x2    -5  -0.7

  B =

       u1

   x1   0

   x2   1

  C =
```

```
       x1  x2
   y1   1   0
  D =
       u1
   y1   0
Continuous-time state-space model.
>> discr=c2d(SS,0.1,'Tustin')
discr =
  A =
           x1       x2
   x1   0.9761  0.09547
   x2  -0.4773   0.9093
  B =
            u1
   x1  0.004773
   x2   0.09547
  C =
           x1       x2
   y1   0.9881  0.04773
  D =
            u1
   y1  0.002387
Sample time: 0.1 seconds
Discrete-time state-space model.
```



Block Parameters: Discrete State-Space

DiscreteStateSpace

Discrete state-space model:
x(n+1) = Ax(n) + Bu(n)
y(n) = Cx(n) + Du(n)

Main | State Attributes

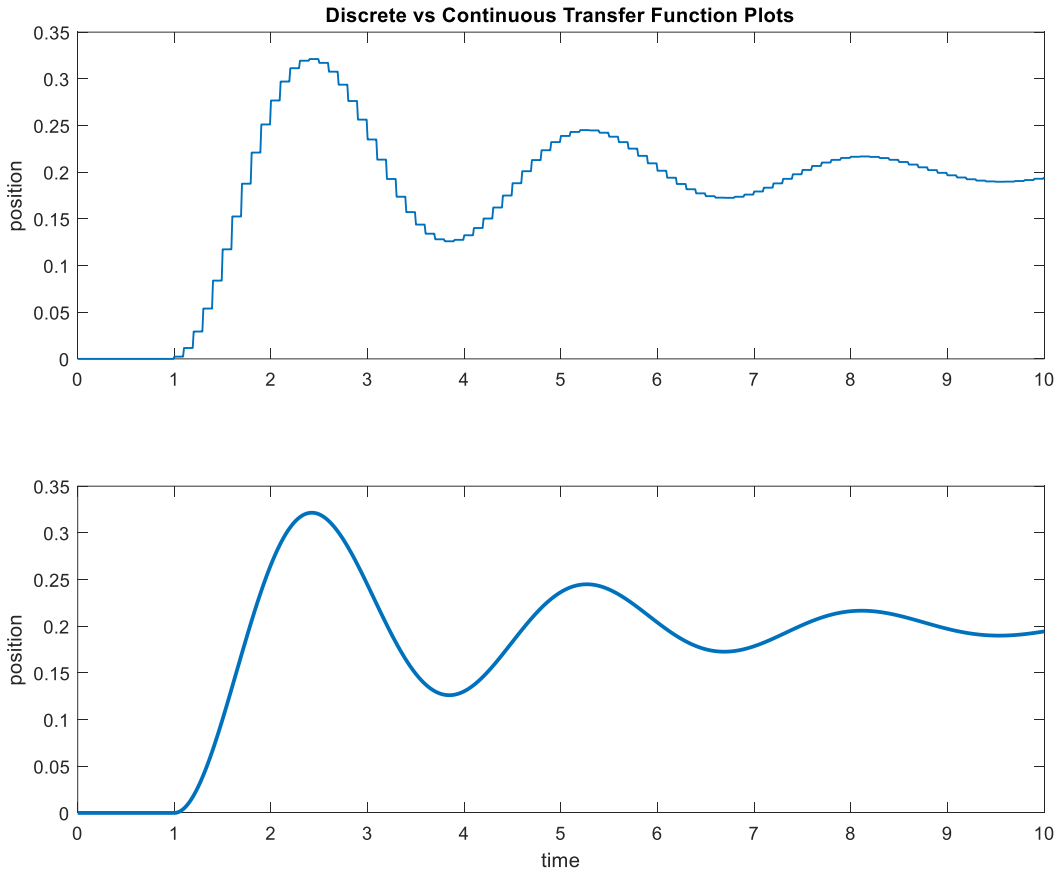
A:
discr.A

B:
discr.B

C:
discr.C

D:
discr.D

Initial conditions:
0

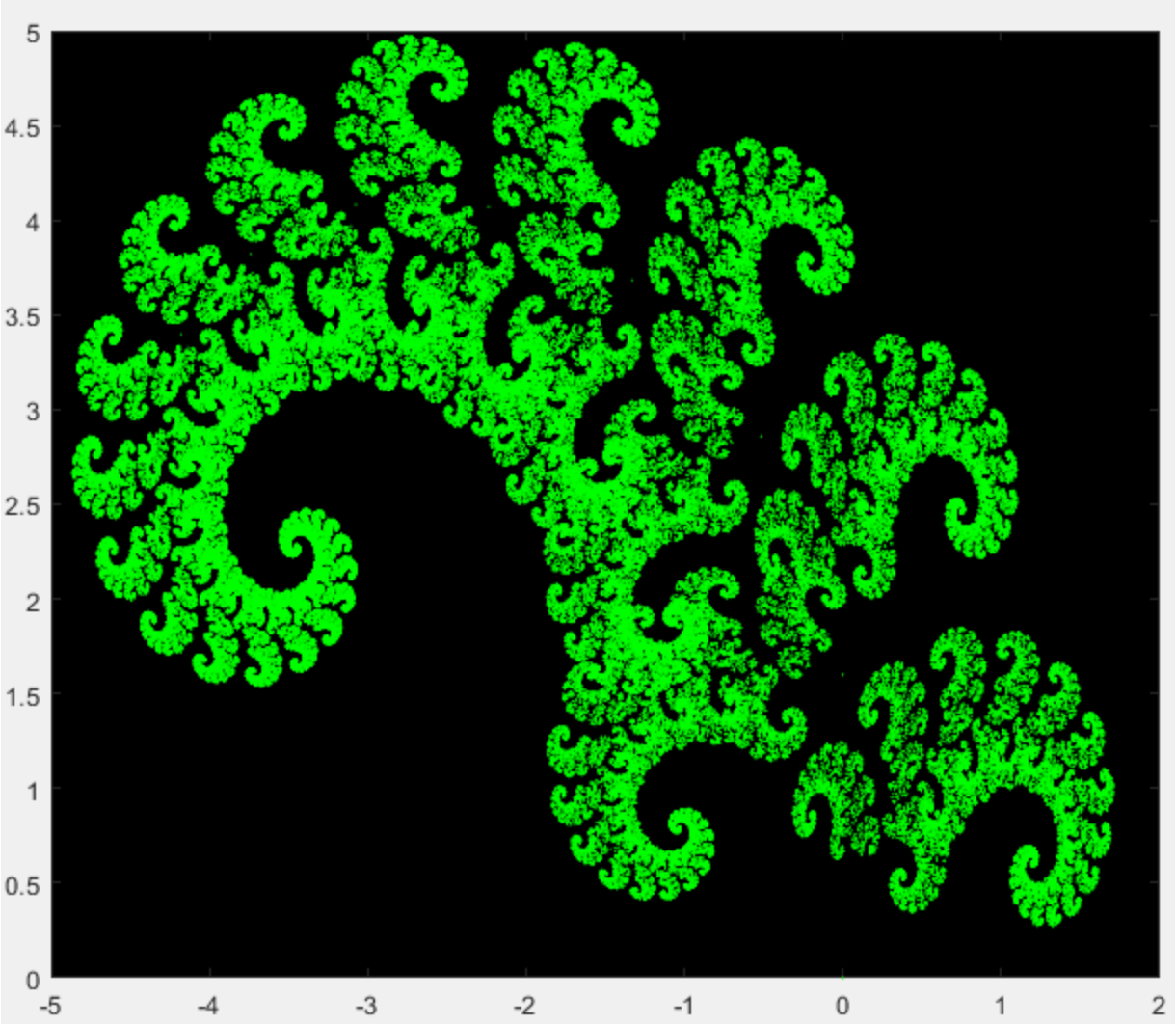



```
DF=c2d(TF,0.1)

DF =

  0.004865 z + 0.004753
  ----------------------
  z^2 - 1.884 z + 0.9324

Sample time: 0.1 seconds
Discrete-time transfer function.
```

Implementing Barnsley Ferns equation for 100,000 points in MATLAB

MATLAB Code for generating the plot:

```
function barnsleyfern
x=[0;0];
A=[0.7873 -0.3230;0.3230 0.7873];
B=[0.0841 -0.3286;0.2930 0.0895];
C=[-0.2458 0.1523;0.1722 0.3358];
b1=[0;1.6];
b2=[0;0.44];
p=[0.8 0.9 1.0];
i=0;
set(gca,'color',[0 0 0])
plot(x(1),x(2),'g.','markersize',1) hold on
while i<100001
    r=rand;
    if r < p(1)
        x=((A*x)+b1);
    elseif r < p(2)
        x=((B*x)+b1);
    else
        x=((C*x)+b2);
    end
    J=x(1,1);
    K=x(2,1);
    set(gca,'color',[0 0 0])
    plot(J,K,'g.','markersize',1)
    i=i+1;
end
end
```


5
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
-5
-4
-3
-2
-1
0
1
2