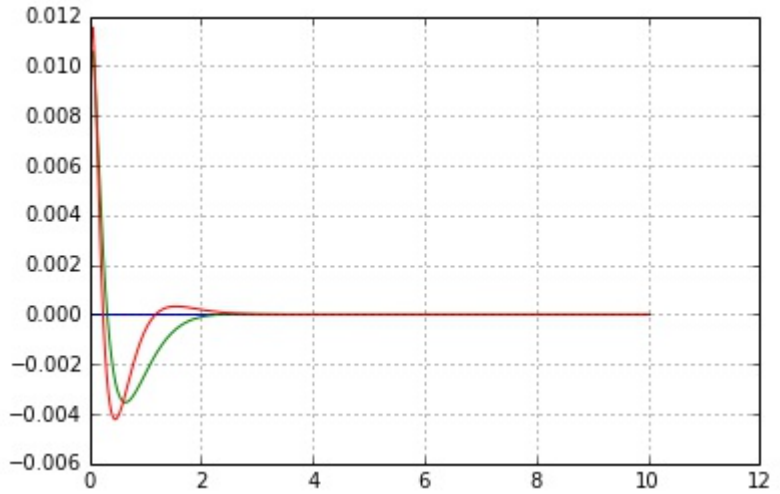
```
Python 2.7.6 (default, Feb 26 2014, 00:34:35)
Type "copyright", "credits" or "license" for more information.

IPython 1.2.1 -- An enhanced Interactive Python.
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra details.
%guiref   -> A brief reference about the graphical user interface.

In [1]:
########################################################################
   ...: # Determine physical parameters
   ...:
########################################################################
#
   ...: # project:            pendulum, mechatronics laboratory, SUPSI
   ...: # pend_id_.m:         script for controller design
   ...: # author:             Roberto Bucher / Silvano Balemi
   ...: # date:               05.03.2008
   ...: # E-mail:             {roberto.bucher,silvano.balemi}@supsi.ch
   ...:
########################################################################
#
   ...:
   ...:
########################################################################
#
   ...: # state-space model
   ...:
########################################################################
#
   ...: # State space matrices with state=[phi omega x v]' from first
principles;
   ...:
   ...: from supsictrl.yottalab import *
   ...: import numpy as np
   ...: import scipy as sp
   ...: from matplotlib.pyplot import *
   ...: from control import *
   ...: from supsictrl.RCPblk import *
   ...:

In [2]: ts=0.01
   ...: M=0.4874257
   ...: m=0.19211
   ...: r=0.323
   ...: rp=0.0286524
   ...: g=9.81
   ...: Theta=0.0294049
   ...: kt=0.0603/1000
   ...: d=1.6800357
   ...: N=1
   ...:

In [3]: A=[[ 0,    1,   0,   0 ], \
   ...:     [m*r*(M+m)*g/(Theta*(M+m)-m*m*r*r), 0, 0, -m*r*d/(Theta*(M+m)-
m*m*r*r)], \
   ...:     [0,    0,   0,   1], \
   ...:     [m*m*r*r*g/(Theta*(M+m)-m*m*r*r), 0, 0, -Theta*d/(Theta*(M+m)-
m*m*r*r)]]
   ...:
```

```
   ...: B=[[0], [kt*N*m*r/(Theta*(M+m)-m*m*r*r)/rp], [0],
[kt*N*Theta/(Theta*(M+m)-m*m*r*r)/rp]]
   ...:
   ...: C=[[0, 0, 1, 0], [1, 0, 0, 0]]
   ...: D=[[0], [0]]
   ...:
   ...: sys = ss(A,B,C,D)
   ...:
   ...: print sys
   ...:
A = [[  0.           1.           0.           0.        ]
 [ 25.64275974   0.           0.          -6.46252132]
 [  0.           0.           0.           1.        ]
 [  2.34155832   0.           0.          -3.06245137]]

B = [[ 0.        ]
 [ 0.00809543]
 [ 0.        ]
 [ 0.00383625]]

C = [[0 0 1 0]
 [1 0 0 0]]

D = [[0]
 [0]]


In [4]: # discrete-time model
   ...: sysd = bb_c2d(sys,ts)
   ...:
   ...: print sysd
   ...:
A = [[  1.00127991e+00   1.00042681e-02   0.00000000e+00  -3.19921193e-04]
 [  2.55787928e-01   1.00127991e+00   0.00000000e+00  -6.36730526e-02]
 [  1.15916698e-04   3.87339786e-07   1.00000000e+00   9.84842241e-03]
 [  2.30705879e-02   1.15916698e-04   0.00000000e+00   9.69837182e-01]]

B = [[  4.00756714e-07]
 [  7.97615284e-05]
 [  1.89877190e-07]
 [  3.77841545e-05]]

C = [[0 0 1 0]
 [1 0 0 0]]

D = [[0]
 [0]]

dt = 0.01


In [5]: # LQR Controller design
   ...:
################################################################################
#
   ...: Q = diag([300,10,400,20]);
   ...: R = [4e-6];
   ...: [k_lqr, S, E] = bb_dlqr(sysd.A,sysd.B,Q,R)
   ...:
   ...: print k_lqr
   ...:
[[ 30810.14625918   6572.24922887  -8882.0598767   -8176.27708664]]
```

```
In [6]: # Observer design
   ...:
##############################################################################
#
   ...: # Control dedign - Reduced order observer
   ...: (preg,pvect) = sp.linalg.eig(A-B*k_lqr)
   ...:
   ...: rho=max(abs(preg));      # process spectral radius
   ...: rhoamp=10;                # amplification of observer poles
   ...:
   ...: angle1 = 15*1j*np.pi/16
   ...: poles_c = rhoamp*rho*sp.exp([angle1, -angle1])
   ...: obs_poles = sp.exp(poles_c*ts)
   ...:
   ...: T=[[0,1,0,0],[0,0,0,1]]
   ...: obs=red_obs(sysd,T,[obs_poles])
   ...:
   ...: print obs
   ...:
A = [[ 0.23884856 -0.06793048]
 [ 0.06793048  0.23884856]]

B = [[  4.86675064e-05  -7.24049832e+00  -5.73891535e+01]
 [  2.64502546e-05  -5.61304373e+01   1.03618821e+01]]

C = [[ 0.  0.]
 [ 1.  0.]
 [ 0.  0.]
 [ 0.  1.]]

D = [[  0.           0.           1.        ]
 [  0.           2.90795652  76.21049474]
 [  0.           1.           0.        ]
 [  0.          74.00363899  -6.78142885]]

dt = 0.01


In [7]: # Controller in compact form
   ...: ctr = comp_form(sysd,obs,k_lqr)
   ...:
   ...: print ctr
   ...:
A = [[-0.08100642  0.32998853]
 [-0.10590718  0.45511317]]

B = [[  4.86675064e-05   2.17091003e+01  -8.59633723e+01]
 [  2.64502546e-05  -4.03966487e+01  -5.16789171e+00]]

C = [[-6572.24922887  8176.27708664]]

D = [[  1.00000000e+00   5.94844503e+05  -5.87131353e+05]]

dt = 0.01


In [8]: # Simulation
   ...: PosRef =constBlk (1, 0)
   ...: Controller =dssBlk ([1,2,3], 4, ctr, 0)
   ...: Saturation =saturBlk (4, 5, 3000, -3000)
   ...: Plant = cssBlk(5,[2,3], sys,[0.01,0,0.01,0])
```

```
   ...: Out =printBlk ([1,2,3])
   ...:
   ...: blks = [PosRef, Controller, Saturation, Plant, Out]
   ...:
   ...: fname = 'pendulum'
   ...: genCode(fname, ts, blks)
   ...: genMake(fname)
   ...:
   ...: simul('pendulum')
   ...:
```




```
In [9]: # Real controller for the real plant
   ...: # Some safety blocks have been introduces
   ...: # The plant is connected using a Peaks USB dongle
   ...: # to connect the epos motor + encoder and the baumer
   ...: # encoder for the pole
   ...:
   ...: PosRef =constBlk (1, 0)
   ...: posX =epos_EncBlk (2, 0x08, 1024)
   ...: posPhi =baumer_EncBlk (3, 0x3FF, 500)
   ...: zeroX =init_encBlk (2,4, 1, 0, 0)
   ...: zeroPhi =init_encBlk (3, 5, 1, 0, -np.pi)
   ...: a2x = matmultBlk (4, 6, -rp/N)
   ...: a2a = matmultBlk (5, 7, 1)
   ...: Controller =dssBlk ([1,6,7], 8, ctr, 0)
   ...: Delay =stepBlk (9, 5, -1)
   ...: Safety =prodBlk ([8,9], 10)
   ...: Saturation =saturBlk (10, 11, 3000, -3000)
   ...: ZERO =constBlk (12, 0)
   ...: ABSX = absBlk(6,13)
   ...: SW =switchBlk ([11,12,13], 14, 0, 0.3, 1)
   ...: Motor =epos_MotIBlk (14, 0x08, 3873, 1122)
   ...:
   ...: blks = [PosRef, posX, a2x, a2a,
zeroX,posPhi,zeroPhi,Controller,Delay,Safety, Saturation,Motor,ZERO,ABSX,SW]
   ...:
   ...:
   ...: fname = 'pendulum'
   ...: genCode(fname, ts, blks)
   ...: genMake(fname, '_rt')
   ...:

In [10]:
```