#-------------------------------------------------------------------------------

# Name: freefall (4.1)

# Purpose:

#

# Author: Clarissa Joyce

#

# Created: 02/10/2014

#-------------------------------------------------------------------------------

#!/usr/bin/env python

from matplotlib import pyplot as plt

g=9.81

xx=[]

yy=[]

def euler(y,vy,vt):

t=0

dt=0.01

plt.xlabel("Time, s")

plt.ylabel("Height, m")

while y>=0:

plt.plot(t,y,"bo")

xx.append(t)

yy.append(y)

#ay = -g\*(1+vy/vt) #stokes acceleration

ay = -g\*(1+vy\*abs(vy)/(vt\*vt)) #Newton's law of drag

vy += ay\*dt

y+= vy\*dt

t+= dt

done = 0

while not done:

y,vy = input("initial height, velocity")

vt=input("terminal velocity")

euler(y,vy,vt)

turnIdx = yy.index(max(yy))

print "The max height is ",max(yy)

print "Ascent time: ",xx[turnIdx]

print "Descent time: ",(xx[len(xx)-1]-xx[turnIdx+1])

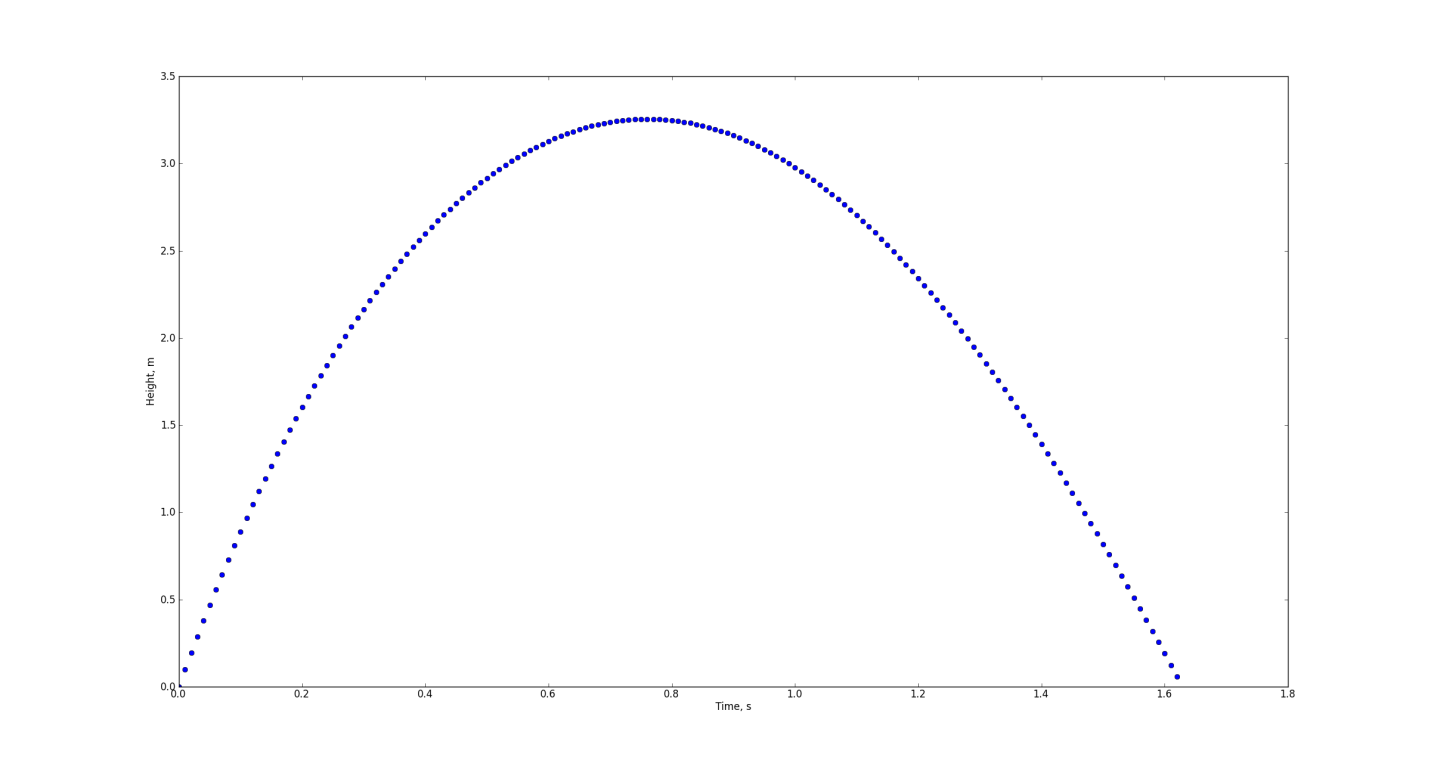
plt.show()

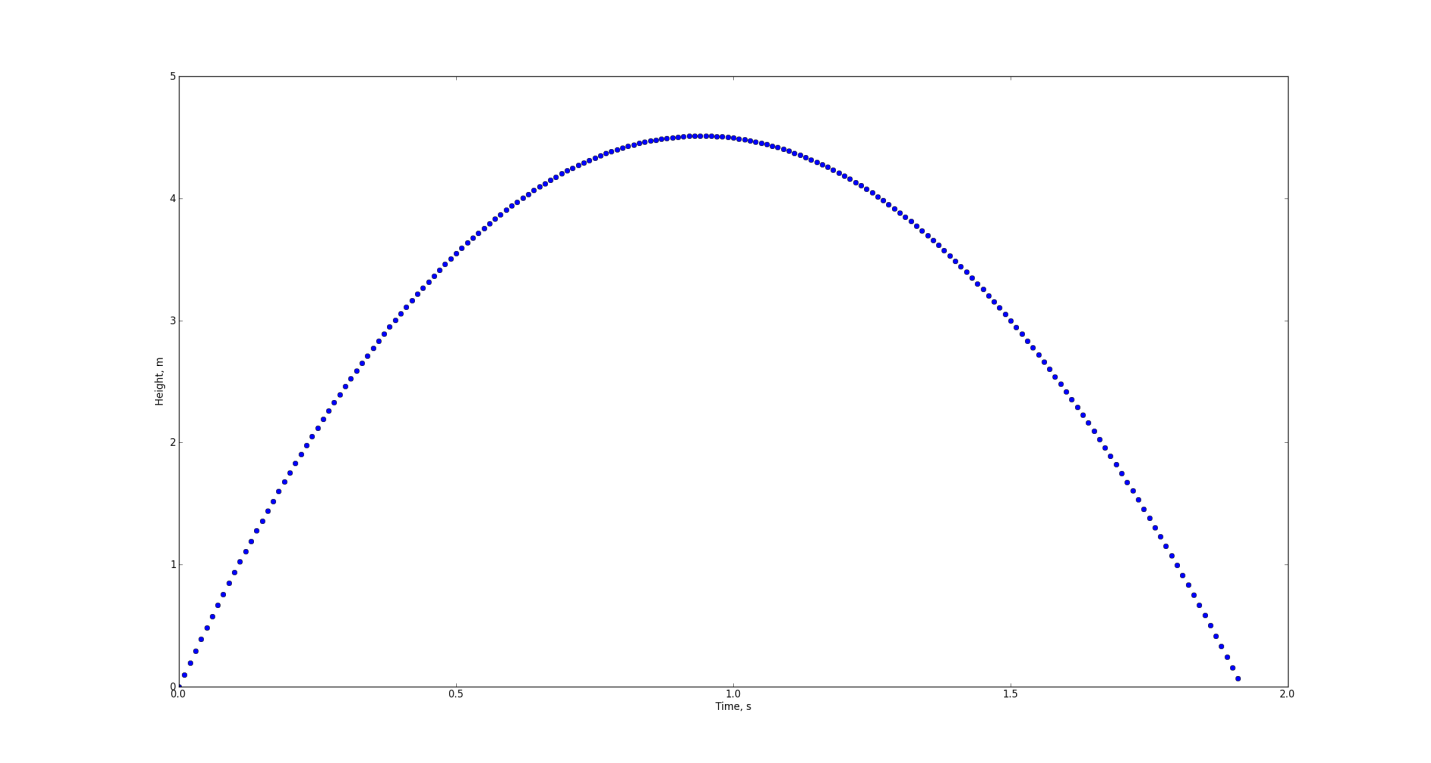
reply = raw\_input("Again? y or n")

if reply == 'n':

done = 1

print "Fin"





#-------------------------------------------------------------------------------

# Name: football

# Purpose:

#

# Author: Claire Joyce

#

# Created: 10/10/2014

#-------------------------------------------------------------------------------

#!/usr/bin/env python

import matplotlib.pyplot as plt

from math import \*

Vo = 30

Tv = 45

g = 9.81

xx = []

yy = []

thth = []

ranges = []

y = 2

#thstep = 1\*pi/180

Maxt = 0

def euler(y, vy, vt ):

t = 0

dt = .1

plt.xlabel("TIme (s)")

plt.ylabel("Height(m)")

while(y>=0):

xx.append(t)

yy.append(y)

ay = -g\*(1+vy\*abs(vy)/(vt))

vy = vy + ay\*dt

y = y + vy\*dt

t = t+dt

Maxt = t

plt.xlabel("Angle")

plt.ylabel("Range")

plt.title("Tony Romo's throw range")

th = 0

while th<90:

vy = Vo\*sin(th\*pi/180)

euler(y, vy, Tv)

vx = Vo \* cos(th\*pi/180)

plt.plot(th,max(xx)\*vx,'bo')

thth.append(th)

ranges.append(max(xx)\*vx)

th+=.5

maxRange = max(ranges)

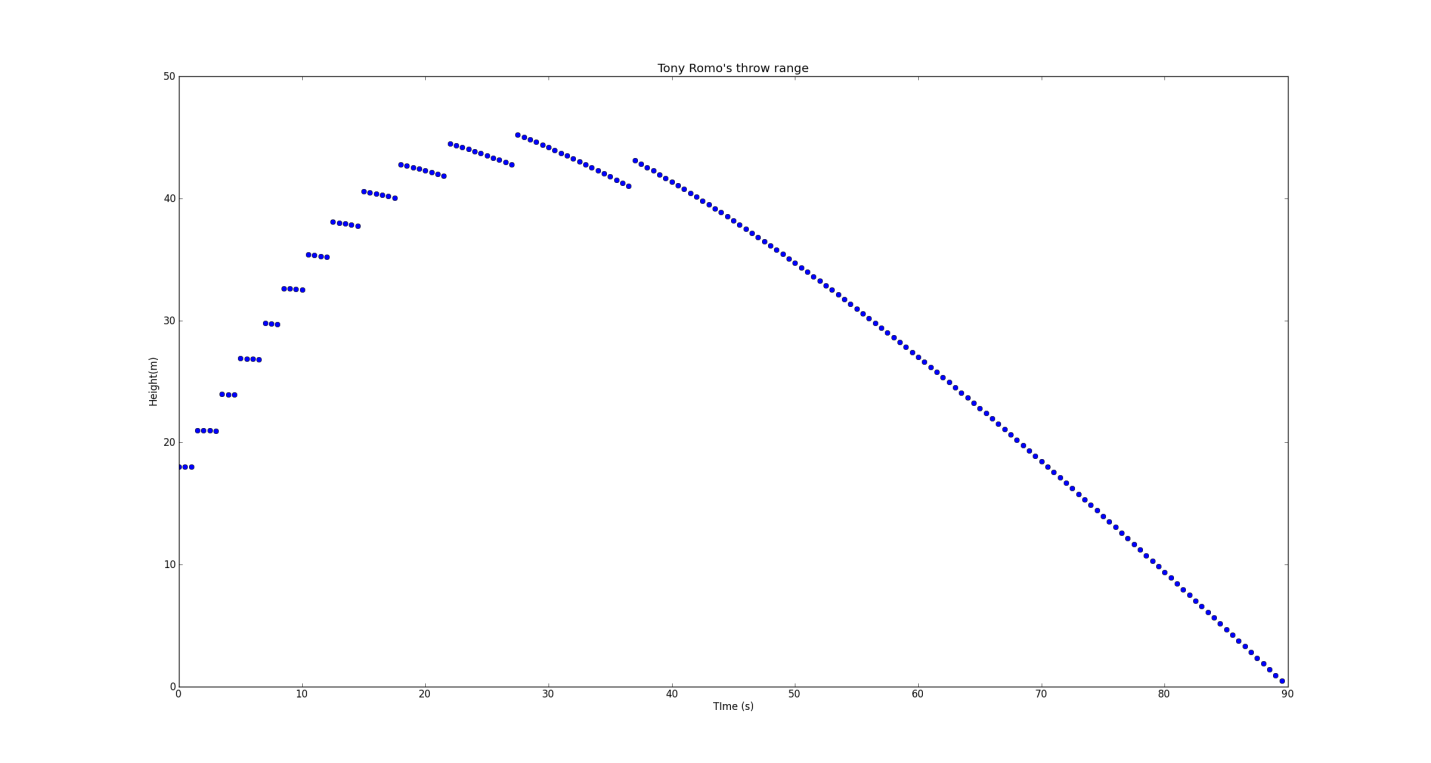
optTheta = thth[ranges.index(max(ranges))]

print "Biggest range: ",maxRange

print "Optimum angle: ",optTheta

print "My results do not agree with the given observation."

plt.show()



#-------------------------------------------------------------------------------

# Name: superball (4.3)

# Purpose:

#

# Author: Claire Joyce

#

# Created: 07/10/2014

#-------------------------------------------------------------------------------

#!/usr/bin/env python

#The distances between impact points increase with each bounce.

#Air resistance causes a decrease in the height of each bounce.

#import libraries

import matplotlib.pyplot as plt

from math import \*

#define and get initial condition and constants

g=9.81

h=input("Enter the height of the ball:")

theta = 30\*pi/180.0 #conv to radians

v=sqrt(2\*g\*h) #init speed of ball at incline

vx=0

vy=-v

x=0

y=0

t=0

dt=0.01

while t<10:

ay=-g

vy+=ay\*dt

vx=vx

y+=vy\*dt

x-=vx\*dt

yIncline=x\*tan(theta)

if y<=yIncline:

print "new bounce"

v=sqrt(vx\*\*2+vy\*\*2)

vx=v\*cos(theta)

vy=v\*sin(theta)

t+=dt

plt.plot(x,y,"c+")

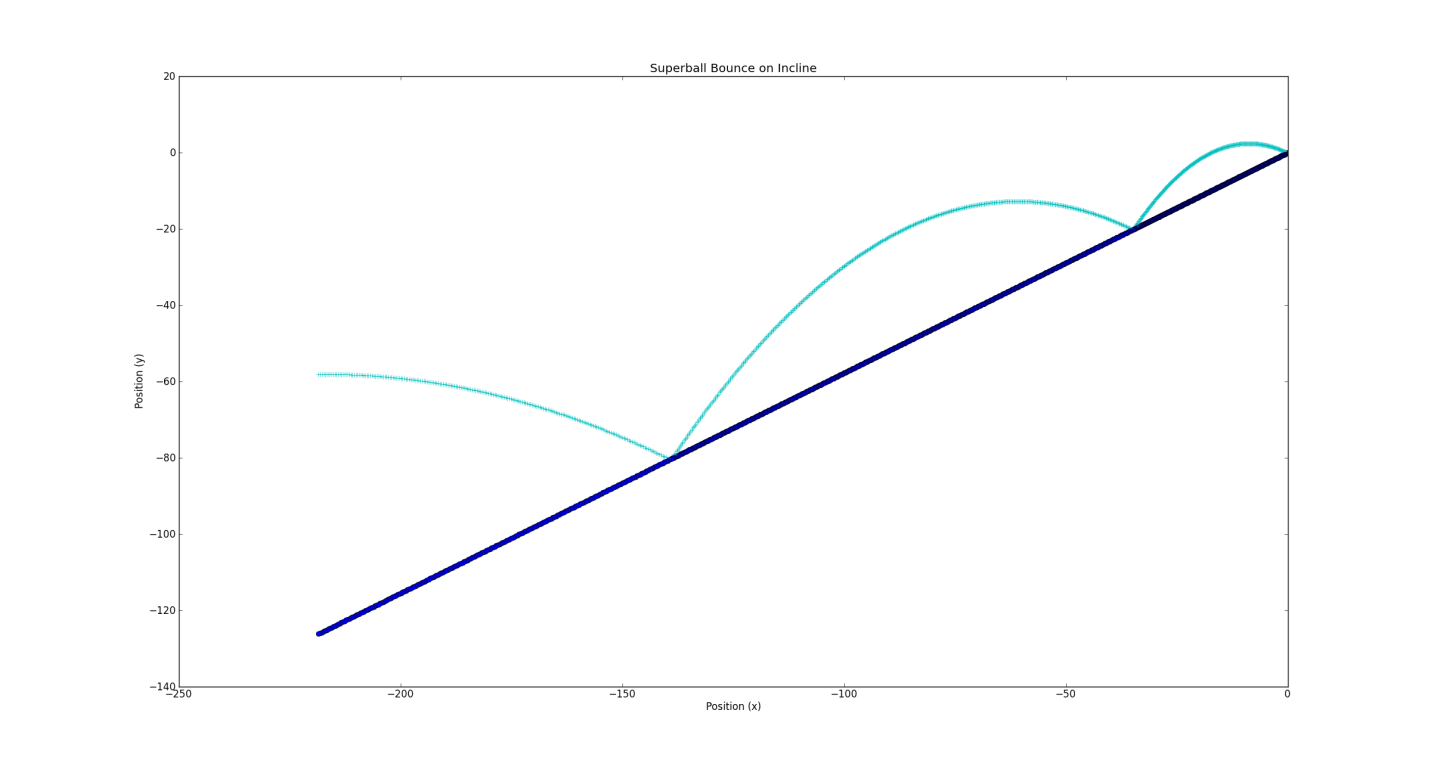
plt.plot(x,x\*tan(theta),"bo")

plt.xlabel("Position (x)")

plt.ylabel("Position (y)")

plt.title("Superball Bounce on Incline")

plt.show()



#-------------------------------------------------------------------------------

# Name: moreBaseball (4.4)

# Purpose:

#

# Author: Claire

#

# Created: 09/10/2014

#-------------------------------------------------------------------------------

#!/usr/bin/env python

from numpy import \*

from math import \*

from mpl\_toolkits.mplot3d import \*

import matplotlib.pyplot as plt

cy = 0

cz = 0

done = 0

def kD(v): #define drag coefficient a la Giordano

delta = 5.

vd = 35.

return 0.0039+0.0058/(1.+exp((v-vd)/delta))

def euler(vx,vy,vz):

x=0

y=0

z=input("pitcher's release height:")

t=0

dt = 0.001

kL=4.1E-4

omega = 1800#spin of ball, rotation rate in rpm

omega = omega\*2.\*pi/60. #converts spin to rad/s

while x<= 18.44:

X.append(x)

Y.append(y)

Z.append(z)

v=sqrt(vx\*\*2 + vy\*\*2 + vz\*\*2)

ax = -kD(v)\*v\*vx + kL\*omega\*(vz\*sin(phi)-vy\*cos(phi))

ay = -kD(v)\*v\*vy + kL\*omega\*vx\*cos(phi)

az = -kD(v)\*v\*vz - kL\*omega\*vx\*sin(phi)-g

vx += ax\*dt

vy += ay\*dt

vz += az\*dt

x+= vx\*dt

y+= vy\*dt

z+= vz\*dt

while not done:

X=[]

Y=[]

Z=[]

TYPE = raw\_input("Type of pitch (fastfball[f]/curveball[c]/slider[s]/screwball[w]: ").lower()

if TYPE == "c":

v=34.5 #pitch speed

phi=45\*pi/180. #spin direction

elif TYPE == "s":

v=37.5

phi = 0.

elif TYPE == "f":

v=41.6

phi=225\*pi/180.

elif TYPE == "w":

v=34.5

phi = 135\*pi/180.

else:

v=42.

phi=0.

theta = 3.\*pi/180. #set throwing angle 3 deg above horizontal

vx=v\*cos(theta)

vy=0.

vz=v\*sin(theta)

g=9.80665

euler(vx,vy,vz)

fig = plt.figure()

Ax = Axes3D(fig)

Ax.plot(X,Y,Z)

Ax.set\_xlim3d(0.,18.44)

Ax.set\_ylim3d(-1.,10.)

Ax.set\_zlim3d(0.,10.)

Ax.plot\_wireframe(array([[18.4,18.4],[18.4,18.4]]),

array([[-0.22,0.22],[-0.22,0.22]]),

array([[0.5,0.5],[1.1,1.1]]),

color = "r")

cy =Y[len(Y)-1]-Y[0]

cz =Z[len(Z)-1]-Z[0]

plt.show()

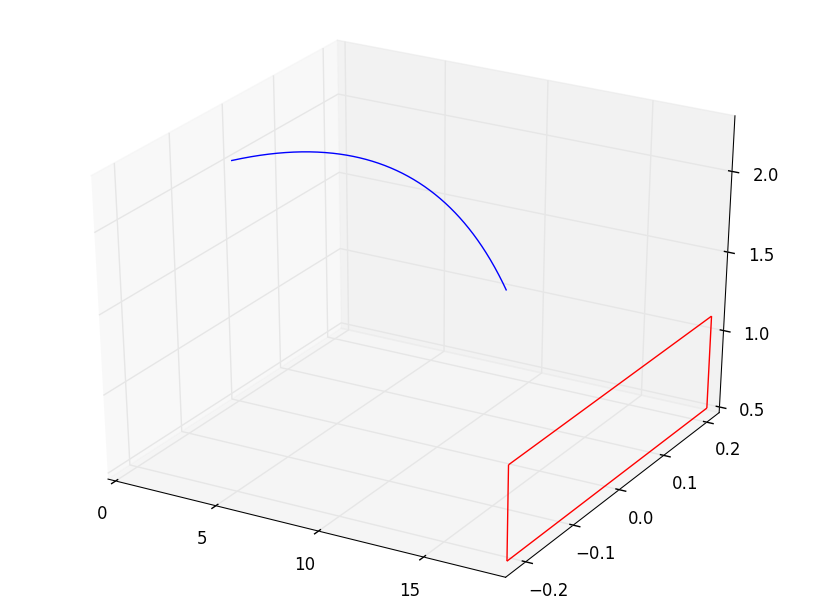
print TYPE,": ", cy," ",cz

reply = raw\_input("Again? y or n").lower()

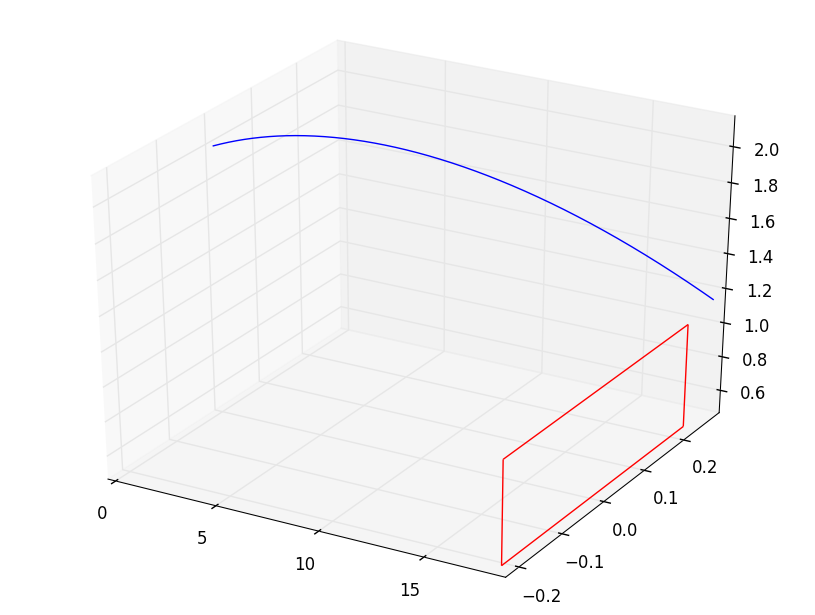
if reply == "n":

done = 1

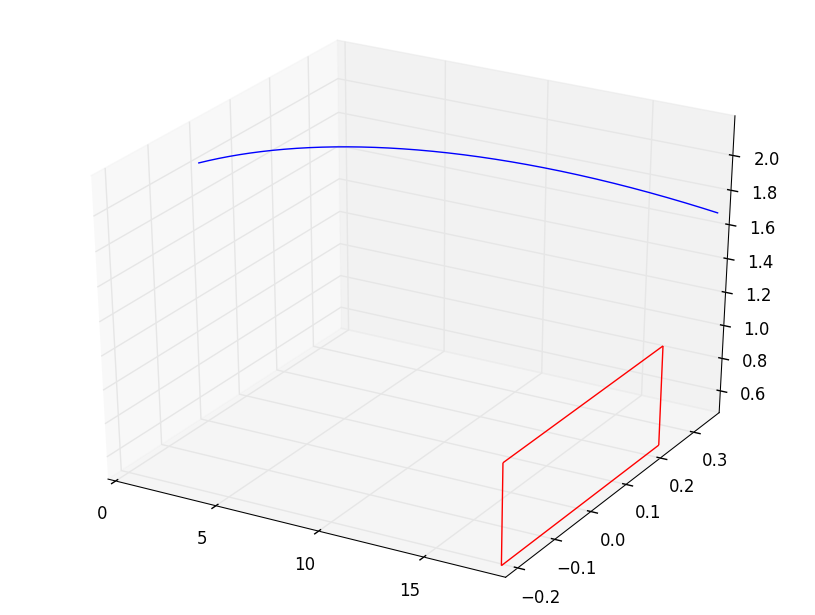
#f : -0.231797184313 0.162539398516



#c : 0.282342528896 -0.856854044879



#s : 0.365557514645 -0.324090825981



#w : -0.282342528896 -0.856854044879

