def getData(fileName):
    dataFile = open(fileName, 'r')
    distances = []
    masses = []
    discardHeader = dataFile.readline()
    for line in dataFile:
        d, m = line.split()
        distances.append(float(d))
        masses.append(float(m))
dataFile.close()
    return (masses, distances)

def plotData(fileName):
    xVals, yVals = getData(fileName)
    xVals = pylab.array(xVals)
    yVals = pylab.array(yVals)
    xVals = xVals*9.81  # acc. due to gravity
    pylab.plot(xVals, yVals, 'bo', label = 'Measured displacements')
    pylab.title('Measured Displacement of Spring')
    pylab.xlabel('|Force| (Newtons)')
    pylab.ylabel('Distance (meters)')

def fitData(fileName):
    xVals, yVals = getData(fileName)
    xVals = pylab.array(xVals)
    yVals = pylab.array(yVals)
    xVals = xVals*9.81
    a, b = pylab.polyfit(xVals, yVals, 1)
    estYVals = a*pylab.array(xVals) + b
    k = 1/a
    pylab.plot(xVals, estYVals, label = 'Linear fit, k = ' + str(round(k, 5)))
    pylab.legend(loc = 'best')
def tryFits(fName):
    distances, heights = getTrajectoryData(fName)
distances = pylab.array(distances)*36
totHeights = pylab.array([0]*len(distances))
for h in heights:
    totHeights = totHeights + pylab.array(h)
pylab.title('Trajectory of Projectile (Mean of 4 Trials)')
pylab.xlabel('Inches from Launch Point')
pylab.ylabel('Inches Above Launch Point')
meanHeights = totHeights/float(len(heights))
pylab.plot(distances, meanHeights, 'bo')
a,b = pylab.polyfit(distances, meanHeights, 1)
altiltudes = a*distances + b
pylab.plot(distances, altitudes, 'r',
           label = 'Linear Fit')
a,b,c = pylab.polyfit(distances, meanHeights, 2)
altiltudes = a*(distances**2) + b*distances + c
pylab.plot(distances, altitudes, 'g',
           label = 'Quadratic Fit')
pylab.legend()

def rSquare(measured, estimated):
    """measured: one dimensional array of measured values
    estimate: one dimensional array of predicted values""
    EE = ((estimated - measured)**2).sum()
mMean = measured.sum()/float(len(measured))
MV = ((mMean - measured)**2).sum()
return 1 - EE/MV

def tryFits1(fName):
    ...
    altitudes = a*distances + b
    altitudes = a*(distances**2) + b*distances + c
    pylab.plot(distances, altitudes, 'g',
               label = 'Quadratic Fit' + ', R2 = ' + str(round(rSquare(meanHeights, altitudes), 4)))
    a,b,c = pylab.polyfit(distances, meanHeights, 2)
    altitudes = a*(distances**2) + b*distances + c
    pylab.plot(distances, altitudes, 'g',
               label = 'Quadratic Fit' + ', R2 = ' + str(round(rSquare(meanHeights, altitudes), 4)))
    pylab.legend()
6.00SC Introduction to Computer Science and Programming
Spring 2011

For information about citing these materials or our Terms of Use, visit: http://ocw.mit.edu/terms.