2013

INTRODUCTION TO NUMERICAL ANALYSIS

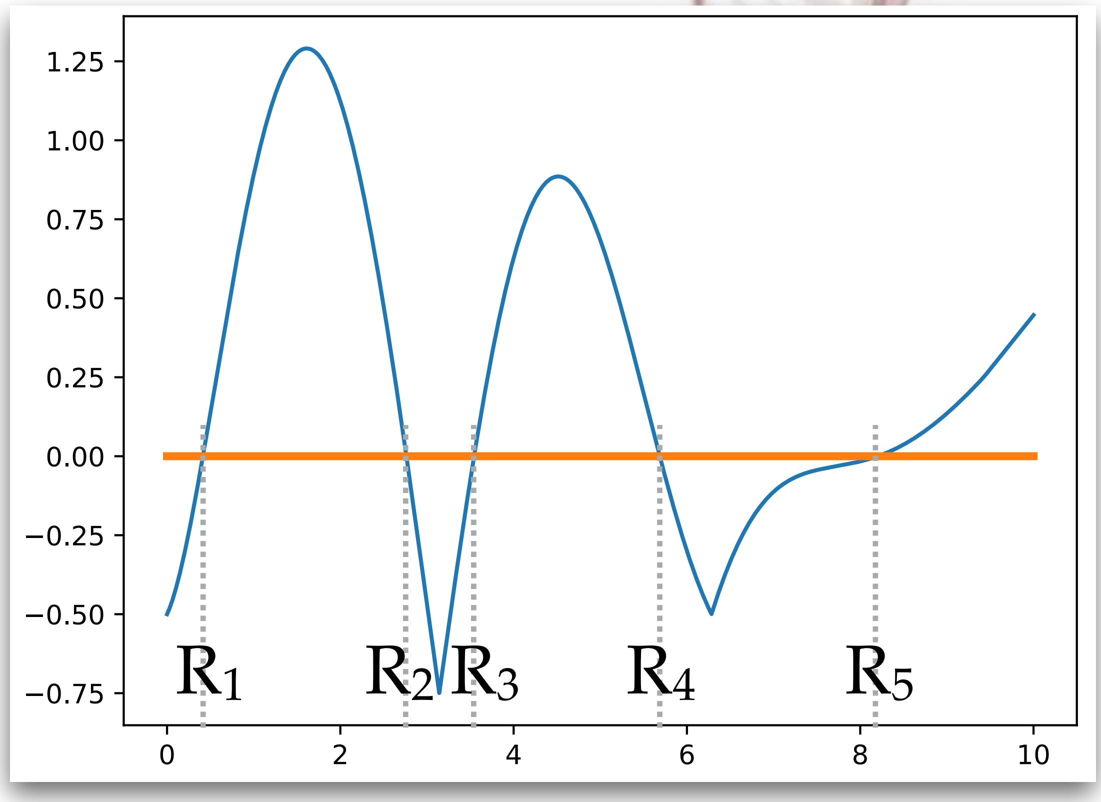
Assignment 4

ASSIGNMENT 4-1

Finding the roots

- Suppose we have a function of ***x*** (**0<*x*<10**) can be expressed as

$$f(x) = \left| \frac{\sin(x)}{\left(\frac{x}{2\pi}\right)^x + \frac{\pi}{8}} \right| - \frac{x}{2\pi} + \left(\frac{x}{2\pi}\right)^2 - \frac{1}{2}$$

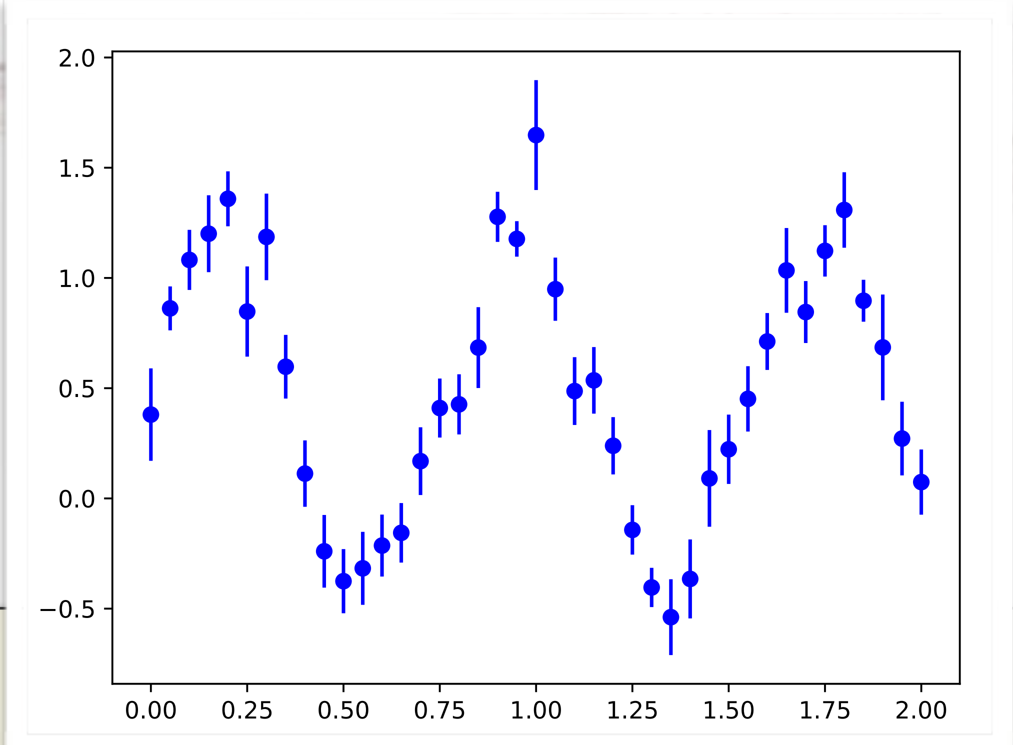


- Construct a function which takes no input arguments and return an array of 5 elements which consist of the 5 roots of the function above ie. [R_1,R_2,R_3,R_4,R_5]. You can use any numeric method presented in the main lecture.

ASSIGNMENT 4-2

Parameter extraction with curve fit

- Suppose we have the following data points:




```
xmin, xmax, xbinwidth = 0.,2.,0.05
vx = np.linspace(0.,2.,41)
vy = np.array(
[+0.3802, +0.8620, +1.0819, +1.2007, +1.3590, +0.8477, +1.1862, +0.5973,
 +0.1126, -0.2399, -0.3754, -0.3172, -0.2139, -0.1561, +0.1690, +0.4099,
 +0.4267, +0.6841, +1.2772, +1.1771, +1.6481, +0.9490, +0.4869, +0.5355,
 +0.2388, -0.1428, -0.4039, -0.5386, -0.3652, +0.0908, +0.2229, +0.4515,
 +0.7118, +1.0343, +0.8454, +1.1228, +1.3083, +0.8966, +0.6850, +0.2714,
 +0.0741])
vyerr = np.array(
[+0.2097, +0.0996, +0.1363, +0.1744, +0.1246, +0.2046, +0.1963, +0.1444,
 +0.1506, +0.1648, +0.1455, +0.1655, +0.1405, +0.1350, +0.1537, +0.1338,
 +0.1364, +0.1834, +0.1136, +0.0805, +0.2492, +0.1433, +0.1538, +0.1510,
 +0.1299, +0.1121, +0.0890, +0.1720, +0.1791, +0.2193, +0.1572, +0.1482,
 +0.1290, +0.1922, +0.1405, +0.1165, +0.1712, +0.0951, +0.2398, +0.1670,
 +0.1479,])
```

assignment-402.py (partial)

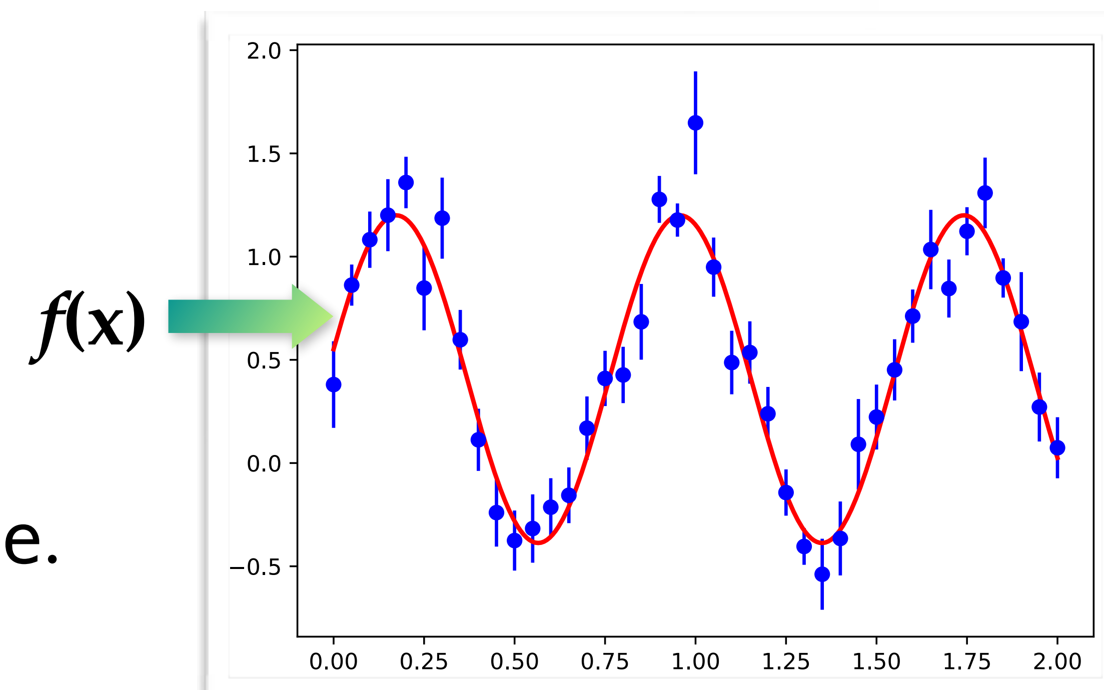
ASSIGNMENT 4-2 (CONT.)

- And the data points can be modeled by the following function:

$$f(x) = N\sin(kx + \phi) + b$$

  where ***N*,*k*,*ϕ***, and ***b*** are the parameters to be determined.

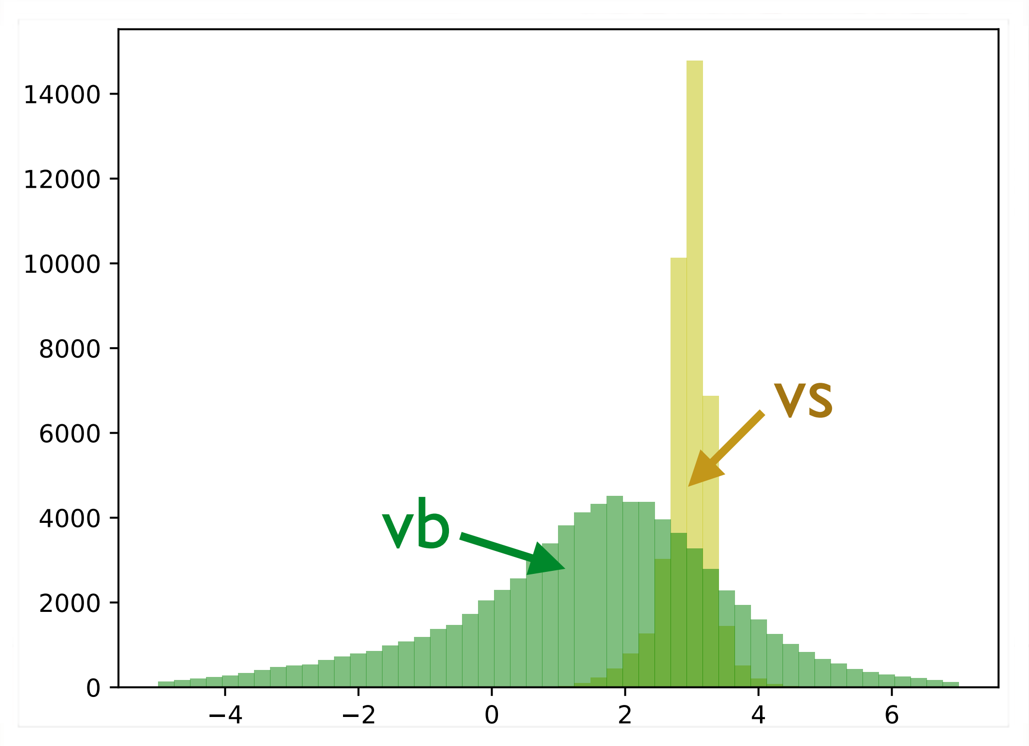
- Construct a function which takes no input arguments, perform a χ^2 fit to the data points with the model given above, and return an array of 4 elements which consist of the best fitted parameters used in the function above ie. [***N*,*k*,*ϕ*,*b***]. You can use either `minimize()` or `curve_fit()` from `scipy.optimize`.



ASSIGNMENT 4-3

Finding the best window

- Consider a "signal" data (**vs**) and a "background" data (**vb**) can be generated below:

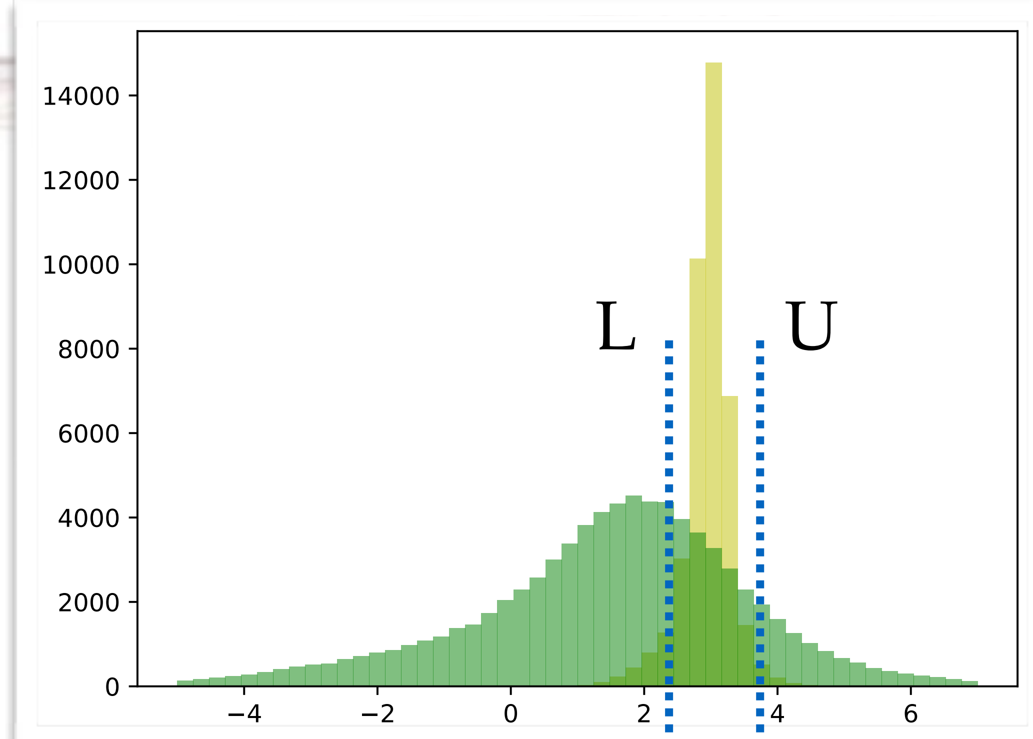



```
nbase = 10000
np.random.seed(1234)
vs =
np.hstack((np.random.randn(nbase*3)*0.2+3.,np.random.randn(nbase)*0
.6+2.8))
vb =
np.hstack((np.random.randn(nbase*4)*1.2+2.,np.random.randn(nbase*4)
*2.8+1.0))

# uncomment below for displaying the histogram plots
plt.hist(vs, bins=50, color='y', range=(-5.,7.), alpha=0.5)
plt.hist(vb, bins=50, color='g', range=(-5.,7.), alpha=0.5)
plt.show()
```

assignment-403.py (partial)

ASSIGNMENT 4-3 (CONT.)



- Construct a function which takes no input arguments, try to look for the best lower and upper bound according to the following criteria:
  - N_s: count how many "signal" (from **`vs`**) that are within ***L*<`vs`<*U***.
  - N_b: count how many "background" (from **`vb`**) that are within ***L*<`vb`<*U***.
  - Maximize $Z = N_s/\sqrt{N_s + N_b}$ to find the best ***L*** and ***U***, up to 2 digits below zero, e.g. **2.34**.
- The function should just return an array of 2 elements, ie. [***L***, ***U***].