Computational Physics, 2nd edition

Exercises for Chapter 11

Exercise 11.1: The photoelectric effect:

In the online resources you will find a file called `millikan.txt`. The file contains two columns of numbers, giving the x and y coordinates of a set of data points.

a) Write a program or modify the one in `hooke.py` to fit these data with a straight line, print out the values of the slope m and intercept c, and make a plot showing the data and the fitted line on the same axes.

These data come from a historic experiment by Robert Millikan that measured the *photoelectric effect*. When light of an appropriate wavelength shines on the surface of a metal, the photons in the light can strike conduction electrons in the metal and, sometimes, eject them from the surface into the free space above. The energy of an ejected electron is equal to the energy of the photon that struck it minus a small amount ϕ called the *work function* of the surface, which represents the energy needed to remove an electron from the surface. The energy of a photon is $h\nu$, where h is Planck's constant and ν is the frequency of the light, and we can measure the energy of an ejected electron by measuring the voltage V that is just sufficient to stop the electron moving. Then the voltage, frequency, and work function are related by the equation

$$V = \frac{h}{e}\nu - \phi,$$

where e is the charge on the electron. This equation was first given by Albert Einstein in 1905.

b) The data in the file `millikan.txt` represent frequencies ν in hertz (first column) and voltages V in volts (second column) from photoelectric measurements of this kind. Using the equation above and your results from the fit, and given that the charge on the electron is 1.602×10^{-19} C, calculate from Millikan's data a value for Planck's constant. Compare your value with the accepted value of the constant, which you can find online or in books. You should get a result within a couple of percent of the right answer.

This calculation is essentially the same one that Millikan himself performed to determine the value of Planck's constant in 1915, although, lacking a computer, he fitted his straight line to the data by eye. In part for this work, Millikan was awarded the Nobel prize in physics in 1923.

Exercise 11.2: The file `kmeans.txt` in the online resources contains the data that appear in Fig. 11.13. Write a program to read these data and carry out the k-means clustering algorithm on them to divide the data into two groups. You will have to perform the following steps:

a) Read the data and put them in a list or array.

b) Set up another array to store the variables z_i that say which group each data point belongs to. Though the groups were labeled 1 and 2 in our description of the algorithm, you might want to use 0 and 1 in your program, in traditional Python style. Assign random groups to all the data points using for instance the `randrange` function (Section 10.1).

c) Calculate the mean value of the data points in each of the two groups.

d) Go through each data point in turn and calculate how far it is from each mean, i.e., calculate $|x_i - \mu_1|$ and $|x_i - \mu_2|$, then assign the data point to whichever group has the smaller distance. (Again, your groups may be called 0 and 1, rather than 1 and 2.) Note whether the new group is different from the old one.

e) If any of the new group assignments were different from the old ones, go back to step (c) and repeat the whole process.

f) If none of the assignments changed then they have reached their final values. Now calculate the value of σ from Eq. (11.68), then make a histogram of the original data and plot on the same axes the two Gaussian distributions, Eq. (11.61). You should get a plot that looks similar to Fig. 11.13. Hint: To make the clearest plot, you should also scale the heights of the Gaussians to match the fraction of measurements that fall in each group.

Exercise 11.3: Earthquakes

The file `earthquakes.txt` in the online resources contains x, y coordinates for each of the 783 earthquakes shown in Fig. 11.14. Write a program to read these data from the file and then perform two-dimensional k-means clustering on them, dividing the data into four groups. Make a scatter plot of the data points, similar to Fig. 11.14, in which the color of each point indicates which group it is assigned to. You should get results similar to those shown in the figure.

Exercise 11.4: EM algorithm for exoplanet sizes

The file `exoplanets.txt` in the online resources contains data on various properties of known exoplanets, as described in Example 509.

a) Read the data on planetary radii (which are the fifth column of numbers in the file) and make a histogram of the results.

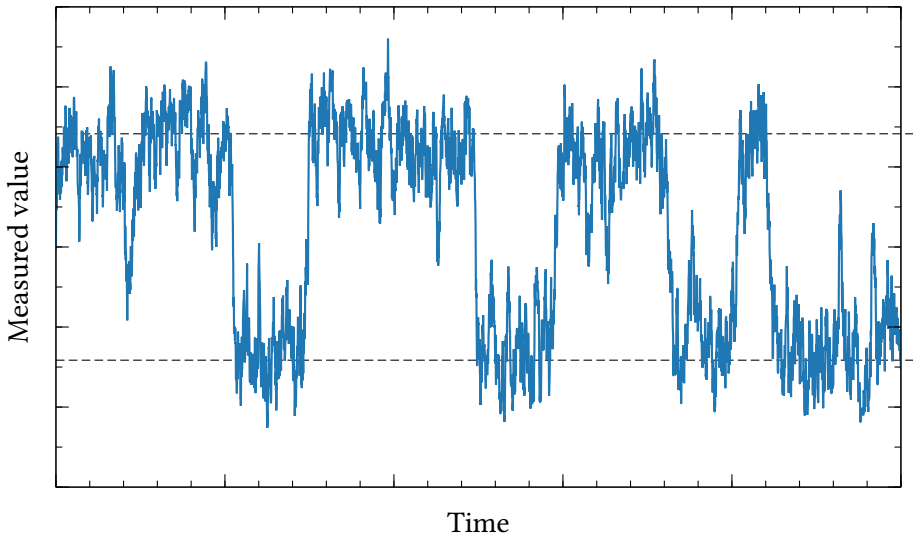
b) Fit the data using a Gaussian mixture model with two Gaussians. What are the average radii of the two groups found by the algorithm?

c) Plot the fitted Gaussians on the same axes as the histogram and compare your results with Fig. 11.16. You should get good agreement.

d) For each of the first 20 planets in the file, print out their radii and the probabilities q_z^i that each planet is a rocky world (rather than a gas giant). You should find that the algorithm is quite confident in its identification of each planet.

Exercise 11.5: A bistable physical system

A bistable system is one with two energy minima, so that it can be stable (or metastable) in either one. At finite temperature such a system can switch back and forth between the two states as it undergoes thermal fluctuations, like this:



(The dashed lines represent the typical values in each of the minima.)

This kind of behavior is encountered widely in condensed matter and biophysical systems. An example is the Ising model of Exercise 10.10 (page 489) and similar magnetic systems: because of its up-down symmetry, the Ising model (in its ferromagnetic phase) has two equilibrium states, one with net positive magnetization and one with net negative. However, it is not always straightforward to say which of the two states the system is in at any moment. Because of the noise in the system, the two can blur together and become hard to distinguish. We can resolve the dispute using a mixture model.

a) The file `bistable.txt` contains time-series data of 100 000 measurements for such a system. Make a histogram of the numbers in the file. You should find that they have a two-peaked distribution.

b) Fit the data to a mixture of two Gaussian distributions using the EM algorithm.

c) Add code to your program to produce a plot as follows. Plot the first 5000 measurements as a time-series, using the `scatter` function to represent the data as a set of points. The `scatter` function allows you to specify an optional argument `c=colors`, where `colors` is a list or array containing floating-point values that specify the color of each point in the plot, in order. Color each point with the value of the quantity q_1^i that represents the probability that the point falls under Gaussian number 1 (or Gaussian number 0 if you numbered yours in Python style, starting from zero). You should see that the algorithm has neatly labeled the points according to which state it thinks the system is in at that time.

Exercise 11.6: The Poisson mixture model

In the hallway outside the author's office there is a machine that dispenses candy at the push of a button. Allegations have swirled for some time that the machine is not fair. Sometimes it dispenses a generous handful of candies, sometimes it dispenses a miserly few. The file `candies.txt` in the online resources contains a record of the number of candies dispensed by the machine on 853 pushes of the button. (I am not making this up. These are real data.) Let us test the truth of the rumors against hard data.

a) Make a histogram of the numbers in the file. You should find that the distribution is somewhat suspicious. It is clearly non-Gaussian, although on the other hand it does not provide strong evidence of bimodal behavior (generous versus miserly output).

b) Let us look for bimodal behavior by fitting a mixture model. The data are integers, not floats, and for such data an appropriate model is a mixture of two Poisson distributions, which describe the probability of dispensing n candies given two means μ_1, μ_2:

$$P(n|\mu_1) = \frac{\mu_1^n}{n!}\mathrm{e}^{-\mu_1}, \qquad P(n|\mu_2) = \frac{\mu_2^n}{n!}\mathrm{e}^{-\mu_2}.$$

By following the steps in Section 11.6.4 for the Gaussian model, but replacing the Gaussian distribution with the Poisson distribution, show that the equations of the EM algorithm for the Poisson model are

$$q_z^i = \frac{\gamma_z \mu_z^{n_i}\mathrm{e}^{-\mu_z}}{\sum_z \gamma_z \mu_z^{n_i}\mathrm{e}^{-\mu_z}}, \qquad \mu_z = \frac{\sum_i q_z^i n_i}{\sum_i q_z^i}, \qquad \gamma_z = \frac{1}{N}\sum_i q_z^i.$$

c) Write a program to fit the Poisson mixture model to the data in the file using the EM algorithm. What are the means of the two distributions? What fraction of the time does the dispenser dispense generous versus miserly allocations of candy?

d) Add code to your program to show on the same axes the histogram of the measurements and the overall distribution predicted by the model:

$$P(n|\mu,\gamma) = \sum_z \gamma_z P(n|\mu_z).$$

How well do the distribution and the histogram match? Do your results support the hypothesis that the machine indeed has two modes of operation, one generous and one miserly?