Phys 512 Problem Set 3

Due on github Monday September 30 at 11:59 PM. You may discuss problems, but everyone must write their own code.

Problem 1: We'll do a linear least-squares fit to some real data in this problem. Look at the file dish_zenith.txt. This contains photogrammetry data for a prototype telescope dish. Photogrammetry attempts to reconstruct surfaces by working out the 3-dimensional positions of targets from many pictures (as an aside, the algorithms behind photogrammetry are another fun least-squares-type problem, but beyond the scope of this class). The end result is that dish_zenith.txt contains the (x,y,z) positions in mm of a few hundred targets placed on the dish. The ideal telescope dish should be a rotationally symmetric paraboloid. We will try to measure the shape of that paraboloid, and see how well we did.

a) Helpfully, I have oriented the points in the file so that the dish is pointing in the $+z$ direction (in the general problem, you would have to fit for direction the dish is pointing in as well, but we will skip that here). For a rotationally symmetric paraboloid, we know that

$$z - z_0 = a\left((x - x_0)^2 + (y - y_0)^2\right)$$

and we need to solve for x_0, y_0, z_0, and a. While at first glance this problem may appear non-linear, show that we can pick a new set of parameters that make the problem linear. What are these new parameters, and how do they relate to the old ones?

b) Carry out the fit. What are your best-fit parameters?

c) Estimate the noise in the data, and from that, estimate the uncertainty in a. Our target focal length was 1.5 metres. What did we actually get, and what is the error bar? In case all facets of conic sections are not at your immediate recall, a parabola that goes through $(0, 0)$ can be written as $y = x^2/(4f)$ where f is the focal length. When calculating the error bar for the focal length, feel free to approximate using a first-order Taylor expansion.

BONUS: Of course, we have just assumed that the dish is circularly symmetric. In real life, we'd obviously need to check that. The leading order correction would give us a dish that looked like $z = ax'^2 + by'^2$ if the vertex (bottom) of the dish was at $(0, 0, 0)$ *and* our coordinate system was aligned with the principal axes of the dish. We won't usually have the benefit of being aligned like that; instead we'll usually be rotated by some (unknown) angle θ, so our observed coordinates x, y will be related to the original coordinates x', y' by a rotation: $x = \cos(\theta)x' + \sin(\theta)y'$ and $y = -\sin(\theta)x' + \cos(\theta)y'$. Find the focal lengths of the two principal axes (and don't forget we can still have arbitrary offsets x_0, y_0, z_0). Is the dish round?

Problem 2: Let's start with data d and a diagonal noise matrix N that we

rotate into a new basis with an orthogonal matrix V. In particular, we have

$$\tilde{\mathrm{N}} = \mathrm{VNV}^T$$

$$\tilde{d} = \mathrm{Vd}$$

Assume there is no signal present in the data, only noise, so

$$\langle d \rangle = 0$$

$$\langle d_i d_j \rangle = \mathrm{N}_{ii}\delta_{ij}$$

Show analytically that under these assumptions we have:

$$\left\langle \tilde{d}_{ij} \right\rangle = \tilde{\mathrm{N}}_{ij}$$

This result confirms that we can write down χ^2 in the presence of correlated noise the same way we wrote it with uncorrelated noise, as long as our noise matrix contains the correlations.

Problem 3:

Part a) Generate correlated noise using the Cholesky decomposition of a Gaussian correlation function with a σ of 20 samples and a length of 1000 samples. You'll want to add a small offset to the diagonal (let's use 1e-2, so the diagonal entries of your matrix are 1.01) to avoid roundoff error making the matrix singular. You might find the function

`scipy.linalg.toeplitz`

useful, but it is by no means required.

Part b) We'll now try to use a stretch of our simulated data to estimate unmeasured points. Say you meant to measure 20 data points, but your equipment during the run, and the last value was zero. We want to find the maximum likelihood value of that point given the data we did measure. If we did have the last point, we know that χ^2 for the whole data set is

$$\chi^2 = d^T \mathrm{C}^{-1} d$$

We can use the framework of least-squares fitting to estimate the missing point by setting

$$d = d_{measured} - \mathrm{A}m$$

where $d_{measured}$ is our measured data values *with a zero* in the final location, A is a column vector of zeros, but with a -1 in the final location, and m is the value of our missing data point. You'll see that forming $d - \mathrm{A}m$ now gives us the full data vector where we've put in m for the final data point. We can group the least-squares solution as follows:

$$m = \left[\left(\mathrm{A}^{\mathrm{T}} \mathrm{C}^{-1} \mathrm{A} \right)^{-1} \mathrm{A}^{\mathrm{T}} \mathrm{C}^{-1} \right] d$$

where we've treated used our covariance C as the noise. If we work out the term in square brackets, we have a set of weights we apply to the data to predict the missing value. Show that the term inside the inverse is just the bottom-right element of $C^{-}1$ and the term $A^{T}C^{-1}$ is just the last row of the covariance not including the final entry (because we set the final unmeasured value of d to be zero). That means the weights become

$$-C^{-1}[-1,:-1]/C^{-1}[-1,-1]$$

Print out these 19 weights for the covariance matrix from part A.

Part C: Use the weights you calculated to reconstruct the data you simulated in part A. Each point in the output (starting with index 20) should be the sum of your weights times the previous 19 data points. Plot the reconstructed data over your original data - what is the RMS error between your reconstruction and the true data?

Note - this procedure is called *kriging* and was originally developed to predict the gold content of mines around Johannesburg in South Africa. Every data point required digging another mine, so there was a powerful incentive to squeeze every drop of information out of the data they had. Kriging is a powerful tool for predicting unmeasured/missing data, interpolating between unevenly sampled points, and many other uses. Much of the information online is at best confusing (and often wrong), but you should be able to work out what you should do by coming back to the least-squares equations.