Computational Physics / PHYS-UA 210 / Special Problem Set
Due December 15, 2023

You *must* label all axes of all plots, including giving the *units*!!

1. Exercise 5.5 of Newman
2. Exercise 5.10 parts (a) and (b) of Newman, but use the potential of a pendulum of length $l = 2$ instead.
3. Exercise 5.13 in Newman. Then: (d) Perform the calculation using Gauss-Hermite quadrature (`scipy` can give you the right roots and weights to use). Can you make an exact evaluation (meaning zero approximation error) of the integral?
4. Exercise 5.20 of Newman
5. This problem demonstrates an application of linear algebra to signal analysis. Download a "signal" as a function of time from this file. Assume that all the measurements have the same uncertainty, with a standard deviation of 2.0 in the signal units.
   (a) Plot the data.
   (b) Use the SVD technique to find the best third-order polynomial fit in time to the signal. Pay attention to the scaling of the independent variable (time).
   (c) Calculate the residuals of the data with respect to your model. Argue that this is not a good explanation of the data given what you know about the measurement uncertainties.
   (d) Try a much higher order polynomial. Is there any reasonable polynomial you can fit that *is* a good explanation of the data? Define "reasonable polynomial" as whether the design matrix has a viable condition number.
   (e) Try fitting a set of sin and cos functions plus a zero-point offset. As a Fourier series does, use a harmonic sequence with increasing frequency, starting with a period equal to half of the time span covered. Does this model do a "good job" explaining the data? Are you able to determine a typical periodicity in the data? You may have noticed a periodicity from the first plot.

   This last fit is a version of the *Lomb-Scargle* technique for detecting variability in unevenly sampled data, which is designed to be a very close approximation to fitting with a set of Fourier modes. Implementing the method the way we do here (explicitly decomposing the design matrix) is not the usual method, since it is slower than other techniques, but it is the simplest to implement.
6. Here we will perform a simple Principal Components Analysis on a real data set. Note this problem is not as daunting as it might look at first; there are many words because I am

walking you through all the steps one by one. Download this file which contains the central optical spectra of 9,713 nearby galaxies from the Sloan Digital Sky Survey (note that although this file is 150 Mb, this is a small sample! larger data sets exist of millions, though they are lower quality). I have done some the work for you by interpolating all of the spectra onto the same restframe wavelength grid. Now do the following:

(a) Read the data in using the `astropy` package, in particular using `astropy.io.fits`. This data set is a special format called the Flexible Image Transport System (FITS) format, common in astronomy. Its only virtue is that it is a standard in astronomy. You should be able to `pip install astropy`, and then:

```
hdu_list = astropy.io.fits.open('specgrid.fits')
logwave = hdu_list['LOGWAVE'].data
flux = hdu_list['FLUX'].data
```

`logwave` will be $\log_{10}\lambda$ for λ in Angstroms. `flux` will be in 10^{-17} erg s^{-1} cm^{-2} Å^{-1}, and is the spectrum. Plot a handful of the galaxies. Being a physicist and knowing something about the transitions in the Hydrogen atom, do you notice any features that relate to it?

(b) There are two processing steps that will make the PCA more meaningful. First, all of these galaxies are at different distances, so their fluxes span a large dynamic range; so first normalize all the fluxes so their integrals over wavelength are the same. It is close enough to sum over the values to estimate the integral (i.e. don't worry about the fact that $\Delta\lambda$ varies across the grid). Keep each normalization available.

(c) Second, the mean flux at every wavelength is positive; this will mean that a PCA will spend an eigenvector to explain the mean offset from zero. Instead, we will first subtract off the mean $\vec{f}_m$ of the normalized spectra. This will leave residuals $\vec{r}_i = \vec{f}_i - \vec{f}_m$ of all the galaxies i varying around zero. Keep the mean spectrum available.

(d) Now perform the PCA. The idea of the PCA is to find the eigenvectors of the covariance matrix of the distribution. This covariance matrix *can* be calculated as follows:

$$\mathbf{C} = \frac{1}{N_{\mathrm{gal}}} \sum_{ij} \vec{r}_i \vec{r}_j \qquad (1)$$

where i and j index the galaxies. If I recast the residuals as a matrix R_{ij} this is $\mathbf{C} = \mathbf{R} \cdot \mathbf{R}^T$. So construct this matrix (it should be $N_{\mathrm{wave}} \times N_{\mathrm{wave}}$), and find its eigenvectors. Make a plot of the first five eigenvectors.

(e) It is also possible to find these eigenvectors directly from $\mathbf{R}$ using SVD. Consider the linear problem, which finds a set of coefficients $\vec{x}$ to multiply the given spectra by, to explain some spectrum $\vec{f}$:

$$\mathbf{R} \cdot \vec{x} = \vec{f} \qquad (2)$$

We know the SVD decomposition of $\mathbf{R}$ yields a rotation $\mathbf{V}$ into the space where the covariance matrix of the uncertainties in $\vec{x}$ is diagonal. This covariance matrix is $\mathbf{R}^T \cdot \mathbf{R}$. Another way to see this is that the covariance can be written:

$$\mathbf{R}^T \cdot \mathbf{R} = \mathbf{V} \cdot \mathbf{W} \cdot \mathbf{U}^T \cdot \mathbf{U} \cdot \mathbf{W} \cdot V^T \tag{3}$$

The central structure is a diagonal matrix, and $\mathbf{V}$ is unitary (its inverse is its transpose), so the matrix $\mathbf{V}$ is composed of the right eigenvectors of $\mathbf{R}^T \cdot \mathbf{R}$ (with eigenvalues which are the square of the singular values). So now find the eigenvectors using an SVD decomposition of $\mathbf{R}$ and show that the vectors are equivalent to what you found before. Compare the computational cost of this method to the method in the previous bullet.

(f) Can you think of reasons you might want to use SVD instead of constructing the covariance matrix and finding its eigenvectors? How does the condition number of $\mathbf{C}$ compare to that of $\mathbf{R}$?

(g) A common use case for PCA is to have components that can compactly approximate data. This can be a form of compression (e.g. storing only the first few PCA coefficients of a spectrum instead of the full spectrum) though it can also be useful in fitting models (e.g. the redshift-fitting algorithm of SDSS uses a PCA basis fit at each redshift and minimizes χ^2 over redshift). Each original input spectrum can be expressed as the mean spectrum, plus the full set of coefficients c_i multiplying the eigenspectra, times the original normalization. But you can see what happens if you keep the first $N_c = 5$ coefficients. Create approximate spectra based on keeping only the first five coefficients. Note that if you think about the math, you don't need to perform *another* matrix inversion or decomposition to find the coefficients, you just have to rotate the spectra into the eigenspectrum basis.

(h) Plot c_0 vs c_1 and c_0 vs c_2.

(i) Now try varying $N_c = 1, 2, \ldots, 20$ and calculate the squared fractional residuals between the spectra and the reconstituted, approximate spectra as a function of N_c. You should see the squared residuals declining. How small is the fractional error for $N_c = 20$?

7. Implement Brent's 1D minimization method. Test it on this function: $y = (x-1)^4 \exp(-x^2/2)$. Compare to the `scipy.optimize.brent` implementation results.

8. Use the `scipy.optimize` package to minimize the quadratic function:

$$f(x, y, z) = (x-1)^2 + (y-2)^2 + (z+3)^2 \tag{4}$$

(whose minimum should be obvious already so you can check the right answer)! Use three different methods: CG, L-BFGS-B, and Newton-CG. Note that they require the derivatives, and in the case of Newton-CG, the Jacobian too. Check how many iterations each takes—comment on the differences. Can you speculate on reasons why you would ever use methods *other* than Newton-CG?