# Homework 7

Due Friday, 16 Mar, at the beginning of class. Print out and turn in any Python/IDL/Matlab code you write.

Reading Assignment: Read Wall & Jenkins, Ch. 8.1-8.3, on data transformations and Fourier transforms.

## 1 Spectral Line Analysis

This problem was the Comps I problem from January 2008. Test yourself by attempting to complete it in less than one hour.

### Potentially Useful Formulas

A symmetric positive definite matrix

$$M = \begin{pmatrix} a & b \\ b & c \end{pmatrix}$$

has an inverse

$$M^{-1} = \frac{1}{ac - b^2} \begin{pmatrix} c & -b \\ -b & a \end{pmatrix}.$$

Common probability distribution functions (PDF's):

Binomial

$$f(x; p, q) = \frac{n!}{x!(n-x)!} p^x q^{n-x}$$

Poisson

$$f(x; \mu) = \frac{e^{-\mu} \mu^x}{x!}$$

Normal (Gaussian)

$$f(x; \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]$$

Chi-square

$$f(x; \nu) = \frac{x^{(\nu/2-1)}}{2^{\nu/2}\Gamma(\nu/2)} \exp(-x/2)$$

There are many problems in astrophysical and planetary sciences (and more broadly, in science in general) that involve estimating the amplitude of a signal in the presence of noise. Our goal here is to estimate the amplitude of such a signal, and characterize the uncertainty in our estimate.

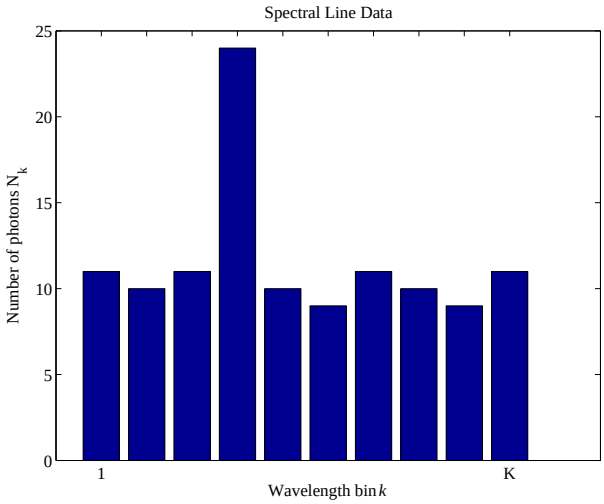


Figure 1: Simulated spectral line data, with the signal assumed to be a delta function in a known wavelength bin $l$ (fourth bin).

Take for example the spectral line data in Figure 1 (This might also be a stellar source in a CCD, or a geological feature in an image of a planet surface). The vertical axis is number of photons $N_k$, and the horizontal axis is wavelength bin $k$. The data are the set $\{N_k\}$ for $K$ bins.

Assume for simplicity that the spectral line feature has a known central wavelength and is sufficiently narrow that it can be represented by a delta function. Assume also that the background has a flat spectrum of unknown amplitude, and that the data are independent (and therefore uncorrelated).

Then the model for the expected number of photons in bin $k$ is given by

$$D_k = A\delta_{kl} + B$$

Where $A$ is the signal amplitude and $B$ is the background, both in units of photon counts, and the signal is in (known) bin $l$.

a) Given the model for the expected number of photons $D_k$ in bin $k$, write down the probability distribution function (PDF) for the number of photons $N_k$ in bin $k$, $P(N_k|D_k)$

b) Write down the (discrete) likelihood function for the data, given the model parameters $A, B$, $\mathcal{L}(A, B) \equiv P(\{N_k\}|A, B)$, where $\{N_k\}$ is the set of $K$ data.

c) Now, analytically solve for the maximum likelihood (ML) values of the parameters, $\hat{A}, \hat{B}$, by maximizing $\ln \mathcal{L}(A, B)$. Interpret your results in terms of fundamental statistical concepts.

Now you need to characterize the uncertainties in your ML estimators $\hat{A}, \hat{B}$.

d) Calculate the $2 \times 2$ curvature matrix for the parameters $A, B$, evaluated at the ML values $\hat{A}, \hat{B}$, as an estimate of the Fisher Information matrix, $\hat{F}$.

e) Using the Fisher matrix estimator $\hat{F}$, find the parameter covariance matrix

$$C = \begin{pmatrix} \sigma_A^2 & \sigma_{AB} \\ \sigma_{AB} & \sigma_B^2 \end{pmatrix}$$

f) Interpret your results for the marginal uncertainties $\sigma_A^2$ and $\sigma_B^2$, in terms of fundamental statistical concepts.

g) Are your parameter estimators $\hat{A}, \hat{B}$ correlated? Conceptually explain why you expect them to be correlated or uncorrelated.

## 2 Minimum-$\chi^2$ model fitting of a 2-D Gaussian to a CCD Stellar Image

In this problem you will use minimum-$\chi^2$ parameter estimation perform parameter fit and uncertainty estimation for a non-linear model using Levenberg-Marquardt minimization algorithm discussed in class. The data is a CCD image of a star field taken at the Sommers Bausch Observatory. Your goal is to fit a 2-D Gaussian flux model to a single star in the image.

You can download the FITS file with the image, `f61final.fits`, from the `Files/Homework/Data` folder on Canvas.

The data units for the image array are integer outputs of an analog-to-digital converter in in the CCD readout electronics. These units are known as DN (data number) or ADU (analog digital units). (Note: this image has been flat-fielded, so DN are not integers here.)

Simply speaking, the CCD performs the following operations: it converts a photon to an electron (with less than unity efficiency), integrates or stores electrons, transfers the electrons to the edge of the device during readout, and converts electron charge to a voltage. The voltage is then converted to DN by an analog-to-digital converter.

The *gain* of the CCD is typically determined (or given in the FITS header) in units of "electrons per DN". You must therefore be careful which units you are working in, particularly when calculating Poisson noise (due to fluctuations in electron counts, *not* DN!) The gain is $G = 2.72$ electrons per DN for this image.

Read in the FITS file data and header information using the appropriate functions in Python/IDL/Matlab. Select a $10 \times 10$ pixel subset of the data around the star of interest.

In Python:

First you need to download and install the `pyfits` library from the Space Telescope Science Institute (STScI): http://www.stsci.edu/institute/software_hardware/pyfits/release

Then,

```
>>> import pyfits
>>> hdulist = pyfits.open('f61final.fits')
```

```
>>> img = hdulist[0].data
>>> hdulist.close()
>>> data = img[54:64,136:146]
```

In IDL:

```
IDL> img=readfits('f61final.fits',hdr)
IDL> data=img(136:145,54:63)
```

In Matlab:

```
>> img = fitsread('f61final.fits');
>> hdr = fitsinfo('f61final.fits');
>> data = img(55:64,137:146);
```

(Note: the selected star is the same in Python, IDL and Matlab. The different indices reflect the fact that Matlab array indexing starts at 1, IDL and Python at 0, and that Matlab and Python arrays are column major, vs. row major for IDL.)

There are several sources of noise in CCD images, which we will cover in more detail later in the semester:

- The CCD introduces noise in the process of readout (called Readout Noise, or RON) and for many observations this places a fundamental limit on the observability of faint objects.
- Dark current is an excess electron count in each pixel that increases linearly with exposure time, and is due to thermal noise in the detector. It has noise associated with it.
- Sky noise is unwanted photons (and associated Poisson noise) due to atmospheric airglow at visible wavelengths.
- Last but not least, there is Poisson noise in the photon (electron) counts from the astronomical source itself.

In the CCD data analysis homework later in the semester, you will estimate values for all these sources of noise by analyzing bias and dark frames. For now, however, take as a given that

$$\sigma_{\rm RON} = 20 \text{ electrons}$$

is the readout noise (square root of the variance) in units of electrons. Neglect the dark current noise:

$$\sigma_{\text{dark current}} = 0 \text{ electrons}$$

since this is a relatively short exposure.

a) Estimate the noise in each pixel. Let $D_k$ be the DN of pixel $k$ in the image, and $G$ be the gain, in electrons per DN. Write down the estimated noise $\sigma_k$ in pixel $k$, due to photon statistics

(from astronomical source and sky noise), and readout noise, assuming the sources of noise are independent, in units of DN.

b) Write down the $\chi^2$ statistic for the data which you will minimize. Assume a 5-parameter circularly symmetric Gaussian model for the stellar source,

$$g(x_k, y_k; \vec{\theta}) = B + A \exp\left(\frac{-[(x_k - x_0)^2 + (y_k - y_0)^2]}{2\sigma^2}\right)$$

where

$$\vec{\theta} = \{A, B, x_0, y_0, \sigma\}.$$

are the five model parameters. (Note: $\sigma$ is the width of the Gaussian, not to be confused with the uncertainty in the data.)

c) Find the minimum-$\chi^2$ values of the parameters using the Levenberg-Marquardt method or Newton method. You do NOT need to write your own code to do this. In Python, try `mpyfit.py`. In IDL, use `mpfit2dfun`. In Matlab, use `fminunc`. Report your minimum-$\chi^2$ values of the parameters, $\hat{\vec{\theta}}$, your value of $\chi^2$ at miminim, $\chi^2_{\rm min}$, and the PTE for this $\chi^2$ value, as a goodness-of-fit check. Is the model a good fit to the data?

Play around with the initial parameter inputs into the minimization function. How sensitive is the minimization algorithm to your initial parameters? Can you force the algorithm to return an erroneous result? (Optionally for IDL users, also try out the function `gauss2dfit` and/or `curvefit` to see how their robustness compares to `mpfit2dfun`.)

d) In four plots, plot the image data, the model (at the same resolution, or number of pixels, as the image data), the difference of (image − model) called the "residual", and the term in the $\chi^2$ sum corresponding to each pixel location. Looking at the residual and/or $\chi^2$ plot, identify where in the image the fit is bad. Suggest ideas for a model that would better fit the image. (Note: In IDL, try using the `tvscl` command to plot the images, in matlab, try `imagesc`. Also be sure to plot a colorbar to show the color scale of the image.) Report your values of the estimated parameter covariance matrix. This is easy, since `mpyfit.py` and `mpfit2dfun` return this, and for matlab users, `fminunc` returns the Hessian at minimum.

e) Plot the 68% confidence ellipse for the parameter pair $x_0, y_0$, in units of pixel width, marginalizing over all other parameters. Quantify your error ellipse width compared to a pixel width. Explain conceptually how we get sub-pixel-size uncertainties for the $x_0, y_0$ centroid location.