PHSX815_Project3:
Characterizing instrumental noise

Ryan Low

March 2021

1 Introduction

Modern astronomy relies on Charged Coupled Devices (CCDs) and other such imaging sensors for recording astronomical data. All of these technologies rely on photons exciting the electrons in some semiconducting material. Counting those electrons becomes a proxy for the number of photons detected. Because of this, recording astronomical data is a counting problem, and thus we can expect the number of photons recorded on a CCD to be distributed as a Poisson distribution. As with all electronic measurements, we must also be aware of sources of noise. Since the noise appears in our counts, we can also expect it to be distributed as a Poisson distribution. However, the rate at which noise occurs, λ_{noise}, depends nontrivially on other confounding factors. The determination of λ_{noise} is an important problem that all astronomers face. Yet, it is often told that a good rule of thumb for determining λ_{noise} is taking the per-pixel median of the calibration images. We will demonstrate how well this rule of thumb holds.

2 Problem Statement

We would like to determine the best-fitting λ_{noise} given a set of data. Because we are dealing with a semiconducting system, how the electrons are distributed in energy depends on the Fermi-Dirac distribution (Equation 1).

$$P(E) = \frac{1}{1 + \exp\left((E - E_F)/k_B T\right)} \tag{1}$$

For silicon, the band gap is about $1.12\,eV$ and the Fermi energy is approximately half of the band gap energy. Using this distribution, we can model the number of noise electrons that we count, which in turn gives us a distribution for λ_{noise}. Using this, we will generate sample data for our detected counts (see Section 3).

Using the generated data, we must be able to infer λ_{noise}. Each sample has probability given by Equation 2.

$$P(\lambda|x) = \frac{\lambda^x e^{-\lambda}}{x!} \tag{2}$$

Here, however, we do not know λ. Instead, we can input the data into Equation 2 and calculate the Log-Likelihood (Equation 3) as a function of λ.

$$\mathrm{LL}(\lambda) = \sum_i \log\left(P\left(\lambda|x_i\right)\right) \tag{3}$$

The value of λ that maximizes the LL will be our best estimate of λ.

3 Algorithm Analysis

To generate the sample data, we perform Gibbs sampling. In Gibbs sampling, this integration is approximated by taking intermediate samples of these nuisance parameters according to their own distributions. Then, using those samples, we generate samples of our rate parameters and feed those rate parameters into a Poisson distribution. This final set of Poisson-distributed samples is a simulated set of data with these nuisance parameters integrated out. We use this when constructing the simulated data for the Log-Likelihood Ratio (LLR) of each model.

So that our computation time remains reasonable, we use `numpy` methods wherever possible. This includes both array operations and random number generation. `numpy` methods are faster than their pure-Python counterparts because they pass execution to an underlying `C` implementation. Since compiled `C` code is much faster than interpreted Python code, using `numpy` affords greater computational speed, which allows us to get away with some less efficient methods.

In our noise model, a noise electron is counted if it is excited into the conduction band. For our purposes, this occurs when the electron's energy is above the Fermi level. The probability that an electron has this energy is

$$P_{detected} = \int_{E_f}^{\infty} P\left(E\right) dE$$

We can easily perform this integral numerically, and do so using Monte Carlo integration. Since the tail probability in Equation 1 is extremely small, it is sufficient to just integrate up to a reasonable upper bound, in our case $\left(E - E_f\right)/k_B T = 1$. Once this probability is calculated, we can produce a uniformly-distributed number from 0 to 1 for each free electron in the pixel and decide whether the electron is excited or not. The total number of excited electrons is our noise rate parameter, λ_{noise}. For our purposes, we will assume that the number of free electrons is fixed.

Although the LL is a one-parameter function, it is a complicated one. Therefore, we will numerically optimize the LL to find λ. Equivalent to maximizing the LL is minimizing the negative LL. Therefore, we can take advantage of all the neat numerical minimization technology that exists. We use the `scipy` optimization package to perform numerical minimization and obtain uncertainties.