Computational Problems in Experimental Particle Physics

1 Categories

1.1 Simulation

First, how is Monte Carlo simulation used in experimental particle physics?

- Predicting background: This is particularly important when searching for a new processes. What known processes can create a signal that is indistinguishable from the new process? Examples: Search for the Higgs boson (`atlas.cern/discover/physics`), search for proton decay.
- Developing identification techniques: Suppose you want to develop an algorithm that distinguishes between two different types of particles in your detector. How will you know how well your technique works, i.e. how will you know how often your algorithm mis-identifies a particle?
- Estimating efficiencies: Closely related to above. If you have to make any cuts (i.e. eliminate some events) from your data set in order to isolate signal events from background events, how will you know how many signal events are accidentally thrown out as background?
- Evaluating statistical significance: How often does a random data set pulled from the same distribution yield a better fit than your data?
- And more...

The different aspects of the experiment that need to be simulated are:

- Particle source (examples: simulate how the neutrinos are created, simulate proton beam, simulate cosmic ray spectrum)
- Interactions (simulate neutrinos interacting with nucleons in the detector, simulate protons smashing together)
- Outgoing particles (simulate how all the produced particles interact with the detector material, i.e. scintillation, ionization, scattering, etc)
- Detection (simulate how those interactions get detected, i.e. scintillation light gets detected by photo-multiplier tubes, electronics, etc)

1.2 Reconstruction

What does reconstruction entail?

- Treat the simulated data and the real data in the same way
- Hits to clusters of hits to tracks and showers
- Based on tracks and showers or other event information, determine what kind of interaction happened (was a muon or electron produced in the event? can you tell by identifying which track belongs to a muon or electron?)
- Reconstruct the particle energy and any other important quantities

Most detectors are spatially segmented in some way to allow for determining the position of the interaction in the detector. A "hit" is usually defined as an electronic signal from one segment of the detector, which indicates the presence of a particle passing through. In NOvA, the detector is segmented into long bars filled with liquid scintillator; a scintillation signal in one bar is considered a hit. In the Super-Kamiokande water Cherenkov detector, the sides of the detector are covered by photomultiplier tubes to detect the Cherenkov light; a signal in one photomultiplier tube is considered a hit.

Each hit carries information about the energy deposited by the particle. For example, in NOvA, the amount of scintillation is proportional to the energy deposited. Understanding how the electronic signal correlates with the particle's energy is called calibration, i.e. for a given size of the electronic signal, how much energy deposited in MeV does that correspond to? Calibration can be accomplished using a source of known energy, or by comparing to simulations, or (usually) some combination of both.

Hits also include a time. This can be useful in many ways. In NOvA, for example, it is possible for more than one neutrino interaction to occur in the same spill (the beam isn't continuous, but comes in short bursts). The time of the hits helps to separate which hits belong to which neutrino interaction.

The next task is to figure out how many particles there were in the event based on the hits. First, hits can be clustered together, based on both spatial and temporal information. Then, these clusters of hits can be formed into tracks or showers. Tracks are configurations of hits along a straight line. Showers are groups of close hits, that also have a general direction, but are not confined along a line. Muons tend to make tracks in the detector. Electrons tend to make showers.

Now we want to understand what type of interaction occured. If there is an easily identifiable muon track, this was likely a muon neutrino interaction. If there is an easily identifiable electron shower, this was likely an electron neutrino interaction. But it is not always so simple. For example, if the energy transferred from the neutrino to the muon is small, then the muon track is short and not as easily identfied. There's another type of interaction,

called a neutral-current interaction, where a neutrino scatters off a nucleon but doesn't create a charged lepton. When these types of events create neutral pions, which decay into two photons, the signature can be misidentified as a electron neutrino interaction.

Then quantities necessary for physics analysis are reconstructed. For example, the incoming neutrino energy is an important quantity for neutrino oscillations, since the oscillation probability is dependent on the energy. (Calormetric vs kinematic energy construction.) Beam neutrino experiments have a fixed baseline (distance between neutrino source and detector), but for atmospheric neutrino measurements, you also need to know the distance the neutrino traveled because the oscillation probability also depends on this distance. You can figure this out from the zenith angle of the neutrino's direction.

Neutrino oscillation experiments (and particle physics experiments in general) have turned to increasingly more sophisticated computational techniques to identify interactions in their detectors, like machine learning. (Another topic that one could teach an entire course on!) Machine learning algorithms improve automatically through experience. These algorithms build a model based on sample data, known as training. Since our use of machine learning is for event classification, we typically use simulated events for training, then apply the classification to data or a different set of simulated events. (We must have a statistically separate sample for training vs application.) Some examples: artificial neural networks, decision trees, convolutional neural networks.

1.3 Analysis

What are the final steps to producing a physics result?

- Select the types of events you want to analyze based on the identification algorithms developed above, removing as much background from your sample as possible
- Predict the remaining amount of background
- Analyze to measure a physics quantity - this likely involves fitting the data to some model and determining parameters
- Evaluate all the sources of uncertainty that you can think of
- Evaluate the significance of your result (could be goodness-of-fit, some other hypothesis test, confidence intervals for your fitted parameter)

2 Fitting

A common computational problem encountered in physics is fitting data. There are many occasions when we have some set of data points that we want to fit to a curve. Common uses include finding a mathematical relationship between two quantities and finding parameters when the theoretical mathematical relationship is known.

Section 10, Exercise 2

2.1 Method of Least Squares

Let's assume we have a collection of N points (x_i, y_i). x is an independent variable (one we control) and y is a dependent variable (one we measure). Suppose we want to fit a line to this data, $y = ax + b$, where a and b are the parameters to be determined. In linear *least squares* fit, we choose a and b such that the quantity

$$\Delta_{\rm LS} = \sum_{i=0}^{N-1} (y_i - y(x_i))^2 \tag{1}$$

is at a minimum. Therefore the best-fit values of a and b are the values which minimize $\Delta_{\rm LS}$, and can be found by solving:

$$\frac{\partial \Delta_{\rm LS}}{\partial a} = 0 \tag{2}$$

$$\frac{\partial \Delta_{\rm LS}}{\partial b} = 0 \tag{3}$$

Writing out the function and taking the derivatives:

$$\begin{aligned}
\Delta_{\rm LS} &= \sum_{i=0}^{N-1} (y_i - y(x_i))^2 \\
&= \sum \left(y_i - (ax_i + b)\right)^2 \\
&= \sum \left(y_i^2 + (ax_i + b)^2 - 2y_i(ax_i + b)\right) \\
&= \sum \left(y_i^2 + a^2x_i^2 + b^2 + 2abx_i - 2ax_iy_i - 2by_i\right) \\
&= \sum y_i^2 + a^2 \sum x_i^2 + Nb^2 + 2ab \sum x_i - 2a \sum x_iy_i - 2b \sum y_i
\end{aligned} \tag{4}$$

$$\begin{aligned}
\frac{\partial \Delta_{\rm LS}}{\partial a} &= 0 \\
2a \sum x_i^2 + 2b \sum x_i - 2 \sum x_iy_i &= 0 \\
a \sum x_i^2 + b \sum x_i &= \sum x_iy_i
\end{aligned} \tag{5}$$

$$\begin{aligned}
\frac{\partial \Delta_{\rm LS}}{\partial b} &= 0 \\
2Nb + 2a \sum x_i - 2 \sum y_i &= 0 \\
Nb + a \sum x_i &= \sum y_i
\end{aligned} \tag{6}$$

The system of equations given by Equations 5 and 6 can be solved for a and b:

$$a = \frac{\overline{xy} - (\bar{x})(\bar{y})}{\overline{x^2} - \bar{x}^2} \tag{7}$$

$$b = \frac{(\overline{x^2})(\bar{y}) - (\bar{x})(\overline{xy})}{\overline{x^2} - \bar{x}^2} \tag{8}$$

where the line means an average value (sum and divide by N).

In general, the measurements y_i will have some associated uncertainty, σ_i. (Note that for simplicity, we are assuming the dependent variable (y) has some uncertainty, but the uncertainty in the independent variable (x) is negligible.) The function to be minimized in this case is

$$\Delta_{\mathrm{LS}} = \sum_{i=0}^{N-1} \frac{(y_i - y(x_i))^2}{\sigma_i^2} \tag{9}$$

The larger the uncertainty, the smaller the weight $1/\sigma_i^2$, which means that point affects the measurement of the parameters less. In this case, the solution for parameters a and b is given by:

$$a = \frac{\sum \frac{1}{\sigma_i^2} \sum \frac{x_i y_i}{\sigma_i^2} - \sum \frac{x_i}{\sigma_i^2} \sum \frac{y_i}{\sigma_i^2}}{\sum \frac{1}{\sigma_i^2} \sum \frac{x_i^2}{\sigma_i^2} - \left(\sum \frac{x_i}{\sigma_i^2} \right)^2} \tag{10}$$

$$b = \frac{\sum \frac{x_i^2}{\sigma_i^2} \sum \frac{y_i}{\sigma_i^2} - \sum \frac{x_i}{\sigma_i^2} \sum \frac{x_i y_i}{\sigma_i^2}}{\sum \frac{1}{\sigma_i^2} \sum \frac{x_i^2}{\sigma_i^2} - \left(\sum \frac{x_i}{\sigma_i^2} \right)^2} \tag{11}$$

Example: Least Squares with Uncertainties

The fact that there is scatter and uncertainty in our measurements of y leads to uncertainty on the estimated parameters a and b. Assuming the uncertainties σ_i are independent (i.e. not correlated between points), then the uncertainty in the parameters can be calculated as

$$\begin{aligned} \sigma_a^2 &= \sum \left[\sigma_i^2 \left(\frac{\partial a}{\partial y_i} \right)^2 \right] \\ &= \frac{\sum \frac{1}{\sigma_i^2}}{\sum \frac{1}{\sigma_i^2} \sum \frac{x_i^2}{\sigma_i^2} - \left(\sum \frac{x_i}{\sigma_i^2} \right)^2} \end{aligned} \tag{12}$$

$$\begin{aligned} \sigma_b^2 &= \sum \left[\sigma_i^2 \left(\frac{\partial b}{\partial y_i} \right)^2 \right] \\ &= \frac{\sum \frac{x_i^2}{\sigma_i^2}}{\sum \frac{1}{\sigma_i^2} \sum \frac{x_i^2}{\sigma_i^2} - \left(\sum \frac{x_i}{\sigma_i^2} \right)^2} \end{aligned} \tag{13}$$

Example: Uncertainties in Fitted Parameters

When the function is nonlinear in the parameters, for example:

$$y(x) = ae^{-x/b} + c\sin(dx)$$

then the system of equations that must be solved are not linear and analytical solutions are not guaranteed. There are many methods of nonlinear fitting, ranging from a simple grid search (try different values until you find the minimum) to more complicated algorithms. In the case of a grid search, the uncertainties in the parameters are estimated by taking numerical derivatives in the region of the minimum.

2.2 Method of Maximum Likelihood

The method of least squares can be considered a special case of the more general *method of maximum likelihood*. This method requires us to know something about the probability distribution for each y_i. For example, if the variable is Gaussian, then we assume that the measured value of y_i is drawn randomly from a Gaussian distribution with a mean of $y(x_i)$ (the value of the function we're trying to find based on the data) and a standard deviation of σ_i. Then the probability P_i for making the observed measurement y_i is

$$P_i = \frac{1}{\sigma_i\sqrt{2\pi}} \exp\left[\frac{(y_i - y(x_i, a, b))^2}{-2\sigma_i^2}\right] \tag{14}$$

The probability for making the observed set of measurements of the N values of y_i is the product of the probabilities for each observation:

$$\begin{aligned} P &= \prod P_i \\ &= \prod \left(\frac{1}{\sigma_i\sqrt{2\pi}} \exp\left[\frac{(y_i - y(x_i, a, b))^2}{-2\sigma_i^2}\right]\right) \\ &= \exp\left[\sum \frac{(y_i - y(x_i, a, b))^2}{-2\sigma_i^2}\right] \prod \left(\frac{1}{\sigma_i\sqrt{2\pi}}\right) \end{aligned} \tag{15}$$

This is called the likelihood function. The goal is to find the value of the parameters a and b that correspond to the maximum probability of pulling measurements y_i, i.e. maximize the likelihood function. We change this into a minimization problem by calculating the quantity $-2\ln P$. If the function P is at a maximum for particular values of a and b, then the function $\ln P$ will be maximized at those same values, and the function $-\ln P$ will be at a minimum at those same values. (The reason for the factor of 2 will be obvious.) We then set out to minimize the function $-2\ln P$:

$$\begin{aligned} -2\ln P &= -2\ln\left\{\exp\left[\sum \frac{(y_i - y(x_i, a, b))^2}{-2\sigma_i^2}\right] \prod \left(\frac{1}{\sigma_i\sqrt{2\pi}}\right)\right\} \\ &= -2\ln\left\{\exp\left[\sum \frac{(y_i - y(x_i, a, b))^2}{-2\sigma_i^2}\right]\right\} - 2\ln\left\{\prod \left(\frac{1}{\sigma_i\sqrt{2\pi}}\right)\right\} \\ &= -2\left[\sum \frac{(y_i - y(x_i, a, b))^2}{-2\sigma_i^2}\right] - 2\ln\left\{\prod \left(\frac{1}{\sigma_i\sqrt{2\pi}}\right)\right\} \\ &= \sum \frac{(y_i - y(x_i, a, b))^2}{\sigma_i^2} - 2\ln\left\{\prod \left(\frac{1}{\sigma_i\sqrt{2\pi}}\right)\right\} \end{aligned} \tag{16}$$

Since the second term is a constant (in terms of parameters a and b), the parameter values that minimize the first term will minimize the whole function. Therefore, we set out to minimize the first term, which is called χ^2:

$$\chi^2 = \sum \frac{(y_i - y(x_i, a, b))^2}{\sigma_i^2} \tag{17}$$

You'll notice that this is exactly the function we minimized in our uncertainty-weighted least squares fit.

A word of warning here: least squares is the most common fitting algorithm. But as we see from this derivation, it should be used for situations where the measured variable has an underlying Gaussian distribution. This is usually a pretty good assumption, or at least a good starting point, but may not always be appropriate. The maximum likelihood method can be applied for *any* underlying probability distribution, as long as you know what it is. The problem always amounts to maximizing $P = \prod P_i$, where P_i is the probability to get a particular measurement y_i.

In the case of a variable that obeys Poisson probability, we assume the measured value of y_i is drawn randomly from a Poisson distribution with a mean of $y(x_i)$. Then the probability P_i for making the observed measurement y_i is given by

$$P_i = \frac{[y(x_i, a, b)]^{y_i}}{y_i!} e^{-y(x_i,a,b)} \tag{18}$$

The probability for making the observed set of measurements of the N values of y_i is the product of the probabilities for each observation:

$$\begin{aligned} P &= \prod P_i \\ &= \prod \left(\frac{[y(x_i, a, b)]^{y_i}}{y_i!} e^{-y(x_i,a,b)} \right) \\ &= \exp\left(-\sum y(x_i, a, b) \right) \prod \left(\frac{[y(x_i, a, b)]^{y_i}}{y_i!} \right) \end{aligned} \tag{19}$$

To maximize P, we can minimize the function $-2 \ln P$:

$$\begin{aligned} -2 \ln P &= -2 \ln \left\{ \exp\left(-\sum y(x_i, a, b) \right) \prod \left(\frac{[y(x_i, a, b)]^{y_i}}{y_i!} \right) \right\} \\ &= -2 \ln \left\{ \exp\left(-\sum y(x_i, a, b) \right) \right\} - 2 \ln \left\{ \prod \left(\frac{[y(x_i, a, b)]^{y_i}}{y_i!} \right) \right\} \\ &= 2 \sum y(x_i, a, b) - 2 \sum \ln \left(\frac{[y(x_i, a, b)]^{y_i}}{y_i!} \right) \\ &= 2 \sum y(x_i, a, b) - 2 \sum \left(y_i \ln y(x_i, a, b) - \ln(y_i!) \right) \\ &= 2 \sum \left[y(x_i, a, b) - y_i \ln y(x_i, a, b) \right] + \text{constant} \end{aligned} \tag{20}$$

where again, the last term is a constant with regard to parameters a and b, and therefore does not affect the minimization.

Minimizing the equation above with respect to a and b for a linear fit produces equations that are nonlinear in the parameters, so this is an example of a case where the nonlinear methods must be used.

However, using Poisson probabilities with the maximum likelihood method is fairly common. The Poisson distribution gives the probability of a given number of events occurring in a fixed interval of time if the events occur with a known constant rate and are independent. Poisson statistics are appropriate for counting experiments where the data represent the number of items or events observed per unit interval. It is important in the study of random processes, and is also applied to data that have been sorted into ranges to form a histogram.

2.3 Fitting Histograms

Let's assume that you are conducting an experiment to measure the lifetime of a particular radioactive isotope. Every time a decay happens, the resulting particles are detected, and you can record the time of the decay. If you make a histogram of these times, it will have an exponential shape. This exponential function can be characterized in terms of the mean lifetime, τ as

$$N(t) = N_0 e^{-t/\tau} \tag{21}$$

We want to fit the data to determine the lifetime.

We care about determining τ, but not N_0. The probability density function for an exponential distribution is given by

$$P(t) = \frac{e^{-t/\tau}}{\tau} \tag{22}$$

Recall that this is the probability per unit on the x-axis, which in this case is time. To transform the counts per bin in a histogram to a probability density, we divided by the total number of entries in the histogram n to make it a probability and then divided by the bin width dt to make it a probability density. To get the number of predicted events in one bin of the histogram, we'll do the inverse. Take the probability density, and multiply by the total number of entries and the bin width:

$$N_{\text{pred}} = \frac{ndte^{-t/\tau}}{\tau} \tag{23}$$

In other words, in the equation for $N(t)$ above, $N_0 = ndt/\tau$.

The number of events in one bin of a histogram is the type of data that is described by Poisson statistics. The Poisson distribution is a discrete distribution, meaning it is only

defined at integral values of the variable in question. If the random variable is n, the Poisson distribution is given by

$$P(n;\mu) = \frac{\mu^n}{n!}e^{-\mu} \tag{24}$$

The parameter μ is the mean of the distribution, and the standard deviation is $\sigma = \sqrt{\mu}$. For this reason, the statistical uncertainty in binned data is usually approximated as the square root of the number of entries in that bin. Figure 1 shows the Poisson distribution for different values of μ. As μ gets larger, the Poisson probability approaches a Gaussian distribution.

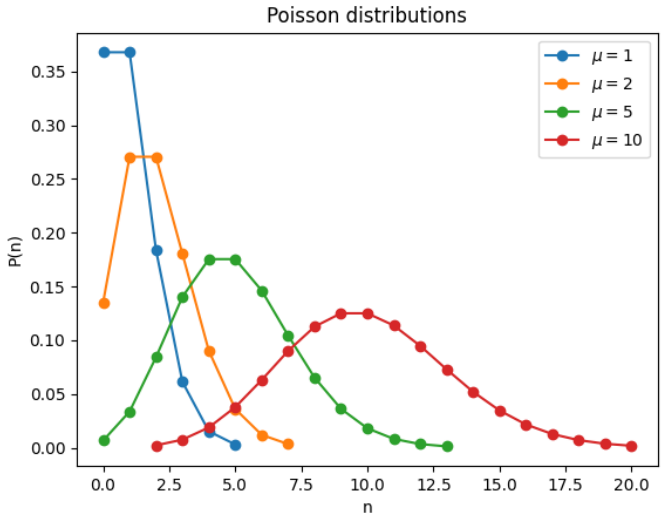

Figure 1:

Now we will fit our radioactive decay data with a χ^2 fit, using a grid search to find the minimum.

$$\chi^2 = \sum_i \frac{(N_i - N_{\text{pred}})^2}{\sigma_i^2} \tag{25}$$

where N_i is the observed number of entires in bin i and N_{pred} is given above. The uncertainty in the number of entries in each bin N_i will be approximated by the square root of the number of entries, $\sqrt{N_i}$, so that $\sigma_i^2 = N_i$. Note that we have to skip bins that have 0 events, to avoid dividing by zero. So if we are using a χ^2 in situations with low statistics, you have to throw out data (usually not a good thing).

Example: Fit Histogram with χ^2

Now we fit the same data using the maximum likelihood method, assuming the data in each

bin is distributed with a Poisson probability.

$$-2\ln L = -2\sum \left(N_i \ln N_{\text{pred}} - N_{\text{pred}} - \ln(N_i!)\right) \tag{26}$$

The maximum likelihood fit does much better with low statistics, as expected.

Example: Fit Histogram with Maximum Likelihood

Now, let's try using a built-in Python function. This time we will fit for both parameters, τ and N_0.

Example: Fit Histogram with scipy

2.4 Goodness of Fit

Section 10, Exercise 3

In the previous examples, we've been comparing the result of the fit to the "true" answer, i.e. the parameter value that was used to generate the distribution. In any real situation, we don't know what the true answer is, so we need some way of evaluating our results without it.

The variance of the fit, s^2 is given by

$$s^2 = \frac{(1/(N-m))\sum(1/\sigma_i^2)\left[y_i - y(x_i)\right]^2}{(1/N)\sum 1/\sigma_i^2} \tag{27}$$

This is the sum of the squared differences between the data points and the fitted values, weighted by the inverse of variance σ_i^2 that describes the uncertainties in each point, normalized to the sum of all the weighting factors. N is the total number of points, and m is total number of parameters, so that $N-m$ is the number of degrees of freedom. The variance is the data can be measured as the sum of the squared individual variances, weighted and normalized in the same way:

$$\begin{aligned}
\langle\sigma_i^2\rangle &= \frac{(1/N)\sum(1/\sigma_i^2)\sigma_i^2}{(1/N)\sum 1/\sigma_i^2} \\
&= \frac{(1/N)\sum 1}{(1/N)\sum 1/\sigma_i^2} \\
&= \frac{(1/N)N}{(1/N)\sum 1/\sigma_i^2} \\
&= \frac{1}{(1/N)\sum 1/\sigma_i^2}
\end{aligned} \tag{28}$$

The variance of the fit is an estimate of the variance in the data. So for a good fit, the ratio

of the two should be close to 1:

$$\begin{aligned}\frac{s^2}{\langle\sigma_i^2\rangle} &= \frac{(1/(N-m))\sum(1/\sigma_i^2)\left[y_i - y(x_i)\right]^2}{(1/N)\sum 1/\sigma_i^2}\left[\frac{1}{(1/N)\sum 1/\sigma_i^2}\right]^{-1} \\ &= (1/(N-m))\sum(1/\sigma_i^2)\left[y_i - y(x_i)\right]^2 \\ &= (1/(N-m))\chi^2 \end{aligned} \tag{29}$$

We see that this ratio is proportional to the χ^2 that we used to minimize the function. This quantity is called the *reduced chi-square*, the chi-square per degree of freedom: $\chi^2_\nu = \chi^2/\nu$, where $\nu = N - m$ is the degrees of freedom. At the best fit point, where χ^2 is at a minimum, we expect $\chi^2_\nu \approx 1$ for a good fit. In this way, χ^2 is used for testing the goodness of fit. If χ^2_ν is much greater than one, this indicates that the fitting function is not appropriate for describing the data, since the estimated variance s^2 is too large. A χ^2_ν smaller than one does not necessarily indicate a better fit. However, a value that is very small may indicate a problem in the assignment of the uncertainties σ_i.

Let's distinguish between the two uses of χ^2: parameter estimation and goodness of fit. If you are certain that you have the correct model and error estimates, a "bad" value of χ^2 at the minimum is simply bad luck and doesn't tell you anything about how accurate your parameter estimates are.

- Goodness of fit: Is the data consistent with having been drawn from a specified distribution?
- Parameter estimation: Which of the following limited set of hypotheses is most consistent with the data?

Parameter estimation is generally more powerful, at the expense of being more model-dependent. You can almost always find a function that fits the data perfectly, but does it tell you something about the underlying physical process?

One of the advantages of χ^2 fitting (least squares fitting) is that it provides a built-in goodness of fit test: χ^2 is the function that's being minimized, so once you find the minimum, you already have the value of χ^2 to use for goodness of fit. The maximum likelihood method does not, in general, yield a goodness of fit parameter.

The probability distribution function for χ^2 with ν degrees of freedom is given by

$$p_\chi(x^2;\nu) = \frac{(x^2)^{(1/2)(\nu-2)}e^{-x^2/2}}{2^{\nu/2}\Gamma(\nu/2)} \tag{30}$$

where the Γ function is equivalent to the factorial function $n!$ extended to non-integer arguments. Figure 2 shows the χ^2 distribution for different values of ν, the degrees of freedom.

For goodness of fit purposes, we are interested in the integral of this function:

$$P_\chi(\chi^2;\nu) = \int_{\chi^2}^{\infty} p_\chi(x^2;\nu)dx^2 \tag{31}$$

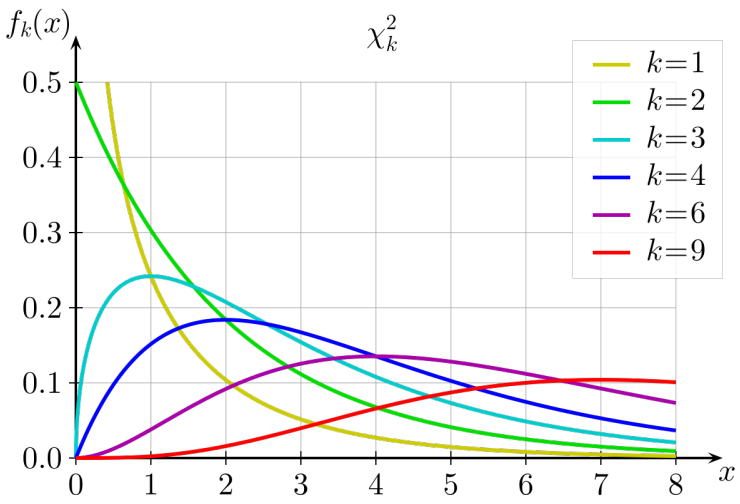

Figure 2: Figure from `https://en.wikipedia.org/wiki/Chi-square_distribution`

This gives the probability that a random set of n data points drawn from the parent distribution (your best fit curve) would yield a value of χ^2 equal to or greater than the given value. If your fitting function is a good approximation to the true function, the value of χ^2_ν should be close to one and the probability above should be approximately 0.5.

Let's unpack that a bit: A small probability indicates that the integral was computed with a lower limit being a value of χ^2 in the right-side tail of the χ^2 distribution. This means for most data sets drawn from the parent distribution, the resulting χ^2 is going to be less than what you got. This is an indication that the fit isn't very good, meaning that your data set seems unlikely to have been drawn from the proposed parent distribution (your best fit curve). In practice, this test is used to identify particularly bad fits, not good ones. In other words, if the probability is very small you should worry, but if it's not small, don't worry about how close to 0.5 it is.

This is an example of a *p-value*. In the language of hypothesis testing, the p-value is the probability of obtaining results at least as extreme as the results actually observed, under the assumption that the null hypothesis is correct. The null hypothesis for a goodness of fit test is that the data set was drawn from the best fit curve. A small p-value is taken as evidence against the null hypothesis.

For a histogram with n bins, this goodness of fit test is usually performed using Pearson's χ^2 test statistic given by

$$\chi^2 = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i} \tag{32}$$

where O_i is the number of observed events in bin i and E_i is the number of expected events in bin i (based on the best fit curve). The distribution of this quantity is generally considered

close enough to a true χ^2 distribution to test for goodness of fit when the number of expected events is at least 5 in all bins.

There's no analytical solution to the integral above. There are existing functions in Python and other languages that can calculate this probability for you.

Example: χ^2 for Goodness of Fit

Given the definition of the p-value, we can actually calculate it using a Monte Carlo simulation. For our best-fit value of τ, we can draw many sets of similar data (pseudo-experiments). We treat each of these sets of data exactly like the original set, calculating the best fit and the χ^2 statistic. The fraction of pseudo-experiments that yield a worse (larger) χ^2 statistic than our observation gives the p-value.

Example: Monte Carlo Tests

There are many different methods to calculate p-values, and you should always choose the one that's most appropriate for your data and what hypothesis you are trying to test. One could teach a whole class just on this subject, so we won't go into detail. But when using existing libraries for statistical testing of data, take the time to study the algorithm and make sure it's appropriate for what you want to use it for.

2.5 Confidence Intervals

Typically, parameter uncertainties are quoted as $a \pm \sigma_a$, and σ_a is interpreted as the standard deviation of a Gaussian distribution with mean a. If you integrate a Gaussian pdf between $a - \sigma_a$ and $a + \sigma_a$, you find that 68.3% of the area underneath the curve falls in this region. Therefore, the range $a - \sigma_a$ to $a + \sigma_a$ is called the 1σ or 68.3% *confidence interval* of the parameter. It might also be stated as a is in the range $a - \sigma_a$ to $a + \sigma_a$ at the 68.3% *confidence level*. Similarly, the region $a - 2\sigma_a$ and $a + 2\sigma_a$, encompasses 95.4% of the area and is called the 95.4% confidence interval, and so on.

We've seen that the χ^2 function is closely related to the Gaussian pdf (in that χ^2 can be derived minimizing $-2 \ln L$ where L is the likelihood function for a Gaussian random variable). It can be shown that an $n\sigma$ confidence interval is determined by a change in χ^2 of $\Delta\chi^2 = n^2$. For example, to calculate the $1-\sigma$ confidence interval for a parameter, you find the the values of the paramter for which the χ^2 value is ± 1 different from the minimum. Table 1 gives the parameters for commonly quoted confidence intervals. Note these $\Delta\chi^2$ values are only for a one parameter fit; you can find tabulated values for different numbers of degrees of freedom.

This is almost always how parameter uncertainties are defined, but keep in mind, it is only really true for the case where the data has Gaussian uncertainties. If your data has lopsided uncertainties (i.e. larger in one direction than the other), for example, then the uncertainties are not Gaussian, and defining uncertainties this way may not be appropriate.

Confidence level	σ	$\Delta\chi^2$
68.3%	1.0	1.00
90.0%	1.6	2.71
95.5%	2.0	4.00
99.0%	2.6	6.63
99.7%	3.0	9.00

Table 1: Confidence intervals for a one-parameter fit

Example: Confidence Intervals

2.6 References

The following two textbooks are excellent references on these topics:

- *Data Reduction and Error Analysis for the Physical Sciences* by Philip R. Bevington and D. Keith Robinson
- *Statistical Methods in Experimental Physics* by Frederick James

3 Examples

3.1 Event Reconstruction

We're going to look at a very simple example of a machine learning technique called k-Nearest-Neighbors. Basically, we take some simulated events divided into two populations based on the "truth" information (i.e. it's simulation, so we know what kind of interaction it was), and pull out some variables that we think will help to distinguish between two different types of events. The variables have to be things that we can measure with the real data (because we want to apply this identification algorithm to real data, where we don't know the truth information).

Let's say we want to determine whether a test event is a type I event or a type II event. We have a sample of simulated events that are all either type I or type II. For the test event that we want to identify, we find the k closest events in our simulated sample. For example, if the two variables we are using to classify are a and b, our test event has values $a = a_{\rm test}$ and $b = b_{\rm test}$, and our simulated sample has values a_i, b_i where i is an index for every event in our simulated sample. Then we calculate the "distance" as

$$d_i = \sqrt{(a_{\rm test} - a_i)^2 + (b_{\rm test} - b_i)^2} \tag{33}$$

for every event in the simulated sample. Then we examine the events in the simulated sample with the k smallest distances to our test event. If more of these k events are type I, we classify the test event as type I. If more are type II, then we classify the test event as type II. The "score" for each test event is the fraction of the closest sample events that are type

I or type II.

We're going to test this out on a sample of simulated atmospheric neutrino interactions in DUNE, a future neutrino oscillation experiment similar to NOvA, but with a different detector technology. We have a sample of 2000 atmospheric interactions, half of which are muon neutrino interactions and the other half which are electron neutrino interactions. We have four variables we want to use to classify the events: the number of reconstructed showers, the number of reconstructed tracks, the number of reconstructed clusters, and the length of the longest track (if there is a track).

Let's take a look at some of the events.

Example: Event Display

Now let's perform the kNN classification. We have 1000 additional events that we are going to classify, and determine how well our kNN algorithm works.

Example: kNN

This is just a simple example. The selection of variables can be better optimized. In selecting variables for a kNN or any type of neural network, you should be careful not to select variables that are model-dependent, i.e. highly dependent upon the model used in your simulation. Since the classification is based purely on simulated events, if your model is very different from reality, your classification scheme won't be very effective. So you should choose variables that are well-modeled by your simulation.

The algorithm can be optimized as well. In this example, we loop over every single event in the simulated sample for every test event. This is fine for the number of events in our example, but if your library of events is larger (perhaps tens or hundreds of thousands), with a similar number of events to classify, this method will take too long. Your could save time, for example, by grouping the sample events into bins, and then starting for the search for nearest neighbors in the bin where the test event lies, plus nearby bins.

3.2 Analysis

As an example, we are going to consider the problem of electron neutrino appearance in a muon neutrino beam to measure the neutrino oscillation parameter θ_{13} in the NOvA experiment. The probability of flavor change is governed by four parameters in the neutrino mixing matrix: θ_{12}, θ_{13}, θ_{23}, and δ. The θ's are called mixing angles and δ is a phase parameter. The probability also depends on the difference in the squared neutrino masses. With three neutrinos, there are only two independent mass-squared differences: $\Delta m^2_{21} = m^2_2 - m^2_1$, $\Delta m^2_{32} = m^2_3 - m^2_2$. The other difference is just a sum of the other two: $\Delta m^2_{31} = m^2_3 - m^2_1 = m^2_3 - m^2_2 + m^2_2 - m^2_1 = \Delta m^2_{32} + \Delta m^2_{21}$.

The probability for a neutrino that is created as a muon neutrino to be observed as an electron neutrino at distance L depends on all these parameters. However, for simplicity, we are going to approximate it as the leading order term:

$$P(\nu_\mu \to \nu_e) \approx \sin^2\theta_{23} \sin^2(2\theta_{13}) \sin^2\left[\left(1.27 \text{ GeV}/(\text{km eV}^2)\right) \Delta m^2_{32} L/E\right] \tag{34}$$

where $L = 810$ km is the distance between the creation point of the neutrino beam at Fermilab and the NOvA detector located in Ash River, MN. E is the neutrino energy; the beam neutrinos observed in the NOvA detector have a range of neutrino energy that peaks around 2 GeV. Typically, a neutrino experiment will fit all the parameters simultaneously, but in our example, we're going to assume the measured values of $\theta_{23} = 0.86$ radians and $\Delta m^2_{32} = 2.41 \times 10^{-3}$ eV^2 and only fit for θ_{13}.

We are going to fit the observed energy spectrum of electron neutrinos to the predicted spectrum, where the predicted spectrum is different for different values of θ_{13}. The predicted spectrum includes a prediction for the signal (the number of electron neutrino events from muon neutrino oscillations) plus a background (other interactions that are misidentified as electron neutrino interactions). We are given the signal prediction for a value of $\theta_{13} = 0.3$. To get the signal prediction at a different value of θ_{13}, we're going to scale by the ratio

$$R = \frac{P(\theta_{13})}{P(\theta_{13} = 0.3)} \tag{35}$$

where P is the probability from Equation 34. We're also going to assume that the background does not depend on the value of θ_{13}. (Both assumptions are not entirely true, but work well enough for what we're doing here.)

Example: Predicted Spectrum

Now we can find the best fit value of θ_{13}, by scanning over values of θ_{13} and finding the value that minimizes the difference between the predicted spectrum and the observed spectrum. For the function to minimize, we're going to use $-2 \ln L$, where L is the Poisson likelihood function, since the bins can have very few events.

Example: NOvA Fit

Section 10, Exercise 4

We're going to evalute the confidence intervals using a technique that we call "Feldman-Cousins" in my field after this paper: "Unified approach to the classical statistical analysis of small signals," by Gary J. Feldman and Robert D. Cousins, Phys. Rev. D 57, 3873, 1998. (Full disclosure: Feldman is a NOvA collaborator!)

Before the value of θ_{13} was known, previous experiments had shown that it was small by not being able to observe oscillation modes for which θ_{13} is the dominant parameter in the probability. The Feldman-Cousins method (as noted in the paper) is particularly applicable

for the θ_{13} search, as it works well for the case where you expect a small signal on top of a large background, and where there's a physical boundary in parameter space. Negative values of $\sin^2(2\theta_{13})$ are unphysical, and meaningless in the model. But imposing a boundary in the fit interferes with using the traditional $\Delta\chi^2$ values to set confidence intervals. In this method, we don't allow unphysical values of θ_{13}, but can set the confidence intervals

The Feldman-Cousins method works like this:

- Make a grid of values of θ_{13} and get the predicted spectrum for each value.
- At each of these values of θ_{13}, calculate the $-2\ln L$ between the observed distribution and the predicted distribution at this value. From this number, subtract the $-2\ln L$ between the data and the predicted distribution at the best fit value. We'll call this

$$\Delta\chi^2_{\text{obs}} = (-2\ln L)_i^{\text{observed}} - (-2\ln L)_{\text{best}}^{\text{observed}} \tag{36}$$

  where the subscript i indicates the value of θ_{13}.
- At each value of θ_{13}, do a Monte Carlo simulation a bunch of times (we'll do 1000). Each of these pseudo-experiments will draw a random "pseudo" observed distribution from the predicted distribution. That pseudo-observation will be treated the same as the data. It will be fit to find the best fit value of θ_{13}, and then we'll find $\Delta\chi^2$ the same way we did for the data:

$$\Delta\chi^2_{\text{pseudo}} = (-2\ln L)_i^{\text{pseudo}} - (-2\ln L)_{\text{best}}^{\text{pseudo}} \tag{37}$$

- At each value of θ_{13}, we find the fraction of pseudo-experiments that have a better (i.e. smaller) $\Delta\chi^2$ than the observed one. Points where this fraction is $< \alpha\%$ are included in the $\alpha\%$ confidence interval for θ_{13}.

Example: Feldman-Cousins

NOvA was designed specifically to measure the appearance of electron neutrinos, so the background is not so large. Let's see an example with a large background. In this case, we want to see at what confidence level we can exclude $\theta_{13} = 0$.

Example: Feldman-Cousins with a Large Background

If you were to have a case where the best fit value of θ_{13} was zero, the same method can be used to calculate an upper limit, i.e. we can say $\theta_{13} < x$ at the 90% confidence level.