Case Study 4:

Snow Gauge Calibration

Caitlyn Bryant - Applied Mathematics, 4th year
Edward Chao - Mathematics: Statistics, 4th year
Kieran Mann - Mathematics/Economics, 4th year
Matthew Kucirek - Applied Mathematics, 4th year

Table of Contents

Introduction

In this study, we will provide a simple procedure for converting snow gain into density using measurements of polyethylene from a calibration run of the United States Department of Agriculture (USDA) Forest Service's snow gauge located in the Sierra Nevada mountain range near Soda Springs. We will do this by: (1) using the data to fit the gain to density, (2) developing a procedure for adding bands around our least squares line that can be used to make interval estimates for the snowpack density from gain measurements, and (3) checking how well our procedure works by omitting the set of measurements corresponding to the block of density 0.508, applying our calibration procedure to the remaining data and providing an interval estimate for the density of a block with an average reading of 38.6.

The main source of water for the northern region of California comes from the Sierra Nevada mountain range, and in order to help monitor the water supply, the USDA Forest Service operates a gamma transmission snow gauge, which is used to determine a death profile of snow density. This snow gauge can measure the same snowpack repeatedly because it does not disturb the snow in the measurement process, and with these measurements, researchers can study snowpack settlement and the dynamics of rain on snow throughout the winter season. Snow absorbs rain water up to a certain point; the denser the snow pack, the less water it can absorb. The snow gauge measures gamma ray emissions and then converts that into a density reading.[1] However, there may be changes over the seasons in the functions used to cover the measured values into density readings due to radioactive source decay and instrument wear. In order to adjust the conversion method, a calibration run is made each year at the beginning of the winter season. This is where our study comes in, where we will develop a procedure to calibrate the snow gauge.

Our data comes from a calibration run of the USDA Forest Service's snow gauge located in the Sierra Nevada mountain range near Soda Springs. The calibration run consists of placing polyethylene blocks (used to simulate snow) of known densities between the two poles of the snow gauge and then taking readings on the blocks. 30 measurements of the polyethylene blocks are taken, but only the middle 10 measurements are reported. These reported measurements are amplified versions of the gamma photon count made by the detector, and we call the gauge measurement the "gain".

The big question that we will answer in our data analyses is: What simple procedure can we use for converting snow gain into density from the provided calibration run of the given USDA Forest Service's snow gauge?

[1] McGurk, Bruce, and David Azuma. "Correlation and Prediction of Snow Water Equivalent from Snow Sensors." United States Department of Agriculture Forest Service (2000).

Research and Analysis

2.1 Fitting Gain ~ Density

From the 10 measurements of 9 density points, we created a scatterplot to analyze the relationship between the density and the gain. As the process of calibrating the snow meter involves measuring gamma gain from fixed blocks of polyethylene of known density, our model holds density as the explanatory variable and gain as the response. Upon visual inspection of the shape of the scatter, it became apparent that a polynomial model would likely provide a good fit.

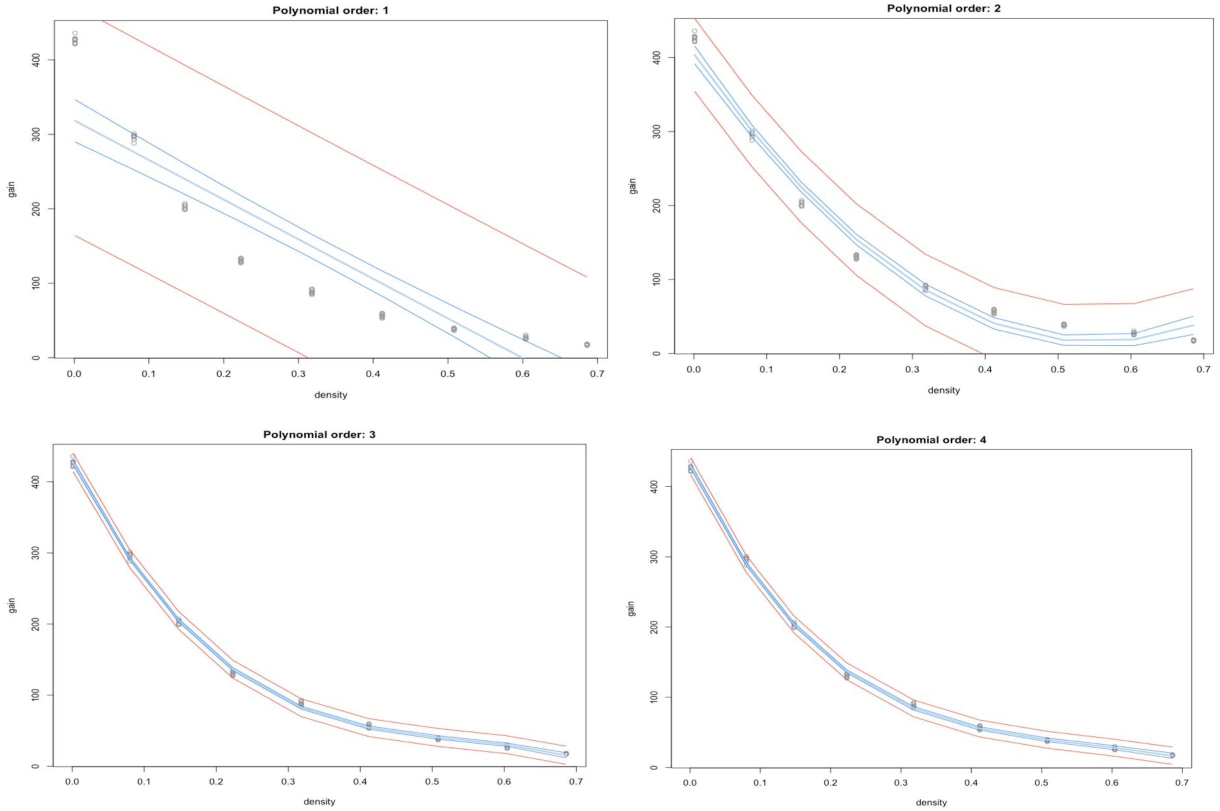

Figure 2.1.1: *Scatter plots showing regressions for a 1st, 2nd, 3rd and 4th order polynomials. Prediction band is shown in orange and confidence intervals are shown in the darker blue*

In fitting the suspected polynomial, it is of course important to determine the proper order of polynomial to use. Polynomials are of the form: $a + bx + cx^2 + \ldots + mx^n$ where n is said to be the “order of the polynomial.” Because the underlying physics in this experiment are complicated and hard to observe, it is true that we are selecting the model out of its goodness of fit, not out of a solid model of the physical occurrences we are observing. The risk in this process is overfitting, where as a researcher, one goes to far and chooses a polynomial of a high order simply because it fits. This can lead to choosing a model tuned to explain randomness or noise in your specific sample, rather than properly estimating the underlying distribution. With this in mind, we settled on a polynomial of third order fitted by Ordinary Least Squares to relate gain and density for the purpose of later using the model we fit to reverse the process, and make estimations on snow density given gain readings from the field.

Using R to obtain parameter estimates for a 3rd order polynomial model yields an intercept estimate of 429.373. This value suggests that when there is no density provided, the gauge measurement, on average, is expected to have a value of around 430, and as the density of measured samples increases increases, we would expect to see a decrease in the observed value of gain.

To further analyze the relationship between the two variables, we took a look at the residuals surrounding the data. Using the same polynomial of order 3, we were able to provide a QQ-Plot, and a graph of the residuals along with a best fit line

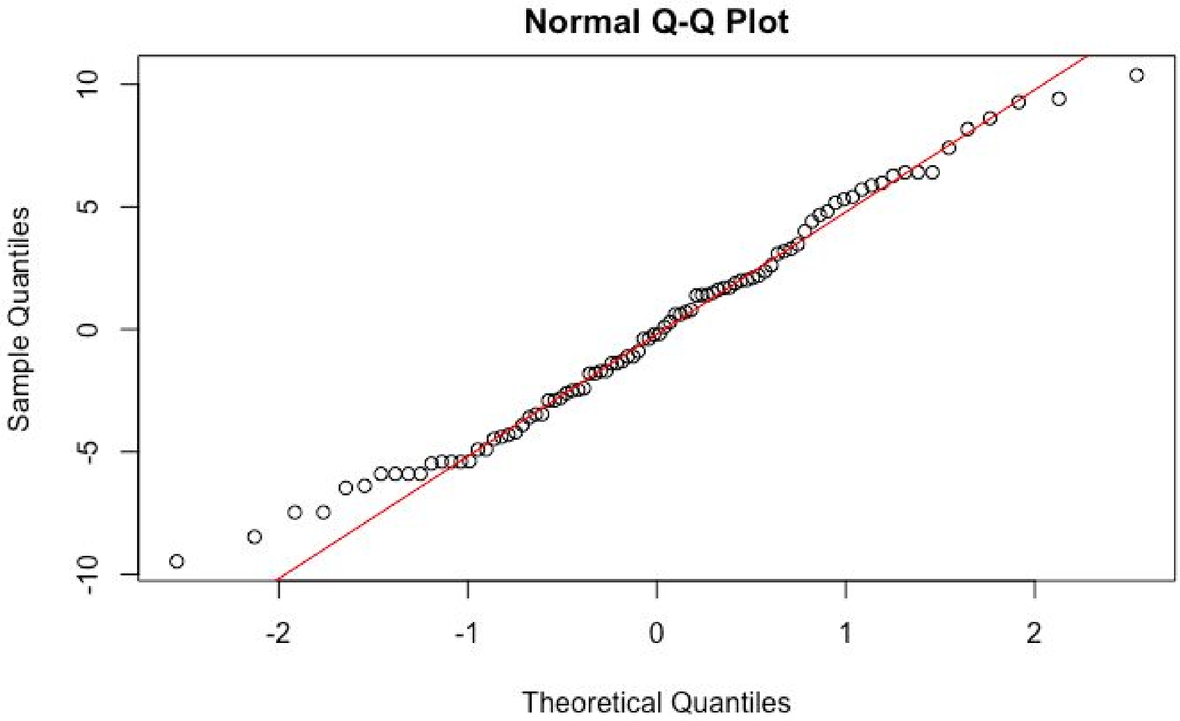

Figure 2.1.2: *A Quantile-Quantile Plot of the residuals*

The best fit line (red) in Figure 2 implies that normality does exist for the residuals. Moreover, there does not seem to exist any large or influential outliers that would affect our data. The resulting conclusion is that there are no unusual observations that would cast doubt on the trend of the data.

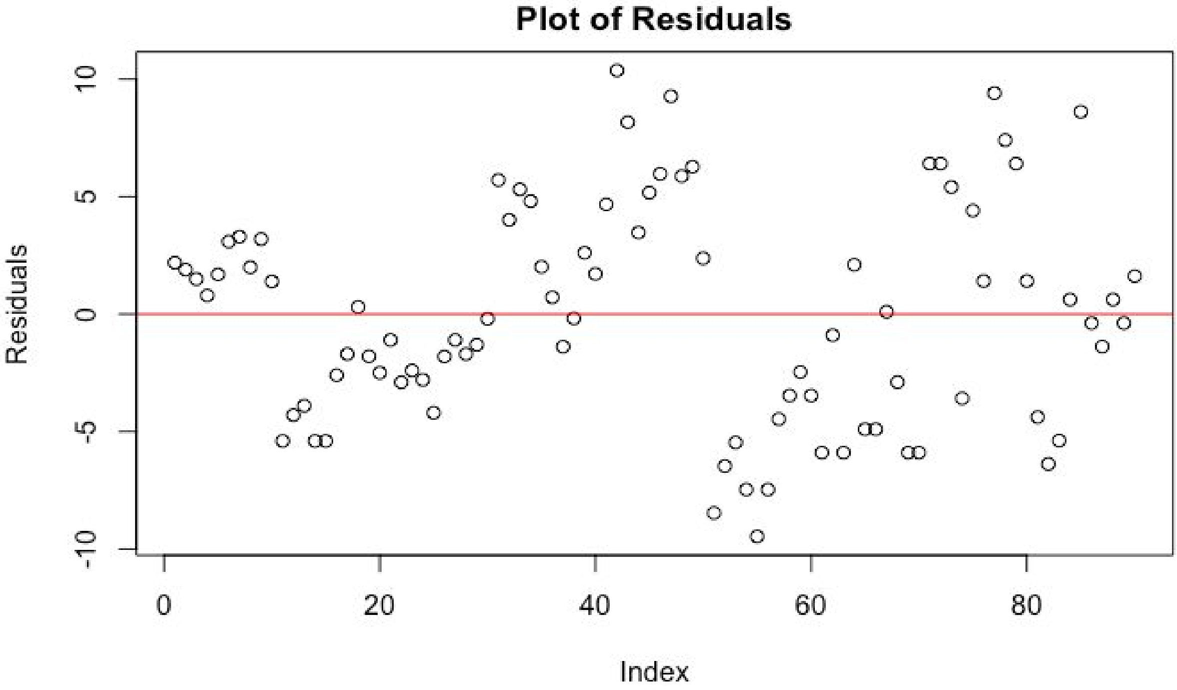

Figure 3: *A plot of the residuals*

This plot (Figure 3) suggests a high residual normality. All the points appear randomly scattered throughout the space–there is no immediate pattern recognized by these researchers. Further observation suggests that the residuals are consistent with random error, since the residuals are neither systematically high nor systematically low. The red line is at the value 0 for the y-axis, and all the points tend to be centered around the zero-line with a constant spread throughout.

To finally evaluate the fit of the linear model, we looked the value of our R^2 coefficient. The value of our R^2 is 0.9988, which suggests that 99% of the variability is explained by our model, promising a good fit of our data.

2.2 Predicting - Density from Gain

The model we use to predict density from gain is the third order degree polynomial. We use his model because we get a reasonable y-intercept and we want to be cautious in preventing overfitting. Refer to figure 2.1.1 to see the graph of the 3rd degree polynomial fit. From this model, we can find the equation of the cubic being used to fit our data. We find the fitting model to be as follows;

$$\textit{Gain} = 429.373 - (1989.918 \bullet \textit{density}) + (3521.061 \bullet \textit{density}^2) - (2186.525 \bullet \textit{density}^3)$$

While this information is great, it is not exactly what we are looking for. This model will predict a gain based on a density, however, we are interested in predicting density from gain. To do this we take advantage of R's uniroot function, using it to construct a function that will calculate our inverse. Once we have obtained our inverse function, we can make our predictions. The two gain values we are interested in are gains of 38.6 and 426.7. When predicting for these values, we keep in mind that 38.6 is the average gain reading for densities of 0.508 and 426.7 is the average gain reading for densities of 426.7. Plugging 38.6 into our predictive function and we obtain a density of 0.5267 and plugging in 426.7 we obtain a density of 0.00134. Given the average readings, we see that these are some decent predictions; however we would like to get a prediction interval for our estimates.

Since we want to predict density, in order to obtain interval estimates, we must flip our regression model. This allows us to use R's predict function to retrieve lower and upper bounds of a 95% prediction interval based around each point of fit. The length of the interval varies ever so slightly depending on where we are along the fitting model. In order to get the length of our prediction interval, we take the average of the lengths of our intervals that we got from the predict function. So the lower bound of our interval will be our predicted value minus one half of our interval length. The upper bound is calculated analogously. When calculating our prediction intervals, we keep track of not only the upper and lower bound, but the predicted value as well.

Figure 2.2.1: *Prediction and Intervals*

Gain	*Density Prediction*	*95% Prediction Interval*
38.6	*0.5267*	*(0.4797, 0.5735)*
423	*0.0032*	*(-0.044, 0.05)*
426.7	*0.0013*	*(-0.046, 0.0483)*
429	*0.000186*	*(-0.0467, 0.0471)*

Clearly, we cannot have a negative density, so any negative density is regarded as zero.

2.3 Cross-Validation

We have obtained 95% prediction intervals for our estimates of density based on gain. The next step is to cross-validate the data by leaving out a subset of entries and performing the tests again. The data we leave out is all the data associated with the block of density 0.508. These are the blocks with an average gain of 38.6. After subsetting the data, we flip the regression again, and obtain our 95% prediction intervals for the fitting. The results are in the following table

Figure 2.3.1: *Cross-Validation Prediction and Intervals*

Gain	*Density Prediction*	*95% Prediction Interval*
38.6	*0.5367*	*(0.4913, 0.582)*
423	*0.0032*	*(-0.0422, 0.0490)*
426.7	*0.0013*	*(-0.0441, 0.0467)*
429	*0.000186*	*(-0.0452, 0.0456)*

We see that Figure 2.3.1 resembles the same data that we saw in Figure 2.2.1. This suggests that we have a valid prediction model, seeing as in each case the true density has been captured by our prediction interval.

Conclusion

Our pursuit was to create an algorithm to go from measurements of gain from polyethylene blocks of known density to actual measurements of snow density from measured gain in the wild, thereby calibrating the snow gauge in question. Our findings from the calibration data set lead to what appears to be a fairly solid algorithm for obtaining realistic density measurements in practice and understanding their levels of significance via confidence intervals.

Of course, if the densities of the polyethylene blocks are not exactly as we assumed, it would seriously impact the validity of the calibration we performed in this case study. Depending on how different the actual densities were from our assumptions and whether or not the difference is systematic, this would impact our model and by extension the accuracy of later readings made with the snow gauge in different ways. All in all, it is surely paramount to the pursuit of calibrating the snow gage that consistent and accurate calibration polyethylene blocks are used.

Looking at the residuals we observed by index, it becomes clear that the observations made early are more thoroughly explained by our residual line. If the pattern that appears to exist in the residual by index graph does in fact exist in actuality, failing to take measurements of the polyethylene blocks in random order would introduce error into our model that varies as a function of x–that is if the blocks were measured sequentially and early measurements experience less noise, then measuring sequentially will introduce heteroskedastic error to our fit. Though R uses underlying algorithms designed to be robust to heteroskedastic error, it is certainly preferable and will increase the final accuracy of the model to account for the "warm up" or whatever factor is introducing greater variance as observations are taken by simply measuring blocks of different densities in random order.

To make our algorithm available to scientists or other users looking to calibrate snow meters, a standard approach would be to develop an R package and submit it to the CRAN repositories for widespread open-source use. Although our method of instrument calibration is sound, we are hardly the first team to approach this problem. Some cursory research on the topic after-the-fact yields a number of great packages that include full-featured calibration functions, such as chemCal[2] and investr[3]. Ultimately it is our group's recommendation to scientists that they use the tried and tested tools made available by the R community when performing something as straightforward as an instrument calibration.

[2] See: https://cran.r-project.org/web/packages/chemCal/index.html

[3] See: https://cran.r-project.org/web/packages/investr/index.html

Appendices

Theory

1. **Ordinary Least Squares**

The least squares line we used is represented by the equation:

$$y = a + bx + cx^2 + dx^3$$

We arrived at an order of polynomial 3 because we wanted a least-squares line that would minimize the sum of the squared residuals:

$$e_1^2 + e_2^2 + \ldots + e_n^2$$

Looking through Figure 2.1.1., we see that an order of 1 does not properly fit the data entirely as the residuals display a significant distance from our data points. However, a polynomial of order 4 would have caused an overfit, thereby describing random error instead of giving us the information about the relationship between our explanatory variable (density) and our response variable (gain). An order of 3 gave us the a nearly perfect fit to begin to examine the underlying relationship of our variables.

2. **Residuals**

The residual is calculated by evaluating the difference the observed value and the predicted value:

$$e_i = y_i - \widehat{y_i}$$

3. ***R^2* with n degrees of freedom**

R^2 is a measure of the total deviance of the observed sample from the regression line or how much of the observed sample can be explained by our model. It is most simply formulated as the sum of explained squares over the total sum of squares. The higher the percentage is (with a maximum value of 1), the better our linear model fits the data.

Code

(See attached r markdown file for replicable code.)