Penalized Spline Regression and its Applications

Whitney Griggs

May 13, 2013

Abstract

Regression analysis is a branch of statistics that examines and describes the relationship between different variables of a dataset. In this paper, we investigate penalized spline fits, a nonparametric method of regression modeling, and compare it to the commonly used parametric method of ordinary least-squares (OLS). Using data from our neuroscience research, we demonstrate several different applications of these penalized splines as compared to linear regression. We conclude the paper with some exploration of statistical inference using bootstrapping and randomization of our spline models.

Contents

List of Figures

1 Introduction

Most simply defined, regression analysis is branch of statistics that characterizes the relationships between different random variables of a sample or population. Throughout the biological sciences and other disciplines, there are large datasets collected where the researchers want to assess the dependence of random variables upon other variables. One simple example of this is analyzing the relationship between distance traveled and time. We know from common experiences that distance traveled is directly proportional to time. The longer one walks in one direction, the further one has traveled. Assuming that the person walks at constant speed, it is easy to assign a direct linear relationship between distance traveled and time because distance is just Speed × Time. However, it is unlikely that someone will walk at a constant speed and there won't be a perfect linear relationship between distance traveled and speed. However, if we were to fit a line through as many of the points as possible, we would still see a general linear trend to the data. This line is an example of a parametric method called Ordinary Least-Squares (OLS) or simple linear regression. In this paper, we discuss regression analysis and its applications from a nonparametric viewpoint. Whereas OLS assumes several things about the raw data, such as an underlying linear relationship and normally distributed error, nonparametric regression reduces or eliminates the assumptions upon the data and allows more general conclusions to be made to a greater variety of data. In particular, we examine penalized spline regression, which uses splines, or piecewise continuous polynomials, paired with a mathematical penalization to find the best fit to the data. To help the reader to understand the concepts being discussed, we will use several datasets from a 2012 neuroscience experiment exploring the human oculomotor system.

2 Background Concepts

Before we more formally define what regression analysis is, we provide some basic statistical background and definitions from [8] that are needed for an understanding of the paper.

2.1 Basic Statistical Definitions

A *random variable*, X, is any observed numerical value from the population of interest. This random variable can be continuous (such as height) or discrete (such as heads or tails) depending on its probability mass or density function. A *probability mass* function, P(X), is all of the probabilities associated with each of the outcome of a discrete random variable. For example, for a single coin flip, the two possible outcomes are heads or tails with both outcomes having a 50% chance of occurring. Therefore the probability mass function is

$$P(X) = \begin{cases} 0.5 & : \quad \text{X is heads} \\ 0.5 & : \quad \text{X is tails} \end{cases}$$

Similarly, the probability density function is the continuous analog to the probability mass function where the height of the graph represents the probability density at that point. By integrating under the probability density function for a range of x-values, it is possible to find the probability of getting a result within the specified x-range. As the name implies, this function describes the probability density at any given point, and thus the probability of any specific outcome is 0, because there are an infinite number of possible outcomes. An example of this is the uniform distribution, where all of the outcomes are equally likely, i.e.

$$P(x) = \begin{cases} \frac{1}{(b-a)} & x \in [a,b] \\ 0 & x \notin [a,b] \end{cases}$$

If we wanted to find the probability that $x = 0.5$, i.e. $P(x = 0.5)$, the probability would be 0, because there are an infinite number of possible outcomes and no single point can have any weight. As we are taught in Calculus, $\int_a^a = 0$, so we can easily confirm that $P(x = 0.5) = 0$ by definition. As these two examples demonstrate, the probability density function and probability mass function both must have a total area under the curve of 1, essentially meaning that 100% of the possible outcomes are accounted for. A collection of random variables is said to be *independently distributed* or *statistically independent* if the variables do not affect the outcome of any of the other variables. For example, during coin flips, the previous flips do not affect the next flip. Every time one has a 50% chance of getting heads or tails (for a fair coin). To contrast this, the probability of picking a red marble from a bag of mixed marbles without replacing the marbles is not independent. As marbles are removed from the bag, the marbles removed affect the probability of drawing a red marble from the bag. A collection of random variables is said to be *identically distributed* if all the variables come from the same probability distribution or density function. An example would be the heights of people at Whitman. Everyone comes from the same height distribution. However, it would not be identically distributed if we were looking at heights of Whitman students and heights of NBA players. The NBA players would have a much different height distribution than students at Whitman. It is a common assumption in statistics that our variables are independent and identically distributed (i.i.d).

2.2 Regression Model

A regression model is of the form

$$Y = f(X_1, X_2, \dots, X_n) + \varepsilon$$

where Y is called the *response* variable, $X_1, X_2, \dots, X_n$ are the *predictor* variables, and ε is the error, or difference between our model and actual values [1]. One of the goals of any regression model is to minimize the error (ε) for all y values without introducing extraneous and arbitrary random variables. Throughout this paper, we focus solely on the relationship between a single predictor variable and the response variable, so we will use X in place of X_1 to remove possible confusion and write our regression models as

$$Y = f(X) + \varepsilon \tag{1}$$

for some function f.

2.3 Parametric vs. Non-Parametric

Depending on the branch of statistics, specific assumptions are made about the distribution of the data, such as the data being independent and identically distributed. One of the most common statistical branches used is *parametric* statistics, which assumes a specific distribution for the data. This distribution is frequently assumed to be a normal, or Gaussian, distribution, because many natural processes and phenomena are approximately normal. Although this branch of statistics is widely used in the sciences and is the branch of statistics most often taught in introductory statistics classes, it is not always the most appropriate. If the data fails to follow the assumed distribution, then the conclusions drawn are based upon false assumptions and are frequently inaccurate and misleading. If the data is not sufficiently "normal" then there a couple of options. If the true distribution, such as an exponential, is known, there are specific parametric tests and models for that distribution that can be applied. Another common option is to take a transform of the data and see if the transformed data is normal or follows another known distribution. If the data is still not normal or easily identifiable as a specific known distribution, then *non-parametric* statistics are needed for the analysis. As the name implies, this branch of statistic uses fewer assumptions about the data, such as only assuming a continuous distribution for the data. Overall the conclusions drawn are typically more rigorous because of the weakened assumptions about the data's distribution. However, as a consequence of these weakened assumptions, the distribution and regression models applied are typically a lot more complicated and harder to interpret. In other words, for data that is somewhat close to a known distribution, non-parametric methods add unnecessary complexity. This is undesirable, since the analyst wants as simple a model as possible and to be able to interpret their model's terms and parts. To evaluate if a parametric model is appropriate, there are distributional tests that assess the validity. Despite this increased complexity, nonparametric tests are often a good choice because of the increased flexibility and applicability of the models.

3 The Steps of Regression Analysis

With any regression analysis, there is a sequence of informal steps for the analysis. The authors of [1] do a good job of summarizing it in the following list:

Steps in Regression Analysis

1. State the problem
2. Select potentially relevant variables
3. Collect data
4. Specify a possible regression model
5. Choose a fitting method
6. Fit the model
7. Critique the model and repeat as needed

8. Use the chosen model to answer the initial problem

We do not address or explain everything in this list, but it is a good framework to refer back to as we derive and explain different regression models for our data.

3.1 State The Problem

From summer research at Whitman College conducted by Dr. Thomas Knight and the author, we have a large collection of data from an experiment examining oculomotor control in human subjects. The research was focused on understanding the neural pathways underlying the control of saccadic gaze movements. A *saccade* is any rapid change in eye position that realigns the subject's vision with a new visual target. The researchers were specifically interested in large-amplitude gaze movements, or those that involve a head movement in addition to an eye movement. Because gaze is defined as the summation of eye-in-head position and head-in-space position, any given gaze shift can be completed with an infinite number of different combinations of eye and head movements ($\Delta G = \Delta H + \Delta E$). For example, a 45° shift could be completed entirely with eye movement ($\Delta E = 45°$), entirely with a head movement ($\Delta H = 45°$), or a combination of eye and head movements ($\Delta H + \Delta E = 45°$). It has been previously shown, but unexplained, that there is large intersubject variability in the amount of head movement for any given amplitude gaze shift. The researchers hypothesized that the variability was due to a variable called the customary oculomotor range (COMR). To test this hypothesis, they restricted the visual field of 16 human subjects to the same ±5° aperture to see if the variability was eliminated for 45°and 90°gaze shifts. As a control condition, the same visual shifts were tested without any visual impairments. All of the data was recorded using the EyeSeeCam Gaze, a high-speed, portable videooculography system that recorded eye and head velocities, eye positions, and eye and head accelerations (angular and linear) at 300 frames/sec. To obtain head position, the head velocity was numerically integrated using the Trapezoidal Rule and a reference starting head position as the integration constant.

With this research in mind, we formulated two different regression problems to examine in one of the human subjects for the control condition.

The first problem (Example One) is the relationship between change in gaze position (ΔG) and change in head position (ΔH). This is a direct measure of the intersubject variability between subjects because the graph demonstrates the range of gaze movements that are completed without or with head movements. By comparing this across subjects in the Aperture experiment, it would be possible to see if the variability had been removed or reduced. By fitting a regression model for the control condition, we are creating a protocol to assess intersubject and intrasubject variability that can be easily applied to all 16 subjects or other similar oculomotor experiments.

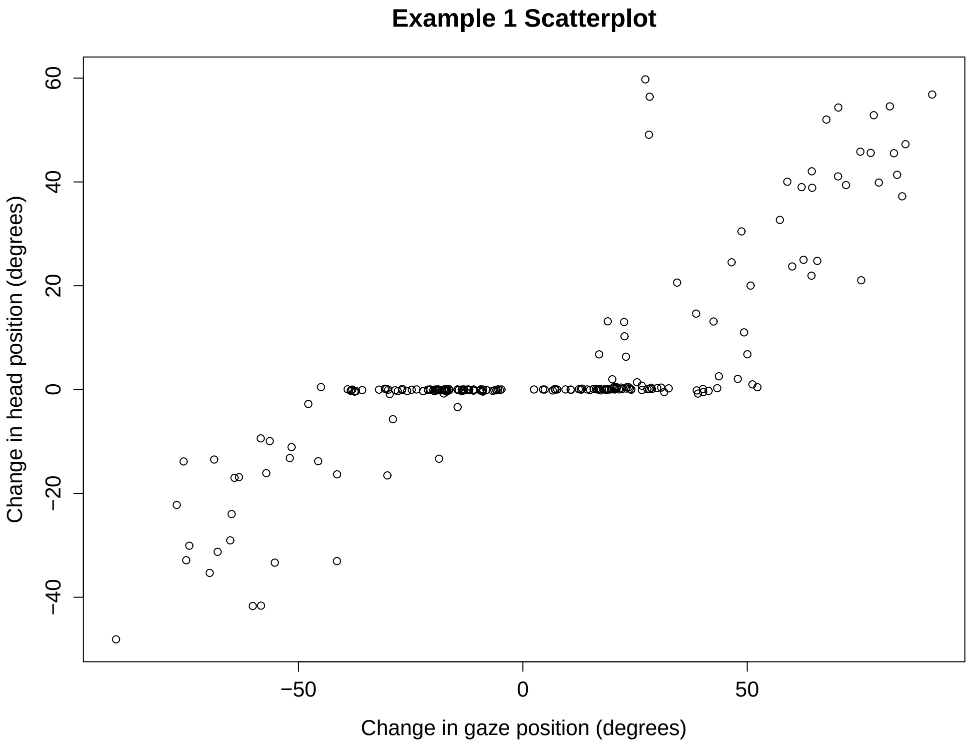

Figure 1: Examining the relationship between amplitude of gaze shift and change in head position. Middle region corresponds to a gaze shift completed entirely with eye movements, whereas the two end regions are completed with large head movements paired to eye movements.

The second problem (Example Two) is the relationship between change in gaze (ΔG) and the duration of the gaze shift (ΔT or Gdur). This relationship was explored to confirm that we had a valid neuroscience dataset that agreed with previous research.

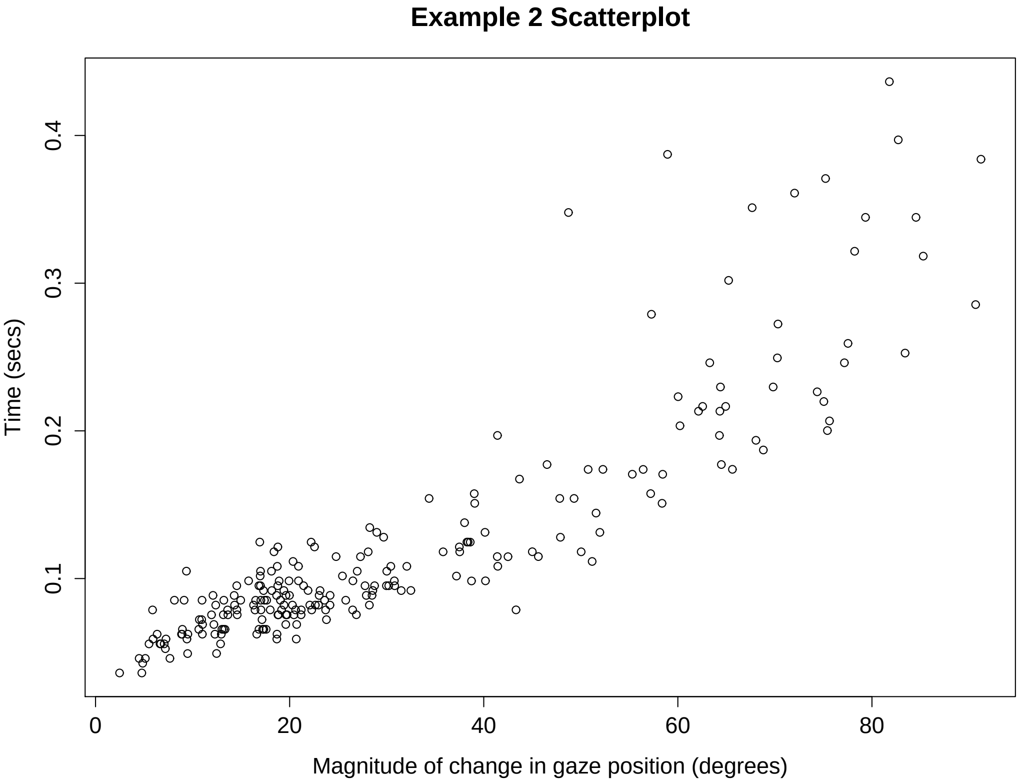

Figure 2: Examining the relationship between amplitude of gaze shift and time to complete the shift. The initial part could be modeled with a linear equation, but as the amplitude increases, the relationship becomes nonlinear.

3.2 Specify a Possible Regression Model

Throughout the course of this paper, we will use four different regression models: simple linear model, second-order linear model, unpenalized spline model, and penalized spline model. We begin with simple linear regression because it explains the concepts of regression well and is the most commonly used regression method in the sciences. We then demonstrate the advantages and disadvantages of unpenalized splines, a simple nonparametric method. Finally, we show how this basic nonparametric method can be improved by introducing a penalization term into the spline models.

4 Simple Linear Regression

4.1 Theory

Simple linear regression, or ordinary least-squares (OLS), as the name implies fits a straight line to the dataset of interest. Looking back at Equation 1, we see that our $f(x)$ is a straight line, i.e. $f(x) = \beta_0 + \beta_1 x$, thus

$$Y = \beta_0 + \beta_1 X + \varepsilon \qquad (2)$$

where ε is the error term accounting for the difference between the predicted and observed Y values. As part of the model, we assume that the error has a mean of zero, a constant variance of σ^2, and is independently distributed. This is essentially saying that each error term is 'centered' about the line of best fit (mean of 0) and that there is a constant amount of spread or deviation of the error terms from the line of best fit (constant variance). These assumptions must later be confirmed for each application of the model. The results are not considered accurate if these assumptions are found to be false. To find the line of best fit through our scatterplot of (x, y)-values, we want to minimize ε for all of our experimental y-values. To do this we define the following modified linear equation to describe our n datapoints:

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i \tag{3}$$

where x_i and y_i for $i = 1, 2, 3 \ldots n$, are in the two collections of random variables being compared. To minimize the overall sum of the squared errors ($\sum \varepsilon_i^2 = 0$) while estimating our two parameters, we apply a method called the method of least squares to estimate the parameters β_0 and β_1. Because we are now attempting to find these population parameters, we notate estimated parameters or random variables with a hat, i.e. the estimated β_0 is notated $\hat{\beta}_0$. To formalize the minimization of the error, let $S(\beta_0, \beta_1)$ represent the sum of the squared errors for all the random samples, i.e.

$$S(\beta_0, \beta_1) = \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i)^2.$$

In order to find the overall summation of the error, each ε is squared so that we remove the sign of the error, but don't have to deal with the non-differentiability of absolute value functions. In order to minimize S, we need to find the critical points of this function, or when the partial derivatives with respect to β_0 and β_1 are zero. This gives us two equations that can be simplified to the following:

$$n\hat{\beta}_0 + \hat{\beta}_1 \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i \tag{4}$$

$$\hat{\beta}_0 \sum_{i=1}^{n} x_i + \hat{\beta}_1 \sum_{i=1}^{n} (x_i)^2 = \sum_{i=1}^{n} y_i x_i \tag{5}$$

These two equations, Eqs. (4) and (5), are collectively referred to as the least-square normal equations and have the following two solutions ([8]):

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}$$

and

$$\hat{\beta}_1 = \frac{\displaystyle\sum_{i=1}^{n} y_i x_i - \frac{1}{n}\left(\sum_{i=1}^{n} y_i\right)\left(\sum_{i=1}^{n} x_i\right)}{\displaystyle\sum_{i=1}^{n} x_i^2 - \frac{1}{n}\left(\sum_{i=1}^{n} x_i\right)^2}.$$

Luckily these two formulas can be simplified based upon the observations that the numerator and denominator of the formula for $\hat{\beta}_1$ are the corrected sum of cross products of x_i and y_i and corrected sum of squares of x_i respectively [8]. This is represented as:

$$S_{xy} = \sum_{i=1}^{n} y_i x_i - \frac{1}{n}\left(\sum_{i=1}^{n} y_i\right)\left(\sum_{i=1}^{n} x_i\right) = \sum_{i=1}^{n} y_i(x_i - \bar{x})$$

and

$$S_{xx} = \sum_{i=1}^{n} x_i^2 - \frac{1}{n}\left(\sum_{i=1}^{n} x_i\right)^2 = \sum_{i=1}^{n}(x_i - \bar{x})^2.$$

Although our source [8] claimed these relations were true, we confirm that the denominator is S_{xx}.

$$S_{xx} = \sum_{i=1}^{n}(x_i - \bar{x})^2 \overset{?}{=} \sum_{i=1}^{n} x_i^2 - \frac{1}{n}\left(\sum_{i=1}^{n} x_i\right)^2$$

To begin, we expand the left side, distribute the summation through the terms, and pull the constants outside of the summations.

$$\begin{aligned} S_{xx} = \sum_{i=1}^{n}(x_i - \bar{x})^2 &= \sum_{i=1}^{n} x_i^2 - 2\bar{x}x_i + \bar{x}^2 \\ &= \sum_{i=1}^{n} x_i^2 - \sum_{i=1}^{n} 2\bar{x}x_i + \sum_{i=1}^{n} \bar{x}^2 \\ &= \sum_{i=1}^{n} x_i^2 - 2\bar{x}\sum_{i=1}^{n} x_i + n\bar{x}^2 \end{aligned}$$

Next, we make a substitution for $\bar{x}$ using the definition of $\bar{x}$ and simplify the resulting equation.

$$\begin{aligned} S_{xx} &= \sum_{i=1}^{n} x_i^2 - 2\bar{x}\sum_{i=1}^{n} x_i + \bar{x}\sum_{i=1}^{n} x_i \qquad \text{because} \qquad \bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \\ &= \sum_{i=1}^{n} x_i^2 - \bar{x}\sum_{i=1}^{n} x_i \end{aligned}$$

Finally, we make another substitution for $\bar{x}$ to generate the final equation and confirm that the denominator of $\hat{\beta}_1$ is S_{xx}.

$$\begin{aligned} S_{xx} &= \sum_{i=1}^{n} x_i^2 - \frac{1}{n}\sum_{i=1}^{n} x_i \sum_{i=1}^{n} x_i \\ &= \sum_{i=1}^{n} x_i^2 - \frac{1}{n}\left(\sum_{i=1}^{n} x_i\right)^2 \\ S_{xx} = \sum_{i=1}^{n}(x_i - \bar{x})^2 &= \sum_{i=1}^{n} x_i^2 - \frac{1}{n}\left(\sum_{i=1}^{n} x_i\right)^2 \end{aligned}$$

A similar proof can be done for S_{xy}.

4.2 Matrix Representation

Before we explain how to evaluate the fit of our model and provide some examples of the whole process, we want to reformulate our linear model in terms of matrices to be more notationally concise and to expose the reader to a common notation for regression models.

From linear algebra, we know that we can represent our simple linear model as $\hat{\mathbf{y}} = \mathbf{X}\mathbf{b}$ where

$$\hat{\mathbf{y}} = \begin{bmatrix} \hat{y}_1 \\ \hat{y}_1 \\ \vdots \\ \hat{y}_n \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}, \quad \text{and} \quad \mathbf{b} = \begin{bmatrix} \hat{\beta}_0 \\ \hat{\beta}_1 \end{bmatrix}. \tag{6}$$

where $\hat{\boldsymbol{\beta}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{Y}$

4.3 Validating the Assumptions

Having explored and confirmed the basic equations behind the method of least squares, we move onto assessing whether the model is appropriate. As [8] points out, there are three common questions raised

1. How well does the model fit the data?
2. How accurate is the model at predicting future values?
3. Are any of the basic assumptions, such as constant and independent variance, violated?

Because questions one and two can be answered by first examining whether the basic assumptions are violated, we begin by checking our assumptions. Although [8] mentions there are better, more advanced methods for evaluating if the assumptions are met, we use the common practice of residual analysis and the *quantile-quantile* (Q-Q) plot.

A *residual* is the difference between the observed and predicted values, i.e. $\varepsilon_i = y_i - \hat{y}_i$. Residual analysis thus examines the error for each predicted y-value. By graphically viewing the residuals it is possible to detect if the assumptions are violated. If there is a clear pattern to the residual plots rather than a random distribution of the residuals, it indicates one of the assumptions may be violated. For example, if the residuals are initially quite small but fan out and get increasingly large, then it would indicate that the variance is not constant.

Another way to graphically check the assumptions is through a Q-Q plot, which plots the sample quantile against the theoretical quantile to determine if two data sets come from a population with a specified distribution. Because we are interested if our errors follow a normal distribution, we want a normal Q-Q plot. As the name implies, we are comparing the quantiles of our first dataset against the quantiles of our second dataset, where quantile refers to how much of the data falls below that point. If there is a normal distribution then we would expect there to be a 45° through the data.

4.4 Evaluating the Model's Fit

There is a measure of the model's fit called the coefficient of determination, or R^2. This measure can range from 0 to 1 and is defined as

$$R^2 = \frac{SS_R}{SS_T} = 1 - \frac{SS_{Res}}{SS_T} \tag{7}$$

where SS_T is the corrected total sum of squares, SS_R is the regression sum of squares, and SS_{Res} is residual sum of squares.

To help understand these three sums of squares and how they explain the model's fit to the data, we briefly digress from regression. The residual sum of squares is defined as

$$SS_{Res} = \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2.$$

Using this formula, we see that SS_{Res} can be visualized as the error, or unexplained variance, in the model. The regression sum of squares is defined as

$$SS_R = \sum_{i=1}^{n} (\bar{Y} - \hat{Y}_i)^2$$

and can be conceptualized as the amount of variability in the data set explained by the linear model. Finally, the corrected total sum of squares is defined as

$$SS_T = \sum_{i=1}^{n} (Y_i - \bar{Y})^2$$

and is the total variance in the model. It is called the corrected total sum of squares, because it is corrected for the mean. Similarly the uncorrected total sum of squares is

$$SS_T = \sum_{i=1}^{n} Y_i.$$

In Equation 7, SS_T refers to the corrected total sum of squares.

Through algebraic manipulation we see that the corrected sum of squares is equal to the residual sum of squares plus the regression sum of squares, i.e.

$$SS_T = SS_{Res} + SS_R.$$

From an intuitive standpoint this makes sense because if SS_R is total explained variance and SS_{Res} is total unexplained variance, then the total variance should be the total variance explained by the model plus the total unexplained variance unexplained by the model.

Returning to the topic of the coefficient of determination, we are now ready to break the formula apart and understand why it is a measure of the model's fit to the data. The first half of Equation 7, $R^2 = SS_R/SS_T$ is the amount of explained variability over the amount of total variability. As more and more of the variability is explained by the model, then this ratio will approach 1. Similarly, the second half of Equation 7 $R^2 = 1 - SS_E/SS_T$ subtracts the fraction of variance not explained by the model, so as less and less variability can be explained by the model, this quantity approaches zero. This helps give a more natural understanding of what R^2 represents. This also explains why the coefficient of determination is frequently called "proportion of variation explained by the regressor" [8]. Although R^2 is usually a good measure of the model's fit, like anything, it has its disadvantages. One of the largest such problems is that R^2 will be increased by just adding more terms to the model, even if those terms are not appropriate for the relationship being examined. Logically it follows that although R^2 may be close to one, the model may not be an accurate predictor of future events. The model generated can explain most of the variability in the current data but a high R^2 value does not imply it can predict future events well. Despite these drawbacks, we will use R^2 to help evaluate the fits of our linear models.

4.5 Application

4.5.1 Example 1: Relationship between change in head and change in gaze

We are investigating the relationship between change in head (response) and change in gaze (predictor). This example is predicted to have a linear relationship but be poorly fit by the linear model. We expect for any gaze shift over $20 - 40°$ that there will be a head movement and that it will be linearly proportional to the change in gaze. The issue will be the center where there is no head movement. During that region, the gaze shift can be completed entirely with eye movements and a head movement would be a needless waste of energy. The data and the line of best-fit can been seen in Fig. 3.

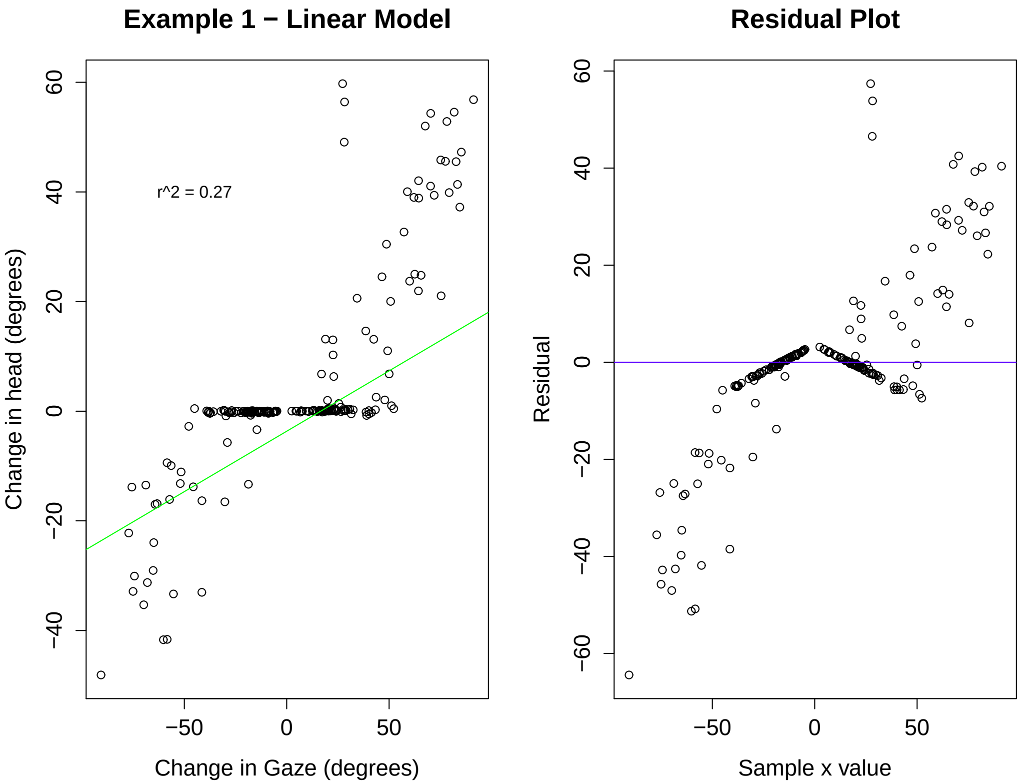

Figure 3: Simple linear regression applied to sample dataset along with residual plot.

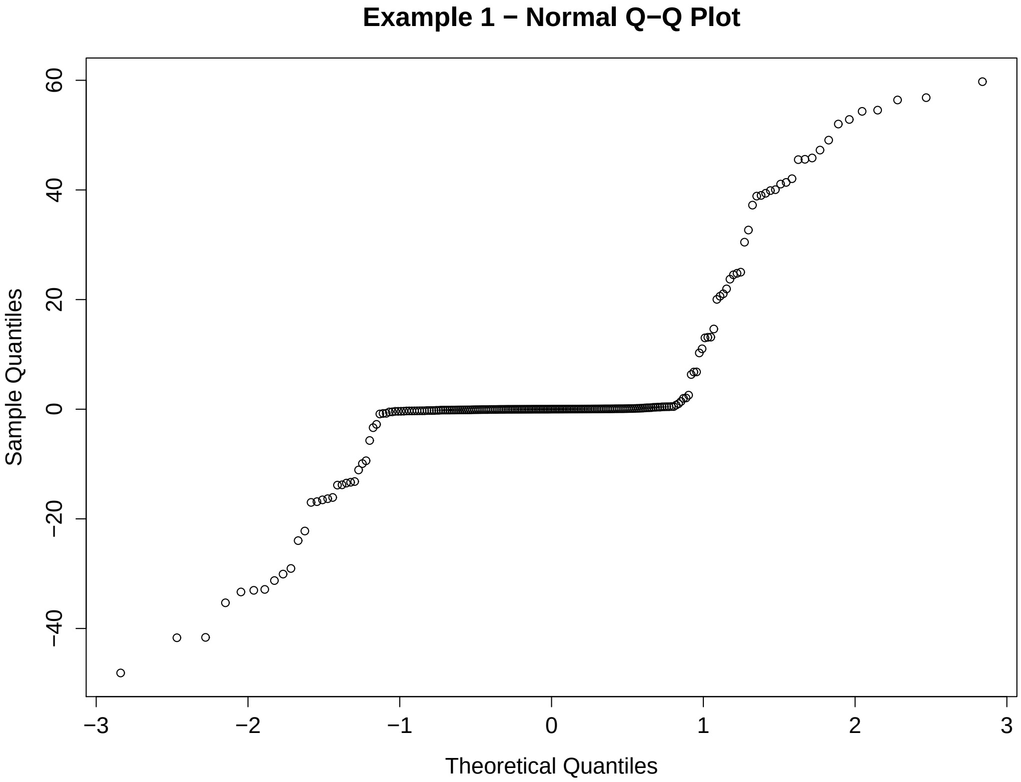

Figure 4: Normal Q-Q plot for linear model 1

Because this model appears to have a piecewise linear fit, the simple linear model does not fit it well. This is reflected in both the residual and Q-Q plots. In the residual (Fig. 3), we see a clear pattern to the residuals with the only small residuals being close to $x = 0$. In the Q-Q plot (Fig. 4), we clearly see 3 distinct parts that fail to meet the normality assumptions, but upon breaking up, may satisfy the normality conditions. This conclusion about the poor fit is further supported by the weak R^2 value of 0.27. This value indicates that the linear model does not explain most of the variability in our data.

4.5.2 Example 2: Relationship between gaze amplitude and time

We expect a strong linear relationship because as size of the shift increases, it should take longer to execute the gaze shift. However, the model may fall apart towards the high end of the dataset because head velocity is not constant throughout the shift, but rather continues to increase throughout most of the gaze shift. Figure 5 shows the relationship between size of the gaze shift (predictor variable) and gaze duration (response variable) along with the linear model fit to the data.

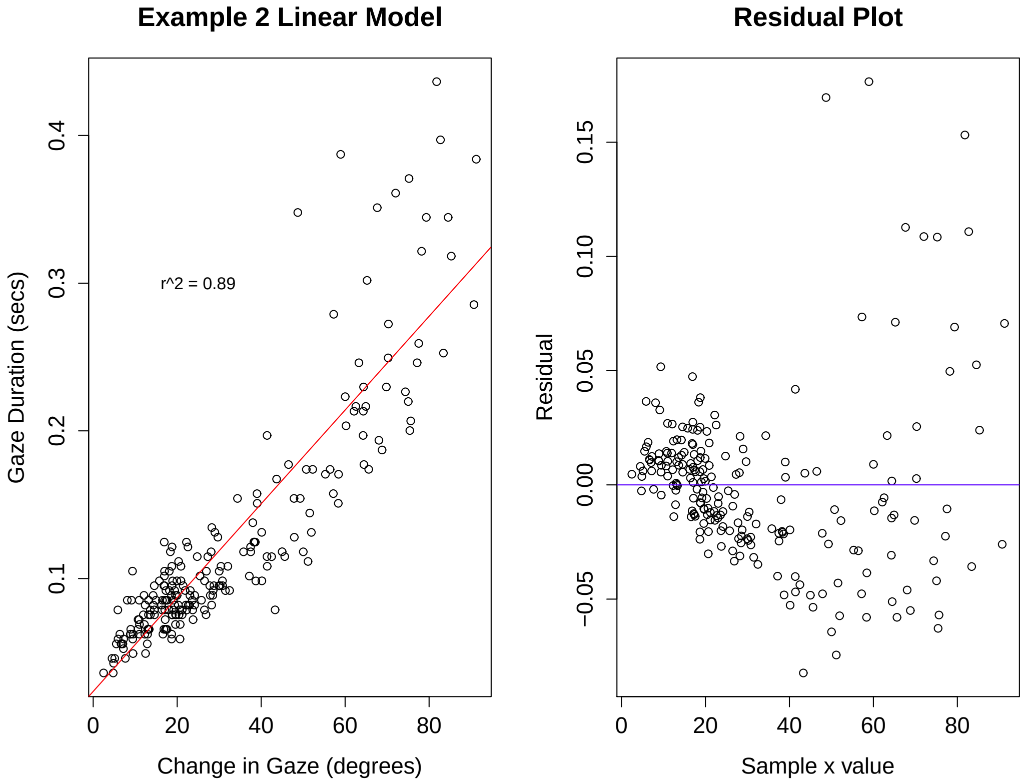

Figure 5: Simple linear regression applied to sample dataset along with residual plot.

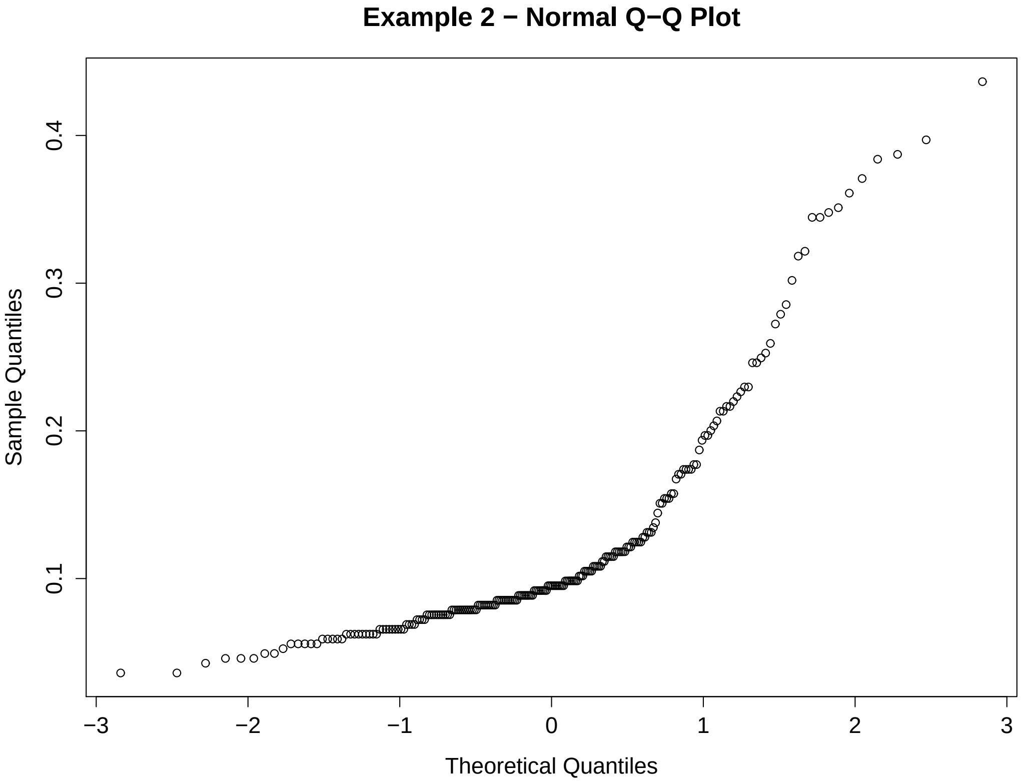

Figure 6: Normal Q-Q plot for linear model 1

As expected, there is an overall upward and linear trend to the data. However, when we start to check our assumptions by looking at the residual and Q-Q plot, we see problems. In the residual plot (Fig. 5), we see a pattern to the model's errors. Initially there are quite small residuals, but at the high end, the residuals become more variable, supporting our belief that the model falls apart towards the upper end. Finally, by looking at the Q-Q plot (Fig. 6), we see that it fails to satisfy the normality conditions because a straight diagonal line cannot be drawn through the plot. All of these indicate that a linear model does not fit the data as well as predicted and may only be valid for the short interval of gaze shifts near zero.

Overall, by examining these previous two models, we see some of the uses and limitations of simple linear regression. This also starts to motivate the need for other models.

5 Inference using the Simple Linear Regression Model

To assess whether a simple linear model and its parts (slope and y-intercept) are statistically significant, we can build confidence intervals and run statistical tests to assess the probability of obtaining the observed results. As a brief reminder, confidence intervals describe a range of values that would contain the true parameter $(1-\alpha)\%$ of the time. In other words, if we collect an infinite number of sets of randomized data from a known distribution and then make 95% confidence intervals for each randomized set, 95% of the intervals generated will contain the true parameter. A statistical test assesses the likelihood of the results under a null hypothesis. As part of this test, an acceptable α value must be chosen, where α is the probability of a Type I error, or probability of falsely rejecting the null hypothesis.

First, let us begin by comparing the slope of our simple linear model to a constant β_N. This is equivalent to saying that our null and alternative hypothesis are:

$$H_0 : \beta_1 = \beta_N$$
$$H_A : \beta_1 \neq \beta_N.$$

As discussed in a prior section, the error, ε, for each data point is assumed to be normally and independently distributed with a mean of zero and constant variance, σ^2. Because of the assumption about normality, we can use a parametric test to quantify the likelihood of the null hypothesis. Because the z-test needs σ^2, a value rarely known, we use an unbiased estimator that follows a known distribution. We then use a t-test to test the stated null hypothesis. As a brief reminder about the general t-test, the test generates a t statistic that follows a t distribution with a specified number of degrees of freedom. By finding the probability of obtaining a t-statistic that or more extreme, the likelihood of results under the null hypothesis is quantified. Based upon the significance level, α, the null hypothesis can then be rejected or failed to be rejected according to the p-value associated with the test statistic. As part of this process, the p-value may need to be duplicated if a two-sided alternative is specified. This comes from the fact that the t-distribution has two tails, and the t-statistic only reflects the probability of more extreme events for one tail. Therefore to find the probability of having the alternative hypothesis be "not equal to" as opposed to "greater than" or "less than", we need to double the p-value obtained.

As [8] points out, the normal distribution of the errors implies that the observations y_i are normally distributed about $\beta_0 + \beta_1 x_i$ with variance, σ^2. It logically follows that $\hat{\beta}_1$ is also normally distributed with a mean of β_1 and variance of σ^2/S_{xx}. To estimate σ^2, we use MS_{Res}, which is an unbiased estimator and follows a χ^2_{n-2} distribution. Using this estimator, we get that our t-statistic (under the distribution for H_0) for comparing slope is of the form:

$$t_0 = \frac{\hat{\beta}_1 - \beta_N}{\sqrt{MS_{Res}/S_{xx}}}. \tag{8}$$

Frequently, to remind the user that the denominator is an estimate for σ, or the standard deviation, the quantity $\sqrt{MS_{Res}/S_{xx}}$ is referred to as the standard error of $\hat{\beta}_1$, or $\mathrm{se}(\hat{\beta}_1)$. Using this test statistic, we can now easily quantify the probability of obtaining a slope that or more extreme under our null hypothesis. One of the most common questions is whether the slope is different than zero, i.e.

$$H_0 : \beta_1 = 0$$
$$H_A : \beta_1 \neq 0.$$

Referring to Equation 8, we see the equation simplifies to

$$t_0 = \frac{\hat{\beta}_1}{\sqrt{MS_{Res}/S_{xx}}}$$

and is quite simple to apply. This hypothesis also corresponds to determining if there is a linear relationship between our predictor and response variable. If there is no linear relationship between the variables, there should be a slope of zero. Similarly, if there is a relationship between the variables, then we would expect a slope that is statistically different from zero. Although failing to reject this hypothesis supports there being no linear relationship, rejecting the null does not necessary mean a linear relationship. (As a quick reminder, we reject the null hypothesis based upon the accepted level of Type 1 error, or probability of incorrectly rejecting the null hypothesis. This level is denoted by α, and for the sciences is typically $\alpha \leq .05$.) A second-order function will have a positive slope, but the relationship is definitely not linear. This reinforces the earlier idea that the residuals and QQ-plot both need to be checked to confirm that the assumptions are met and there is no pattern to the errors.

Next, we want to find the 95% confidence interval for our model's slope. This is quite similar to finding the t-statistic and is given by

$$\hat{\beta}_1 - t_{\alpha/2,n-2}\text{se}(\hat{\beta}_1) \leq \beta_1 \leq \hat{\beta}_1 + t_{\alpha/2,n-2}\text{se}(\hat{\beta}_1)$$

where the t distribution has $n-2$ degrees of freedom and $t_{\alpha/2,n-2}$ is the t-statistic with $\alpha/2$ area in the tail beyond the t-statistic. Finally, we want to be able to predict future interior values using our simple linear model. To accomplish this, we need to find the confidence interval on the mean response at the point x_0. In simpler terms, this corresponds to finding the confidence interval for any interior point of our SLM. Following from the earlier note that each observation y_i is normally distributed, it is relatively simple to build our confidence interval. The $100(1-\alpha)$ % confidence interval of the mean response at the point $x = x_0$ is given by:

$$\hat{\mu}_{y|x_0} \pm t_{\alpha/2,n-2}\sqrt{MS_{Res}\left(\frac{1}{n} + \frac{(x_0-\bar{x})^2}{S_{xx}}\right)}$$

where $\hat{\mu}_{y|x_0} = \hat{\beta}_0 + \hat{\beta}_1 x_0$ is an unbiased point estimator of $E(y|x_0)$. For an explanation of this formula, see [8]. It is possible to plot these confidence intervals for all of the values and generate confidence bands about the simple linear model to indicate the approximate range of expected observations from the line.

Having explored the formulas, we now apply these three parts to the prior neuroscience dataset examining the relationship between gaze amplitude and duration. Previously we saw that the residuals for this graph had a clear pattern and that a linear fit might not be optimal. We suggested that a second-order model may have fit better. This fit can be seen in Figure 7. For an explanation of how this was fit, see [1].

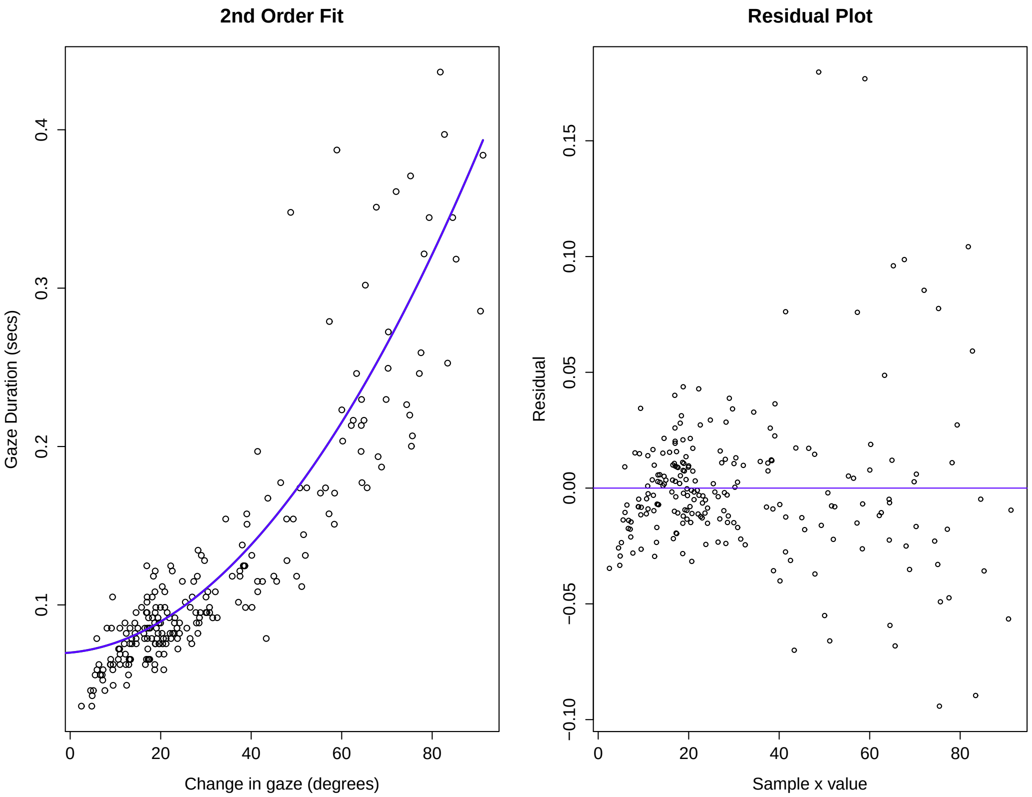

Figure 7: Quadratic fit and residuals for scatter plot of gaze duration vs. amplitude. Adjusted $R^2 = 0.8277$

From this figure, it appears that the second-order fit is more representative of the data. This conclusion is further supported by looking at the residuals which no longer have a clear pattern and the adjusted R^2 value of 0.8277 which is higher than the adjusted R^2 value of 0.7868 for the simple linear model. The adjusted R^2 is quite similar to the previously explained R^2, but corrects for the extra parameters used in the model. The R^2 value will always increase with added model parameters, but adjusted R^2 will only increase if the added parameter has any additional explanatory power [5]. This can be seen in the following equation:

$$R^2_{\text{adj}} = 1 - \frac{(1 - R^2)(n - 1)}{n - k - 1}$$

where n is number of datapoints, k is number of different parameters in the regression model, and R^2 is defined above.

Due to the good fit of the second-order model, we were curious whether the slope t-test would suggest that a linear fit was inappropriate. As part of this, we plotted the confidence bands to see how well these bands explain the observed values. These confidence bands can be observed in Figure 8. At the lower end, the confidence bands contain a large proportion of the points, but as the upper tail is reached, the bands contain a much smaller number of points. This would strengthen our earlier conclusion that the simple linear model is appropriate for gaze shifts under 30 °in amplitude, but is not as appropriate for larger shifts.

Interestingly, the p-value for the two-sided null hypothesis $H_0 : \beta_1 = 0$ was $2*10^{-6}$ indicating that the relationship does have an overall linear trend. As we discussed prior, having a non-zero slope does not actually mean the relationship is linear. The residual and QQ-plot both need to be checked for normality to check for abnormal patterns to the data.

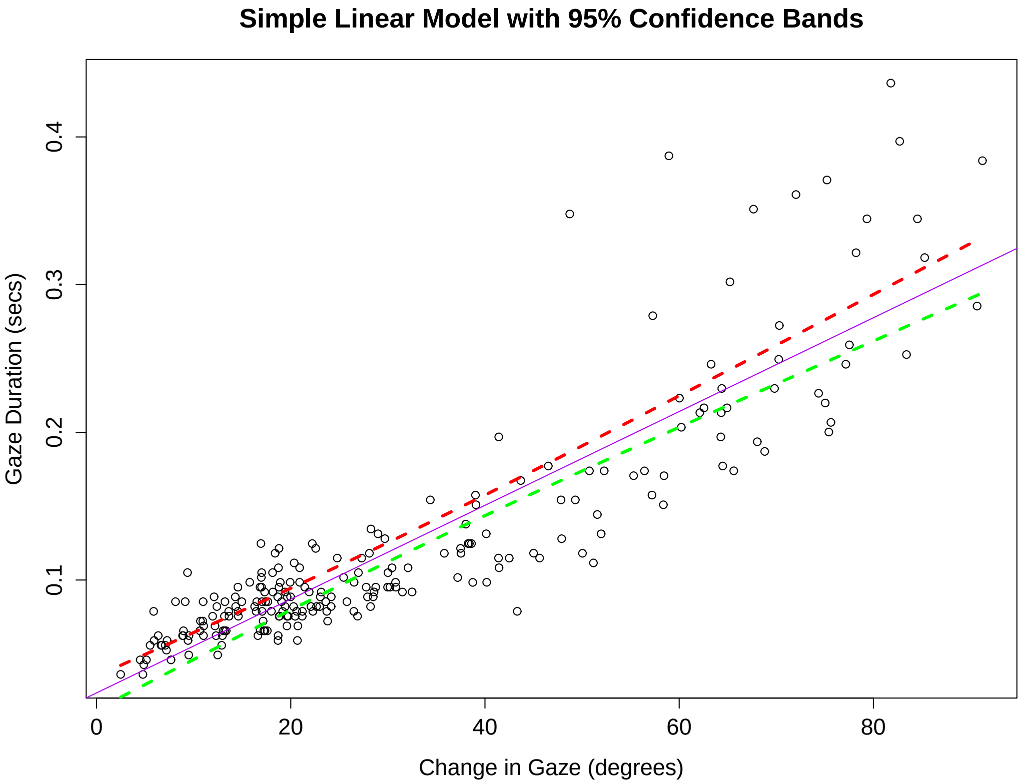

Figure 8: Simple linear model with confidence bands for scatter plot of gaze duration vs. amplitude. Adjusted $R^2 = 0.7868$

6 Spline Regression Models

Having looked at simple linear regression and using it to introduce the concept of regression, we want to expand our simple linear regression model to be more flexible. From before, we know that we can represent our simple linear model as $\hat{\mathbf{y}} = \mathbf{X}\mathbf{b}$ where

b= beta

$$\hat{\mathbf{y}} = \begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_n \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}, \quad \text{and} \quad \mathbf{b} = \begin{bmatrix} \hat{\beta}_0 \\ \hat{\beta}_1 \end{bmatrix}. \tag{9}$$

Y = y

where $\hat{\boldsymbol{\beta}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{Y}$. From linear algebra, you may recall that a *basis* of a vector space is "a set of V of elements of the vector space such that any element of the space can be expressed uniquely as a linear combination of elements of V" [9]. By looking at the equation for $\hat{\mathbf{y}}$, we see that $\hat{\mathbf{y}}$ is just a unique linear combinations of 1 and the x-values, thus our basis for the simple linear model is 1 and x. This means that we can only generate linear solutions. If we have analyzed our data and know that our data's relationship is not linear, but might be a higher-order polynomial, it is relatively easy to add an x^2 or x^3 to our basis by adding two columns to $\mathbf{X}$ and the associated parameters β_2 and β_3 to $\mathbf{b}$. However, this is still limiting and computationally hard. If we were to have a group of data that involved one or more absolute values and had at least one non-differentiable point, then none of these three methods would be able to adequately fit the plot. We know that many absolute value functions can easily be fit using piecewise functions though. Consider $f(x) = |x|$, which is defined as

$$f(x) = \begin{cases} x & : x > 0 \\ -x & : x < 0 \end{cases} .$$

Although there are ways to transform this dataset and still use simple linear or higher-order regression, such as splitting the dataset into two separate datasets or assuming that the function is symmetric and looking at y versus $|x|$, it would be nice to have a single regression model that could be applied without modifying the dataset. For example, $|x|$ could easily be fit using two lines joined together at 0. Luckily, there is such a method called *spline regression*, where the regression model is a piecewise continuous polynomial function. Traditionally a spline is a thin strip of wood that draftsmen or artists used to draw a smooth curve through a set of given points. Mathematically, a *n-degree spline* is a piecewise continuous function that joins multiple n-degree polynomials to generate a smooth curve through a set of points. The junctions of these polynomials are called *knots*, because they tie the functions together into a smooth curve. An example of a spline can be seen in Figure 9, where we fit an interpolating spline to a set of arbitrary data points.

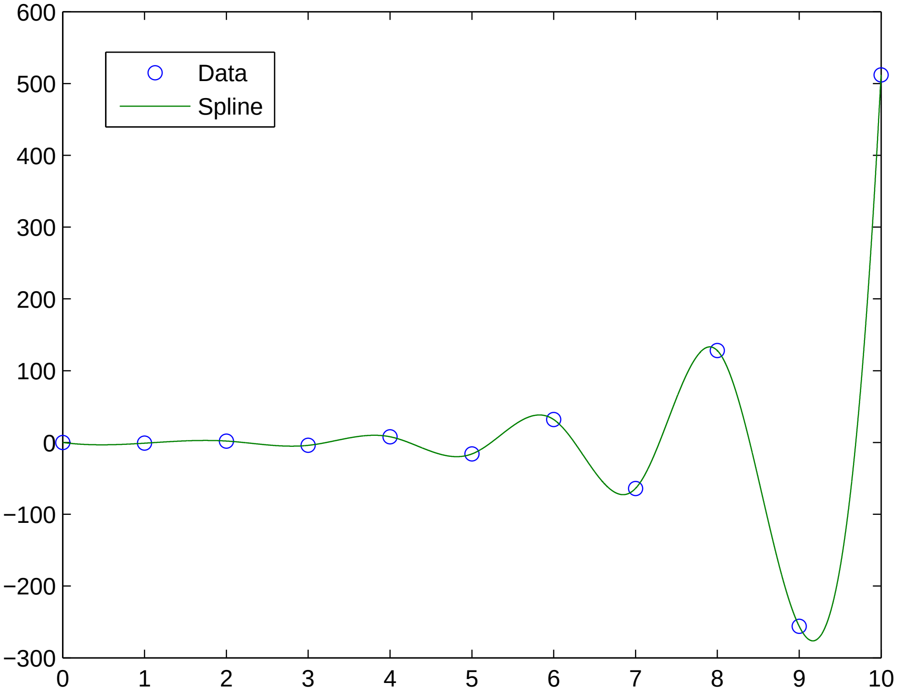

Figure 9: Example of cubic spline

Frequently there are smoothness assumptions added to the spline model, such as differentiability at the knots in addition to interior points, but for a spline regression model, these assumptions are relaxed to allow for linear splines. To explain the underlying theory, we will focus on linear splines.

6.1 Linear Splines

A linear spline is defined as

notation is likely wrong, "k" is suppose to be "i"

$$f(x) = \beta_0 + \beta_1 x + \sum_{i=1}^{K} b_k (x - \kappa_k)_+ \tag{10}$$

where b_k refers to the weight of each linear function and $(x - \kappa_i)_+$ refers to the i^{th} linear function with a knot at κ_i. The parenthetical notation is used to indicate that below κ_i, the linear function is defined to be zero, and only above that point does it have positive value, i.e.

$$(x - \kappa_i)_+ = \begin{cases} x - \kappa_i & : \quad \text{if} \quad x - \kappa_i > 0 \\ 0 & : \quad \text{if} \quad x - \kappa_i \leq 0 \end{cases} .$$

Referring back to our discussion of the basis of our regression models, we see that the basis of our spline model becomes

$$\mathbf{b} = \begin{bmatrix} 1 & \mathbf{x} & (\mathbf{x} - \kappa_1)_+ & (\mathbf{x} - \kappa_2)_+ & \ldots & (\mathbf{x} - \kappa_n)_+ \end{bmatrix} . \tag{11}$$

This allows for a wide variety of shapes to be fit. Looking at Figure 10 from [9], we seen an example of a 'whip' model and the corresponding linear spline basis. The 'handle' is the initial portion for $x \in [0, .5]$ and the 'whip' is the portion for $x \in (.5, 1]$. The whip portion of the data can be well fit by the 10 knots, but it is apparent that we could add additional knots if we were not satisfied with the spline fit. This model is a specific example, because most datasets examined do not have this observed pattern with the initial linear portion followed by a oscillatory portion. Despite this, the example still demonstrates the versatility of spline models and how they can easily be modified to fit a wide variety of data. If there are portions that are linear, those portions only need knots at the beginning and end to define them, but if there are portions that oscillate, then more knots are defined within the portion to fit those sections.

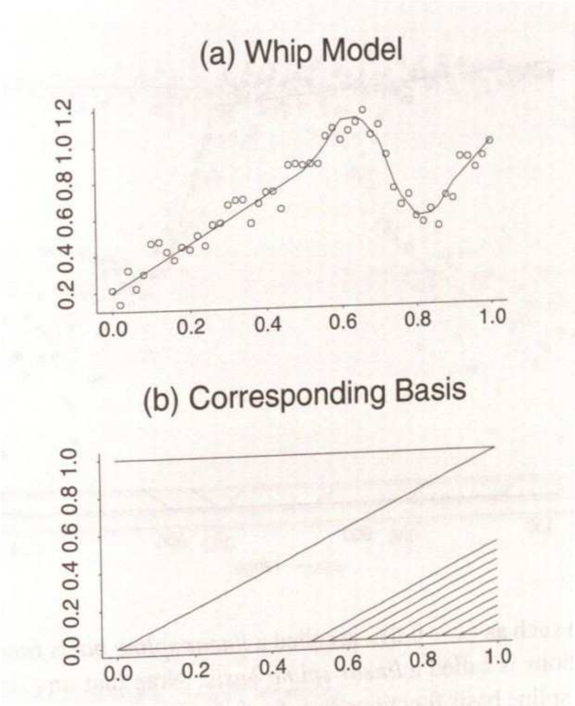

Figure 10: 'Whip' model and its basis. From [9].

By defining these new knots or shifting the existing knots, the linear spline can be easily modified to better fit the data of interest. This is analogous to the idea of adding more predictor variables to our simple linear model. By adding more parameters, the fit can be improved at the cost of increased complexity. However as we will show later, it is possible to add too many knots and over fit the data, picking up random fluctuations that are not significant.

6.2 Examples

6.2.1 Example 1

We first begin with a randomly generated dataset to demonstrate the versatility of spline regression models. We generate 80 points from the non-linear function $f(x) = 10+5\sin\left(\frac{\pi}{24}x\right)$ and add normally distributed error ($\sigma^2 = 9$) to each y-value. Having generated this dataset, we fitted a simple linear model and two linear spline models with varying number of knots. The knots were equally distributed from 0 to 80. The data and fit lines can be seen in Figure 11.

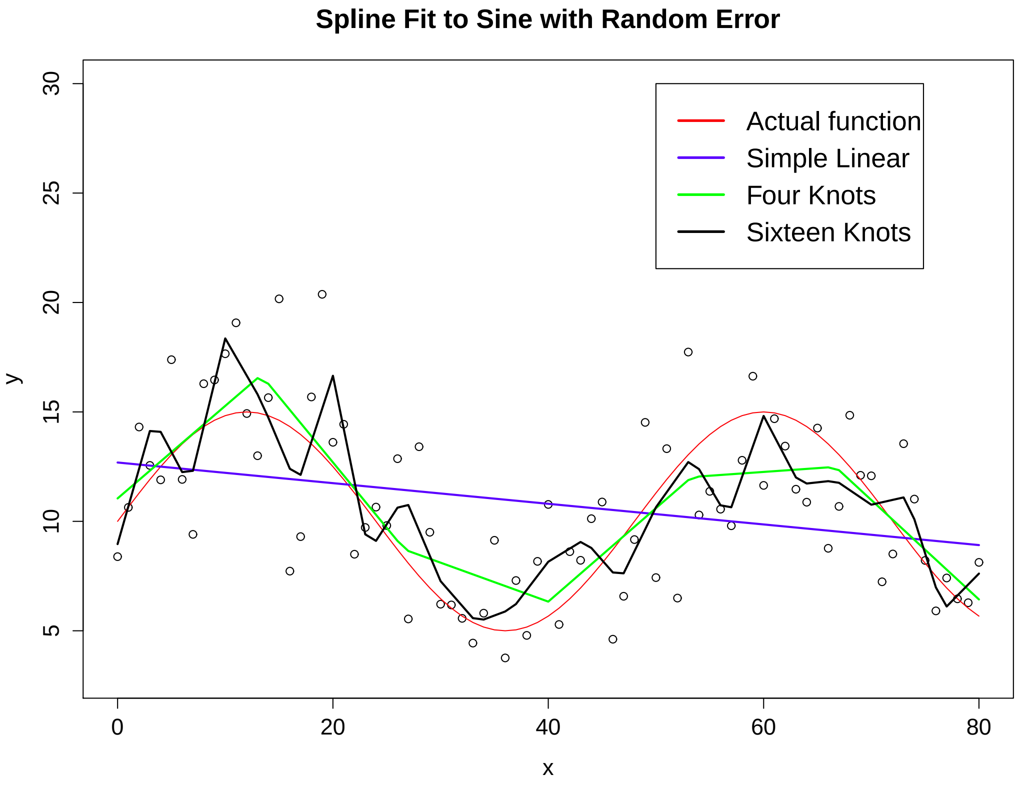

Figure 11: Simple linear and linear spline fits to non-linear dataset

We first see that the simple linear has an abysmal fit to the data as would be expected. This justifies switching to a different method. Although we could consider higher polynomials before using our linear spline, we know that it would need to be of at least degree four to accurately fit the data from 0 to 80. We also know that it would have little predictive power beyond 80. Because of these issues, we move directly to linear splines, because it is easier to add more piecewise functions as our data range increases rather than define a new interpolating polynomial. Although we could start with fewer knots, we know that the simplest spline fit must have at least 3 knots to fit the data's peaks and troughs. To determine this minimum number of knots, we plotted the scatterplot of the data and assess how many times the data changed direction. To add a little bit more flexibility, we begin with 4 knots. Because there are approximately 1.5 periods of the sine function in our plot, we expect a decent fit to be returned by this spline. It fits the overall data pattern quite well, but it is quite jagged and choppy at its knots. Because it appears like we could just increase the number of knots and we would get a much better fit, we also fit a linear spline with 16 knots. Although we expected a good fit with this model, we see that we over fit the data and pick up random fluctuations in the data. This is obviously not ideal, because it is assigning significance to non-significant results. Although this is a computer-generated example, it briefly illustrates some of the issues that may be improved by switching to other types of splines, such as higher-order splines. A higher-order spline would be differentiable at the knots and would thus appear smooth and more aesthetically pleasing. In addition, for most natural datasets, there is rarely an instantaneous transition in the data and having a smooth fit avoids this issue. In the case of the randomized sine function, a higher-order spline would also be able to match the curves at the extremes.

6.2.2 Example 2

We return to our first neuroscience example examining the relationship between the amplitude of a gaze shift and the concomitant change in head position. The spline fit can be seen in Figure 12. Four knots were used at -50°, -25°, 25°, and 50°.

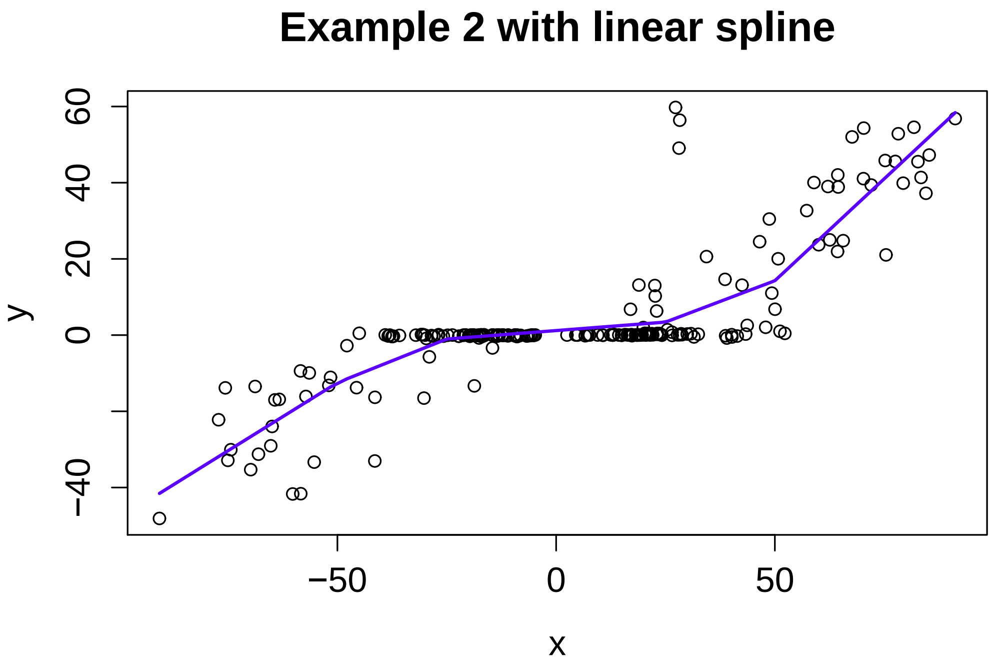

Figure 12: Spline fit for ΔH vs. ΔG with four-evenly spaced knots

From this fit, we immediately see that the spline model is a much better choice than the simple linear model. It accurately follows the overall shape of the graph and has predictive power. However, it does have its issues. In the central region, approximately $[-25, 25]$, we expected a flat line because there are no head movements associated with the gaze shifts.

Although we do not get this line, it is fairly close and with some minor modification of the knot locations, it should be possible to fix this issue. As with the previous example, it is also aesthetically unpleasing to have the curve be non-smooth at the knots. Both of these issues, the poor fit in the central region and the non-smoothness, are minor though because they can be avoided by increasing the order of the polynomials and/or adding more knots to the function and refitting the model.

6.2.3 Example 3

We finally examine the second neuroscience example to quantify the relationship between the amplitude of a gaze shift and the gaze shift duration. The spline fit can be seen in Figure 13. Four knots were used at 20°, 40°, 60°, and 80°.

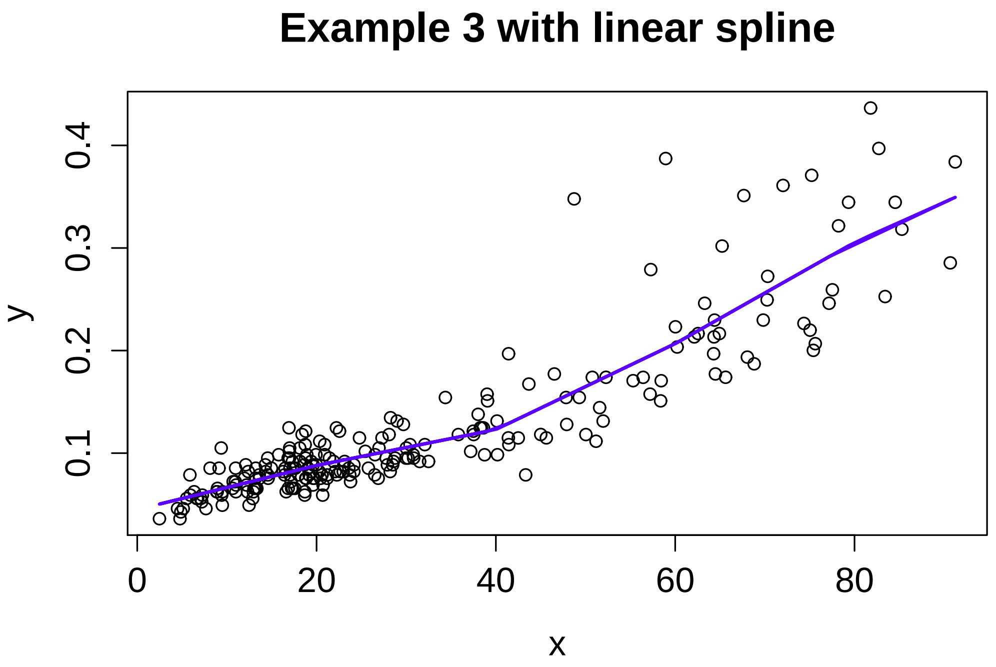

Figure 13: Spline fit to Neuroscience Relationship 2

As with the previous two examples, the spline fit appears to fit the data well with only four knots. It does have the same issues of being non-smooth at the knots and not following the data perfectly, but overall it does fit the data better than the simple linear model. It is hard to tell how it compares to the second-order fit in Figure 7 though. Both of these issues can also be fixed by increasing the order of the spline model and by adding more knots.

6.2.4 Comments

Although all three examples could benefit from increasing the polynomial order, we continue to look at first-order spline models to keep the comparison similar. We instead focus on changing the knots or weights upon each linear component to improve the fit. We saw in

Example 1 that we cannot just increase the number of knots and expect a better fit though. At some point, the spline model acquires extraneous linear components that fit random fluctuations and the model begins to over fit the data. To avoid this issue and optimize the fit, it is possible to manually select the optimal number and location of knots. This has some obvious disadvantages though. As the datasets get increasingly large and complex, it will take longer and longer to manually select the number of knots and locations of each knot. Also, as the number of datasets increases, the time to fit these models also drastically increases. This motivates the next section, penalized splines. As the name implies, a penalized spline imposes a penalization upon the piecewise polynomial components to optimize the fit. Using this method, we can choose a large number of knots (30-40 for an intermediate sized dataset) and penalize the splines for over fitting the data. This makes the application as simple as simple linear regression, but we get a substantially better fit.

7 Penalized Spline Estimation

As just discussed, there is an optimal number of knots that leads to an intermediate amount of smoothing that avoids under- or over-fitting the data. This optimal number can be found by experimentation, but this can time intensive, especially if there are numerous, large, complicated datasets. Luckily there is another method, that of penalized splines, where in our model (Eq. 10), $b_1, b_2, b_3, \ldots, b_K$ are constrained in order to optimize the fit and avoid over fitting the data. In other words, weights are put onto each of the splines to penalize over fitting of the data while still allowing the splines to fit the data well. The weights are determined based upon some penalization criteria. There are several options for the penalization criteria, but the easiest to implement is to choose a C such that

$$\sum_{i=1}^{K} b_i^2 < C.$$

This is an excellent minimization criteria, because it reduces the overall effect of individual piecewise functions and avoids over-fitting the data. This minimization criteria is more formally stated as minimizing the equation $|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}|^2$ subject to $\boldsymbol{\beta}^T\mathbf{D}\boldsymbol{\beta} \leq C$, where

$$\mathbf{D} = \begin{bmatrix} 0 & 0 & 0 & 0 & \ldots & 0 \\ 0 & 0 & 0 & 0 & \ldots & 0 \\ 0 & 0 & 1 & 0 & \ldots & 0 \\ 0 & 0 & 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \ldots & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{0}_{2\,\times\,2} & \mathbf{0}_{2\,\times\,K} \\ \mathbf{0}_{K\,\times\,2} & \mathbf{I}_{K\,\times\,K} \end{bmatrix}. \tag{12}$$

Using Lagrange multipliers, this is equivalent to minimizing

$$|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}|^2 + \lambda^2\boldsymbol{\beta}^T\mathbf{D}\boldsymbol{\beta} \tag{13}$$

for some $\lambda \geq 0$ with respect to $\boldsymbol{\beta}$.

8 Solution to Penalized Spline Equation

Because we now have our minimization problem, we need to solve the equation to find the optimal $\hat{\boldsymbol{\beta}}$ value for a given λ value.

8.1 Derivation of Two Common Matrix Equations

Before we derive the solution, we first show that

1. $\partial(\mathbf{a}^T\boldsymbol{\beta})/\partial\boldsymbol{\beta} = \mathbf{a}$
2. $\partial(\boldsymbol{\beta}^T\mathbf{A}\boldsymbol{\beta})/\partial\boldsymbol{\beta} = 2\mathbf{A}\boldsymbol{\beta}$

where $\mathbf{a}$ is a 2×1 vector, $\mathbf{A}$ is a 2×2 symmetric matrix, $\boldsymbol{\beta} = \begin{bmatrix}\beta_0 & \beta_1\end{bmatrix}^T$, and the partial of $g(\boldsymbol{\beta})$ with respect to $\boldsymbol{\beta}$ is defined as

$$\frac{\partial g(\boldsymbol{\beta})}{\partial\boldsymbol{\beta}} \equiv \begin{bmatrix}\partial g(\boldsymbol{\beta})/\partial\beta_0 \\ \partial g(\boldsymbol{\beta})/\partial\beta_1\end{bmatrix}.$$

1. By matrix multiplication, we know that

   $$\mathbf{a}^T\boldsymbol{\beta} = a_1\beta_0 + a_2\beta_1.$$

   Therefore

   $$\frac{\partial(\mathbf{a}^T\boldsymbol{\beta})}{\partial\boldsymbol{\beta}} = \begin{bmatrix}\partial(a_1\beta_0 + a_2\beta_1)/\beta_0 \\ \partial(a_1\beta_0 + a_2\beta_1)/\beta_1\end{bmatrix} = \begin{bmatrix}a_1 \\ a_2\end{bmatrix} = \mathbf{a}$$

2. By matrix multiplication, we see that

   $$\mathbf{A}\boldsymbol{\beta} = \begin{bmatrix}a_1 & a_2 \\ a_2 & a_3\end{bmatrix}\begin{bmatrix}\beta_0 \\ \beta_1\end{bmatrix} = \begin{bmatrix}a_1\beta_0 + a_2\beta_1 \\ a_2\beta_0 + a_3\beta_1\end{bmatrix}$$

   $$\boldsymbol{\beta}^T\mathbf{A}\boldsymbol{\beta} = \begin{bmatrix}a_1\beta_0^2 + 2a_2\beta_0\beta_1 + a_3\beta_1^2\end{bmatrix}$$

   Using the definition of partial derivatives given above, we get

   $$\begin{aligned}\frac{\partial(\boldsymbol{\beta}^T\mathbf{A}\boldsymbol{\beta})}{\partial\boldsymbol{\beta}} &= \begin{bmatrix}\frac{\partial(a_1\beta_0^2 + 2a_2\beta_0\beta_1 + a_3\beta_1^2)}{\beta_0} \\ \frac{\partial(a_1\beta_0^2 + 2a_2\beta_0\beta_1 + a_3\beta_1^2)}{\beta_1}\end{bmatrix} \\ &= \begin{bmatrix}2a_1\beta_0 + 2a_2\beta_1 \\ 2a_2\beta_0 + 2a_3\beta_1\end{bmatrix} \\ &= 2\begin{bmatrix}a_1 & a_2 \\ a_2 & a_3\end{bmatrix}\begin{bmatrix}\beta_0 \\ \beta_1\end{bmatrix} \\ &= 2\mathbf{A}\boldsymbol{\beta}\end{aligned}$$

8.2 Derivation of Penalized Spline Solution

To find the solution to the minimization of Equation 13, we need to find when all the partial derivatives with respect to β_0 and β_1 are 0. This is equivalent to saying

beta-hat contains beta0 beta1 and all b_i

$$\frac{\partial}{\partial\boldsymbol{\beta}}(||\mathbf{y}-\mathbf{X}\hat{\boldsymbol{\beta}}||^2)+\frac{\partial}{\partial\hat{\boldsymbol{\beta}}}(\lambda^2\hat{\boldsymbol{\beta}}^T\mathbf{D}\hat{\boldsymbol{\beta}})=0.$$

Because differentiation is linear, we can split this into two separate derivatives as follows.

1. By expanding $\partial\hat{\boldsymbol{\beta}}(||\mathbf{y}-\mathbf{X}\hat{\boldsymbol{\beta}}||^2)/\beta$, we get

$$\frac{\partial}{\partial\boldsymbol{\beta}}(\mathbf{y}^T\mathbf{y})-2\frac{\partial}{\partial\boldsymbol{\beta}}(\mathbf{X}^T\mathbf{y}\boldsymbol{\beta})+\frac{\partial}{\partial\boldsymbol{\beta}}(\boldsymbol{\beta}^T\mathbf{X}^T\mathbf{X}\boldsymbol{\beta}).$$

Using the two identities proven above, we see that

$$\frac{\partial}{\partial\hat{\boldsymbol{\beta}}}(||\mathbf{y}-\mathbf{X}\hat{\boldsymbol{\beta}}||^2)=2\mathbf{X}^T(\mathbf{y}-\mathbf{X}\hat{\boldsymbol{\beta}})$$

because $\mathbf{X}^T\mathbf{y}$ is our vector $\mathbf{a}^T$ and $\mathbf{X}^T\mathbf{X}$ is our matrix $\mathbf{A}$.

2. Starting with

$$\frac{\partial}{\partial\hat{\boldsymbol{\beta}}}(\lambda^2\hat{\boldsymbol{\beta}}^T\mathbf{D}\hat{\boldsymbol{\beta}}),$$

we see that by the linearity of differentiation, λ factors out and leaves us with

$$\lambda^2\frac{\partial}{\partial\hat{\boldsymbol{\beta}}}(\hat{\boldsymbol{\beta}}^T\mathbf{D}\hat{\boldsymbol{\beta}}).$$

Because $\mathbf{D}$ is symmetric, we can apply one of our differentiation identities and obtain

$$\frac{\partial}{\partial\hat{\boldsymbol{\beta}}}(\lambda^2\hat{\boldsymbol{\beta}}^T\mathbf{D}\hat{\boldsymbol{\beta}})=2\lambda^2\mathbf{D}\hat{\boldsymbol{\beta}}.$$

3. Combining these partial derivatives, we now see that

$$\lambda^2\mathbf{D}\hat{\boldsymbol{\beta}}-\mathbf{X}^T(\mathbf{y}-\mathbf{X}\hat{\boldsymbol{\beta}})=0.$$

From this, the solution easily comes with some linear algebra manipulations as follows.

$$\begin{aligned}\lambda^2\mathbf{D}\hat{\boldsymbol{\beta}}&=\mathbf{X}^T(\mathbf{y}-\mathbf{X}\hat{\boldsymbol{\beta}})\\ \lambda^2\mathbf{D}\hat{\boldsymbol{\beta}}&=\mathbf{X}^T\mathbf{y}-\mathbf{X}^T\mathbf{X}\hat{\boldsymbol{\beta}}\\ \mathbf{X}^T\mathbf{X}\hat{\boldsymbol{\beta}}+\lambda^2\mathbf{D}\hat{\boldsymbol{\beta}}&=\mathbf{X}^T\mathbf{y}\\ (\mathbf{X}^T\mathbf{X}+\lambda^2\mathbf{D})\hat{\boldsymbol{\beta}}&=\mathbf{X}^T\mathbf{y}\\ \hat{\boldsymbol{\beta}}&=(\mathbf{X}^T\mathbf{X}+\lambda^2\mathbf{D})^{-1}\mathbf{X}^T\mathbf{y} \qquad (14)\end{aligned}$$

Having derived the solution to find the $\hat{\boldsymbol{\beta}}$ vector, we can now fit our penalized spline to our dataset, because we know that $\hat{\mathbf{y}} = \mathbf{X}\hat{\boldsymbol{\beta}}$, i.e.

$$\hat{\mathbf{y}} = \mathbf{X}(\mathbf{X}^T\mathbf{X} + \lambda^2\mathbf{D})^{-1}\mathbf{X}^T\mathbf{y}.$$

However, this is dependent upon λ, a value that must be estimated for our datasets. Although this value can be subjectively determined by a guess-and-check method, that method is not ideal for large datasets or for fitting multiple similar datasets. A great example of where manually choosing λ is almost impossible is during large intersubject comparisons, such as the type that we completed last summer during our neuroscience research. We have 16 subjects to compare, and we don't have time to manually go through every dataset to find the ideal λ for every non-parametric relationship we wish to examine. This is where automatic λ selection becomes important and desirable.

9 Automatic Scatterplot Smoothing

There are algorithms that automatically choose knot position, degree of the splines, and the λ. Because we know that a cubic spline will easily describe any patterns in our neuroscience data, we do not need the algorithms that compute the ideal spline degree. In a similar fashion, we can layer our plot with about thirty-fifty evenly-spaced knots, and not worry about the number of knots either. By introducing an appropriate λ, we will get a good fit of our spline to the data even with many knots. We are therefore most interested in how to automatically choose λ.

9.1 Finding the Optimal λ value

The residual sum of squares (RSS) is usually a good measure of the "goodness-of-fit" because this summation finds the overall amount of error between the regression curve and the actual data. As previously mentioned, residual sum of squares is defined as:

$$\text{RSS} = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2.$$

If we try to minimize RSS in order to get the best fit for our penalized spline, we run into an issue. With many knots, minimizing RSS results in no smoothing because the curve that minimizes RSS is the curve that is closest to each data point, i.e. over fitting the data by having each piecewise polynomial fit primarily the closest point. This is obviously problematic and means we need other measures and methods to evaluate the goodness of fit, such as cross-validation or Akaike's Information Criteria.

9.1.1 Cross-Validation

Cross-validation assesses the fit of the model with a particular λ in a very similar manner to RSS, but removes the y_i point and assesses how well the fit predicts that removed point.

In other words, it attempts to minimize RSS while ignoring the closest point, hence it is frequently referred to as the "leave-one-out" strategy. This is formally defined as

$$CV(\lambda) = \sum_{i=1}^{n} \{y_i - \hat{f}_{-1}(x_i;\lambda)\}^2 \tag{15}$$

where $\hat{f}_{-1}(x_i;\lambda)$ refers to the spline fit without the (x_i, y_i) point. In essence, this allows us to find the λ for a given spline basis that minimizes this value, while taking the prediction of new points into account and avoiding the danger of over-fitting. However, there is one large issue with this method. It is very computationally intense. For every iteration, a new spline fit has to be found, which takes a substantial amount of computing time for large datasets. This computation time can be significantly reduced by using an approximation, which generally holds. This approximation is

$$\hat{f}_{-1}(x_i;\lambda) = \frac{\displaystyle\sum_{i=1, j\neq i}^{n} S_{\lambda,ij} y_j}{\displaystyle\sum_{i=1, j\neq i}^{n} S_{\lambda,ij}} \quad [9],$$

where $\mathbf{S}_\lambda$ is the smoothing matrix of a penalized linear spline described previously ($\mathbf{S}_\lambda = \mathbf{X}(\mathbf{X}^T\mathbf{X} + \lambda^2\mathbf{D})^{-1}\mathbf{X}^T$). Although [9] did not explain when or why this approximation is true, it allows us to rewrite the cross-validation criteria as

$$\text{CV}(\lambda) = \sum_{i=1}^{n} \left(\frac{y_i - \hat{y}_i}{1 - S_{\lambda,ii}} \right)^2 .$$

This significantly reduces computation time because it uses the normal residual of the original fitted model and only needs the diagonal entries of the smoothing matrix. This eliminates about half of the steps in Equation 15.

9.1.2 Akaike's Information Criterion (AIC) and Corrected AIC ($AIC_{\mathbf{C}}$)

Two other methods that are commonly used are Akaike's information criterion (AIC) and corrected AIC (AIC$_{\text{C}}$) with the difference between the AIC and the AIC$_C$ is that the corrected AIC introduces additional terms to correct for sample size. These two criteria are defined as [9]:

$$\begin{aligned} \text{AIC}(\lambda) &\equiv \log[\text{RSS}(\lambda)] + 2\frac{\text{df}_{\text{fit}}(\lambda)}{n} \\ \text{AIC}_{\text{C}}(\lambda) &\equiv \log[\text{RSS}(\lambda)] + 2\frac{\text{df}_{\text{fit}}(\lambda) + 1}{n - \text{df}_{\text{fit}}(\lambda) - 2} \end{aligned}$$

9.2 Using Algorithm to Find Optimal λ

Although the previous formulae work in theory, they can be quite computationally intense when directly calculated. [9] discusses this issue and provides an algorithm and sample code to drastically reduce the computational cost of generalized cross-validation (GCV). Ruppert, Wand, and Carroll recommend using a Demmler-Reinsch Orthogonalization to compute our fitted values, $\hat{f}$, for each λ value [9]. This would allow us to compute several matrices that make the computation of the RSS much faster. This is accomplished using the identity

$$\text{RSS}(\lambda) = \mathbf{y}^T\mathbf{y} - 2\mathbf{y}^T\hat{f}_\lambda + \hat{f}_\lambda^T\hat{f}_\lambda.$$

The Demmler-Reinsch Orthogonalization is described below [9].

- Assumption

  1. $\hat{f}_\lambda$ is of the form $\hat{f}_\lambda = \mathbf{C}(\mathbf{C}^T\mathbf{C} + \lambda\mathbf{D})^{-1}\mathbf{C}^T\mathbf{y}$

- Process

  1. Obtain matrix $\mathbf{R}$ using Choleskvy decomposition: $\mathbf{C}^T\mathbf{C} = \mathbf{R}^T\mathbf{R}$.
  2. Use singular value decomposition on symmetric matrix $\mathbf{R}^{-T}\mathbf{D}\mathbf{R}^{-1}$ to obtain $\mathbf{U}\text{diag}(\mathbf{s})\mathbf{U}^T$.
  3. Compute $\mathbf{A}$ and $\mathbf{b}$ as follows: $\mathbf{A} \equiv \mathbf{C}\mathbf{R}^{-1}\mathbf{U}$ and $\mathbf{b} \equiv \mathbf{A}^T\mathbf{y}$
  4. The fitted values, $\hat{f}_\lambda$, are
  $$\hat{f}_\lambda = \mathbf{A}\left(\frac{\mathbf{b}}{\mathbf{1} + \lambda\mathbf{s}}\right),$$
  where $\mathbf{s}$ is from the singular value decomposition
  5. The degrees of freedom, $\text{df}_{\text{fit}}(\lambda)$, is
  $$\text{df}_{\text{fit}}(\lambda) = \mathbf{1}^T\left(\frac{\mathbf{1}}{\mathbf{1} + \lambda\mathbf{s}}\right)$$

Having briefly explained the rationale behind this technique, we modified the code from [9] to also compute Akaike's information criterion (AIC) and corrected Akaike's information criterion (AIC_C) in addition to GCV. This code can be seen in Appendix 13.1.

9.3 Comparison of fit to several different random datasets

Having written R code to find the optimal λ using GCV, AIC, and AIC_C, we were curious how the three λ values would differ. To explore this, we fit penalized splines to our previous three examples, the sine curve with normally distributed error and the two neuroscience datasets.

9.3.1 Sine curve with random error

A linear penalized spline with 40 knots was fit to the same portion of the sine curve as in Figure 11. To determine the optimal λ value, we ran the R code seen in Appendix 13.1 with a range of 50 - 200 for the value of λ.

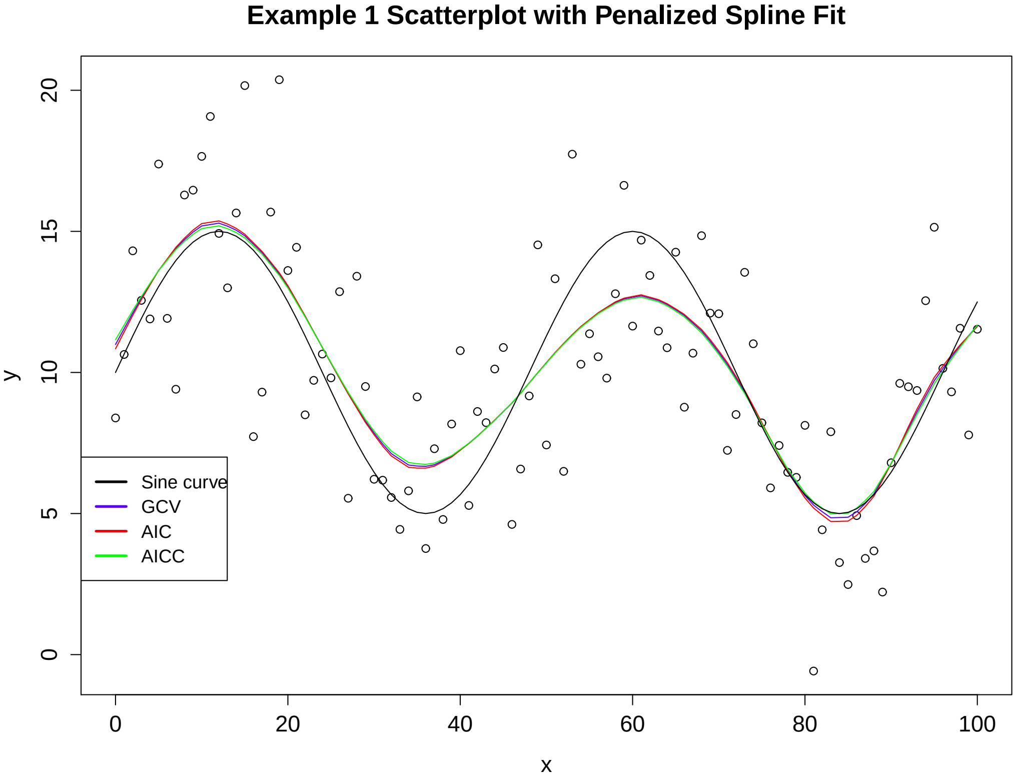

Figure 14: Penalized Spline fit to Example 1

As can be seen in Figure 14, the three fits are quite similar, but fit slightly different to the extrema of the sine curve. The GCV λ was 107.33, the AIC λ was 85.6, and the AIC_C λ was 134.6, reflecting the difference in these criterion's formulas. It is also interesting that all of the penalized splines do an excellent job at fitting the first and last extrema of the graph, but underestimate the interior extrema. Based upon examining the data and increasing the number of knots to 2000 (not shown), we concluded that the less weight at the interior extrema was due to the randomly distributed error itself and not due to the penalized spline. We see that there are more data points below the interior peak and above the interior trough, thus illustrating how an estimated curve can only be as good as the data. If the data does not reflect the true underlying pattern, then the estimated relationship will not reflect the underlying pattern either, demonstrating the value of gathering accurate and representative data. Despite this issue, we see that the penalized spline with 40 knots fits much better than the unpenalized fits considered previously.

9.3.2 Examining ΔH v. ΔG relationship

A linear penalized spline with 9 evenly spaced knots was fit to the same portion of the sine curve as in Figure 11. To determine the optimal λ value, we ran the R code seen in Appendix 13.1 with a range of 1000 - 10000 for the value of λ.

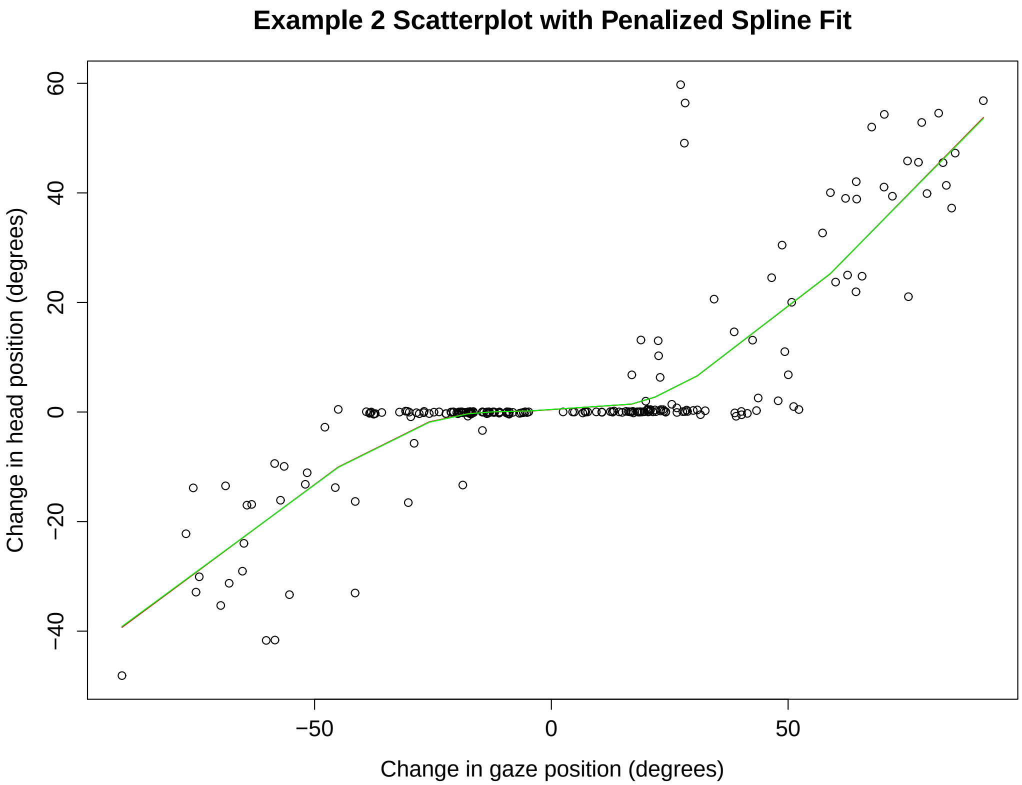

Figure 15: Penalized Spline fit to Example 2

Unlike in Example 1, the three fits here are almost indistinguishable. The GCV λ was 1842, the AIC λ was 1757, and the AIC$_C$ λ was 1930. We also see that the penalized fits fits the data much better than in Fig 12, despite only having 5 more knots and incorporating the penalized term.

9.3.3 Examining ΔT v. ΔG relationship

A linear penalized spline with 40 evenly spaced knots was fit to the same portion of the sine curve as in Figure 11. To determine the optimal λ value, we ran the R code seen in Appendix 13.1 with a range of 1000 - 10000 for the value of λ. The fits using the three λ values can be seen in Figure 16.

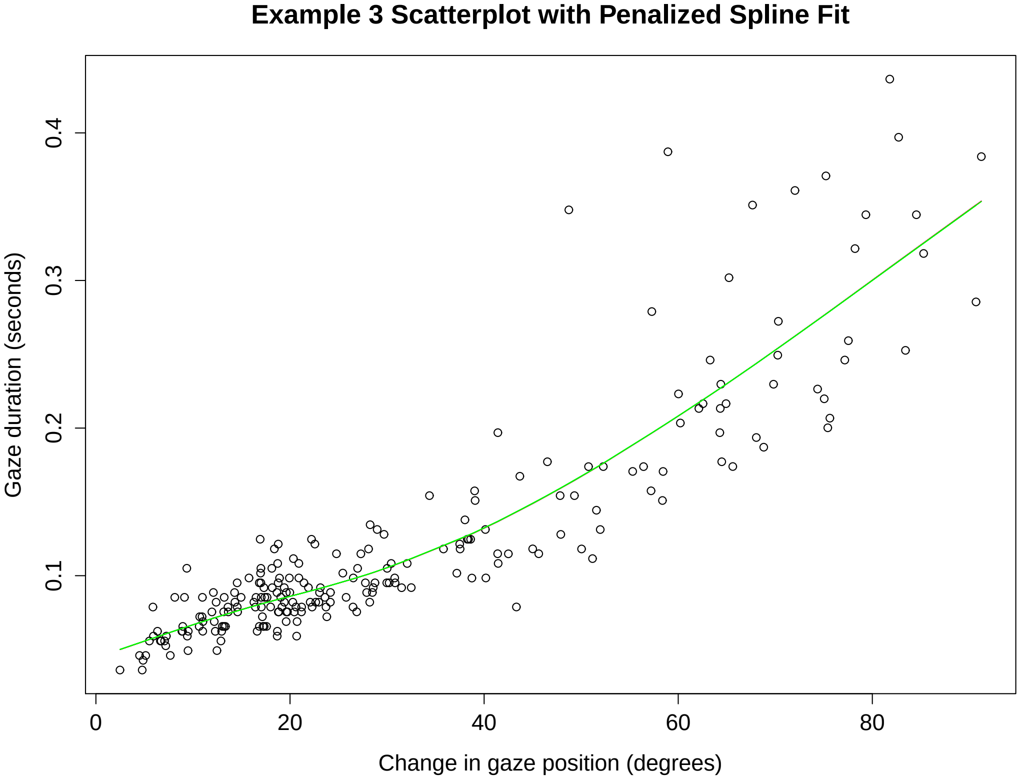

Figure 16: Penalized Spline fit to Example 3

As in Example 2, the three fits are indistinguishable. The GCV λ was 7543, the AIC λ was 7543, and the AIC$_\mathrm{C}$ λ was 7906. We also see that the penalized fits fits the data much better than in Fig 12, despite having many more knots. Looking at these graphs, we see that penalized spline has done an excellent job of fitting the graphs, and that there is no clear or persistent differences between AIC, AIC$_\mathrm{C}$, or GCV for these provided examples and that all three are excellent choices. All three optimized λ were computed rapidly (less than 5 seconds for entire R script to run) for less than 100 evenly-spaced knots and gave representative fits to the provided data.

10 Introduction to Bootstrapping and Randomization

Bootstrapping is a data-based statistical simulation method that resamples randomly from a given dataset to give the researcher hundreds or thousands of randomized replications to assess the likelihood of the actual results. For this type of method, we are looking at the relationship between two or more variables, such as whether aspirin prevents strokes, or any of the regression examples previously examined. In the aspirin example as discussed in [2], a longitudinal study was conducted that tracked whether people that consumed aspirin had more strokes compared to a placebo group. To conduct the bootstrap analysis, a dataset for each of the placebo and aspirin groups was created. In the two datasets, a 1 indicates that the patient had a stroke and a 0 indicates that the patient had no stroke during the study. For the bootstrap, the researchers drew with replacement a sample of n items from the first dataset and m items from the second dataset where n and m are the number of people in the aspirin and placebo groups respectively. By doing each of these random samples, or bootstrap replications, the researchers were able to quantitatively estimate the variability or standard error of the two original samples. Using these standard errors, the researchers could determine confidence intervals for the mean or median and assess whether aspirin may cause strokes in a larger population. As this example illustrates, bootstrapping assumes that the sample is a representative sample of the population and that the data points are all identically and independently distributed. By resampling with replacement, we are assuming that the samples are representative of the whole and that by resampling we are in effect taking a new sample from the population. Another method to assess the likelihood of results is to use randomization. This is quite similar to bootstrap methods, except that instead of sampling with replacement, the methods require sampling without replacement. Because this will generate the same data, just in a different order, the method is primarily useful for quantifying the likelihood of observed differences in datasets. Going back to the aspirin example, a randomization study would combine all of the subjects into one large pool and then draw without replacement to generate the two simulated aspirin and placebo group. This thus randomly distributes the total number of strokes between the two groups, from which the researchers can conclude how likely it is to have the observed difference between the two groups and assess whether aspirin is correlated with increasing or reducing the risk of a stroke as compared to the placebo group.

10.1 Testing Significance of Spline Coefficients using Randomization

In a similar manner to the aspirin example, we are interested in assessing the likelihood of obtaining the the spline coefficients that we obtained from our initial penalized linear spline fit. To accomplish this randomization, we first fit and optimize the penalized spline to our datasets of interest. The penalized spline fit was optimized by using Akaike's Information Criteria Corrected (AIC_C) to find the optimal λ value. To utilize randomization, we sampled without replacement the observed y-values and reassigned them to the observed x-values, i.e. we randomly reordered the y-values. Once these new y-values were assigned, we generated the spline coefficients using the previously found λ. This randomization process was repeated for $10,000$ samples. To assess the likelihood of the observed coefficients, we found the number of samples with the designated spline coefficient greater than or equal to the observed spline coefficient. This allowed us to assess the statistical likelihood of obtaining each of the observed coefficients. This methodology was applied to two datasets previously examined: the dataset comparing size of gaze shift to time and the dataset comparing change in gaze position to change in head position. As previously described these two datasets are both poorly fit by simple linear regression and benefited from nonparametric regression. We will first examine the dataset comparing change in gaze position to change in head position

10.1.1 Neuroscience Example 1

We are interested in whether β_1 is statistically significant. To determine this, we found the number of randomizations in which the β_1 value was more extreme than the observed value.

This allows us to obtain the probability of both tails and estimate the likelihood of having the observed β_1 coefficient. The histogram of the randomized β_1 coefficients is seen in Figure 17. Our observed value from the initial optimized fit was 0.636. In our randomization study, 0% of the values was more extreme than our observed value, indicating that our intercept is statistically significant.

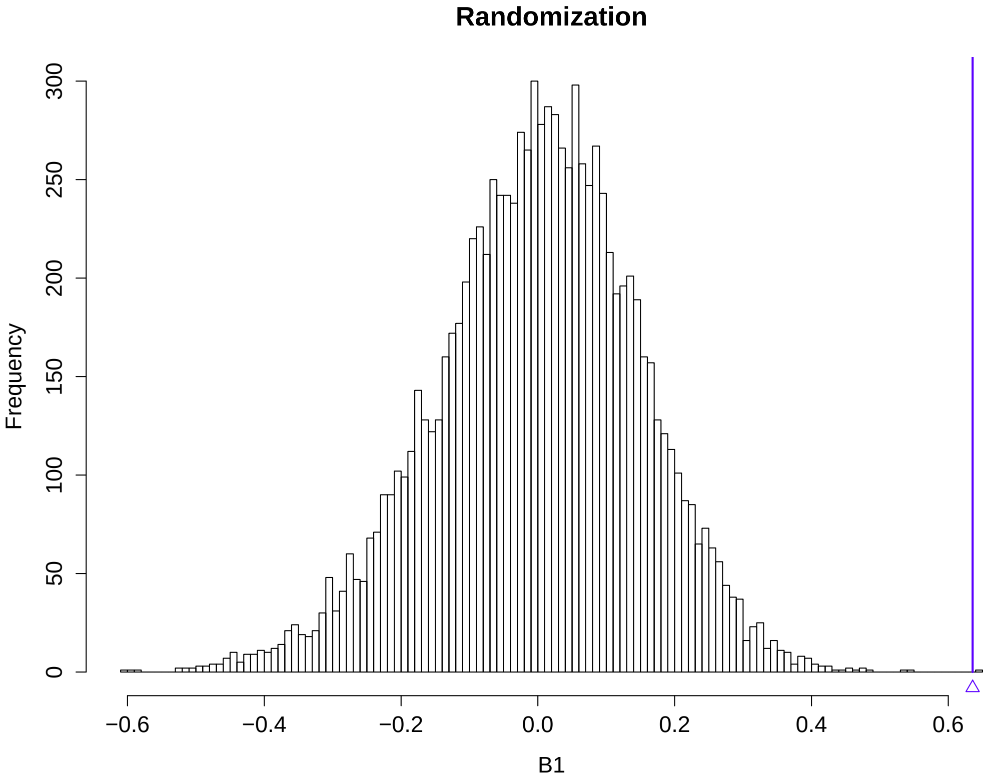

Figure 17: The distribution of randomized β_1 coefficients

10.1.2 Neuroscience Example 2

As you will recall upon examining Figure 2, there appears to be a strong correlation between increasing size of a gaze shift and the time it takes to complete the shift. We showed that this could be well fit with a second degree polynomial, but we were interested in fitting it using our nonparametric methods instead of the previously applied parametric models. After fitting and optimizing our model, we did a similar randomization study to generate our randomized coefficients. To assess the statistical significance of the y-intercept, we examined the likelihood of obtaining a β_1 coefficient equal to or more extreme than the observed value of 0.0022. Because we are examining the relationship between time to complete a shift and size of the shift, we would expect a positive slope, so it seems quite reasonable that our β_1 is greater than 0. By looking at Figure 18, we saw that 10.8% of the randomized values are more extreme than our obtained β_1 value, indicating that it is likely that we would get a slope of 0.0022 by random chance.

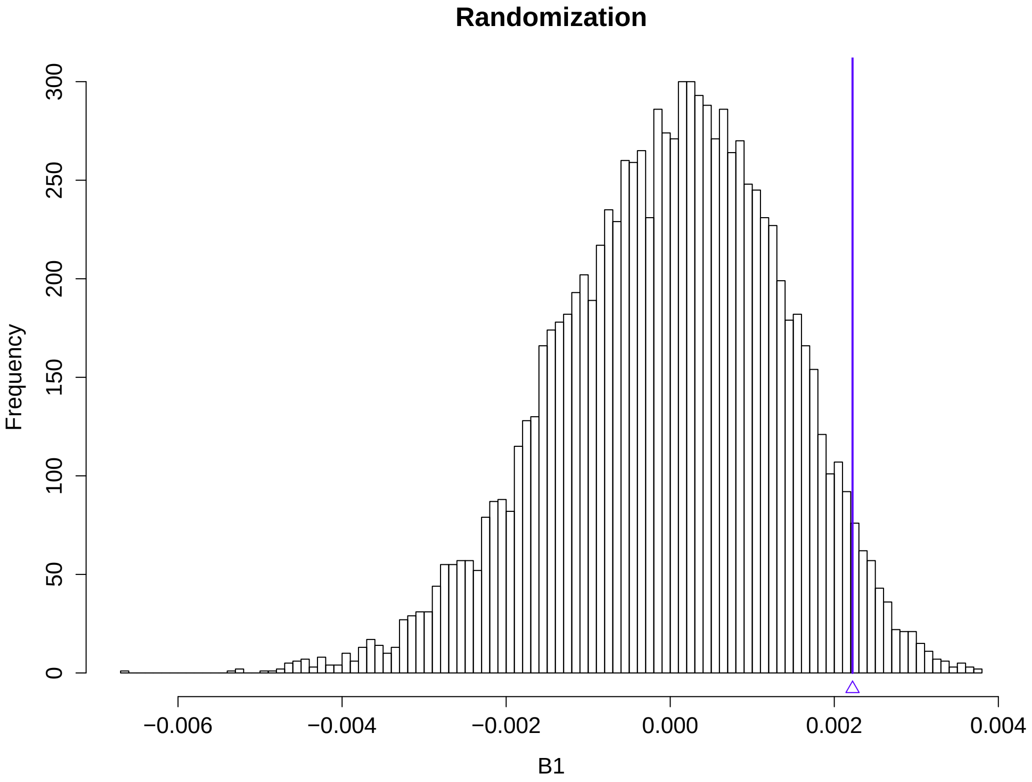

Figure 18: The distribution of randomized β_1 coefficients

10.1.3 Discussion of results

Although we successfully completed a randomization to assess the likelihood of the observed coefficients, it is possible that this is not an accurate method to assess the likelihood of the coefficients by random chance. As several papers have indicated, there are other methods that would reduce the amount of bias in my answers, such as using a method called a wild bootstrap, or using a bootstrap upon the residuals instead of the observed y-values.

10.2 Improvement to Prior Randomization Study

Following the ideas of [2], the residuals, or observed differences between the true y-values and the fitted values, can be randomized instead. This will maintain the overall relationship between the variables, but assumes that the errors in the model are independently and identically distributed. Although this was an assumption that we have tried to avoid through our nonparametric regression fitting, it is a reasonable assumption to be made at this point. Our penalized spline is optimized to fit the data as closely as possible and it is a reasonable assumption that the errors will be fairly constant throughout the model's range. Under this assumption, we can resample from our residuals using a bootstrap method (sample with replacement) or a randomization method (sample without replacement). By reassigning the residuals to the fitted values, we generate a new, very similar dataset that can be refitted and then used to assess the likelihood of the original coefficients. Below we present the same two examples from the previous randomization study, using both the boostrap and randomization approaches.

10.2.1 Neuroscience Example 1

Same as in the randomization of the y-values, $10,000$ resamplings were completed for both the bootstrap and the randomization. The β_1 histograms for both can be seen in Figure 19.

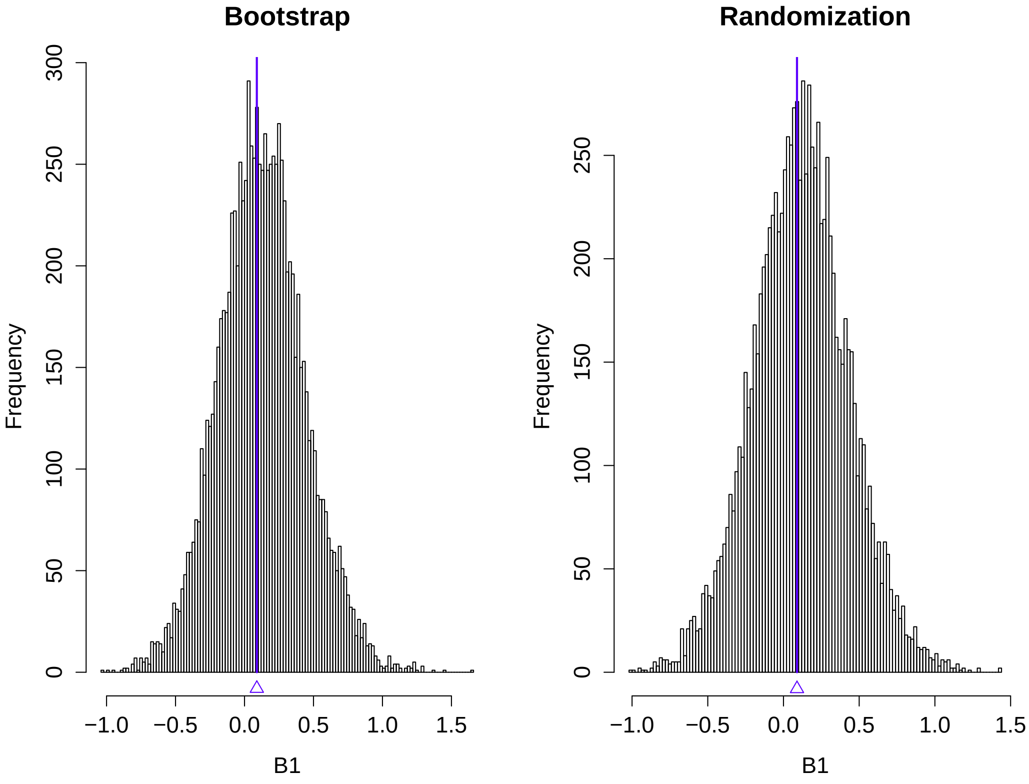

Figure 19: Histograms of the residual bootstrap and randomization. Blue line indicates where the original coefficient is located

As should be evident from this figure, the two methods appear to be quite similar. There might be slightly more weight in the tails of the bootstrap, but it is difficult to determine. Examining these histograms also revealed a possible method to construct confidence intervals for our coefficients. By finding 95% of the values centered about the mean for the coefficients, it should be possible to determine the likely 95% confidence interval for β_1. The validity of this approach is unknown though and would need further investigation. Because the observed coefficients (β_1 can be seen in the histogram) fall very close to the mean, it would appear to indicate that there is some “wiggle” room in the coefficients and that if the observed observations have slightly different errors, then the coefficients would be slightly different, but still quite similar.

10.2.2 Neuroscience Example 2

As in Example 1, the two methods are compared in Figure 20.

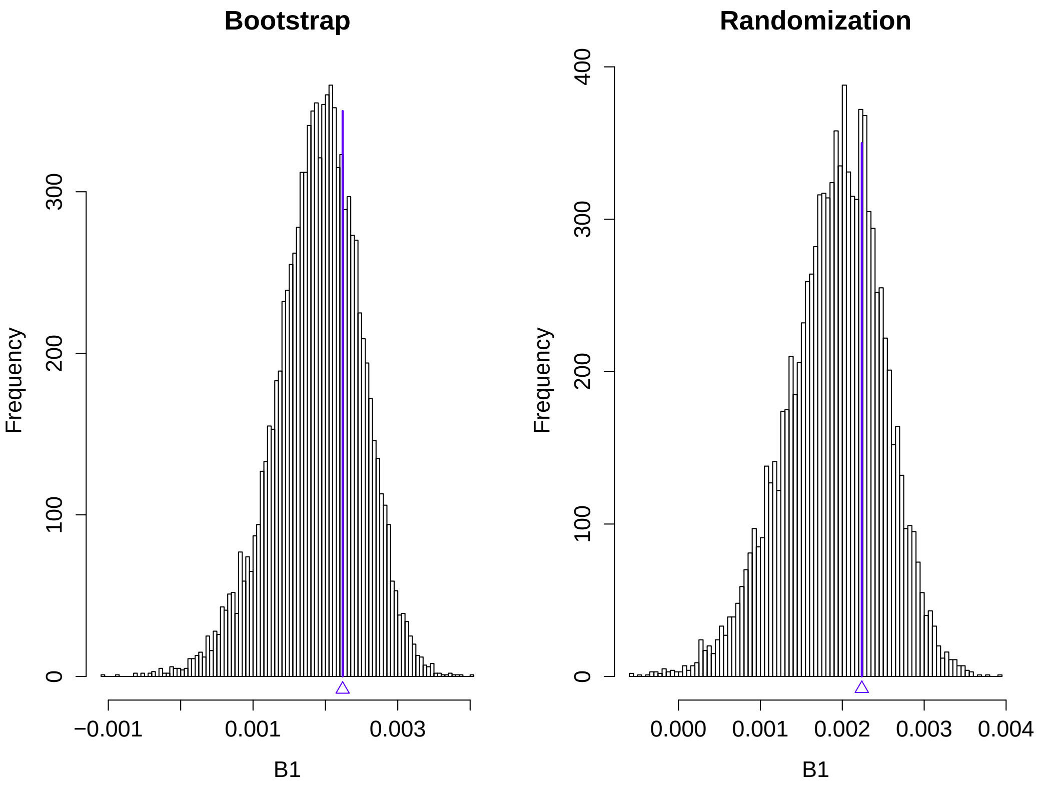

Figure 20: Histograms of the residual bootstrap and randomization. Blue line indicates where the original coefficient is located

The first thing of note is that the distribution is not centered around 0. This supports our conclusion that a slope greater than 0 is probable. A slope of 0 is contained with the approximate 95% confidence interval though, so more research is needed to confirm this is a valid method for assessing the likelihood of the fitted global coefficients

10.2.3 Discussion of results

Although this is a different approach, we are not sure it really addressed any of our previous issues. We were concerned with testing the significance of our coefficients, but the approach of resampling from the residuals does not appear to allow this type of hypothesis test, because the correlation between variables is largely maintained. This new method does allow for confidence intervals to be built for our coefficients though. Previously we mentioned exploring the significance of the spline coefficients themselves (the u_k values for each piecewise linear spline portion), but we realized that the analysis of these coefficients is meaningless from a scientific perspective. The coefficients are chosen to get the best fit and do not reflect any intrinsic relationship between the data. With 9 knots, there is no significance to the obtained values, so hypothesis testing or confidence intervals for these coefficients does not reveal any important information about the relationship between the variables. However, with a different model formulation, it would be possible to test for significance. One such example is in Example 1. If the model did not include the β_0 or β_1 coefficients and had 4 precisely located knots, it would be possible to fit a continuous piecewise linear function to the graph, and reveal information about the two lobes with head contribution and the central region with little to no head contribution. A similar approach using kernel regression in place of penalized spline regression has been used by Dr. John Stahl, a neuroscientist examining the relationship between gaze size and amount of head contribution [10].

11 Conclusion

Over the course of the semester and this research paper, we successfully explored, explained, and applied simple linear regression and penalized spline regression and its extensions, parametric and nonparametric methods respectively. We also began an exploration of inference using randomization and bootstrapping of our y-values paired with penalized spline regression. We successfully examined the relationship between gaze duration and gaze amplitude and obtained results that are quite similar scientific articles from the neuroscience community. Using the penalized spline model, we also characterized the relationship between change in head position (ΔH) and change in gaze position (ΔG), which was examined in the author's second thesis for honors in Biochemistry, Biophysics, and Molecular Biology [4].

12 Acknowledgements

The author would like to acknowledge Dr. Kelly McConville, his thesis advisor, for all of the valuable comments and help as he completed this research project. He would also like to thank Jeremy Batterson, Emily Wooley, and Dr. Balof for reading portions of this paper every week and pointing out the gaps that needed to be filled.

13 Appendix

13.1 Automatic Scatterplot Smoothing R Code

The following script shows this algorithm and also graphically compares the λ obtained from the three methods.

```
##Generates the data I want to fit
x <- seq(1,100,.5) ##Generate our sequence of x-values, i.e 1, 1.5, 2, ..., 100
mu <- m.3(x) ##Generates the basic underlying function from an external function
set.seed(2010) ##Generate the seed number for the random number generator
eee <- rnorm( length(mu) ) ##Generate the random normally distributed error
y <- mu +eee ##Combine the basic function with the error to generate our y-values
##----------Generates the matrices needed to cheaply compute AIC, GCV, and AICC
num.knots <- max(5,min(floor(length(unique(x))/4),35)) ## Number of knots
```

```
knots <- quantile(unique(x),seq(0,1,length=(num.knots+2))[-c(1,(num.knots+2))])
 ##Evenly space the knots across the x-values
lambda.low <- .0001 ##Set our lower bound for lambda
lambda.upp <- 10 ##Set our upper bound for lambda
num.lambda <- 50 ##The number of lambda values to check
lambda.vec <- 10^(seq(log10(lambda.low),log10(lambda.upp),length=num.lambda))
 ##Space the intermediate lambda values logarithmically.
n <- length(x) ## Find number of datapoints in x
X <-cbind(rep(1,n),x)
## Generate our X matrix with column 1 being degree 0, column 2 being degree 1
Z <-outer(x,knots,"-") ##See Comment 1 below for explanation of this function
Z <-Z*(Z>0)
##Define each knot, such that they are 0 below ki where ki is position of ith knot
C.mat <-cbind(X,Z) ##Combine x-values with knots to generate basis matrix
CTC <- t(C.mat) %*%C.mat ##Generate positive definite matrix from basis
D.mat <- diag(c(rep(0,ncol(X)),rep(1,ncol(Z))))
##Generate the D matrix that is used to weight each spline
R.mat <- chol(CTC) ## Completes a Cholesky decomposition
svd.out <- svd(t(solve(t(R.mat),t(solve(t(R.mat),D.mat)))))
##Computes the singular value decomposition of
s.vec <- svd.out$d ##Grabs specific column
U.mat <-svd.out$u ##Grabs specific matrix
A.mat <- C.mat %*%backsolve(R.mat,U.mat) ##Generates A matrix
b.vec <- as.vector(t(A.mat)%*%y) ##Generates b matrix
r.mat <- 1/(1+outer(s.vec,lambda.vec)) ##Generates r matrix
f.hats <-A.mat%*%(b.vec*r.mat)
   ##Each column contains the fitted values for each choice of lambda
y.vecs <- matrix(rep(y,num.lambda),n,num.lambda) ##Creates n columns of the y vector.
RSS.vec <- apply((y.vecs-f.hats)^2,2,sum) ##Generates RSS for each column
df.vec <- apply(r.mat,2,sum) ##Generates degrees of freedom for each column
##----------------------Compute Generalized cross-validation
GCV.vec <- RSS.vec/((1-df.vec/n)^2)  ##Computation of GCV for each column
ind.min <- order(GCV.vec)[1] ##Find the smallest GCV
if (ind.min==1) ##Confirms that our smallest GCV is not an endpoint
  stop("Hit left boundary; make lambda.low smaller.")
if (ind.min==num.lambda) ##Confirms that our smallest GCV is not an endpoint
  stop("Hit right boundary; make lambda.upp bigger.")
lambda.GCV <- lambda.vec[ind.min] ##Records the smallest lambda into another variable
##----------------------Compute AIC
AIC.vec <-log(RSS.vec, base=exp(1))+2*(df.vec)/n ##Computation of AIC for each column
ind.min2 <- order(AIC.vec)[1] ##Find the smallest AIC
if (ind.min2==1) ##Confirms that our smallest AIC is not an endpoint
  stop("Hit left boundary; make lambda.low smaller.")
if (ind.min2==num.lambda) ##Confirms that our smallest AIC is not an endpoint
  stop("Hit right boundary; make lambda.upp bigger.")
```

```
lambda.AIC <- lambda.vec[ind.min2] #Records the smallest lambda into another variable
##----------------------Compute AICC
AICC.vec <-log(RSS.vec, base=exp(1))+2*(((df.vec)+1)/(n-df.vec-2))
  ##Computation of AICC for each column
ind.min3 <- order(AICC.vec)[1]
if (ind.min3==1) ##Confirms that our smallest AICC is not an endpoint
  stop("Hit left boundary; make lambda.low smaller.")
if (ind.min3==num.lambda) ##Confirms that our smallest AICC is not an endpoint
  stop("Hit right boundary; make lambda.upp bigger.")
lambda.AICC <- lambda.vec[ind.min3] #Records the smallest lambda into another variable
##-------Compare the choices of lambda
plot(x,y) ##Plot the initial data that needs to be fit
lines(x,f.hats[,ind.min],col="blue") ##Plot the GCV line
lines(x,f.hats[,ind.min2],col="red") ##Plot the AIC line
lines(x,f.hats[,ind.min3],col="green") ##Plot the AICC line
```

Comments

1. outer(x,y,'function') computes the outer product of x and y using the function specified. This can agree with the linear algebra definition with outer product if the function specified is multiplication, but it can be other functions as well. This may seem confusing at first, so we have provided two example uses of this function. The first uses the standard definition of outer products to generate the 12s times table. The second uses subtraction as the function to generate a smaller, but similar matrix as in the code above.

   (a) **Input:**

   ```
   outer(c(1,12),c(1,12),'*')
   ```

   Output:

   ```
       1  2  3  4  5  6  7  8   9  10  11  12
   1   1  2  3  4  5  6  7  8   9  10  11  12
   2   2  4  6  8 10 12 14 16  18  20  22  24
   3   3  6  9 12 15 18 21 24  27  30  33  36
   4   4  8 12 16 20 24 28 32  36  40  44  48
   5   5 10 15 20 25 30 35 40  45  50  55  60
   6   6 12 18 24 30 36 42 48  54  60  66  72
   7   7 14 21 28 35 42 49 56  63  70  77  84
   8   8 16 24 32 40 48 56 64  72  80  88  96
   9   9 18 27 36 45 54 63 72  81  90  99 108
   10 10 20 30 40 50 60 70 80  90 100 110 120
   11 11 22 33 44 55 66 77 88  99 110 121 132
   12 12 24 36 48 60 72 84 96 108 120 132 144
   ```

   Explanation: Every row and column heading is multiplied together to generate the 12 X 12 multiplication table that is used throughout elementary school. This agrees with our understanding of outer products from linear algebra as well.

(b) **Input:**

```
x<-1:10
knots<-seq(2,8,2)
Z <-outer(x,knots,"-")
```

Output:

```
     2  4  6  8
1   -1 -3 -5 -7
2    0 -2 -4 -6
3    1 -1 -3 -5
4    2  0 -2 -4
5    3  1 -1 -3
6    4  2  0 -2
7    5  3  1 -1
8    6  4  2  0
9    7  5  3  1
10   8  6  4  2
```

Explanation: Looking at this example, we see that each column heading is subtracted from the x-value for that row, generating an unusual subtraction table. Using this matrix, we can easily calculate our spline basis, where each spline should be 0 below the starting knot of that spline.

2. Overall this script is quite easy to implement and use. We have not calculated the actual computational load, but for our purposes (less than 100 knots), it runs almost instantaneously. It will also be easy to convert into a function to allow for easier input of λ ranges.

3. Because the initial cross-validation has still not been fully debugged, this code provides a framework to develop a simpler, and probably faster, method to compute CV.

4. This works for a linear spline basis, but still needs to be expanded to handle quadratic or cubic splines, both of which have the desirable property of having a continuous first derivative at each knot.

13.2 Randomization and Bootstrapping Code

```
##Generates the data I want to fit
WGdata2=read.csv(file="~/Dropbox/Senior Project/Kernel Density Estimation/WG_Ndata.csv",
WGdata2.sort <- WGdata2[order(WGdata2$dG),]; ##Sort the dataset as needed for spline fit
x <- WGdata2.sort$dG ##Generate our sequence of x-values, i.e 1, 1.5, 2, ..., 100
y <- WGdata2.sort$dH ##
##----------Generates the matrices needed to cheaply compute AIC, GCV, and AICC
##num.knots <- max(5,min(floor(length(unique(x))/4),50)) ## Number of knots
num.knots<-9
knots <- quantile(unique(x),seq(0,1,length=(num.knots+2))[-c(1,(num.knots+2))])
```

```
##Evenly space the knots across the x-values
lambda.low <- 1000 ##Set our lower bound for lambda
lambda.upp <- 10000 ##Set our upper bound for lambda
num.lambda <- 50 ##The number of lambda values to check
lambda.vec <- 10^(seq(log10(lambda.low),log10(lambda.upp),length=num.lambda))
##Space the intermediate lambda values logarithmically.
n <- length(x) ## Find number of datapoints in x
X <-cbind(rep(1,n),x)
## Generate our X matrix with column 1 being degree 0, column 2 being degree 1
Z <-outer(x,knots,"-") ##See Comment 1 below for explanation of this function
Z <-Z*(Z>0)
##Define each knot, such that they are 0 below ki where ki is position of ith knot
C.mat <-cbind(X,Z) ##Combine x-values with knots to generate basis matrix
CTC <- t(C.mat) %*%C.mat ##Generate positive definite matrix from basis
D.mat <- diag(c(rep(0,ncol(X)),rep(1,ncol(Z))))
##Generate the D matrix that is used to weight each spline
R.mat <- chol(CTC) ## Completes a Cholesky decomposition
svd.out <- svd(t(solve(t(R.mat),t(solve(t(R.mat),D.mat)))))
##Computes the singular value decomposition of
s.vec <- svd.out$d ##Grabs specific column
U.mat <-svd.out$u ##Grabs specific matrix
A.mat <- C.mat %*%backsolve(R.mat,U.mat) ##Generates A matrix
b.vec <- as.vector(t(A.mat)%*%y) ##Generates b matrix
r.mat <- 1/(1+outer(s.vec,lambda.vec)) ##Generates r matrix
f.hats <-A.mat%*%(b.vec*r.mat) ##Each column contains the fitted values for each choice
y.vecs <- matrix(rep(y,num.lambda),n,num.lambda) ##
RSS.vec <- apply((y.vecs-f.hats)^2,2,sum) ##Generates RSS for each column
df.vec <- apply(r.mat,2,sum) ##Generates degrees of freedom for each column
##----------------------Compute Generalized cross-validation, AIC, and AICC
GCV.vec <- RSS.vec/((1-df.vec/n)^2)  ##Computation of GCV for each column
ind.min <- order(GCV.vec)[1] ##Find the smallest GCV
if (ind.min==1) ##Confirms that our smallest GCV is not an endpoint
  stop("Hit left boundary; make lambda.low smaller.")
if (ind.min==num.lambda) ##Confirms that our smallest GCV is not an endpoint
  stop("Hit right boundary; make lambda.upp bigger.")
lambda.GCV <- lambda.vec[ind.min] ##Records the smallest lambda into another variable
plot(x,y,main="Example 1 Scatterplot with Penalized Spline Fit",xlab="Change in gaze pos
lines(x,f.hats[,ind.min],col="blue")
AIC.vec <-log(RSS.vec, base=exp(1))+2*(df.vec)/n
ind.min2 <- order(AIC.vec)[1]
if (ind.min2==1)
  stop("Hit left boundary; make lambda.low smaller.")
if (ind.min2==num.lambda)
  stop("Hit right boundary; make lambda.upp bigger.")
lambda.AIC <- lambda.vec[ind.min2]
```

```
lines(x,f.hats[,ind.min2],col="red")
AICC.vec <-log(RSS.vec, base=exp(1))+2*(((df.vec)+1)/(n-df.vec-2))
ind.min3 <- order(AICC.vec)[1]
if (ind.min3==1)
  stop("Hit left boundary; make lambda.low smaller.")
if (ind.min3==num.lambda)
  stop("Hit right boundary; make lambda.upp bigger.")
lambda.AICC <- lambda.vec[ind.min3]
lines(x,f.hats[,ind.min3],col="green")
##--------------------Randomization code to assess likelihood of coefficients
trueB<-solve((t(C.mat)%*%C.mat+lambda.AICC*D.mat))%*%t(C.mat)%*%y
##Finds observed coefficients to be compared
B.mat<-matrix(rep(0,110000),nc=11)
for(i in 1:10000){y.star<-sample(y);
                 B.mat[i,] <- solve((t(C.mat)%*%C.mat+lambda.AICC*D.mat))%*%t(C.mat)%*%y
                 ##Generates beta matrix for each randomization
}
sum(B.mat[,1]>=abs(trueB[1])) ##Finds likelihood of obtaining our results for beta0
sum(B.mat[,1]<=-abs(trueB[1]))
sum(B.mat[,2]>=abs(trueB[2])) ##Finds likelihood of obtaining our results for beta1
sum(B.mat[,2]<=-abs(trueB[2]))
```

14 References

References

[1] S. Chatterjee, A.S. Hadi,B. Price, *Regression Analysis By Example*,3rd ed., Wiley, New York, (2000).

[2] B. Efron, R.J. Tibshirani. *An Introduction to the Bootstrap*, Chapman and Hall, New York (1998).

[3] P.H.C. Eilers, B.D. Marx, “Flexible Smoothing with B-splines and Penalties”, *Statistical Science*, 1996, p. 89-121.

[4] W. Griggs, “Eliminating Intersubject Variability of Large Amplitude Gaze Metrics by Reducing the Visual Field”, Biochemistry, Biophysics, and Molecular Biology Honors Thesis, Penrose Library, Whitman College.

[5] N. Henry, “R-square and Standardization in Regression”, Virginia Commonwealth University, Richmond, VA, (2001). http://www.people.vcu.edu/ nhenry/Rsq.htm. Accessed April 22, 2013.

[6] J.J. Higgins, *Introduction to Modern Nonparametric Statistics*, Duxbury Press, (2004) 1-2 and 323-329.

[7] R.V. Hogg, E.A. Tanis, *Probability and Statistical Inference*, 8$^{\text{th}}$ ed., Pearson, Upper Saddle River, NJ, (2010).

[8] D.C. Montgomery, E. Peck, G.G. Vining, *Introduction to Linear Regression Analysis*, 3$^{\text{rd}}$ ed.,Wiley, New York, (2001).

[9] D. Ruppert, M.P. Wand, and R.J. Carroll. *Semiparametric Regression*, Cambridge University Press, New York (2003) p. 1-90.

[10] J.S. Stahl. Amplitude of human head movements associated with horizontal saccades. Experimental Brain Research 1999; 126(1): 41-54.

[11] W. Zucchini, "Applied Smoothing Techniques - Part 1: Kernel Density Estimation", Temple University , Philadephia, PA, (2003). http://isc.temple.edu/economics/Econ616/Kernel/ast_part1.pdf Accessed January 29, 2013.