Spline regression - basic theory

PPS

27/01/2021

NB1: Almost all material below is copied directly from the spline-section of Whitney Griggs' thesis-project on regression splines (Whitman Collage, 2013): https://www.whitman.edu/Documents/Academics/Mathematics/Griggs.pdf

I have correct some typos in equations, removed text and examples and added some explanatory stuff relevant for my understanding.

NB2: After having read this tutorial, I recommend reading Tristan Mahrs excellent bloggpost explaining penalized splines and their connection to random effects in mixed-effect models: https://www.tjmahr.com/random-effects-penalized-splines-same-thing/

//Pontus PS, 2021, NRU, Copenhagen.

Linear regression

$$\hat{Y} = \begin{bmatrix} \hat{y_1} \\ \hat{y_2} \\ \vdots \\ \hat{y_n} \end{bmatrix}, X = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}, \mathbf{b} = \begin{bmatrix} \hat{\beta}_1 \\ \hat{\beta}_2 \end{bmatrix}$$

where $\mathbf{b} = (X^T X)^{-1} X^T Y$.

$\hat{Y}$ can be expressed as 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 the function only have linear solutions.

If data is non-linear we can add polynomials (e.g., x^2 and x^3 and their associated **b**-parameters) to better fit the data. But that is not always practical for reasons not to be to be disclosed to mortal souls. Instead we can move on to *spline regression*, where the regression model is a piece-wise 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 piece-wise 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.

Linear splines

The simplest form of a polynomial function is the 1-degree one, which is... you know... linear. Let's start with that one.

A linear spline is defined as $f(x) = \beta_0 + \beta_1 x + \sum_{k=1}^{K} b_k (x - \kappa_k)_+$.

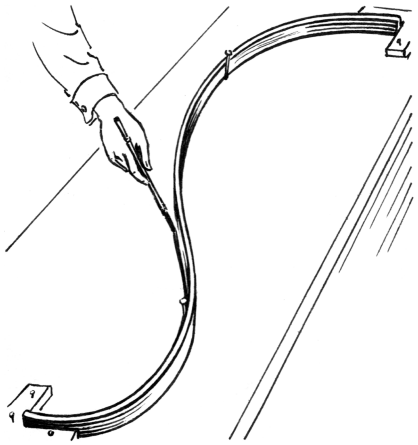

Figure 1: Traditional spline used by shipwrights

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

$$(x - \kappa_i)_+ = \begin{cases} x - \kappa_i : if\ x - \kappa_i > 0 \\ 0 : if\ x - \kappa_i < 0 \end{cases}$$

We see that the basis of our spline model becomes:

$$B = \begin{bmatrix} 1 & \mathbf{x} & (\mathbf{x} - \kappa_1)_+ & (\mathbf{x} - \kappa_2)_+ & ... & (\mathbf{x} - \kappa_K)_+ \end{bmatrix}$$

This allows for a wide variety of shapes to be fit. Looking at Figure 1 below, we seen an example of a 'whip' model and the corresponding linear spline basis. 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.

Every value of f in the top panel is defined as the (beta and b-weighted) sum of this/these basis(es?). For the first values of x up until 0.5, the knot basis(es?) are zero, meaning that data is only fitted by the intercept (β_0) and the slope β_1 in this part.

By defining 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 overfit the data, picking up random fluctuations.

We first begin with a randomly generated dataset to demonstrate the versatility of spline regression models:

Knots are placed uniformly from 0 to 80. 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. This translates to a function with 6 basis(es?).

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.

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.

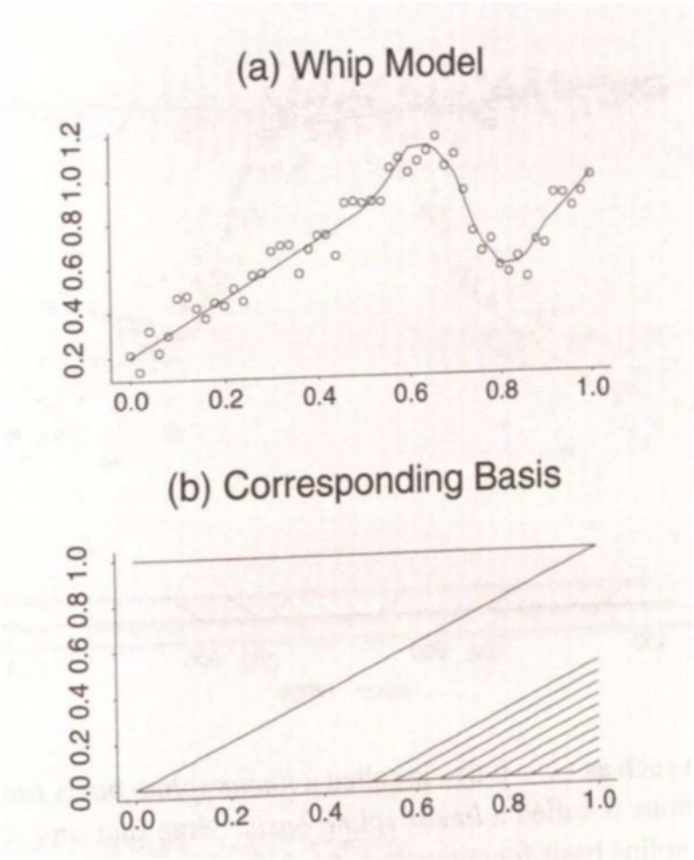

Figure 2: Wip model.

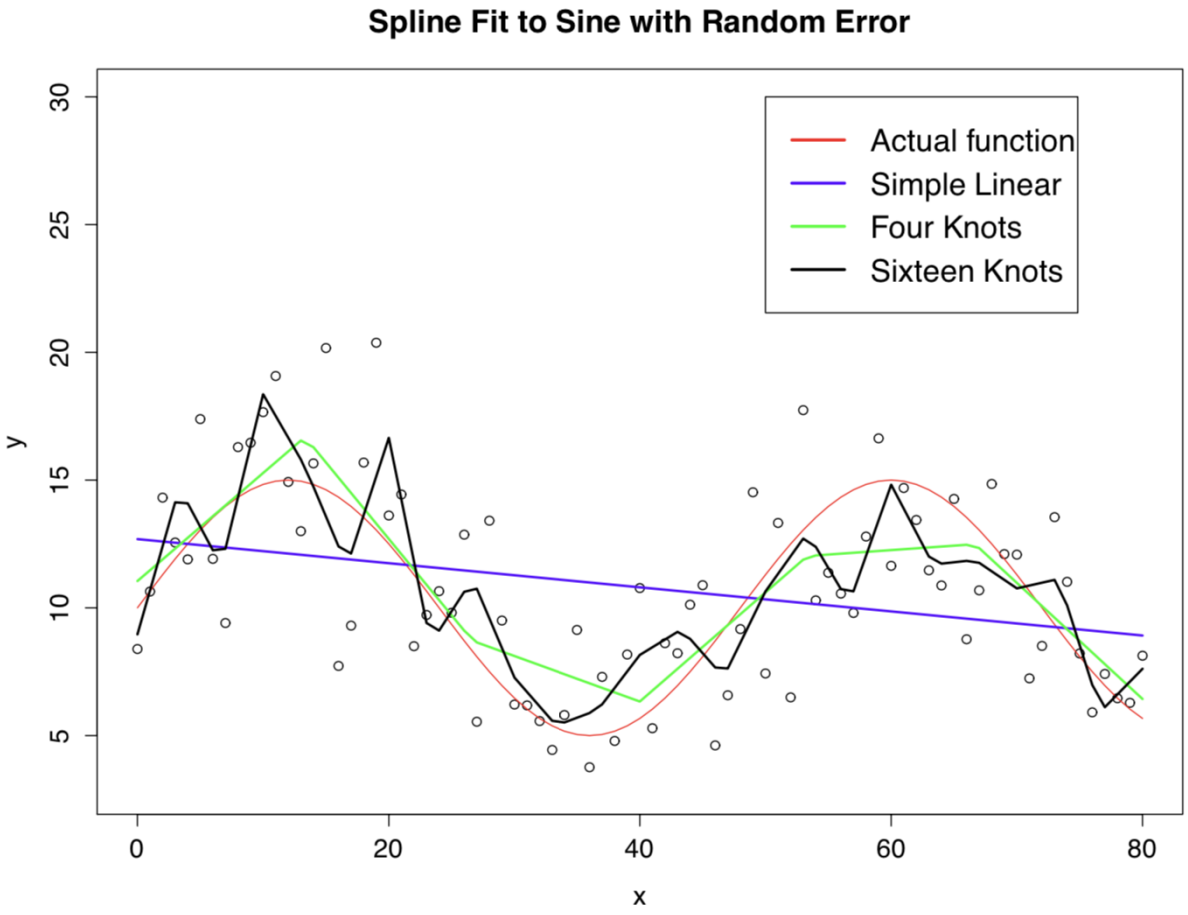

Figure 3: Simulated sine with random noise example

To avoid the overfitting 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.

Penalized splines

As the name implies, a penalized spline imposes a penalization upon the piece-wise 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.

Put a constrain on parameters in $\mathbf{b}$ to find optimize the fit and avoid both under and overfitting. There are several options for the penalization criteria, but the easiest to implement is to choose a C such that:

$$\sum_{k=1}^{K} b_k^2 < C$$

This is a good criteria, because it reduces the overall effect of individual piece-wise functions and avoids over-fitting the data. Formally, we then want to minimize the equation $(Y - X\mathbf{b})^2$ subject to $\mathbf{b}^T D\mathbf{b} \leq C$, where

$$D = \begin{bmatrix} 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{0}_{2x2} & \mathbf{0}_{2xK} \\ \mathbf{0}_{Kx2} & \mathbf{I}_{KxK} \end{bmatrix}$$

Where the first two rows and columns correspond to the first to basis(es?) which in turn correspond to β_0 and β_1. Basically, D just mean that we don't constrain the intercept and the slope before the first knot, but we do penalize the remaining regression weights.

Using Lagrange multipliers (<- Yeah, come on. I don't even know how to pronounce this) , this is equivalent to minimizing

$$(Y - X\mathbf{b})^2 + \lambda \mathbf{b}^T D\mathbf{b}$$

for some $\lambda \geq 0$.

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

To find the solution to the minimization above, we need to find when all the partial derivatives with respect to the elements of $\mathbf{b}$ are 0. After some proofs and reshuffling that I will not show here we can show that:

$$\mathbf{b} = (X^T X + \lambda^2 D)^{-1} X^T Y$$

Which looks pretty darn elegant.

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. This is where automatic λ selection becomes important and desirable.

Cubic splines

In order to get smoother fits at the junction of the knots we can use higher order polynomials in our spline regression. One common approach is to use cubic splines:

$$f(x) = \beta_0 + \beta_1 x + \beta_3 x^2 + \beta_4 x^3 + \sum_{k=1}^{K} b_k (x - \kappa_k)_+^3$$

.

The basis(es) for the regression then becomes:

$$B = \begin{bmatrix} 1 & \mathbf{x} & \mathbf{x}^2 & \mathbf{x}^3 & (\mathbf{x} - \kappa_1)_+^3 & (\mathbf{x} - \kappa_2)_+^3 & \ldots & (\mathbf{x} - \kappa_K)_+^3 \end{bmatrix}$$

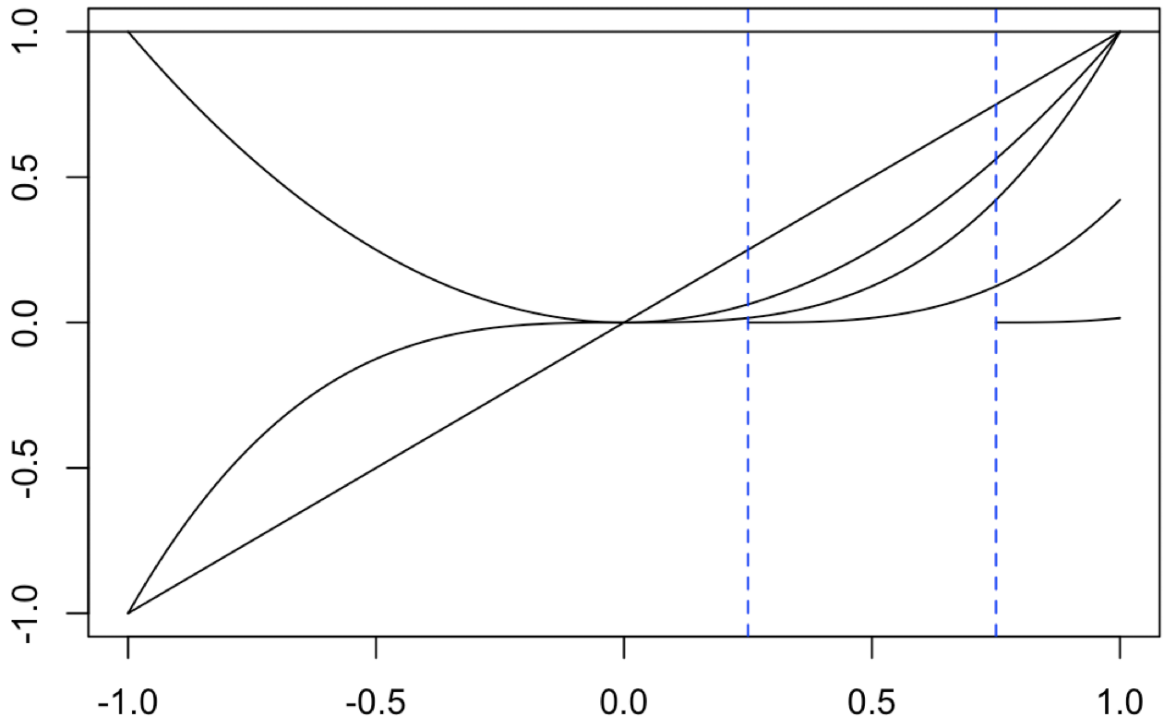

Figure 4: Corresponding basis for cubic spline with two knots at 0.25 and 0.75

To get a smooth and continuous fit of this function to data we have to introduce some constrains:

1. The value of the function is the same on either side of a knot.
2. The first order derivative is the same for two functions on either side of a knot
3. The second order derivative is the same for two functions on either side of a knot

(etc etc with polynomials of degree > 3).

To get better fit at the extremes of the data we can impose an additional constraint stating that the second order derivative of the functions at endpoints must be 0. This is then referred to as a "natural cubic spline" which is one of the most common splines encountered in the wild.

Finding the optimal lambda value

For regression spline we have to decide on 3 main options: 1) the number and placement of knots, 2) the degree of the piece-while polynomials and 3) the value of *lambda*. Commonly we just set a high number of knots and decide to use cubic splines, and with an appropriate λ we can find a good fit regardless. We are therefore most interested in how to automatically choose λ.

The residual sum of squares (RSS) is usually a good measure of the "goodness-of-fit". 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 - we end up with total and utter horseradish of overfitting.

Instead we use leave-one-out cross validation.