An Introduction to Splines

Contents

1 Introduction

One of the fundamental concepts of statistical modelling is the almost omnipresent balance between bias and precision. Models that derive from strong assumptions about the nature of a system produce precise estimates, but the estimates will be biased if the assumptions are not correct. On the other hand, models that derive from weaker assumptions are less prone to bias, but the resulting estimates are less precise. Much of the effort in constructing statistical models entails finding assumptions that strike the proper balance.

Regression methods are used to model changes in a response variable as a function of changes in a predictor variable (or several predictor variables). Standard regression methods belong to the family of parametric models, meaning that they involve strong (parametric) assumptions about the nature of the system being modelled. At the other end of the spectrum, non-parametric regression models are an attempt to make no assumptions at all. Between these extremes lie the semi-parametric methods, which offer a balance by employing very general assumptions: for example, that the relationship between the response and predictor is continuous, or smooth in some sense, without being restricted to a specific shape. Splines belong to the class of semi-parametric techniques.

2 Linear Regression

We will begin be discussing the common methods of parametric regression – including simple linear regression, the method of least squares, and polynomial regression – and then introduce the fundamental concepts of spline smoothing. Throughout the session we will base our examples on daily temperatures recorded in Montreal from 1961 to 1996. This data is freely available in the `fds` library for the `R` software package (hereafter referred to as `R`).

2.1 Simple Regression and the Least Squares Method

Simple Linear Regression Figure 1 depicts the temperatures observed in Montreal daily from April 1 to June 31, 1961. Not surprisingly, the daily temperature shows a consistent, increasing trend over this time period with temperatures near 0C in the early spring and reaching above 20C at the start of this summer. To describe this

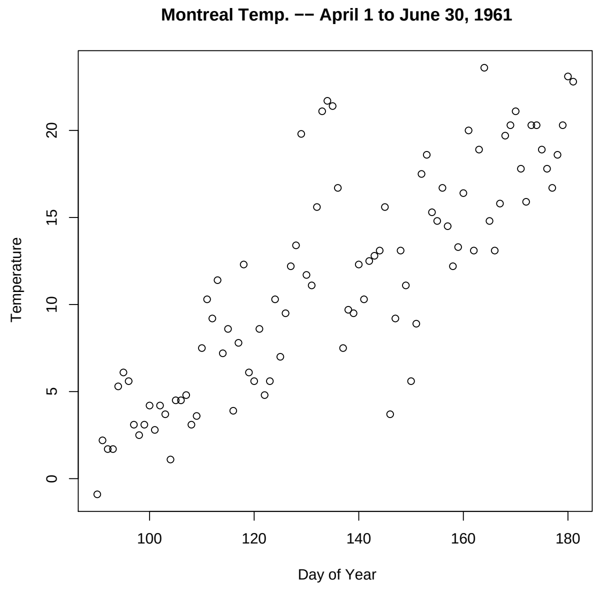

Figure 1: Daily temperatures in Montreal from April 1 (Day 81) to June 30 (Day 191), 1961.

data, we must fit a statistical model which explains first how the mean temperature changes over time and further how the actual observations vary about the mean.

It appears from the plot that the mean temperature increases at an almost constant rate over this time period, and so a straight line is a natural model for this data. In this section, we will discuss fitting the simple, linear regression model to this data by the method of least squares and illustrate how these models can be fit and assessed in `R`.

The primary assumption of the simple, linear regression model is that the average change in the response variable (temperature) is proportional to the change in the predictor variable (day) regardless of the value of the predictor. For the temperature data, this means specifically that the average change in the temperature between two consecutive days is the same whether we days at the beginning of April, in the middle of May, or at the end of June. Secondary modelling assumptions concern the variation of the actual observations about the mean. By fitting a simple linear regression model, we are going to assume that the error for one observation, the distance between the true mean and the actual observed value, does not depend on the error for any other observation and, moreover, that the average magnitude of the errors is the same for all values of the predictor. Suppose that the data contains n observations (for the temperature data $n = 92$ days). We can write the simple, linear regression model in mathematical terms by stating that:

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

for all $i = 1, \ldots, n$ where: y_i and x_i are the values of the response and predictor variables for the i^{th} observation; β_0 and β_1 are the intercept and slope of the line, the parameters that we must estimate; and ϵ_i is the error for the i^{th} observation which we assume to have variance σ^2 for all values of x and to be independent of ϵ_j for all $j \neq i$.

The Least Squares Method To fit the simple, linear regression model we need to estimate the values of the parameters β_0 and β_1. First, we will define a criterion that allows us to assess the fit of the model to the data after we have chosen specific values of β_0 and β_1. We will then compute our estimates by finding the values which minimize this criteria. The specific criterion we will use is called the least squares criterion and the resulting estimates are called the least squared estimates and are commonly denoted by $\hat{\beta}_0$ and $\hat{\beta}_1$.

Suppose that we were to make some guess at values of β_0 and β_1. We could then compute a fitted value for each of the n observations by plugging these values into the model, $\hat{y}_i = \beta_0 + \beta_1 x_i$, and further the error for the i^{th} observation by:

$$e_i = y_i - \hat{y}_i$$

the difference between the observed and fitted value, called the residual. Intuitively, the values of β_0 and β_1 are good if the fitted values are close to the observed values and the residuals are small. It will never be possible to make all of the residuals equal to 0 (except in the unlikely case that the points actually lie along a straight line) so instead we will estimate β_0 and β_1 by making the average of the residuals as small as possible (actually, the average of the square of the residuals). The least squares fitting criterion is:

$$SS = \sum e^2 = \sum (y - (\beta_0 + \beta_1 x))^2,$$

the sum of the squared residuals, and the least squares estimates are the values of β_0 and β_1 which minimize this criterion. Although these estimates can be computed explicitly for the simple, linear regression model, this is not true in general and we focus instead on how to fit this model in `R` .

2.2 Simple Linear Regression in `R`

Least Squares Fitting `R` Linear regression models can be fit in `R` using the `lm` function. Suppose that the values of the response, the observed temperatures, are stored in the vector `y` and values of the predictor, the days from 80 to 191, in the vector `x`. The command to fit the linear regression model is simply: `lm(y~x)`. Executing this command in `R` returns the estimates of the two parameters which are: $\hat{\beta}_0 = -15.4$ and $\hat{\beta}_1 = .2$. This indicates that the mean temperature increases by .2C every day and that *if the linear model applied to the data before April 1* then the mean temperature on day 0 (December 31 of the previous year) would be -15.4C. As we will see, this assumption is not justified so the interpretation of $\hat{\beta}_0$ is not so simple. Instead, it would be better to say that the mean temperature on day 80, the first day of observation, is $-15.4 + .2(80) = .6C$.

The first argument of the `lm` function is a formula which specifies the structure of the regression model. The tilde operator separates the response, on the left, from specification of the predictor variables, on the right. By default, `R` includes an intercept in the structure of the model, so we don't need to specify this explicitly. If we had multiple predictors of y, then we would list them all on the right hand side. For example, the formula `y~x1 + x2` defines y as a linear function of x_1 and x_2:

$$y = \beta_0 + \beta_1 x_1 + \beta_1 x_2.$$

Again, we do not need to list the intercept in the predictors because it is included by default. There are several operators that can be used to define more complicated structures with multiple predictors. For example, the model including the interaction term between the two variables can be written as `x1 + x2 + x1:x2` or more succinctly as `x1*x2`. More information on defining formula can be obtained using the command `help(formula)`.

Plotting the Fitted Curve The `lm` function actually produces much more output which can be accessed by storing the results in a new variable, say `lmfit`: `lmfit=lm(y~x,data)`. A full list of the output can be obtained with the command `attributes(lmfit)` and details for each component are provided in the help file accessed by the command `help(lm)`. One product of the function is the vector of fitted values for each observation, computed as:

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i$$

and can be accessed through the variable `lmfit$fit`. Figure 2 illustrates the fit of the least squares line to the observed temperatures from the spring of 1961. This figure was produced commands:

```
lmfit = lm(y~x,data)
plot(x,y,main="Montreal Temp. -- April 1 to June 30, 1961",
     xlab="Day of Year",ylab="Temperature")
lines(x,lmfit$fit,col="red",lwd=3)
```

which first fit the model, then plot the raw data, and finally overlay the estimated mean from the best linear model.

Residual Diagnostics Another product of the `lm` function is the set of residual values which can be used to assess the fit of the model to the data. As noted above, the residual for an observation is the difference between the fitted and the observed values of the response. Ideally, we would like the residuals not only to be small but

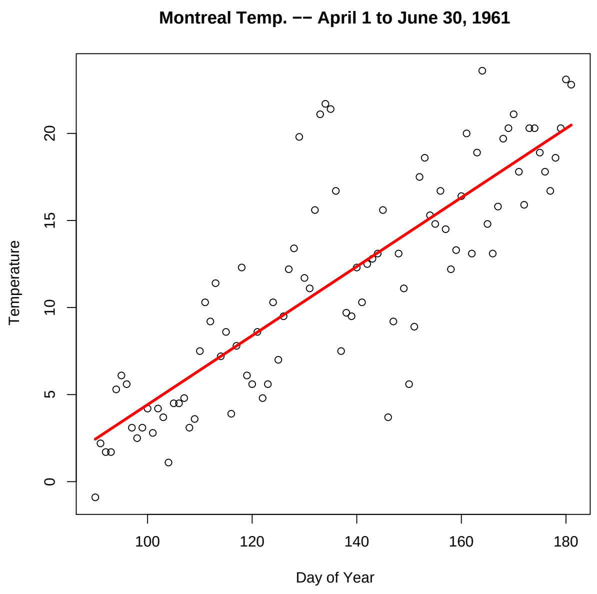

Figure 2: Daily temperatures in Montreal from April 1 (Day 81) to June 30 (Day 191), 1961: simple, linear regression model.

also to have the same size, on average, for all values of the response and the predictor. This can be assessed by constructing plots of the residuals versus x and y. If the model fits the data well, then both plots will show an even scatter of the residuals about 0 with no discernible pattern. If there are regions in which the residuals are consistently above or below 0 in either plot, then this indicates that the model has not properly captured the trend in the mean and that more structure is needed in the model. If the residuals in some regions are much larger than in other regions, commonly visible as an overall funnel or football shape, then this indicates that the variance of the errors is not constant.

Residual plots produced from the fit of the simple, linear regression model to the Montreal temperatures in the spring of 1961 are shown in Figure 3. Both plots show a fairly good scatter of the residuals about 0, with the exception of the residuals for the observations from days 129, 133, 134, 135 and 164, which are much higher than those on the other days, and for the observations from days 146 and 150 which are much lower than those on the other days. These observations are outliers, meaning that their values are not explained properly by the model. It is possible that these values are unusually large because of a phenomenon that is not included in the model, like an unusually warm week that does not fit the slow warming trend. It is also possible that these observations result from error in the measuring device or in data transcription – perhaps the temperature on May 9 was not 19.8C as recorded but in fact 9.8C, which would lie almost exactly on the fitted line. In practice, these outliers can be removed if it is known that they resulted from errors in the data or else the model should be expanded to account for the unusual behaviour. This issue will not be explored further here.

Exercises

1. Montreal Temperature Data – April 1 to June 30, 1961
   Code for fitting the simple linear regression model to the Montreal temperature data from the spring of 1961 is included in the file `montreal_temp_1.R`. Use this code to fit the model, plot the fitted line, and produce the residual plots.

2. Montreal Temperature Data – January 1 to December 31, 1961
   Repeat exercise 1 with the data from all of 1961 using the code in the file `montreal_temp_2.R`.

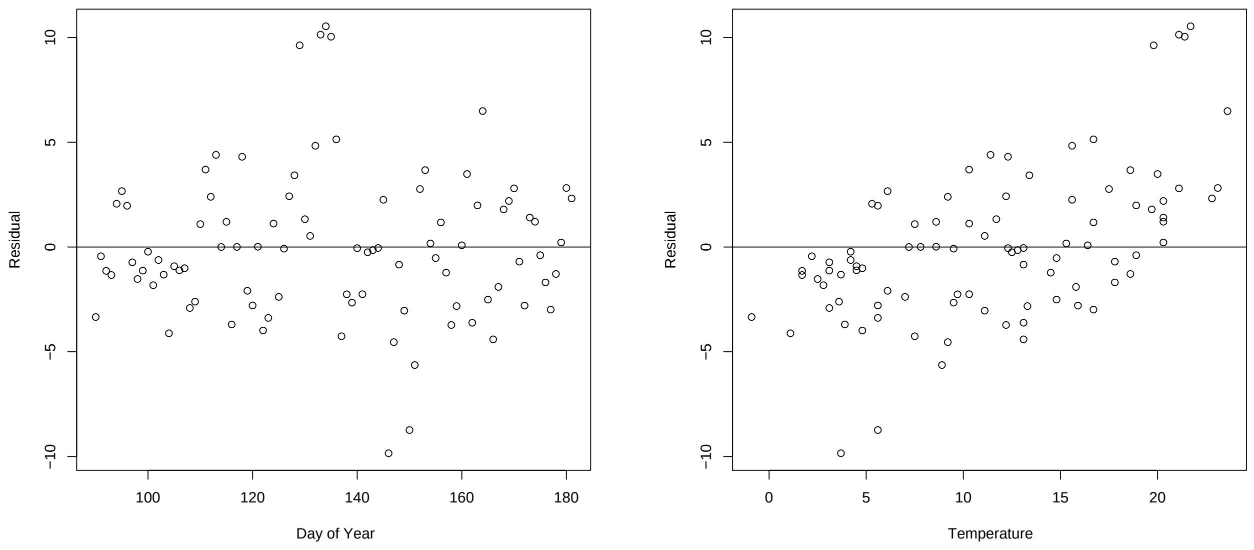

Figure 3: Residual daily temperatures in Montreal from April 1 (Day 81) to June 30 (Day 191), 1961 versus day (left) and temperature (right).

2.3 Polynomial Regression

Polynomial Regression The assumption of linearity is often adequate for describing relationships over small periods of time, but not for modelling long term trends. This is clear in the case of the Montreal temperature data. Although the linear increase describes the changes in mean temperature well for the spring of 1961, the trend over the entire year is clearly not linear. The mean temperature increases during the spring, plateaus in the summer, and decreases again in the fall (After all, Montreal is in the northern hemisphere). The left panel of Figure 4 illustrates the best linear fit to the Montreal temperature data for all of 1961 and shows that the model overestimates the mean temperature at the start of the year, underestimates in the middle of the year, and overestimates again at the end of the year. This behaviour is also clear from the residual plot in the right panel.

One way that the new relationship can be modelled is by constructing the predictor as a polynomial in x: a function that includes powers of x like the quadratic (x^2), cubic (x^3), and quartic (x^4) terms . Suppose that we wish to model the response as a degree D polynomial in the predictor. The mathematical equation is:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \ldots + \beta_D x^D$$

or:

$$y = \beta_0 + \sum_{k=1}^{D} \beta_d x^d$$

using summation notation. Estimates of the coefficients $\beta_0, \ldots, \beta_D$ can again be found by minimizing the least squares criterion:

$$SS = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

where e_i is still the residual but the fitted value is now given by $\hat{y}_i = \beta_0 + \sum_{k=1}^{D} \beta_d x_i^d$. As in the case of simple, linear regression, explicit expressions for the least squares estimates do exist (using matrix notation) but we will instead consider how to fit these models `R` .

The Design Matrix The D^{th} degree polynomial model can again be fit in `R` using the `lm` function. One way to construct the polynomial would be to define new variables containing the values of $x^2, x^3, \ldots, x^D$ and then

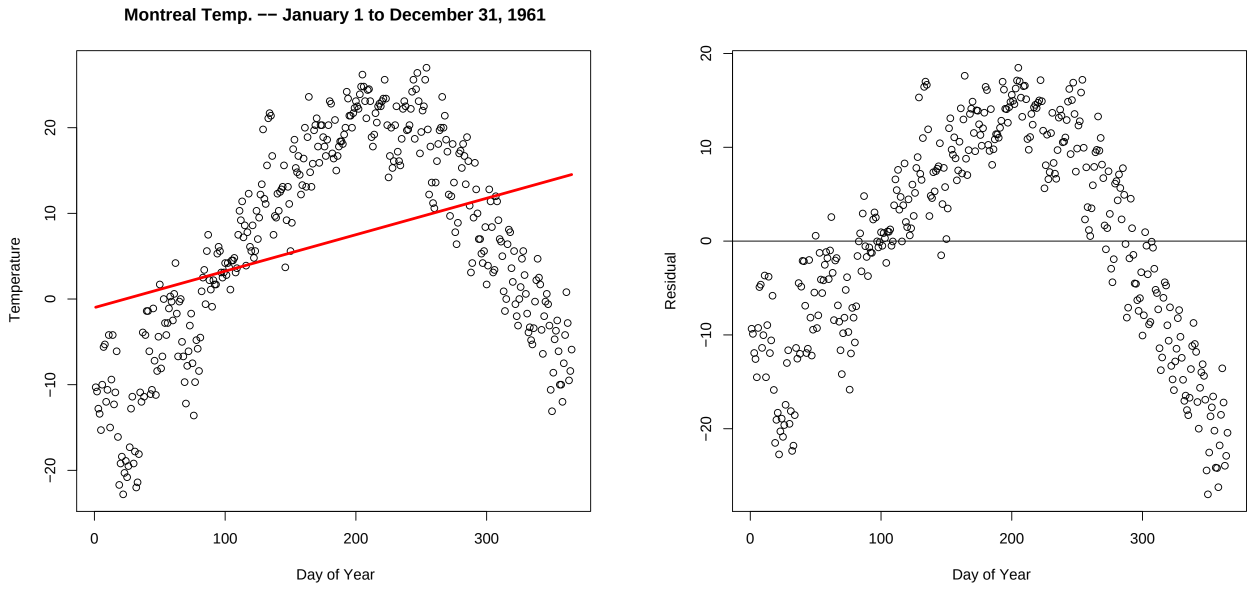

Figure 4: Daily temperatures in Montreal from January (Day 1) to December 31 (Day 365), 1961: fitted simple, linear regression model (left) and corresponding residuals versus day (right).

to list these values on the right hand side of the formula. However, a simpler way to do this is to construct a design matrix which contains all of the values in a single object. In a multiple regression problem including n observations and p predictor variables, the design matrix is the $n \times p$ matrix whose i^{th} row contains the values of each of the predictor variables associated with the i^{th} observation (alternatively, whose j^{th} columns contains the values for the j^{th} predictor for each individual). For the degree D polynomial regression model, the i^{th} row of the design matrix will contain the values $1, x_i, x_i^2, \ldots, x_i^D$. In total, there are in fact $D + 1$ predictor variables and the 1 in the first entry represents the intercept term.

Constructing the Design Matrix in R The design matrix for the degree D polynomial regression model can easily be constructed in R with the function `outer`. This function takes three arguments, two vectors and a binary operator (e.g., addition, subtraction, multiplication etc), and returns a matrix that is formed by applying the specified operator to each pair of elements from the two vectors. To construct the design matrix, we will supply the vector of predictor values (x), the vector of exponents in the model $(1, \ldots, D)$, and the exponentiation operator `^`. The command is:

```
X = outer(x,1:D,"^")
```

and the linear model is then fit as:

```
lmfit = lm(y~X,data=data)
```

Note that the `lm` function includes the intercept term by default, so it is not necessary to put the leading 1 in the design matrix.

Quadratic Model for Montreal Temperature in 1961 The left panel of figure 5 illustrates the fit of the best quadratic model for the 1961 Montreal temperature data. As expected, the fitted model now follows the increasing temperature in spring and the decreasing temperature in fall. However, the residual plot versus day, in the right panel, shows that there are still areas where the model does not fit properly. In particular, the model underestimates the temperature at the very beginning of the year (all residuals are above 0) and also seems to underestimate the summer temperature. The residual plot shows four clusters through the year that follow a high-low-high-low pattern. Better fit to the data can be achieved by including higher order terms in the regression model.

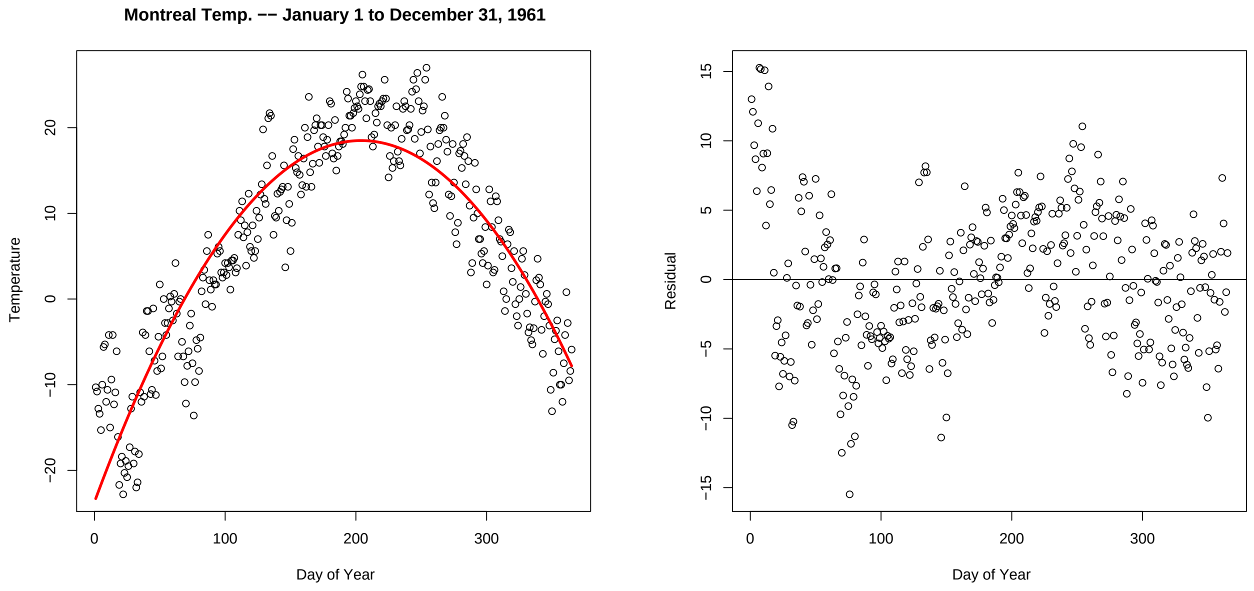

Figure 5: Daily temperatures in Montreal from January 1 (Day 1) to December 31 (Day 365), 1961: fitted quadratic regression model (left) and corresponding residuals versus day (right).

Exercises

1. Montreal Temperature Data – January 1 to December 31, 1961
   Use the code in the file `montreal_temp_3.R` to fit polynomial regression models of varying degree to the data for all of 1961. Models of different degree are constructed by changing the variable `D` (e.g., `D=2` produces a quadratic model and `D=3` a cubic). What is the minimal degree required for the model to fit well?

2. Montreal Temperature Data – January 1, 1961, to December 31, 1962
   Use the code in the file `montreal_temp_4.R` to repeat this exercise using the data recorded in both 1961 and 1962.

3 Smoothing Splines

3.1 Truncated Polynomial Basis

Piecewise Polynomials Figure 6 shows the fit of an 8^{th} degree polynomial to all of the data from 1961 and 1962. The model does fit the mean temperature fairly well (except, perhaps, for an unusually cold period at the end of January, 1962) but is fairly complicated and difficult to interpret. Linear and quadratic models have simple interpretations – a linear model represents a system with a constant rate of change and a quadratic model a system with a rate of change that increases or decreases at a fixed rate (the acceleration). Even a cubic model can be believed to represent some underlying process in the system – the acceleration changes at a constant rate. However, polynomials of higher degree are hard to interpret, and it is difficult to believe that the fitted equation actually represents the underlying behaviour in the system. What does the 8^{th} degree polynomial say about the change in temperature in the spring of 1962, and how does this compare with the spring of the previous year?

An alternative to increasing the degree of the fitted polynomial to account for curvature is to divide the range of the data into smaller pieces and fit simpler models to each of the subintervals. As seen in the previous sections, linear, quadratic or cubic polynomials are sufficient to fit small ranges of the data However, it seems reasonable to stipulate that the resulting function be continuous, without any jumps. Ideally, we would like the separate

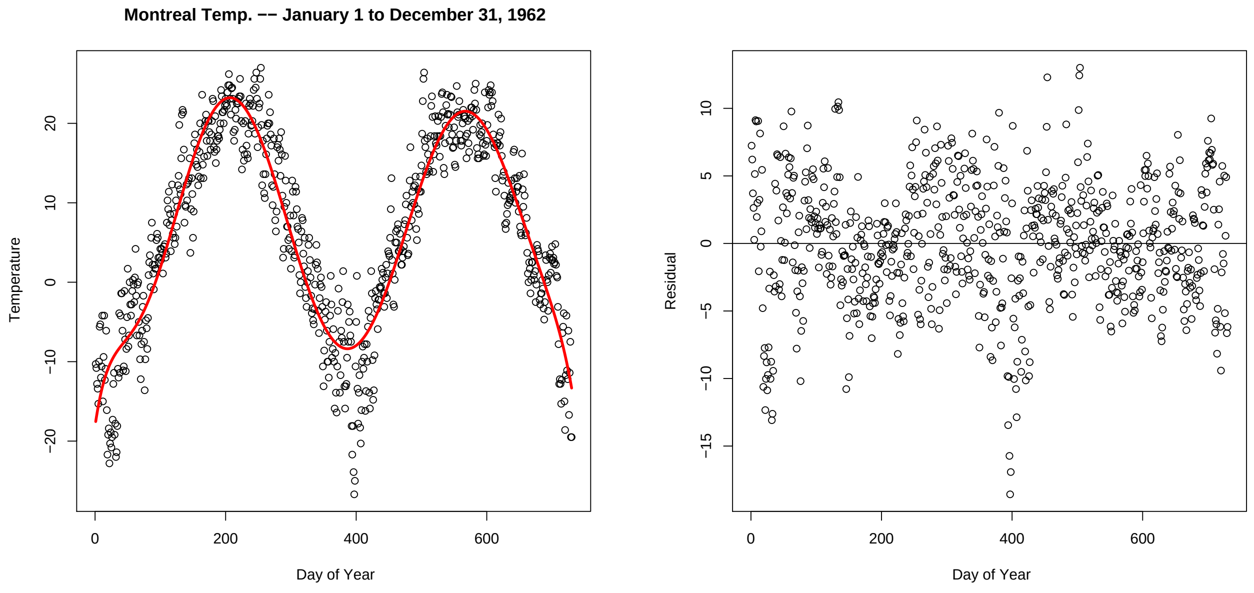

Figure 6: Daily temperatures in Montreal from January 1, 1961 (Day 1), to December 31, 1962 (Day 730): fitted 8^{th} degree polynomial regression model (left) and corresponding residuals versus day (right).

polynomial pieces to connect at the boundaries of the intervals, and, perhaps, to produce a smooth curve over the entire range of the data (whatever we may mean by smooth). This is exactly what a smoothing splines does.

Linear Splines The simplest spline (a spline of degree 1) is formed by connecting linear segments. Figure 7 illustrates the fit of a linear spline to the Montreal temperature data for 1961 and 1962. The range of the data has been divided into 10 subintervals of 73 days each and the fitted function is linear over each of the subintervals. The slope of the line is allowed to change between intervals, but the segments meet exactly at the interval boundaries. In spline terminology, the dividing points are called the knots of the spline. To divide the range into 10 subintervals requires 9 knots, and these are indicated by the red points in the figure. Mathematically, the equation for the linear spline is:

$$y = \beta_0 + \beta_1 x + \sum_{k=1}^{9} b_k (x - \xi_k)_+$$

where $\xi_1 = 73, \xi_2 = 146, \ldots, \xi_9 = 637$ are the knots of the spline. The function $(x - \xi)_+$ is defined as:

$$(x - \xi)_+ = \begin{cases} 0 & x < \xi \\ x - \xi & x > \xi \end{cases}$$

so that its value only affects the fit of the spline to the right of ξ. This is how the slope of the fitted function changes at each knot. For values of x left of the first knot the fitted function is simply $y = \beta_0 + \beta_1 x$. Between the first and second knot the fitted function is $y = \beta_0 + \beta_1 x + b_1(x - \xi_1) = (\beta_0 - b_1\xi_1) + (\beta_1 + b_1)x$. The result is a new linear segment which meets with the first segment at ξ_1, but has a slope of $\beta_1 + b_1$. In general, the slope of the line between the k^{th} and $k + 1^{st}$ knot is $\beta_1 + b_1 + \ldots + b_{k-1}$.

The values of the parameters, including both β_0 and β_1 and the new parameters $b_1, \ldots, b_9$, can once again be estimated by minimizing the least squares criterion and fit in `R` using the `lm` function. Note that the fitted function requires 11 parameters, so in some sense it is more complicated than the 8^{th} degree polynomial which only required 9 parameters. However, the equation at any single point is much simpler and the fitted curve is much easier to interpret. The slope of the linear segment over the spring of 1961 is .24C/day and the slope over the spring of 1962 is .32. This suggests that the temperature rose faster in the spring of 162 than in 1961.

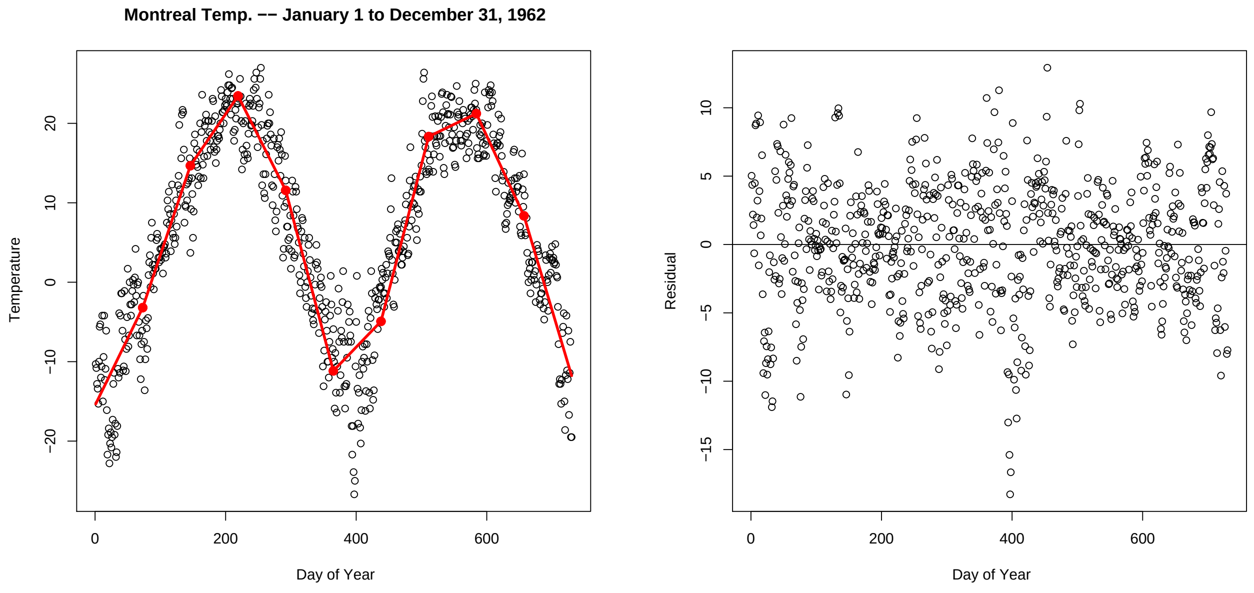

Figure 7: Daily temperatures in Montreal from January 1, 1961 (Day 1), to December 31, 1962 (Day 730): fitted linear spline with 10 knots indicated by the red points (left) and corresponding residuals versus day (right).

Higher Order Splines The residual plot in Figure 7 indicates that the spline fits the data as well as, if not better than, the 8^{th} degree polynomial, but the sudden changes at the knots are not appealing. By constructing the spline from segments of higher degree – quadratic or cubic polynomials – we can obtain a curve that is still simple to interpret but produces a more appealing, smooth fit to the data. Generally, a spline of degree D with K knots is defined by the equation:

$$y = \beta_0 + \beta_1 x + ...\beta_D x^D + \sum_{k=1}^{K} b_k (x - \xi_k)_+^D$$

where the function:

$$(x-\xi)_+^D = \begin{cases} 0 & x < \xi \\ (x-\xi)^D & x > \xi \end{cases}$$

is called a truncated polynomial of degree D. As with the linear spline, these terms only affect the fit of the function to the left of the point ξ. In general, a spline of degree D with K knots is a continuous function formed from $K+1$ polynomial segments of degree D which connect at the knot points. The first $D-1$ derivatives are continuous over the entire range of the data, and the D^{th} derivative is continuous except at the K knot points. In contrast, a polynomial of degree D has D continuous derivatives over the entire range. The number of parameters in the spline is $1+D+K$.

Increasing the degree of the spline makes the fitted function appear more smooth but also increases the complexity of the polynomial segments in each interval. In practice, the most common choice is the cubic spline ($D=3$). Figure 8 shows the fit of a cubic spline with 5 knots to the data from 1961 and 1962. The number of parameters in the spline is $5+3+1=9$, the same as for the 8^{th} degree polynomial, but the fitted function is much easier to understand.

The Design Matrix The design matrix for this model contains the values 1 to x^D in the first $D+1$ columns and then the values of the truncated polynomial terms in the final K columns. The two parts of the matrix can be defined separately in R and then fused together using the function `cbind` which joins matrices columnwise. The first matrix is formed using exactly the same call as above (once again, we drop the column of 1s because the intercept is automatically included by `lm`). The second can be constructed with two calls to the `outer` function:

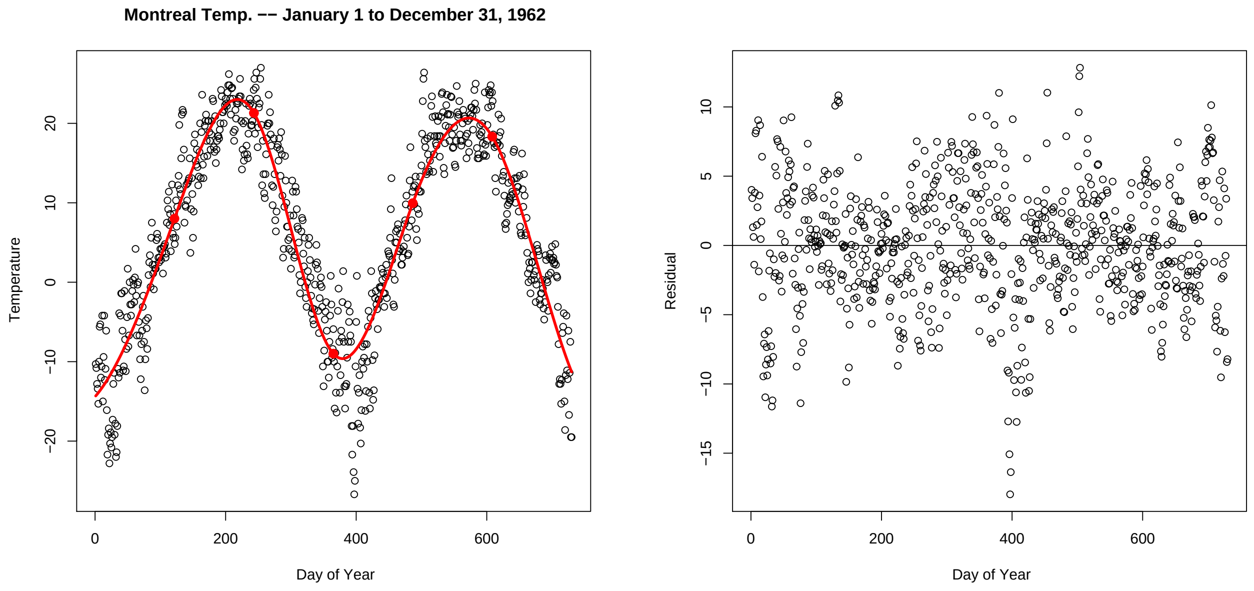

Figure 8: Daily temperatures in Montreal from January 1, 1961 (Day 1), to December 31, 1962 (Day 730): fitted degree cubic spline with 5 knots indicated by the red points (left) and corresponding residuals versus day (right).

the first to create a matrix with entries $(x - \xi_k)^D$ for each value of x and ξ_k, and the second to construct a matrix with 1's where $x > \xi_k$ and 0's otherwise. The full call in R is:

```
X <- cbind(outer(data$x,1:D,"^"),
           outer(data$x,knots,">") * outer(data$x,knots,"-")^D)
```

Exercises

1. Montreal Temperature Data – January 1 to December 31, 1961
   Fit splines to the data from 1961 and 1962 using the code in the file `montreal_temp_5.R`.

3.2 B-spline basis

Troubles with Truncated Polynomials The truncated polynomial terms in the spline add a lot of flexibility to the fitted function because they allow for the fit to be local – meaning that some of the parameters in the model depend on only part of the observed data. However, the use of truncated polynomials can sometimes lead to numerical problems. If two knots are very close then the associated truncated polynomial terms will be very similar for all observations, almost co-linear, and the fitting algorithm will become unstable. As well, the values in the design matrix can be very large which can lead to overflow errors. The largest value in the design matrix for the cubic spline fit to the Montreal temperature data is 730^3, greater than 389 million.

The B-spline Basis These problems can be avoided in practice by using a different set of functions in place of the truncated polynomials. There a several sets of functions which produce the same range of fitted curves, called equivalent bases, but the most common is the set of B-splines. Although any function that can be fit with the truncated polynomials can also be fit with B-splines, and vice versa, the B-spline basis functions are not colinear and their values are always between 0 and 1. This makes the computational algorithms for fitting splines much more stable.

In R , the design matrix for the B-spline basis can be constructed with the function `bs` in the `splines` library. The arguments to the function are: the vector of the predictor values, the vector of knot points, and the degree of the spline. We can also choose to include the intercept column in the design matrix rather than relying on the lm function to automatically include this. To make things simpler in the next section, we will choose to include the intercept term in design matrix and then tell lm not to add the column of ones. The full call is:

```
X <- bs(data$x,knots=knots,degree=D,intercept=TRUE)
lmfit <- lm(y~X-1,data=data)
```

where the first command constructs the design matrix and the second fits the model. The notation -1 in the regression formula tells lm that the intercept is included in the design matrix and so it does not need to be added. Figure 9 shows the fit of a B-spline of degree 3 with 5 equally spaced knots. The fit is exactly the same as the fit of the degree 3 truncated polynomial spline with 5 knots shown in Figure 8.

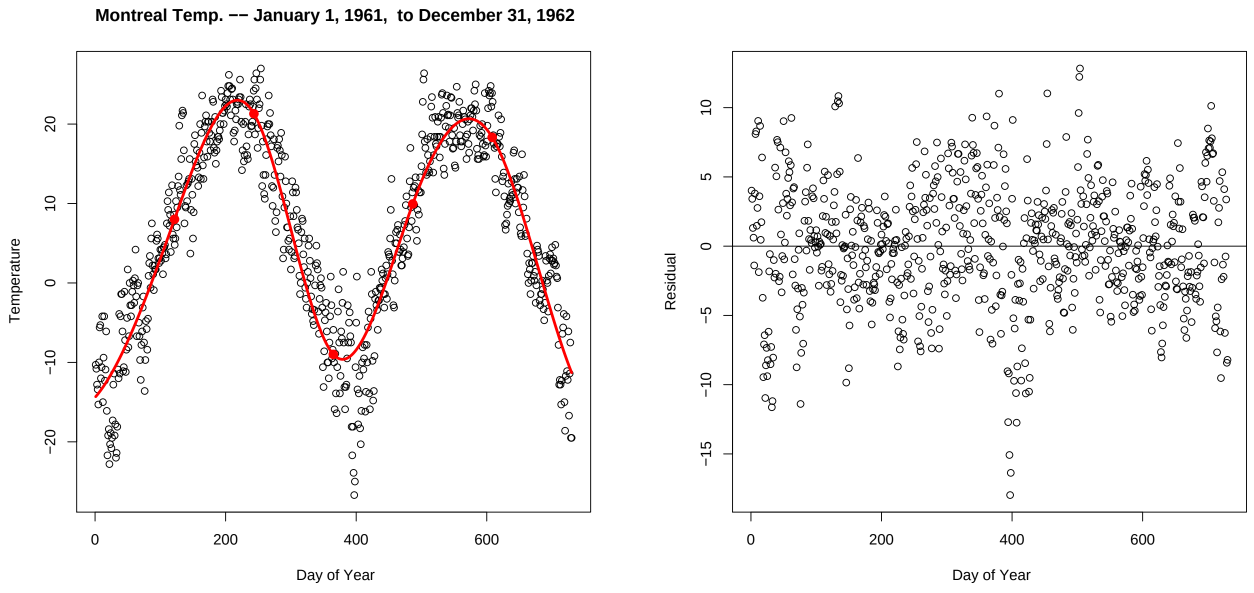

Figure 9: Daily temperatures in Montreal from January 1, 1961 (Day 1), to December 31, 1962 (Day 730): fitted B-spline of degree 3 with 5 knots indicated by the red points (left) and corresponding residuals versus day (right).

Exercises

1. Montreal Temperature Data – January 1 to December 31, 1961
   Fit a B-spline to the data from 1961 and 1962 using the code in the file montreal_temp_5.R. Increase the number of knots to see how this affects the fit of the curve. What happens when the number of knots is very large, say $K = 50$?

3.3 Overfitting, Smoothness and Penalization

Overfitting and Smoothness The spline with 5 knots used in the previous section seems to fit the Montreal temperature data very well. It shows a smooth change in the mean over the two year period and captures the important trends in temperature. When the number of knots is increased further the subintervals contain only a few points each and the polynomial segments begin to interpolate the data. Figure 10 shows the fit of a spline of degree 3 with 50 equally spaced knots. The new curve still captures the overall behaviour, but it is also picking out a lot of the details in the data. We say that the model is overfit. In general, there is a balance between the fit and the smoothness of a spline. A spline with only a few knots will be very smooth but may not be able to adapt to the shape of the data. A spline with many knots can easily adapt to the shape of the data but may lead to overfitting.

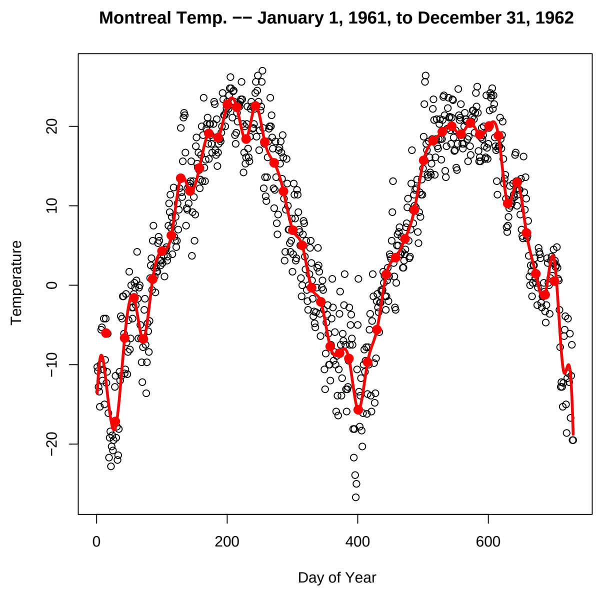

Figure 10: Daily temperatures in Montreal from January 1, 1961 (Day 1), to December 31, 1962 (Day 730): fitted cubic B-spline with 50 knots indicated by the red points.

Knot Selection versus Penalization One way to avoid overfitting is to vary the number of knots (and their location) until we find an arrangement that allows the spline to properly adapt to the shape of the data. However, finding the optimal set of knots can be a very complicated procedure that requires very intensive computing. An alternative strategy is to always use many knots, but to control overfitting by limiting the amount between the segments in neighbouring intervals. Recall that the slope of the the linear spline constructed from truncated polynomials is β_1 to the left of the first knot and $\beta_1 + b_1$ to the right. The difference in the slope may be very severe if b_1 is allowed to vary at will, but if we restrict the value of b_1 then the slope will change slightly and the fitted function will always be smooth.

Penalization for Truncated Polynomials To ensure that the fitted spline doesn't overfit the data we can define a new fitting criterion that attempts to minimize both the residuals and the values of the parameters $b_1, \ldots, b_K$. The new criterion is:

$$\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda \sum_{k=1}^{K} b_k^2$$

and is called the penalized least squares criterion because the second sum penalizes the coefficients. Although larger values of $b_1, \ldots, b_K$ may improve the fit, and decrease the sum of squared residuals, they will increase the penalty term and so may not be preferred. The value λ is called the smoothing parameter and controls the balance between smoothnes and fit. If λ is large, then the criterion weights the penalty more heavily so that large values of b_k are severely penalized and the resulting spline is very smooth. If λ is small then the penalty term will have little effect and the spline will come close to interpolating the data.

Penalization for the B-spline Basis A spline formed in the B-spline basis becomes smoother as the differences between the coefficients get smaller, not as the values of the individual coefficients approach 0. Hence, smoothness and fit will be balanced by a fitting criterion that penalizes the differences between the coefficients instead of their absolute values. For technical reasons that won't be explained here, it actually makes more sense to penalize the second differences (i.e., the differences of the differences) of the coefficients when working with the cubic B-spline basis. The criterion is:

$$\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda \sum_{k=3}^{1+D+K}((b_k - b_{k-1}) - (b_{k-1} - b_{k-2}))^2$$

and the parameter λ again controls the smoothness of the fit. There is no restriction on the smoothness when λ is small and the fit will be very close to a cubic polynomial when λ is large. The fit of the penalized spline constructed from the B-spline basis with 50 equal spaced knots and $\lambda = 5$ is shown in Figure 11.

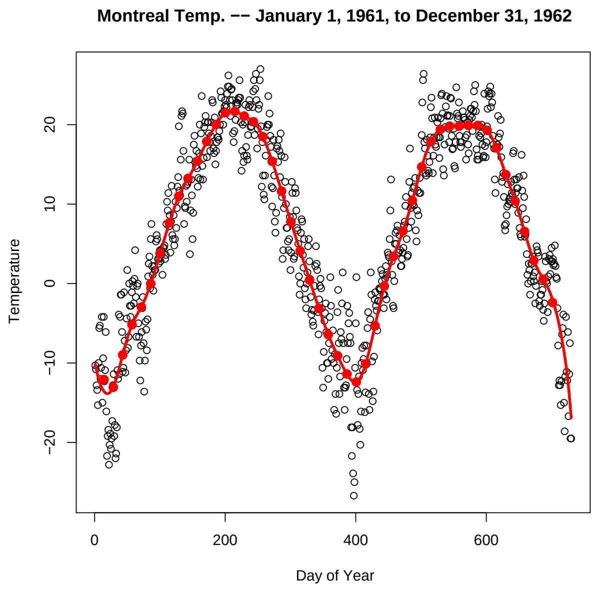

Figure 11: Daily temperatures in Montreal from January 1, 1961 (Day 1), to December 31, 1962 (Day 730): fitted penalized spline of degree 3 with 50 knots, indicated by the red points, and a penalty on the second differences with $\lambda = 5$.

Exercises

1. Montreal Temperature Data – January 1, 1961, to December 31, 1962
   The file `montreal_temp_7.R` contains code for fitting a penalized, cubic spline with the B-spline basis. Vary the smoothing parameter, `lambda`, to see how this affects the shape of the fitted curve.