Implementation of B-Splines via Coox - De Boor algorithm and possibile applications

Christian Riccio P37/02
Antonio Elia Pascarella P37/03
Claudio Ciano P37/06

Abstract

This article shows our approach to the implementation of the **Coox - De Boor algorithm** for the construction of the basis splines, also known as B-Splines. We will also show how to find the best knots positions trough an optimization algorithm and the fitting procedure with a GLM model. In the end a possible application is shown in the field of medical imaging studies.

1 Introduction: B-Splines theory

A spline function of order **n** is a piecewise polynomial function of degree **n-1** in a variable x. The places where the pieces meet are known as knots. The key property of spline functions is that they and their derivatives may be continuous, depending on the multiplicities of the knots.

B-splines of order n are basis functions for spline functions of the same order defined over the same knots, meaning that all possible spline functions can be built from a linear combination of B-splines, and there is only one unique combination for each spline function.

When facing the modelling of data, the underlying function is other than linear, in fact situations where non linearity is involved are very common. Often, the nature of the function which describes the data is unknown and what we try is to model it with non linear functions. A well known solution is to write it as linear combination of basis splines, which allows us to move beyond linearity.
The algorithm for the construction of basis splines (Coox - De Boor) it's described in the following section.

Construction of basis splines

Step 1

Define an augmented knots sequence:
$\xi^* = (\xi_{-p}, ..., \xi_0, \xi, \xi_{k+1}, ..., \xi_{k+p+1})$, where p is the degree of the spline and K is the number of internal knots, so that k+p+1 equals to the total degree of freedom related to the method.

Step 2

Build the first order base for the observed data (p=0), which gives the fundamental for a step-function spline. For $i = 0, ..., k$, let

$$B_{i,0}(x) = \begin{cases} 1 & x \quad \in \quad [\xi_i, \xi_{i+1}) \\ 0 & otherwise \end{cases} \tag{1}$$

Where $B_{i,0}(x) = 0$ if $\xi_i = \xi_{i+1}$.

Step 3

The i-th B-spline basis function of degree j, $j = 1, ..., p$ is given by:

$$B_{i,j}(x) = \frac{x - \xi_i}{\xi_{i+j} - \xi_i} * B_{i,j-1} + \frac{\xi_{i+j+1} - x}{\xi_{i+j+1} - \xi_{i+1}} * B_{i+1,j-1}$$

for $i = -p, .., k + p - j$.

The result can be seen in figure 1, showing a strong non linear composition of the basis functions.

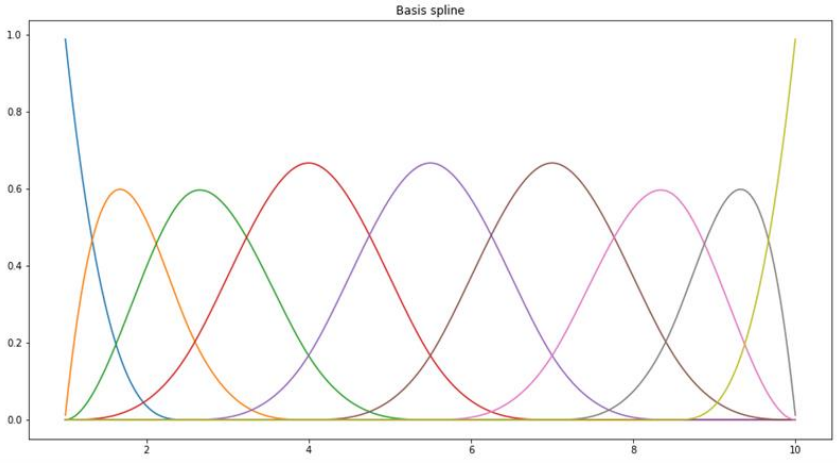

Figure 1: Example output of basis spline of degree 3 with 5 equally spaced internal knots.

N.B. For each x value, the summed components of the basis is equal to 1.

2 The project

The implementation of the code is given as a Jupyter notebook and it is mainly composed by 2 important data structures:

- **B_spline** class
- **my_data** class

The first class allows us to generate and manage the matrix of basis splines, in particular by executing a recursive algorithm after the number of knots have been fixed. This class also includes a procedure that locate autonomously the knots in closer regions as the slope of the resulted spline approximately increases. More in depth, we roughly approximated our function with a regression spline of order 1 and 30 knots. We used the cumulative of derivatives in order to map equally spaced knots into more thickened ones, where the signal has higher slope.

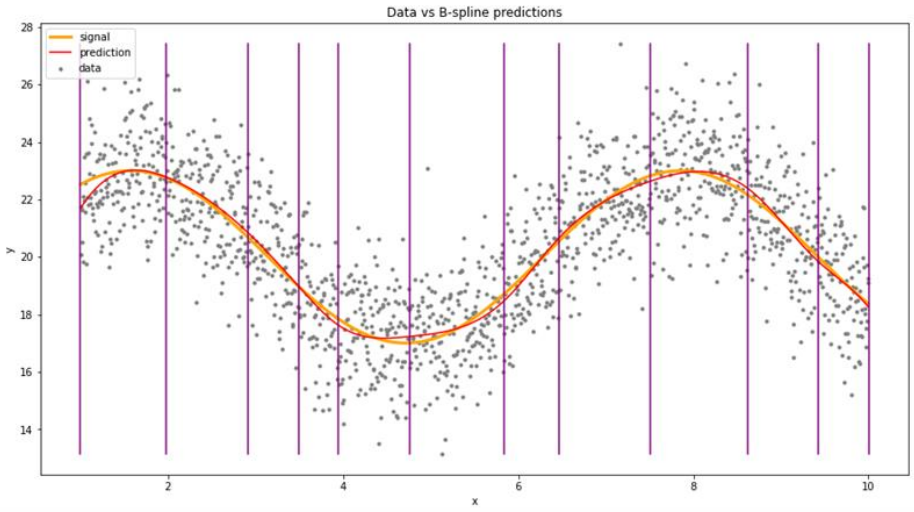

Figure 2: Spline fitted on the data using B-spline basis functions (red line), known a-priori sinusoidal signal (orange line) and position of not equally spaced knots (vertical purple lines).

The second class can hold either simulated or real data together, with some feasible inner operations, such as the addition of a noise to the data or the definition of a signal that the spline should be able to find, etc. We can think about data as a composition of a signal and a noise that may follow several different distributions (poisson, gaussian, ...) depending on the generator process. In

particular, the plots shown in the notebook refer to a Gaussian noise summed to a sinusoidal signal. The basis matrix obtained from the algorithm above is used as a regressor to fit a GLM model; in order to perform a GLM fit, it is mandatory to choose a link function and a family distribution for the response variable. The analytical formulation of the model is the following:

$$\hat{\mathbf{y}} = \mathbf{B} \cdot \mathbf{w}$$

where B represents the matrix of the basis and w the coefficients that have to be estimated as results of the GLM.
Once we accomplished the fitting procedure, we are always capable of building the basis vector for a new point and get an estimation for the signal by computing:

$$\hat{y} = \mathbf{B}_x \cdot \mathbf{w}$$

where $\mathbf{B}_x$ is a row vector for the corresponding base at the point x.
In order to check for the stability of the model we performed a 10-fold Cross Validation. In more details, at each iteration a portion of random points is removed from the set of data and the model is fitted on the remaining points. Each subset of random samples chosen to be removed at each iteration does not overlap with the others. We iterated this procedure 10 times, obtaining 10 different models. We computed the standard deviation and accordingly to the assumption of a normal distribution for the response variable we were able to construct the 95% confidence intervals.

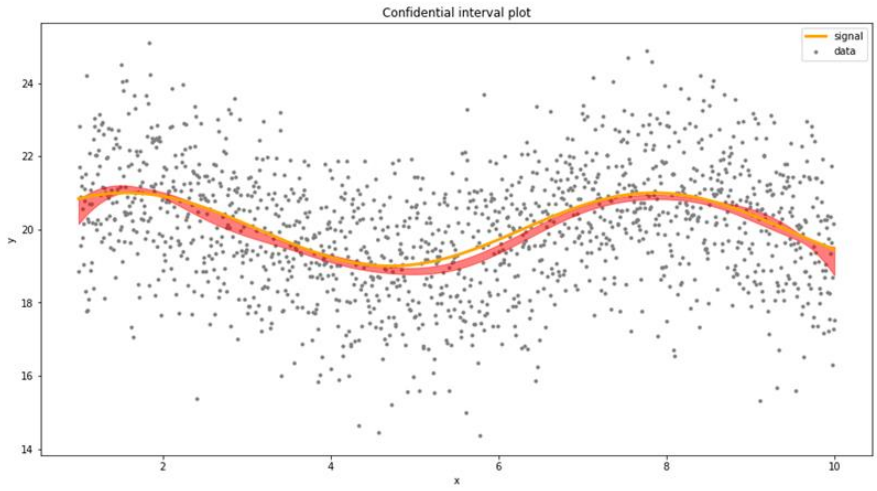

Figure 3: 10-fold cross validation confidence interval.

The tuning parameter of knots number is almost always a not easy task and requires some statistic considerations. In this case we fitted several models iteratively, increasing the number of knots and choosing the parameter that

ensures a good trade-off between higher parsimony and lower residual sum of square (RSS). Eventually, we applied the regression B-spline method on a med-

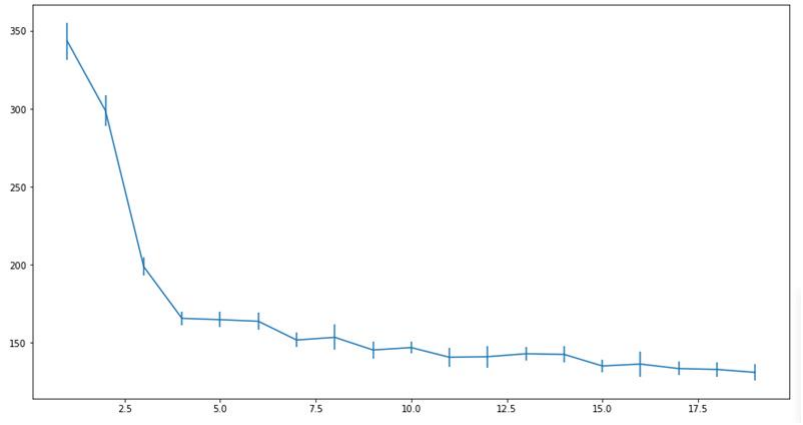

Figure 4: RSS as a function of number of knots.

ical image in which we wanted to outline the edges of a human frontal lobe. We evaluated a GLM assuming a Poisson distribution for the random component, because of the nature of the pixels that are positive integer numbers. To obtain the edge of the frontal lobe we decided to select for each columns of pixels, those first exceeding a fixed threshold of 30 % of the maximum intensity.
In this specific task we managed an image resulted from a TAC, where an anomalous thickening of the skull wall has been found, probably due to the **Paget** disease.

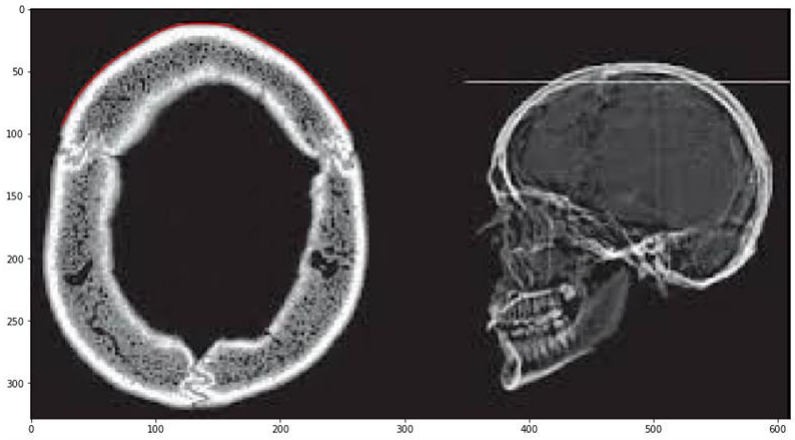

Figure 5: Planar prospective of a human frontal lobe, the red line represents the signal of the edge of the skull.