LAB5 - Cubic Spline Regression

We can finally continue with our lab sessions!

In this lab, we are going to implement a cubic spline regression. A cubic spline, like any other non-linear regression technique, can be implemented using standard multiple linear regression.

IMPORTANT NOTE: We are going to implement this one ourselves, which means no outside packages for regression fitting are allowed! We are of course going to use numpy and matplotlib for computational and plotting purposes.

Our input file is now a different and a bigger one. It is a .csv file with various statistics taken from Wimbledon and Roland Garros grand slam matches (men category) from 2013. We are going to investigate the effects of “net points attempted” to “unforced errors done by the other player”, to see if going for the net affects the opposing player’s unforced error rate. In this case, our “x” is going to be the “NPA1” column in the .csv file, and our “y” is going to be the “UFE2” column.

Just like multiple linear regression, we need an input matrix. But this time, aside from our x, our other independent variables are going to be some function of x. See the image below for details:

$$
\begin{aligned}
b_1(x_i) &= x_i \\
b_2(x_i) &= x_i^2 \\
b_3(x_i) &= x_i^3 \\
b_{k+3}(x_i) &= (x_i - \xi_k)_+^3,
\end{aligned}
\quad \rightarrow \quad
\mathbf{X} = \begin{array}{c}
\begin{matrix} x_i & x_i^2 & & (x_i-\xi_k)^3_+ \\ \downarrow & \downarrow & & \downarrow \end{matrix} \\
\begin{bmatrix}
1 & x_{21} & x_{31} & \dots & x_{k1} \\
1 & x_{22} & x_{32} & \dots & x_{k2} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
1 & x_{2n} & x_{3n} & \dots & x_{kn}
\end{bmatrix}
\end{array}
$$

Where x is our input data, ξ_k is the knot value at the k^{th} knot, and k = 1,2,3,...,K. This means that in your input matrix X, you will have 3 + k + 1 columns (with an extra ones column).

Your task: Implement the cubic spline regression above in a function with three parameters: your x and y (which can be taken from the data sheet), and your knot values in a separate array. Inside this function, create your X matrix and calculate your coefficients using:

$$\hat{\beta} = [\mathbf{X}'\mathbf{X}]^{-1}\mathbf{X}'\mathbf{y}$$

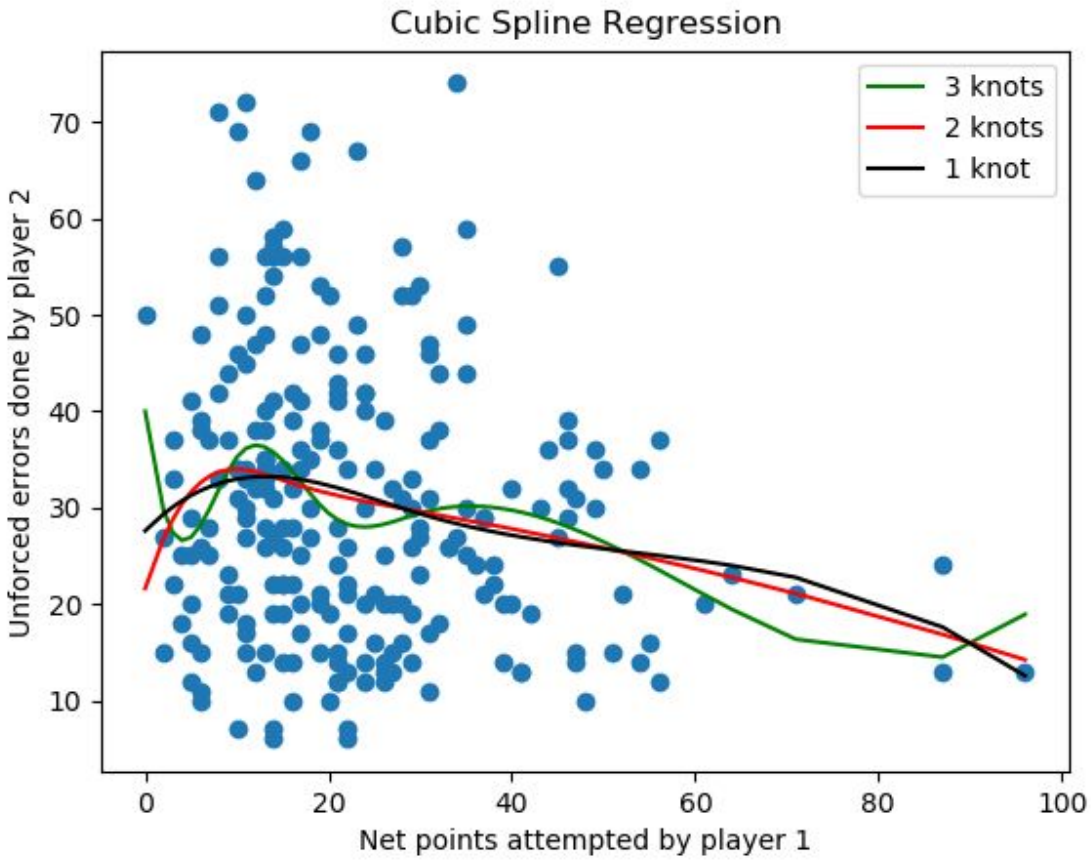
After calculating the coefficients, calculate the regression line as your prediction and return it. You can calculate the line using:

$$\hat{\mathbf{y}} = \mathbf{X}\hat{\beta}$$

In the main function, read your data sheet, extract your x and y vectors. Call the function you implemented three times, using the same x and y, but different knot values. The knot values you should pass to the function are:

1) 3 knots at 10, 20 and 30
2) 2 knots at 15 and 30
3) 1 knot at 30

Since we have a single independent variable x, we can easily plot our regression lines. Plot the data points as a scatter plot, then plot all of the three regression lines, all in different colours. You should get a result like this:



But please feel free to experiment with different knot values to see the the effects of knot placement!