ISC 5315
Applied Computational Science 1
Due on 11/3, 2023 (Thursday)

Smoothing Data with B-Splines

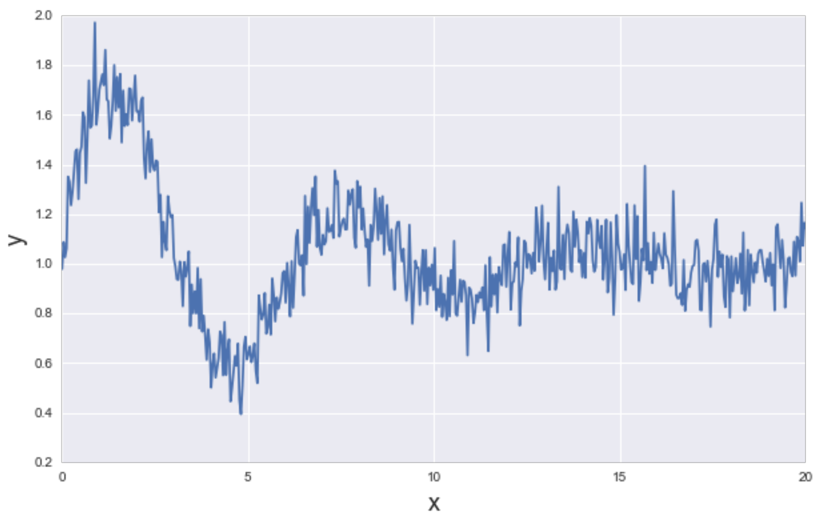
1 Goal

There are many ways to smoothen noisy data. These include techniques like moving averages, Kalman filters, kernel smoothers, local regression, Savitzky-Golay filters etc.

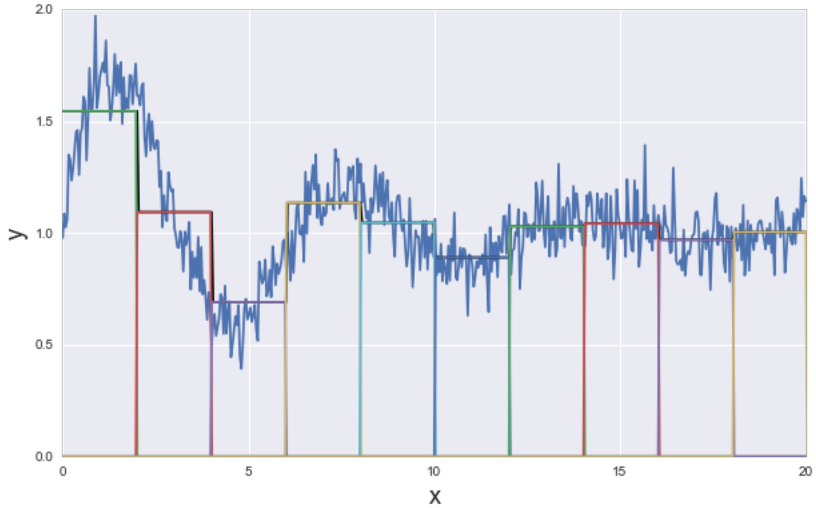
The goal of this lab is to approximate noisy data with smooth B-splines of different orders by solving the corresponding linear-least squares problem.

2 Motivation

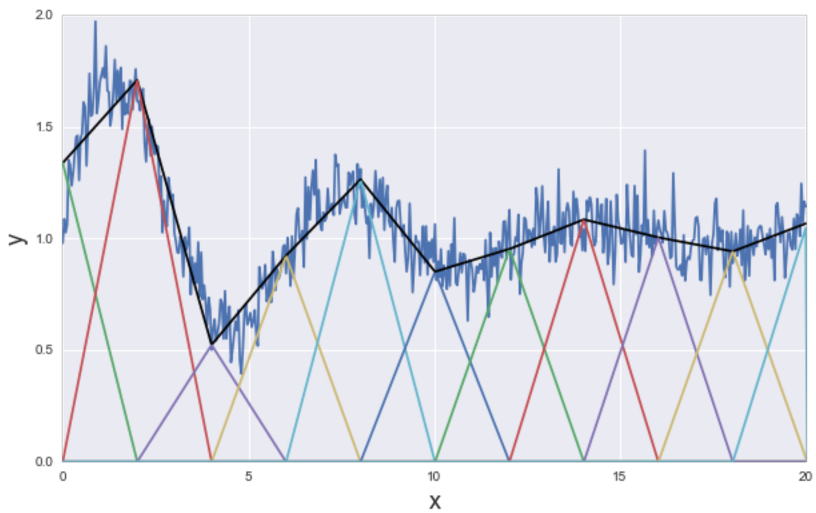
Suppose we have noisy data $y(x)$ that looks like this. You may download this dataset from the file xy.dat. The dependent variable x is equispaced on the domain [0, 20].

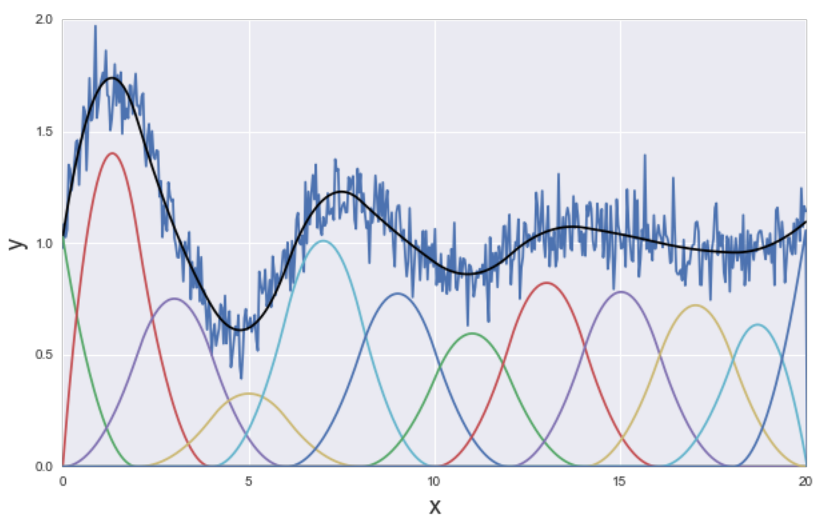


A first approximation may be to simply histogram the data; i.e., divide the domain into equal sized bins, and approximate the underlying function as its mean value in each of the bins.
For example, if we use 10 bins, we may get something like.



If we use higher order B-spline approximation (more on how to do it below), we could try to fit the data to B-splines of order 1 or 2.





3 Approach

The knot-vector will include values on the domain of the dependent variable, $x_0 < x_1 < \cdots < x_m$. Here I assume that the knots are not restricted to lie between 0 and 1. If your software package for B-splines is more restrictive, simply perform a linear transformation to map the data to [0, 1].

We will have to append repeated end-points, depending on the order p of the B-spline required.

Our objective is to find a fitting function

$$\hat{y}(x) = \sum_{i=0}^{n} Y_i N_{i,p}(x), \tag{1}$$

where $N_{i,p}$ are B-spline basis functions of order p, and the Y_i are the $n+1$ control points. Ideally, we would like to assert,

$$\hat{y}(x_k) = \sum_{i=0}^{n+1} Y_i N_{i,p}(x_k) = y_k, \quad k = 0, 1, ..., m, \tag{2}$$

where y_k are the actual noisy data values.

If you think about the problem a little bit, you realize what we really want to do is find the control points Y_i in eqn. 2.

Let us write eqn. 2 in matrix form:

$$\begin{bmatrix} N_{0,p}(x_0) & N_{1,p}(x_0) & \cdots & N_{n,p}(x_0) \\ N_{0,p}(x_1) & N_{1,p}(x_1) & \cdots & N_{n,p}(x_1) \\ & \ddots & & \\ N_{0,p}(x_m) & N_{1,p}(x_m) & \cdots & N_{n,p}(x_m) \end{bmatrix} \begin{bmatrix} Y_0 \\ Y_1 \\ \vdots \\ Y_n \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_m \end{bmatrix}$$

We can think of this as $AY = y$. Since the B-spline basis functions have local support, the matrix A is going to be banded, and have a lot of zeros. Its size is $(m+1) \times (n+1)$, with m presumably much greater than n.

The least-squares problems yields the normal equations $A^T AY = A^T y$.

A python B-spline package "bspline.py" has been upload on Canvas. Import it as "import bspline as bs". Read the description of "Bspline()" in that script to see how to use it.

4 Algorithm

1. Divide the domain $[x_{min}, x_{max}]$ into `nbin` bins of equal size
2. Choose the ends of each of these bins as knots
3. Depending on the order of the B-spline, p, pad the knot-vector with appropriate number of boundary knots.
4. Generate the basis B-splines using the software package
5. Build the matrix A and the vector y
6. Solve the least-squares problem.
7. Plot the data, the fit, and the basis functions as in the plots above

5 Tasks

1. Choose `nbin` $= 10$, and $p = 2$ and find the least-squares B-spline approximation to the data.
2. What is the condition number of the resulting matrix A? Use Cholesky decomposition and QR algorithms to solve the LLS problems.
3. Holding number of bins constant at 10, change the order from $p = 0$ to $p = 5$, and find the least squared error $||\hat{y}(x_k) - y_k||^2$, as a function of p.

4. Holding the order $p = 2$ constant, change the number of bins between 5 and 50, and and find the least squared error $||\hat{y}(x_k) - y_k||^2$, as a function of the number of bins.