Exercise 5 FYSA120 C++ numerical programming Winter 2015

Email the *commented* solution code (*.cpp) to :
address: `FYSY160(at)gmail.com` Subject-line: `demo5`
If you run into trouble, please send questions also to that address.

1. B-spline basis functions are described in Wikipedia (web link). There are B-spline libraries, but for teaching purposes I wrote the header file `numerics/Bspline.hpp`, which computes the basis functions using the deBoer recursive algorithm. Testing is done in `numerics/Bspline.cpp`.

   - Study the behaviour of the basis using `numerics/Bspline.cpp`. First choice is the *order k of the B-splines*, higher k means higher order smoothness. The "black magic" about B-splines is in the choice of the knots (also called breakpoints). The knots must be in ascending order. The more knots there are in a region, the more rapidly varying function can be represented. The continuity of the basis functions at a knot is controlled by *multiplicity* of a knot, that is, how many times a knot is repeated. Try $k = 3$ (cubic B-spline) and knots `(0,1,2,3,4,5,6,7,8,9,10)` `(0,1,2,2,2,3,4,5,6,7,8,9,10)`, and observe how the basis changes.

   - Write a program that generates noisy input data and stores it to two Armadillo `vec`'s,

```
// known points
const int N=100;
vec x(N);
vec y(N);
for(int i=0; i<N; ++i) {
   double xx=i*10.0/(N-1);
   x[i] = xx;
   auto noise=0.2*(rand()*1.0/RAND_MAX-0.5);
   y[i] = 3*exp(-abs(xx-4))+noise;
 }
```

     Do a least squares fit of the data to B-spline basis $\{B_j(x)\}$,

$$y_i(x_i) = \sum_j c_j B_j(x_i)$$

You get a linear set of equations for the coefficient vector $\mathbf{c}$,

$$A^T A\mathbf{c} = A^T \mathbf{y} \ ,$$

where the "model matrix" A has elements

$$A_{ij} = B_j(x_i) \ .$$

Thus, the model matrix stores the basis functions evaluated at the points of the input data.

The noise hides the kink (discontinuous first derivative) at $x = 4$ in the underlying function. Suppose you *know* from some physical insight that there must be a kink at $x = 4$. Adjust the B-spline knot vector so that the kink is built into the basis and do the fitting. The fact that you can set up the B-spline basis like this is one of their assets.

Set up A and solve the coefficients c_j using the Armadillo routine `solve` `(web link)`. Write the noisy data points to a file, and 1000 fitted values between 0 and 10. Plot them together, perhaps using gnuplot.