Statistics and Data Analysis

05.05.2017

Prof Jamie Tattersall and Frederic Poncza
(derived from Jan-Frederik Schulte)

SS 2017

Computer exercise: 05.05.2017, Hand in by 11.05.2017

Week 3

---

1. Interpolation of data (15 points)

Data is provided in the file `data.txt`

(a) Implement a linear interpolation routine and calculate the expected value of the function at,
(i) $x = 0.4$
(ii) $x = -0.128$
(iii) $x = -2.0$
(iv) $x = 3.2$

(b) Implement a cubic spline interpolation routine and calculate the expected value of the function at the same points. For the matrix inversion either write your own simple version or use an external routine (e.g. download the external solver `tridag` from the Numerical Recipes website).

(c) Using both interpolation routines, interpolate the data with a finely spaced grid and plot the results on the same set of axis.

(d) Investigate different boundary conditions for the cubic spline and explain what choice you think is best.

(e) Which interpolation routine appears to give the better solution and why?

2. Simple integration routines (10 points)

Implement the Trapezoidal Rule and Simpson's Rule with 5 sample points,

(a) Compute $\int_1^5 x dx$ with both rules and compare to the analytic result

(b) Compute $\int_1^5 x^2 dx$ with both rules and compare to the analytic result

(c) Which routine performs better and why?

3. *Numerical Integration using the Romberg procedure (optional problem)

The Romberg procedure allows for the numerical integration of a function $f(x)$ in an interval $[a, b]$ by Richardson extrapolation of the trapezoidal rule. The following terms were introduced in the lecture:

$$R(0,0) = \frac{1}{2}(b-a)(f(a) - f(b)), \tag{1}$$

$$R(n,0) = \frac{1}{2}R(n-1,0) + h_n \sum_{k=1}^{2^{n-1}} f(a + (2k-1)h_n), \tag{2}$$

$$R(n,m) = R(n,m-1) + \frac{1}{4^m - 1}(R(n,m-1) - R(n-1,m-1)), \tag{3}$$

where $1 \leq m \leq n$ and $h_n = \frac{b-a}{2^n}$. Integrate the gauss function with $\sigma = 1$ and $\mu = 0$ in the interval $[-1, 1]$ with the Romberg procedure with a precision of at least 10^{-7}. Use romberg.cc as a starting point. The calculation of $R(0,0)$ and $R(n,0)$ is already implemented, exemplifying the use of std::map in C++ to store the elements of R. Proceed as follows:

(a) Add a loop to calculate the elements $R(n,m)$.

(b) The result of the integration will be $R(n,m)$ for a sufficiently large n. Terminate the loop over n with a appropriate termination condition to reach the desired accuracy.