Numerische Methoden für komplexe Systeme, SoSe 2015

Dr. S. Gurevich

Exercises online: http://pauli.uni-muenster.de/tp/

Aufgabe 1: Wettbewerbsmodell

Die Leistungsfähigkeit pseudospektraler Methoden gründet auf der effizienten Implementierung der Fouriertransformation, der als *fast fourier transform* (FFT) bekannte Algorithmus skaliert mit $N \log N$. Subroutinen zur Berechnung der Fouriertransformation sind von vielen Anbietern erhältlich und in den meisten numerischen Bibliotheken implementiert. Eine frei verfügbare FFT, die auf vielen Systemen lauffähig ist und von `C`, `C++` und `Fortran` aus aufrufbar ist, ist unter `www.fftw.org` erhältlich.

a) Besuchen Sie die homepage und machen Sie sich mit den Eigenschaften dieser numerischen Bibliothek vertraut. Versuchen Sie insbesondere, folgende Fragen zu beantworten:

- Wie installiert man diese Bibliothek?
- Wie kompiliert man ein Programm, das auf in der Bibliothek enthaltenen Subroutinen zurückgreift?
- Was ist ein *plan* und wozu dient er?
- Was ist der Unterschied zwischen *in place* und *out of place* Transformationen?
- Welche verschiedenen Arten von Transformationen werden angeboten? Was ist beispielsweise der Vorteil einer *real to real* Transformation?

b) Versuchen Sie nun ein Beispielprogramm zu erstellen, welches einen eindimensionalen *array* (reell oder komplex) in den Fourierraum transformiert. Wie kann man in der Programmiersprache Ihrer Wahl mit komplexen Zahlen umgehen? Wie sind die Fourierkoeffizienten in dem transformierten *array* angeordnet? Was ist bezüglich der Normierung zu beachten? Überprüfen Sie durch Rücktransformation, ob Sie wieder die ursprünglichen Werte erhalten.

Tipp für Frustrierte: In der Dokumentation sind zahlreiche Beispiele zu finden.

Exercise 2: Numerical Derivative using FFT

A particularly simple method to calucate derivatives of functions or to apply a low or high pass filter on them is based on the FFT. To apply this method correctly, it is crucial to be familiar with the order of the fourier coefficients in the transformed *array*. Let f be an L-periodic function, which is evaluated at a discrete set of points resulting in an array $\{f_1, \dots f_N\}$. If f is complex, the transformed array is arranged according to the following ordering:

$$\tilde{f}_0, \tilde{f}_1, \dots, \tilde{f}_{\frac{N}{2}}, \tilde{f}_{-\frac{N}{2}+1}, \tilde{f}_{-\frac{N}{2}+2}, \dots, \tilde{f}_{-1}.$$

If the function f is real valued, the fourier coefficients obtain the symmetry $\tilde{f}_{-j} = \tilde{f}_j^*$. Therefore the array in the fourier space only needs to be half as large as in the complex case to carry the full information. In this case, the transformed array is ordered in the following way:

$$\tilde{f}_0, \tilde{f}_1, \dots, \tilde{f}_{\frac{N}{2}}.$$

Task: In order to calculate a derivate, every coefficient in the fourier space has to be multiplied by the respective wavenumber $k(j) = \frac{2\pi}{L} j$ and the imaginary unit i. Write a short program which initialises a periodic function in the real space, then performs a fourier transformation of this function and finally calculates the derivative in the fourier space. Using simple examples, make sure that after transforming the derivative back you obtain the correct result.