Computational Physics - Project 3

Johannes Scheller, Vincent Noculak, Lukas Powalla

October 18, 2015

Contents

1 Execution

1.1 Gaussian quadrature

Our first task was to calculate the integral in Cartesian coordinates by Gaussian quadrature using only Legendre polynomials. With this method, it is only possible to integrate a function on some finite interval $[a,b]$. Therefore, we had to replace the original integration limits $-\inf$ and $\inf$ by some appropriate values $-a$ and a. As it can be seen in fig. **??**, the one-dimensional wave function of only one particle gets more or less zero at $r=5$. Therefore, we decided to chose $[-5,5]$ as the interval for our first trials.

Our program reads in the desired number of grid points, n, and the desired integration limits a and b. After that, it starts an algorithm to calculate the weights w_i and zeros x_i of a Legendre polynomial of the desired degree n and to save them in arrays. This algorithm is taken from the source-code „exampleprogram.cpp"from the project folder for this course on Github. After calculating the weights and zeros, the algorithm performs a sextuple loop over the arrays and to sum up the weighted values of the function at those points and returns the result:

```
double legendre(int n, double a, double b){
double * x = new double [n];
double * w = new double [n];
gauleg(a, b, x, w, n);
double integral = 0;
for (int i=0; i<n; i++){
for (int j=0; j< n; j++){
for (int k=0; k<n; k++){
for (int l=0; l<n; l++){
for (int y=0; y<n; y++){
for (int z=0; z<n; z++){
        integral += (w[i] * w[j] * w[k] * w[l] * w[y]* w[z]
                                        * function_cartesian(x[i], x[j], x[k], x[l], x[y], x[z]));
}}}}}}
return integral;
delete x;
delete w;
```

The function *function_cartesian* simply returns the function value at a given point.

In tab. **??**, you can see the results of this method for some different values of n and some different intervals $[-a,a]$. It is very obvious that these results did not turn to be stable at all. Nevertheless, the computation time was already very high for $n=20$, as we had to perform roughly n^6 operations! Remember that the analytical value of the integral is $\frac{5\pi}{256}$.

To improve the results, we tried calculating the integral with Gauss-Laguerre quadrature. The standard integration limits of Laguerre polynomials are $[0,\inf]$ and the polynomials are suited for functions of the form $x^{\alpha}\cdot e^{-x}$. This fits perfectly to our case when we change to spherical coordinates:
The integral

$$\int_{-\infty}^{\infty} d\mathbf{r_1} d\mathbf{r_2} e^{-4(r_1+r_2)} \frac{1}{|\mathbf{r_1}-\mathbf{r_2}|} \tag{1}$$

can now be written as

$$\int_{-\pi}^{\pi}\left(\int_{-\pi}^{\pi}\left(\int_{0}^{\pi}\left(\int_{0}^{\pi}\left(\int_{0}^{\infty}\left(\int_{0}^{\infty}\left(r_1^2 r_2^2 \sin(\theta_1)\sin(\theta_2) e^{-4(r_1+r_2)}\frac{1}{|\mathbf{r_1}-\mathbf{r_2}|}\right)dr_1\right)dr_2\right)d\theta_1\right)d\theta_2\right)d\phi_1\right)d\phi_2 \tag{2}$$

which perfectly fits to the form of Laguerre polynomials if we substitute $r_i' = 4r_i$. Note that we have to change the Jacobian accordingly! Nowe we can solve the integral by applying Gauss-Laguerre quadrature for the two integrals over r_i' and by using Gauss-Legendre for the integrals over the angles.

Our algorithm again sets up the weights and zeros for both methods in arrays by using algorithms from the course folder on Github.

2 Comparison and discussion of the results

3 Source-code