UNIVERSITY OF OSLO

COMPUTATIONAL PHYSICS

Project 3

Authors:

Birgitte Madsen

Magnus Isaksen

Soumya Chalakkal

Autumn 2015

UiO : University of Oslo

Department of Physics
University of Oslo
Sem Sælands vei 24
0371 Oslo, Norway
+47 22 85 64 28
http://www.mn.uio.no/fysikk/english/

Course:

Computational Physics

Project number:

3

Link to GitHub folder:

`https://??`

Hand-in deadline:

Friday, October 23, 2015

Project Members:

Birgitte Madsen
Magnus Isaksen
Soumya Chalakkal

Copies: 1
Page count: ??
Appendices: 0
Completed: ??

TABLE OF CONTENTS

CHAPTER

1 INTRODUCTION

CHAPTER

2 METHOD

2.1 Nature of the problem

2.2 Gauss-Legendre Method for Computing the Integral

To be able to use the Gauss-Legendre method to compute the integral in [1], the limits of the integral must be made finite. Since the wave function

$$e^{-2\alpha x} \tag{2.1}$$

rapidly goes toward zero as x is increased (see fig), the integral can be approximated by the same integral with finite limits.

In this project, we [2] have accepted that 10^{-9} is close enough to zero to neglect contributions from the part of the wave function when the wave function gives a value of this order. For $x = 5$ the value of the wave function is $e^{-10\alpha} \approx 2.1 \cdot 10^{-9}$, when $\alpha = 2$. Hence, the considered integral that is to be solved by the Gauss-Legendre method is given by

$$\left\langle \frac{1}{|\mathbf{r}_1 - \mathbf{r}_2|} \right\rangle = \int_{-5}^{5} \frac{e^{-2\alpha x}}{|\mathbf{r}_1 - \mathbf{r}_2|} d\mathbf{r}_1 d\mathbf{r}_2 \tag{2.2}$$

To be able to use the Gauss-Legendre method, the limits have to be -1 and 1. This is, however, easily obtained by a change in variables using the following quantity

$$\int_a^b f(t)dt = \int_{-1}^{1} f\left(\frac{b-a}{2}x + \frac{b+a}{2}\right) dx \tag{2.3}$$

The first step of the Gauss-Legendre method is then to compute the roots of the n'th Legendre polynomial. The i'th root z_i is approximated by

$$z_i = \cos\left(\frac{\pi(4 \cdot i - 1)}{4n+2}\right) \tag{2.4}$$

[1] FiXme Note: ref. to problem eq
[2] FiXme Note: sorry about the "we"

for large n. The n'th [3] Legendre polynomial $L_n(x)$ can then be computed using the recursive relation

$$(j+1)L_{j+1}(z) + jL_{j-1}(x) - (2j+1)zL_j(z) = 0 \tag{2.5}$$

with $L_{-1}(x) = 0$ and $L_0(x) = 1$. The code for generation the Legendre polynomial can be found in sec ?? [4] together with a test of the Legendre polynomial generating algorithm.

The derivative of the Legendre polynomial can then be computed as

$$L'_n(z) = \frac{-n \cdot z \cdot L_n(z) + n \cdot L_{n-1}(z)}{1 - z^2} \tag{2.6}$$

Newton's method is then applied to find the best estimation of the roots by considering the expression

$$z_1 = z - \frac{L_n(x)}{L'_n(x)} \tag{2.7}$$

in which z is the first approximation of the root given by Eq. (2.4), and z_1 is a better approximation for the root. The algorithm for generation of the roots and the Legendre polynomials is run until the difference between z_1 and the root approximated by Eq. (2.4) is ultimately zero, which in this project is set to 10^{-6}.

????? write how we get weights ??????

2.3 Generation of Legendre Polynomials

???? Write smt about the following lines of code ?????

```
//Code for computing the Legendre polynomials to determine the roots of the n'th
    Legendre polynomial and the weights of the roots
    for (int i = 1; i<m;i++){
        root = cos(pi * (4*i-1) / (4*n + 2 ));
        //approximation of the root of the n'th polynomial
    do{
        // This uses the recursive relation to compute the Legendre polynomials
        double L_plus = 1.0 , L = 0.0, L_minus;
        for (int j = 0; j < n; j++)
        {
        L_minus = L;
        L = L_plus;
        L_plus = (2.0*j +1)*root*L - j*L_minus ;
        L_plus /= j+1;
        }
      //derivative of the Legendre polynomial (L_plus)
        dev_L = (-n*root*L_plus + n*L)/(1-root*root);
        // Newton's method
        z = root;
        root  = z - L_plus/dev_L;
     } while(fabs(root - z) > pow(10,-6));
```

[3] FiXme Note: right??
[4] FiXme Note: ref to sec

CHAPTER

3 CONCLUSION

Conclude.... conclude.... conclude....

APPENDIX

A MATLAB CODE FOR SMT....

This is how, we write MatLab code in the report

```
close all
clear all
clc
%I am a comment

filename = 'Results.xlsx';
sheet = 4;
xlRange = 'B3:C12';

[v,T,vT] = xlsread(filename, sheet, xlRange);
x10=v(:,1);y10=v(:,2);

figure
plot(??)
legend(??)

xlim([??])
ylim([??])

title(??)
xlabel('x')
ylabel('y')
```