UiO : Faculty of Mathematics and Natural Sciences
University of Oslo

Computational Physics

FYS4150

Project 1

Introduction to numerical projects

Vegard Rønning
Materials, Energy and Nanotechnology
University of Oslo
Norway
Fall 2016

1 Introduction

In this project we will solve the one-dimensional Poissson equation with Dirichlet boundary conditions by rewriting it as a set of linear equations.

The approximation of the second derivative can be written as

$$-\frac{v_{i+1} + v_{i-1} - 2v_i}{h^2} = f_i$$

2 Method

The approximation of the second derivative can be written as a set of linear equations of the form

$$\mathbf{Av} = \widetilde{\mathbf{b_i}}$$

so that

$$\mathbf{A} = \begin{pmatrix} 2 & -1 & 0 & & & \\ -1 & 2 & -1 & 0 & & \\ 0 & -1 & 2 & -1 & & \\ & 0 & \ddots & \ddots & \ddots & 0 \\ & & & -1 & 2 & -1 \\ & & & 0 & -1 & 2 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ \\ \\ v_n \end{pmatrix} = \begin{pmatrix} \widetilde{b_1} \\ \widetilde{b_2} \\ \widetilde{b_3} \\ \\ \\ \widetilde{b_i} \end{pmatrix}, \qquad \text{where} \qquad \widetilde{b_i} = h^2 f_i$$

this result in a set of linear equations as

$$\begin{array}{ll} b_1 v_1 + c_1 v_2 + 0 + \ldots + 0 & = \widetilde{b_1} \\ a_1 u_1 + b_2 v_2 + c_2 v_3 + 0 + \ldots + 0 & = \widetilde{b_2} \\ 0 + a_2 u_2 + b_3 v_3 + c_3 v_4 + 0 + \ldots + 0 & = \widetilde{b_3} \\ \vdots & \vdots \\ 0 + \ldots + 0 + a_{(n-2)} v_{(n-2)} + b_{(n-1)} v_{(n-1)} + c_{(n-1)} v_n & = \widetilde{b}_{(n-1)} \\ 0 + \ldots + 0 + a_{(n-1)} v_n + b_n v_n & = \widetilde{b_n} \end{array}$$

2.1 Gaussian Elimination

If one look at the set of linear equations

$$\mathbf{Av} = \mathbf{f}$$

and $\mathbf{A} \in \mathbb{R}^{\mathbf{4x4}}$, can be written as:

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{pmatrix}$$

or

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + a_{14}x_4 &= f_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + a_{24}x_4 &= f_2 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 + a_{34}x_4 &= f_3 \\ a_{41}x_1 + a_{42}x_2 + a_{43}x_3 + a_{44}x_4 &= f_4 \end{aligned}$$

Gaussian elimination utilizes that one can find the value of the first unknown in a set of linear equations, and then use this to eliminate the first unknown next equation, the the two first for the next equation, and so on. By doing so one end up with a upper triangular matrix of the form

$$
\begin{aligned}
b_{11}x_1 + b_{12}x_2 + b_{13}x_3 + b_{14}x_4 &= \widetilde{f}_1 \\
b_{22}x_2 + b_{23}x_3 + b_{24}x_4 &= \widetilde{f}_2 \\
b_{33}x_3 + b_{34}x_4 &= \widetilde{f}_3 \\
b_{44}x_4 &= \widetilde{f}_4
\end{aligned}
$$

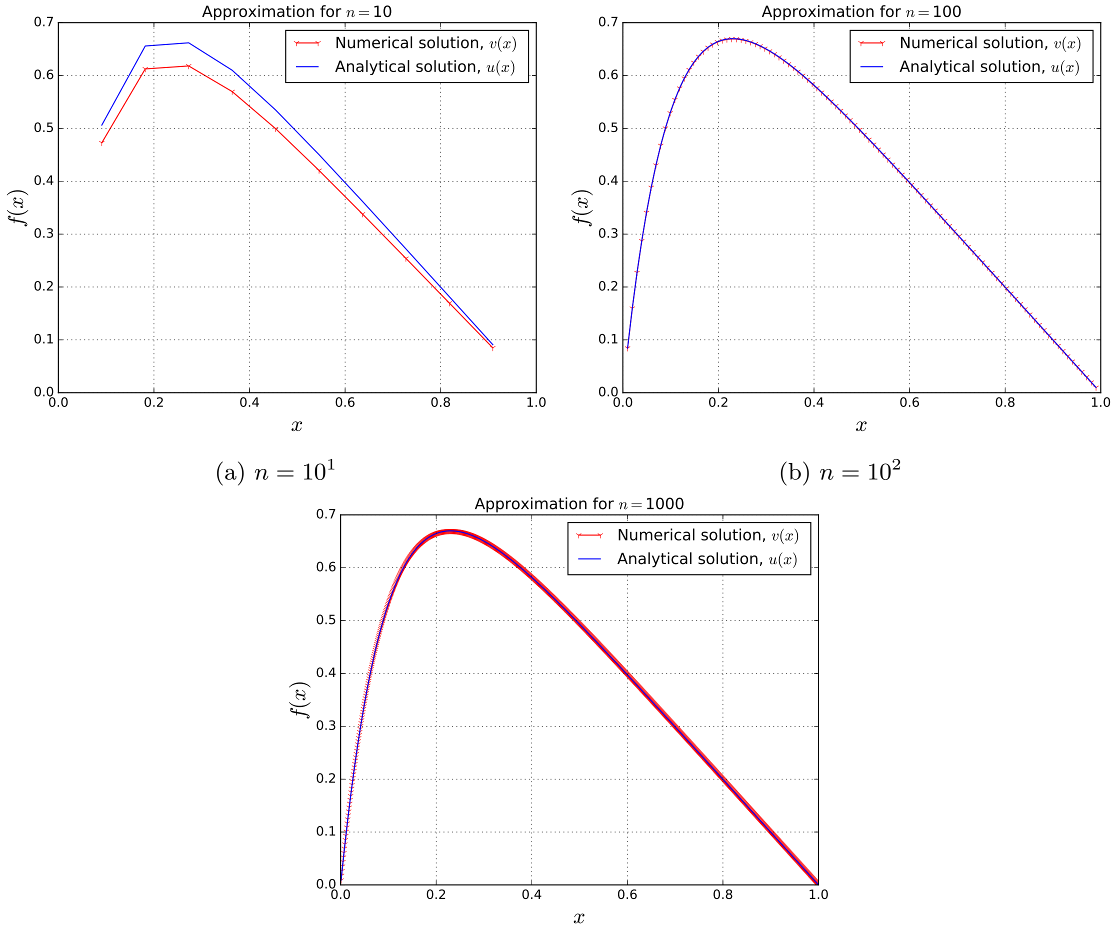
3 Implementation

Programs can be found at:

`https://github.com/vegro90/ComFysProject1`

4 Results

Number of flops for Gaussian elimination:
Number of flops for special case:



(a) $n = 10^1$

(b) $n = 10^2$

(c) $n = 10^3$

Figure 1: Numeric- vs analytic solution of (nxn)-matrix where:

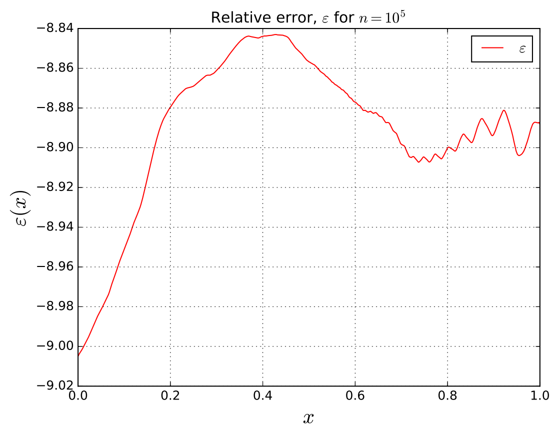
CPU-times

n	General, (s)	Special, (s)
10^1	4.900000E-05	1.000000E-06
10^2	7.000000E-06	3.000000E-06
10^3	6.800000E-05	2.900000E-05
10^4	0.0005120000	0.0004050000
10^5	0.004333000	0.002822000
10^6	0.04849100	0.03443600
10^7	0.5627300	0.3366240

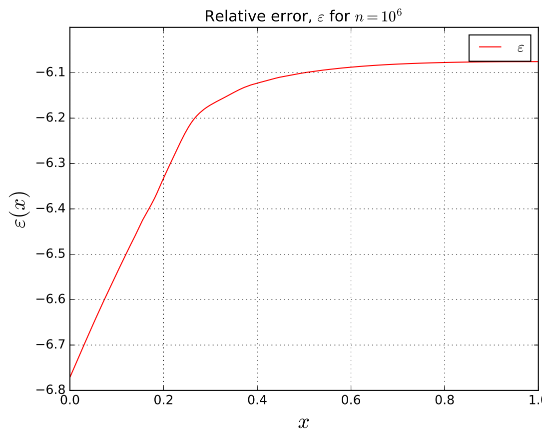
Table 1: CPU time used solving general- and special algorithm for (nxn) matrix

n	relative error
10^1	-1.1796978
10^2	-3.0880368
10^3	-5.0800516
10^4	-7.0792853
10^5	-9.0048965
10^6	-6.7713588
10^7	-13.007004

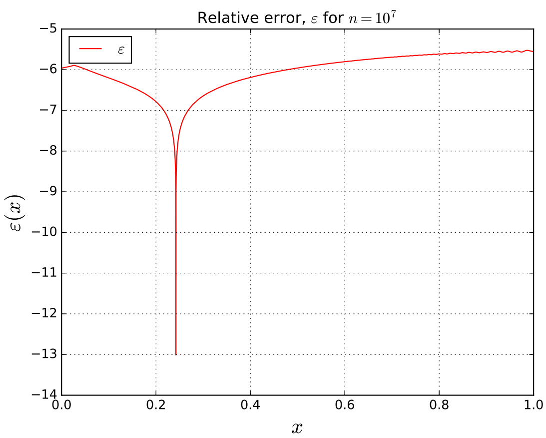
Table 2: Largest relative error, ε between analytic- and numeric solution generated from general algorithm



(a) $n = 10^5$



(b) $n = 10^6$



(c) $n = 10^7$

Figure 2: Plot of relative error, $\varepsilon(x)$ for:

5 Conclusion