FYS3150 - Project 1

Sigurd Sørlie Rustad and Vegard Falmår
(Dated: 9. september 2020)

Abstract om du vil

I. INTRODUCTION

We can describe the evolution of many physical systems with the help of differential equations, but because of their complexity we are unable to find the analytical solutions. Therefore we have to use numerical methods in order to approximate the solution. Computers are limited in both their memory and accuracy, so we have to be careful when both selecting the numerical method and how we implement it. In this report we are going to explore these issues by trying to solve the one-dimensional Poisson equation with Dirichlet boundary conditions, described in equation 3 in the theory section.

The example function we are going to use in our studies is described by the equation (1) and has an analytical solution (2) we can use to compare our results.

$$-\frac{d^2u}{dx^2} = f(x) = 100e^{-10x} \tag{1}$$

$$u(x) = 1 - (1 - e^{-10})x - e^{-10x} \tag{2}$$

In order to solve equation (1) we end up with a set of linear equations described by a tridiagonal matrix multiplied with a vector (see the theory section for further explanation). Now there are many ways we can solve this set of equations, each with their own pros and cons. The methods we are going to explore is one where we solve for a general tridiagonal matrix, one where we specialize the algorithm to our tridiagonal matrix and lastly by using LU decomposition. For each method we will study the accuracy, cpu-time used and the number of floating-point operations (FLOPS).

II. THEORY

A. Matrix formulation of the discrete one-dimensional Poisson equation

The one-dimensional Poisson equation with Dirichlet boundary conditions is given by equation 3.

$$-\frac{\mathrm{d}^2u(x)}{\mathrm{d}x^2} = f(x), \quad x \in (0, 1), \quad u(0) = u(1) = 0 \tag{3}$$

We defnine the discretized approximation to u to be v_i at points $x_i = ih$ evenly spaced between $x_0 = 0$ and $x_{n+1} = 1$. The step length between the points is $h = 1/(n+1)$. The boundary conditions from equation 3 then give $v_0 = v_{n+1} = 0$. An approximation to the second derivative of u, derived from the Taylor expansion, is

$$\frac{-v_{i-1} + 2v_i - v_{i+1}}{h^2} = f_i \quad fori = 1, 2, ..., n \tag{4}$$

where $f_i = f(x_i)$.

Written out for all i, equation 4 becomes

$$\begin{aligned} -v_0 + 2v_1 - v_2 &= h^2 f_1 \\ -v_1 + 2v_2 - v_3 &= h^2 f_2 \\ &... \\ -v_{n-2} + 2v_{n-1} - v_n &= h^2 f_{n-1} \\ -v_{n-1} + 2v_n - v_{n+1} &= h^2 f_n \end{aligned}$$

In general, this can be rearranged slightly so that

$$\begin{aligned} 2v_1 - v_2 &= h^2 f_1 + v_0 \\ -v_1 + 2v_2 - v_3 &= h^2 f_2 \\ &... \\ -v_{n-2} + 2v_{n-1} - v_n &= h^2 f_{n-1} \\ -v_{n-1} + 2v_n &= h^2 f_n + v_{n+1} \end{aligned}$$

This system of equations can be written in matrix form as

$$\boldsymbol{A}\boldsymbol{v} = \tilde{\boldsymbol{b}}, \tag{5}$$

explicitly

$$\begin{bmatrix} 2 & -1 & 0 & 0 & ... & 0 \\ -1 & 2 & -1 & 0 & ... & 0 \\ . & & & & & \\ . & & & & & \\ . & & & & & \\ 0 & ... & 0 & -1 & 2 & -1 \\ 0 & ... & 0 & 0 & -1 & 2 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ . \\ . \\ . \\ v_{n-1} \\ v_n \end{bmatrix} = \begin{bmatrix} h^2 f_1 + v_0 \\ h^2 f_2 \\ . \\ . \\ . \\ h^2 f_{n-1} \\ h^2 f_n + v_{n+1} \end{bmatrix}$$

With $v_0 = v_{n+1} = 0$, the right side reduces to $\tilde{b}_i = h^2 f_i$.

B. Solve tridiagonal matrix equation

In order to solve the tridiagonal matrix below we need to develop an algorithm. As mentioned in the exercise set [1] we first need to do a decomposition and forward substitution.

$$\mathbf{Av} = \begin{bmatrix} b_1 & c_1 & 0 & \dots & \dots & \dots \\ a_1 & b_2 & c_2 & \dots & \dots & \dots \\ & a_2 & b_3 & c_3 & \dots & \dots \\ & \dots & \dots & \dots & \dots & \dots \\ & & & a_{n-2} & b_{n-1} & c_{n-1} \\ & & & & a_{n-1} & b_n \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ \dots \\ \dots \\ \dots \\ v_n \end{bmatrix} = \begin{bmatrix} \tilde{b}_1 \\ \tilde{b}_2 \\ \dots \\ \dots \\ \dots \\ \tilde{b}_n \end{bmatrix}.$$

Looking at the first matrix multiplication we get the following expression.

$$b_1 v_1 + c_1 v_2 = \tilde{b} \implies v_1 + \alpha_1 v_2 = \rho_1, \quad \alpha_1 = \frac{c_1}{b_1} \wedge \rho_1 = \frac{\tilde{b}_1}{b_1} \tag{6}$$

Doing the second matrix multiplication we get

$$a_1 v_1 + b_2 v_2 + c_2 v_3 = \tilde{b}_2 \tag{7}$$

If we multiply equation 6 by a_1, and subtract it from equation 7 the resulting expression becomes

$$\begin{aligned} &(b_2 - \alpha_1 a_1) v_2 + c_2 v_3 = \tilde{b}_2 - \rho_1 a_1 \\ &\implies v_2 + \frac{c_2}{b_2 - \alpha_1 a_1} v_3 = \frac{\tilde{b}_2 - \rho_1 a_1}{b_2 - \alpha_1 a_1} \\ &\implies v_2 + \alpha_2 v_3 = \rho_2 \\ &\text{where} \quad \alpha_2 = \frac{c_2}{b_2 - \alpha_1 a_1} \wedge \rho_2 = \frac{\tilde{b}_2 - \rho_1 a_1}{b_2 - \alpha_1 a_1} \end{aligned}$$

Noticing the pattern in ρ and α we can generalize the terms.

$$\alpha_i = \frac{c_i}{b_i - \alpha_{i-1} a_{i-1}} \quad \text{for} \quad i = 2, 3, ..., n-1 \tag{8}$$

$$\rho_i = \frac{\tilde{b}_i - \rho_{i-1} a_{i-1}}{b_i - \alpha_{i-1} a_{i-1}} \quad \text{for} \quad i = 2, 3, ..., n \tag{9}$$

Inserting the terms into the matrix above, we get a much simpler set of equations.

$$\mathbf{Av} = \begin{bmatrix} 1 & \alpha_1 & 0 & \dots & \dots & \dots \\ 0 & 1 & \alpha_2 & \dots & \dots & \dots \\ & 0 & 1 & \alpha_3 & \dots & \dots \\ & \dots & \dots & \dots & \dots & \dots \\ & & & 0 & 1 & \alpha_{n-1} \\ & & & & 0 & 1 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ \dots \\ \dots \\ \dots \\ v_n \end{bmatrix} = \begin{bmatrix} \rho_1 \\ \rho_2 \\ \dots \\ \dots \\ \dots \\ \rho_n \end{bmatrix}.$$

Now the last step is to do a backward substitution. Starting with $v_n = \rho_n$ we can work our way backward, with the general expression

$$v_{i-1} = \rho_{i-1} - \alpha_{i-1} v_i \quad \text{for} \quad i = n, n-1, ..., 2 \tag{10}$$

Now in this report we are going to consider a matrix with elements $b_n = 2$ and $a_n = c_n = -1$. We can insert this into equations (8) and (9) to get the expressions (11) and (12).

$$\alpha_i = \frac{-1}{2 + \alpha_{i-1}} \tag{11}$$

$$\rho_i = \frac{\tilde{b} + \rho_{i-1}}{2 + \alpha_{i-1}} \tag{12}$$

C. Lower-upper (LU) decomposition

LU decomposition is a method where you factorize a matrix A into two matrices L and U, where L is lower triangular and U is upper triangular. We can use this decomposition to solve a matrix equation.

$$A\mathbf{x} = LU\mathbf{x} = \mathbf{b}$$

Where A and $\mathbf{b}$ is known. First we can solve $L\mathbf{y} = \mathbf{b}$ with the algorithm given by equation (13) (see [2]).

$$y_i = \frac{1}{l_{ii}} \left(b_i - \sum_{j=1}^{i-1} l_{ij} y_j \right), \quad y_1 = \frac{b_1}{l_{11}} \tag{13}$$

Where y_i is element i in $\mathbf{y}$, l_{ij} element ij in matrix L and b_i element i in $\mathbf{b}$. We can then solve $U\mathbf{x} = \mathbf{y}$ with the algorithm from equation (14) (see [2]).

$$x_i = \frac{1}{u_{ii}} \left(y_i - \sum_{j=i+1}^{N} u_{ij} x_j \right), \quad x_N = \frac{y_N}{u_{NN}} \tag{14}$$

Here x_i is element i in $\mathbf{x}$.

D. Floating-point operations

Whenever you do a mathematical operation on a computer you call it a floating-point operation. This includes division, multiplication, subtraction and addition. Floating-point operations (what we from now on will refer to as FLOPS) is a count of the total number of mathematical operations needed for an algorithm.

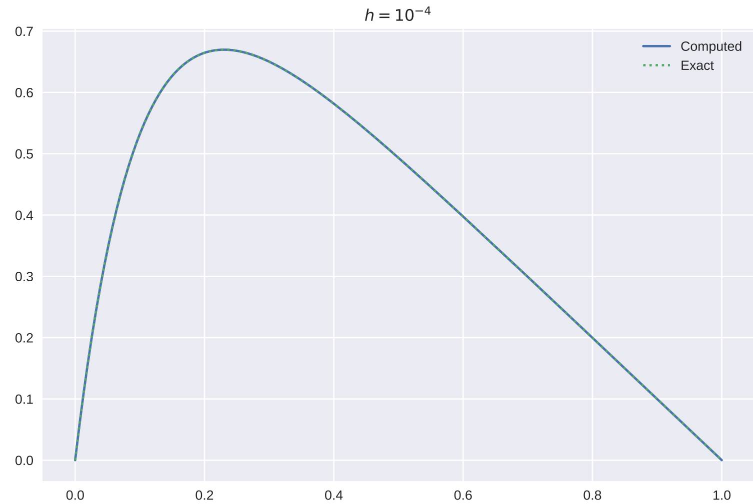
III. METHOD

First numerical test

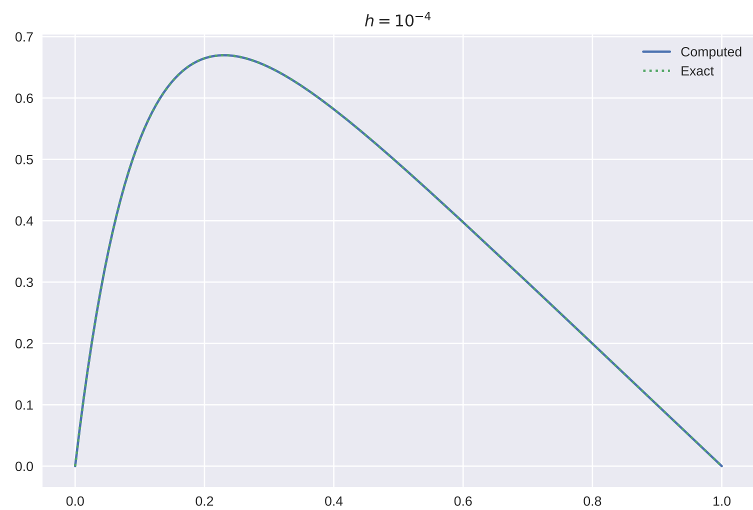
IV. RESULTS

V. DISCUSSION

VI. CONCLUTION



Figur 1. This figure shows the numeric solution for our general algorithm and the exact solution on top (green dashed line). The step size used is $h = 10^{-4}$.

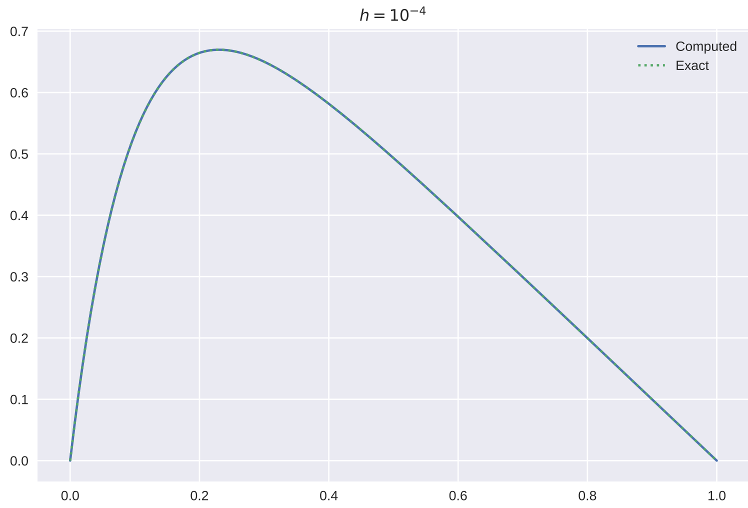


Figur 2. This figure shows the numeric solution for our specific algorithm and the exact solution on top (green dashed line). The step size used is $h = 10^{-4}$.

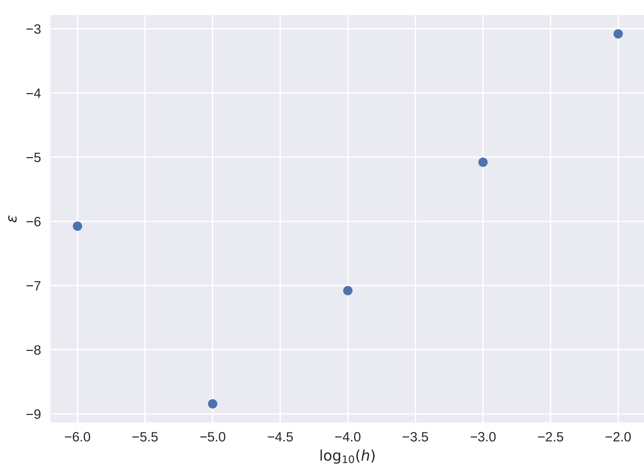
[1] HER REFERERERERERER VI TIL OPPGAVETEKSTEN
[2] HER REFERERERERERR VI TIL LU (https://mathworld.wolfram.com/LUDecomposition.html)

VII. APPENDIX

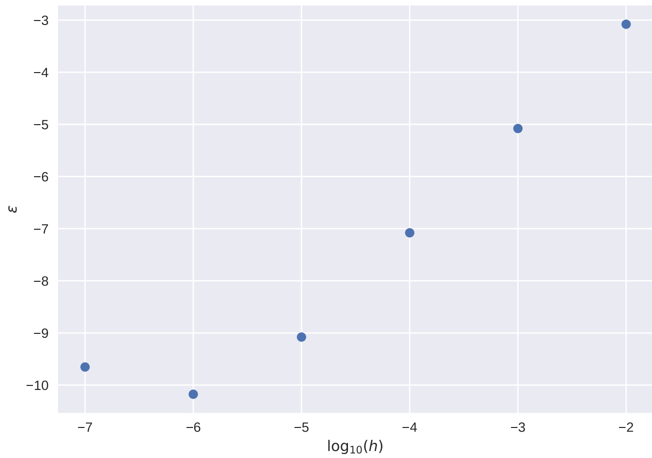
VIII. KODE



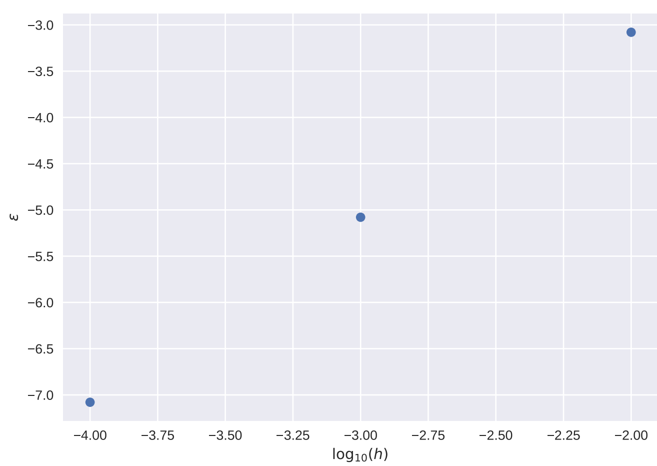
Figur 3. This figure shows the numeric solution for the LU-algorithm and the exact solution on top (green dashed line). The step size used is $h = 10^{-4}$.



Figur 4.



Figur 5.



Figur 6.