Project 1 FYS 4150

Kjetil Karlsen, Vilde Mari Reinertsen, Sigbjørn Lundesgaard Foss

September 11, 2017

Abstract

1 Introduction

2 Theory

In this project we will solve the general one-dimensional Poisson's equation (Eq. 1) for $f(x) = 100e^{-10x}$ and $x \in [0,1]$. The boundary conditions are Dirichlet bondary conditions. That means that $u(0) = u(1) = 0$.

2.1 Rewriting the problem to a matrix problem

$$-u''(x) = f(x) \tag{1}$$

We rewrite the second derivative by switching from the continuous $u(x)$ to n discrete number of points $v(x_i) = v_i$ and then using a Taylor expansion to obtain the new Poisson equation (Eq. 2).

$$-\frac{v_{i+1} + v_{i-1} - 2v_i}{h^2} = f_i \qquad \text{for } i = 1, \ldots, n, \tag{2}$$

where $f_i = f(x_i)$.

The next step is to rewrite Eq. 2 as a linear set of equations of the form of Eq. 3.

$$\mathbf{Av} = \mathbf{b}, \tag{3}$$

where $\mathbf{A}$ is an $n \times n$ tridiagonal matrix:

$$\mathbf{A} = \begin{bmatrix} 2 & -1 & 0 & \cdots & 0 & 0 \\ -1 & 2 & -1 & 0 & \cdots & 0 \\ 0 & -1 & 2 & -1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & -1 & 2 & -1 \\ 0 & 0 & \cdots & 0 & -1 & 2 \end{bmatrix},$$

and $b_i = h^2 f_i$.

Writing out the equations gives us:

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

where $v_0 = 0$ and $v_{n+1} = 0$.

Adding some zeros to the equations for illustrative purposes:

$$\begin{aligned} i = 1 &: \quad 2v_1 - v_2 + 0 + 0 + 0 + 0 + \cdots + 0 = f_1 h^2 \\ i = 2 &: \quad -v_1 + 2v_2 - v_3 + 0 + 0 + \cdots + 0 = f_2 h^2 \\ i = 3 &: \quad 0 - v_2 + 2v_3 - v_4 + 0 + \cdots + 0 = f_3 h^2 \\ &\vdots \qquad\qquad\qquad\qquad \vdots \\ i = n-1 &: \quad 0 + \cdots + 0 - v_{n-2} + 2v_{n-1} - v_n = f_{n-1} h^2 \\ i = n &: \quad 0 + \cdots + 0 + 0 + 0 - v_{n-1} + 2v_n = f_n h^2 \end{aligned}$$

We can now recognize the equations as a product between a matrix and a vector that equals another vector.

We define vectors that fits like this:

$$\mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} \qquad \tilde{\mathbf{b}} = \begin{bmatrix} f_1 h^2 \\ f_2 h^2 \\ \vdots \\ f_n h^2 \end{bmatrix}$$

The matrix is the matrix $\mathbf{A}$. Then we have:

$$\begin{bmatrix} 2 & -1 & 0 & \cdots & 0 & 0 \\ -1 & 2 & -1 & 0 & \cdots & 0 \\ 0 & -1 & 2 & -1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & -1 & 2 & -1 \\ 0 & 0 & \cdots & 0 & -1 & 2 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ \vdots \\ v_{n-1} \\ v_n \end{bmatrix} = \begin{bmatrix} f_1 h^2 \\ f_2 h^2 \\ f_3 h^2 \\ \vdots \\ f_{n-1} h^2 \\ f_n h^2 \end{bmatrix} \implies \mathbf{Av} = \mathbf{b}$$

2.2 An analytical solution

The Poisson equation (Eq. 1) with our function $f(x) = 100e^{-10x}$ has an analytical solution (Eq. 4).

$$u(x) = 1 - \left(1 - e^{-10}\right) x - e^{-10x} \tag{4}$$

That can be shown by putting it into the Poisson equation (Eq. 1). Then we first have to find the second derivative:

$$\begin{aligned} u(x) &= 1 - \left(1 - e^{-10}\right) x - e^{-10x} \\ u'(x) &= \left(1 - e^{-10}\right) + 10e^{-10x} \\ u''(x) &= -100e^{-10x} \end{aligned}$$

Second, we put the second derivative into Eq. 1:

$$-u''(x) = -(-100)e^{-10x} = 100e^{-10x} = f(x)$$

3 Method

- algorithm
- Theoretical models and technicalities

3.1 Algorithm

3.2 The general tridiagonal matrix

The algorithm is first written for an equation with a general tridiagonal matrix on the short form:

$$\mathbf{Av} = \mathbf{b}$$

The full form looks like this:

$$\begin{bmatrix} d_1 & e_1 & 0 & \cdots & 0 & 0 \\ c_1 & d_2 & e_2 & 0 & \cdots & 0 \\ 0 & c_2 & d_3 & e_3 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & c_{n-2} & d_{n-1} & e_{n-1} \\ 0 & 0 & \cdots & 0 & c_{n-1} & d_n \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ \vdots \\ v_{n-1} \\ v_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_{n-1} \\ b_n \end{bmatrix}$$

The method used to solve the equation numerically was to first to a forward substitution to obtain a upper triangular matrix and then a backwards substitution to obtain the solution.

The algorithm preformed on the matrix values was:

The forward substitution:
The values on the diagonal:

$$\tilde{d}_i = d_i - \left(\frac{c_i \cdot e_i}{\tilde{d}_{i-1}} \right)$$

The values on the right side of the equation:

$$\tilde{b}_i = b_i - \left(\frac{\tilde{b}_{i-1} \cdot e_i}{\tilde{d}_{i-1}} \right)$$

for $i = 2, \ldots, n-2$

The backwards substitution:

$$v_i = \frac{\left(\tilde{b}_i + v_{i+1} \right)}{\tilde{d}_i}$$

for $i = 2, \ldots, n-2$. The boundary values are known ($v_0 = v_n = 0$). v_{n-1} was calculated first and like this:

$$v_{n-1} = \frac{\tilde{b}_{n-1}}{\tilde{d}_{n-1}}$$

3.3 This specific tridiagonal matrix

The matrix in our case had the same values on the different diagonals:

$$d_i = 2 \qquad e_i = -1 \qquad c_i = -1$$

We can then pre-calculate some of the operations in the algorithm and if becomes then:

The forward substitution:
The values on the diagonal:

$$\tilde{d}_i = d_i - \left(\frac{c_i \cdot e_i}{\tilde{d}_{i-1}} \right) = 2 - \left(\frac{(-1) \cdot (-1)}{\tilde{d}_{i-1}} \right) =$$

$$2 - \frac{1}{\tilde{d}_{i-1}} = \frac{i+1}{i}$$

The last expression can be shown to be true, by calculating some of the first values:

$$\tilde{d}_2 = 2 - \frac{1}{2} = \frac{3}{2} = \frac{2+1}{2}$$

$$\tilde{d}_3 = 2 - \frac{1}{\frac{3}{2}} = 2 - \frac{2}{3} = \frac{12}{6} - \frac{4}{6} = \frac{8}{6} = \frac{4}{3} = \frac{3+1}{3}$$

The values on the right side of the equation:

$$\tilde{b}_i = b_i - \left(\frac{\tilde{b}_{i-1} \cdot e_i}{\tilde{d}_{i-1}} \right) = b_i - \left(\frac{\tilde{b}_{i-1} \cdot (-1)}{\tilde{d}_{i-1}} \right) = b_i + \left(\frac{\tilde{b}_{i-1}}{\tilde{d}_{i-1}} \right)$$

for $i = 2, \ldots, n-2$.

The backwards substitution:

$$v_i = \frac{\left(\tilde{b}_i + v_{i+1} \right)}{\tilde{d}_i}$$

for $i = 2, \ldots, n-2$. The boundary values are known ($v_0 = v_n = 0$). v_{n-1} was calculated first and like this:

$$v_{n-1} = \frac{\tilde{b}_{n-1}}{\tilde{d}_{n-1}}$$

3.4 LU-composition

4 Result

Vilde's CPU times:

These numbers are for calculating $n = 10$ to $n = 10^7$ and not printing anything.

Unspecialised: 'CPU time: 7.35938 s'

Specialized: 'CPU time: 2.17188 s'

Vilde's error table:

Table 1: This is a table listing the log of the error and the log of the associated h value. We can see that the smallest error, log(RelativeError) = −7, is accomplished when $\log(h) = -4$.

log(h):	log(RelativeError):
-1.04	-1.18
-2.00	-3.09
-3.00	-5.08
-4.00	-7.09
-5.00	-6.57
-6.00	-4.65