Project 1

For the course FYS3150

Erik Grammeltvedt, Erlend Tiberg North and Alexandra Jahr Kolstad

September 7, 2019
Week 35-37

Contents

1 Abstract

We have developed an algorithm that computes...
Thomas algorithm, loss of numerical precision (FLOPS)

2 Introduction

en introduksjon som sier hva målet er og hva man har gjort. på slutten et kort sammendrag av strukturen til rapporten

In this project we are going to get familiar with vectors and matrices in `C++`. Our group is using the Armadillo package to more easily define and compute with matrices. We are also going to work with and try to understand dynamic memory allocation. The main exercise is grouped into smaller sub exercises ranging from **a)** to **e)**.

Må skrive mer

3 Method

teoretiske modeller og teknikaliteter -> metode (noen beregninger/kodeeksempler underveis)
Her skal man vise at koden fungerer
må kommentere alt i hele koden -> hva de gjør, hva de betyr
må inkludere selve koden -> gjøres utenfor selve programmet, gjerne gjennom GitHub
burde prøve å finne analytiske løsninger eller finne grensene for å teste programmet

3.1 Exercise a)

In the exercise we are given the equation

$$-\frac{v_{i+1}+v_{i-1}-2v_i}{h^2} = f_i \qquad \text{for } i = 1, 2, 3, ..., n$$

Rewrites the equation to

$$\begin{aligned} -(v_{i+1}+v_{i-1}-2v_i) = h^2 f_i = \tilde{b}_i \\ -v_{i+1}-v_{i-1}+2v_i = \tilde{b}_i \end{aligned}$$

where in the exercise we are also given the correlation $\tilde{b}_i = h^2 f_i$, which is implemented here.

Defines the equation for different values of the integer i to get a set of equations. The exercise also gives the boundry conditions $v_0 = v_{n+1} = 0$.

$$
\begin{aligned}
i = 1: &\quad -v_{1+1} - v_{1-1} + 2v_1 = -v_2 - v_0 + 2v_1 = -0 + 2v_1 - v_2 = \tilde{b}_1 \\
i = 2: &\quad -v_{2+1} - v_{2-1} + 2v_2 = -v_3 - v_1 + 2v_2 = -v_1 + 2v_2 - v_3 = \tilde{b}_2 \\
i = 3: &\quad -v_{3+1} - v_{3-1} + 2v_3 = -v_4 - v_2 + 2v_3 = -v_2 + 2v_3 - v_4 = \tilde{b}_3 \\
\vdots \\
i = n: &\quad -v_{n+1} - v_{n-1} + 2v_n = -v_{n-1} + 2v_n - 0 = \tilde{b}_n
\end{aligned}
$$

Equations can be rewritten as a matrix equation, which gives a matrix $\mathbf{A}$ with integers as elements, a vector $\vec{v} = [v_1, v_2, v_3, ..., v_n]$ and another vector $\vec{\tilde{b}} = [\tilde{b}_1, \tilde{b}_2, \tilde{b}_3, ..., \tilde{b}_n]$. This gives the matrix equation

$$\mathbf{A}\vec{v} = \vec{\tilde{b}} \tag{1}$$

The matrix and the vectors are given as

$$
\begin{bmatrix}
2 & -1 & 0 & 0 & \dots & 0 \\
-1 & 2 & -1 & 0 & \dots & 0 \\
0 & -1 & 2 & -1 & \dots & 0 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & -1 & 2 & -1 \\
0 & 0 & 0 & 0 & -1 & 2
\end{bmatrix}
\begin{bmatrix}
v_1 \\ v_2 \\ v_3 \\ \vdots \\ v_{n-1} \\ v_n
\end{bmatrix}
=
\begin{bmatrix}
\tilde{b}_1 \\ \tilde{b}_2 \\ \tilde{b}_3 \\ \vdots \\ \tilde{b}_{n-1} \\ \tilde{b}_n
\end{bmatrix}
$$

Therefore the matrix equation has been proved.

3.2 Exercise b)

3.2.1 Calculations

i beregningene til erlend på ark er b diagonalelementene, g er b-vektoren og u er v-vektoren. i pythonkoden er d diagonalelementene, b er b-vektoren og v er v-vektoren. med tilde er algoritmediagonal

For the forward substitution algorithm there are 5 floating point operations.
For the backward substitution algorithm there are 3 floating point operations.
The total number of floating point operations is 8. For one iteration.

For the forward substitution the equations used in the algorithm are

$$\tilde{d}_i = d_i - \frac{a_{i-1}c_{i-1}}{\tilde{d}_{i-1}} \tag{2}$$

$$\tilde{b}_i = b_i - \frac{\tilde{b}_{i-1}a_{i-1}}{\tilde{d}_{i-1}} \tag{3}$$

with the condition that $\tilde{d}_1 = d_1$.

For the backward substitution the equation used in the algorithm is

$$v_i = \frac{\tilde{b}_i - c_i v_{i+1}}{\tilde{d}_i}$$

with the conditions that $v_0 = 0$ and $v_n = \frac{\tilde{b}_n}{\tilde{d}_n}$.

3.2.2 The programming

Her skal man kommentere programmet til oppgaven
First we are going to look at the programme `thomas-algorithm.cpp`. This is the main programme for this exercise, which computes the decomposition and forward substitution and the backward substitution of the given matrices. More disriptive comments are included in the respective programmes on GitHub. This code asks the user for an input for the variable n, which is the dimension for the matrix $\mathbf{A}$ and the length of the vectors $\vec{v}$ and $\vec{b}$. After this it runs through a for-loop to compute the new arrays `ad`, `d_new` and `b_tld_new` for the forward substitution. They are respectively the variable $\frac{a_{i-1}}{\tilde{d}_{i-1}}$, the new diagonalelements to $\mathbf{A}$ and the new array for the $\vec{\tilde{b}}$ vector. The equations for the new variable and arrays are described in (2) and (3).

3.3 Exercise c)

3.3.1 Calculations

The equation for the elements on the diagonal is

$$\tilde{d}_i = d_i - \frac{a_{i-1}c_{i-1}}{\tilde{d}_{i-1}}$$

This equation is derived from forward substitution from **a)**.

With the knowledge that the diagonalelements are $a_1 = a_2 = ... = a_{i-1} = -1$ and $c_1 = c_2 = ... = c_{i-1} = -1$ we can shorten the equation to the form

$$\tilde{d}_i = d_i - \frac{1}{\tilde{d}_{i-1}}$$

When asserting different integer values for i we can compute the elements.

$$\begin{aligned}
\tilde{d}_1 &= d_1 = 2 \\
\tilde{d}_2 &= 2 - \frac{1}{2} = \frac{3}{2} \\
\tilde{d}_3 &= 2 - \frac{1}{\frac{3}{2}} = \frac{4}{3} \\
\tilde{d}_4 &= 2 - \frac{1}{\frac{4}{3}} = \frac{5}{4} \\
&\vdots \\
\tilde{d}_n &= 2 - \frac{1}{1} = 1
\end{aligned}$$

From this we can derive a general formula for the diagonalelements

$$\tilde{d}_i = \frac{i+1}{i}$$

The equation for forward substitution is

$$\tilde{v}_i = \frac{\tilde{b}_i - c_i v_{i+1}}{\tilde{d}_i}$$

We can also shorten this equation with the same knowledge from earlier that $c_1 = c_2 = ... = c_{i-1} = -1$.

$$\tilde{v}_i = \frac{\tilde{b}_i + v_{i+1}}{\tilde{d}_i}$$

3.3.2 The programming

3.4 Exercise d)

3.4.1 Calculations

3.4.2 The programming

3.5 Exercise e)

3.5.1 Calculations

3.5.2 The programming

4 Results and discussion

skal inkludere resultatene enten som figur eller som en tabell
må nummerere/navngi alle resultatene
alle resultatene skal ha relevante titler og merkelapper på aksene
burde evaluere "troverdigheten" (reliability) og den numeriske stabiliteten/presisjonen til resultatene
hvis mulig inkluder en kvalitativ og/eller kvantitativ diskusjon av den numeriske stabiliteten, tap av presisjon osv
prøve å tolke resultatene i svaret til problemene
faget ønsker at man skal kommentere oppgavene. hva som var bra, hva som kan være bedre, hva man kan gjøre annerledes

lime inn resultater av siste gang programmene kjøres

The codes that were used for exercises **c)**, **d)** and **e)** are respectively `c-koden.cpp`, `d-koden.cpp` and `e-koden.cpp`. For **b)** there are two codes called `b-koden.cpp` and `b-koden.py`.

5 Conclusion and perspective

6 Appendix

Kan også linke til ekstra materiale på GitHub

7 Refrences

Må referere til Thomas Algorithm fordi den er mindre kjent
Unngå wikipedia -> se på hvilke artikler/bøker de har linket til
finner bøker/artikler osv ved: universitetsbiblioteket, ”physical review letters of the american physical society”