Title of the document

Henrik Modahl Breitenstein and Carl Petter Duedahl
(Dated: September 6, 2021)

https://github.com/henrikbreitenstein/FYS3150.git

PROBLEM 1

Poisson likningen

$$-\frac{\mathrm{d}^2u}{\mathrm{d}x^2} = f(x)$$

Bytter $f(x)$ med gitt funksjon

$$\begin{aligned}-\frac{\mathrm{d}^2u}{\mathrm{d}x^2} =& 100e^{-10x}\\ -\mathrm{d}^2u =& 100e^{-10x}\ \mathrm{d}x^2\end{aligned}$$

Tar integralene

$$\begin{aligned}-\iint \mathrm{d}^2u =& \iint 100e^{-10x}\ \mathrm{d}x^2\\ -u =& \int -10e^{-10x} + c_1\ \mathrm{d}x\\ -u =& e^{-10x} + c_1x + c_2\\ u =& -e^{-10x} - c_1x - c_2\end{aligned}$$

Bruker initialbetingelsene

$$u(0) = 0 \Rightarrow -1 - c_2 = 0 \tag{1}$$

$$u(1) = 0 \Rightarrow -e^{-10} - c1 - c2 = 0 \tag{2}$$

Med 1 og 2 får vi:

$$\begin{aligned}c2 =& -1\\ c1 =& 1 - e^{-10}\end{aligned}$$

Ved å sette inn for c_1 og c_2 får vi:

$$u = 1 - (1 - e^{-10})x - e^{-10x} \tag{3}$$

PROBLEM 2

I repostory'et under FYS3150/Project1/main.cpp så har vi skrevet koden som regner ut verdiene til den eksakte løsningen, og i FYS3150/Project1/plot.py så tegnes grafen. Kan kjøre programmet 'main.cpp' med commandoen

$ make all
og kjøre 'plot.py' med

$ python plot.py

PROBLEM 5

Problem a

Siden vi gjør en matrismultiplikasjon så må bredden, n, til $\mathbf{A}$ være det samme som lengden m til $\overrightarrow{v}$ og $\overrightarrow{g}$. Så

$$n = m$$

Problem b

Av $A\vec{v} = \vec{g}$ vil vi finne alle verdier mellom grensebetingelsene $u(0) = u(1) = 0$. $\vec{v*}$ er den samme som $\vec{v}$ men vi har lagt til v_0 og v_{n+1} som er grensebetingelsene.

PROBLEM 6

A

Vi har nå en vektor

$$\vec{v} = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix}$$

og en $n \times n$-matrise

$$\mathbf{A} = \begin{Bmatrix} b_1 & c_2 & 0 & 0 & \cdots & 0 & 0 \\ a_2 & b_2 & c_2 & 0 & \cdots & 0 & 0 \\ 0 & a_3 & b_3 & c_3 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & \cdots & \cdots & \cdots & a_n & b_n \end{Bmatrix}$$

Matrisemultipliserer vi disse får vi

$$\mathbf{A}\vec{v} = \begin{matrix} (I) & b_1 v_1 & +c_1 v_2 & & & = g_1 \\ (II) & a_2 v_1 & +b_2 v_2 & +c_3 v_3 & & = g_2 \\ (III) & & a_3 v_2 & +b_3 v_3 & +c_3 v_4 & = g_3 \\ \vdots & & & & & \\ (n) & & & a_n v_{n-1} & +b_n v_n & = g_n \end{matrix}$$

Herfra skal vi radredusere og starter med å ta $(II) - \frac{a_2}{b_1}(I)$ slik at vi får

$$(II*) \quad 0 \quad b_2 - \frac{c_1 \cdot a_2}{b_1} \quad c_2 \quad \cdots \quad g_2 - g_1 \frac{a_2}{b_1}$$

og vi ser da at a_2 går bort. Vi kan også sette $b_2^* \equiv b_2 - \frac{a_2 \cdot c_1}{b_1}$ og $g_2^* \equiv g_2 - g_1 \frac{a_2}{b_1}$. Da har vi mellom (II^*) og (III) noe som ser ganske likt ut som det vi hadde mellom (I) og (II). Derfor gjør vi det samme som vi gjorde før og tar $(III) - \frac{a_3}{b_2^*}(II)^*$ og får

$$(III)^* \quad 0 \quad 0 \quad b_3 - \frac{c_2 \cdot a_3}{b_2^*} \quad c_3 \quad g_3 - g_1 \frac{a_3}{b_2^*}$$

og vi kan igjen definere $b_3^* = b_3 - \frac{c_2 \cdot a_3}{b_2^*}$ og $g_3^* = g_3 - g_1 \frac{a_3}{b_2^*}$. Og vi kan da fortsette med dette nedover som $(k) - \frac{a_k}{b_{k-1}^*}(k-1)^*$. Så har vi fjernet a-ene så da må vi fjerne c-ene. Vi starter nå på siste og neste siste rad, altså $(n)^*$ og $(n-1)^*$ som nå er

$$\begin{matrix} (n-1)^* & 0 & \cdots & b_{n-1}^* & c_{n-1} & g_{n-1}^* \\ (n)^* & 0 & \cdots & 0 & b_n^* & g_{n-1}^* \end{matrix}$$

så hvis vi da tar $(n-1)^* - \frac{c_{n-1}}{b_n^*}(n)^*$ får vi

$$(n\tilde{-}1) \;\; 0 \;\; \cdots \;\; b_{n-1}^* \;\; 0 \;\; g_{n-1}^* - \frac{c_{n-1} g_n^*}{b_n^*}$$

og vi setter $g_{n\tilde{-}1} = g_{n-1}^* - \frac{c_{n-1} g_n^*}{b_n^*}$ og dette gjør vi videre oppover som $(k-1) - \frac{c_k}{b_{k-1}}(k)$ Til slutt står vi da bare igjen med b^*-ene og disse kan vi da dele på seg selv og vi får en identitetsmatrise. Vi kan nå skrive dette som en algoritme. Anta vi har en tridiagonal matrise

$$A = U = \begin{pmatrix} u_{1,1} & u_{1,2} & \cdots & u_{1,n} \\ u_{2,1} & u_{2,2} & \cdots & u_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ u_{n,1} & un, 2 & \cdots & un, n \end{pmatrix}$$

og en som skal løses for vektoren

$$g = h = \begin{pmatrix} h_1 \\ h_2 \\ \vdots \\ h_n \end{pmatrix}$$

Algorithm 1 Radredusering av tridiagonal matrise

for $i = 2, 3, ..., n$ **do** $\triangleright$ Forward substitution, $n - 1$ repetisjoner

$$t \leftarrow \frac{u_{i,i-1}}{u_i - 1, i}$$

$\triangleright$ 1 FLOP

$$u_{i,i-1} \leftarrow u_{i,i-1} - u_{i-1,i-1} \cdot t$$

$\triangleright$ 2 FLOPs

$$u_{i,i} \leftarrow u_{i,i} - u_{i-1,i-1} \cdot t$$

$\triangleright$ 2 FLOPs

$$h_i \leftarrow h_i - h_{i-1} t$$

$\triangleright$ 2 FLOPs

$\triangleright$ Til sammen $7 \cdot (n - 1)$ FLOPs i loopen

for $j = n - 1, n - 2, ..., 1$ **do** $\triangleright$ Backward Substitution, $n - 1$ repetisjoner

$$k \leftarrow \frac{u_{j,j+1}}{u_{j+1,j}}$$

$\triangleright$ 2 FLOPs

$$u_{j,j+1} \leftarrow u_{j,j+1} - u_{j+1,j+1} \cdot k$$

$\triangleright$ 2 FLOPs

$$u_{j,j} \leftarrow u_{j,j} - u_{j+1,j} \cdot k$$

$\triangleright$ 2 FLOPs

$$h_j \leftarrow h_j - h_{j+1} \cdot k$$

$\triangleright$ 2 FLOPs

$\triangleright$ Til sammen $7 \cdot (n - 1)$ FLOPs i loopen

for $l = 1, 2, \cdots n$ **do** $\triangleright$ Deler for å få identitetsmatrise, til sammen n repetisjoner

$$u_l \leftarrow \frac{u_l}{u_l}$$

$\triangleright$ 1 FLOP

$$h_l \leftarrow \frac{h_l}{u_l}$$

$\triangleright$ 1 FLOPs

$\triangleright$ Til sammen $2 \cdot n$ FLOPs

Da ser vi at vi til sammen får $2 \cdot n + 2 \cdot 7 \cdot (n - 1) = 16n - 14$ FLOPs.

PROBLEM 7

Sriptet 'Problem7new.cpp' og 'Problem7func.cpp' bruker den generelle algoritmen til å løse matriselikningen. For å kjøre de sammen:

$ make pr7all

PROBLEM 8

Problem a

problem b

Problem c

PROBLEM 9

Algorithm 2 Spesialisert algoritme

$$a \leftarrow -1$$

$$b \leftarrow 2$$

$$c \leftarrow -1$$

$$g \leftarrow \overrightarrow{0_n}$$

$$\widetilde{b} \leftarrow \overrightarrow{0_n}$$

$$\widetilde{g} \leftarrow \overrightarrow{0_n}$$

$$v \leftarrow \overrightarrow{0_n}$$

$$\widetilde{b}_0 \leftarrow b$$

for $i = 0, 1, 2, \cdots, n$ **do** ▷ Fyller g med verdiene til $f(x)$

$$g_i \leftarrow f(x_i)$$

$$\widetilde{g}_0 \leftarrow g_0$$

$$d \leftarrow a \cdot c$$

for $i = 0, 1, 2, \cdots, n-1$ **do** ▷ Forward elemination

$$\widetilde{b}_{i+1} \leftarrow \widetilde{b}_{i+1} - \frac{d}{\widetilde{b}_i}$$

▷ 2 FLOPs

$$\widetilde{g}_{i+1} \leftarrow g_{i+1} - \frac{a}{\widetilde{b}_i} g_i$$

▷ 3 FLOPs

for $i = n-1, n-2, \cdots, 0$ **do** ▷ Backward elemination

$$v_i \leftarrow \frac{\widetilde{g}_i - c v_{i+1}}{\widetilde{b}_i}$$

▷ 3 FLOPs

Problem a

problem b

Problem c

I filen 'solve.cpp' har vi kodet den spesielle algoritmen. Kjøres ved kommandoen

$ make solveall