FYS3150 - project 1

Sverre Wehn Noremsaune & Frida Marie Engøy Westby
(Dated: September 8, 2021)

https://github.uio.no/comPhys/FYS3150/tree/project1

PROBLEM 1

We have the one-dimensional Poisson equation

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

where $f(x)$ is known to be $100e^{-10x}$. We also assume $x \in [0, 1]$, that the boundary condition are $u(0) = 0 = u(1)$ and $u(x)$ is

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

where $u(x)$ is an exact solution to Eq. (1). We can analytically check this by derivateing $u(x)$ twice.

$$u(x)' = 10x^{-10x} - 1 + \frac{1}{e}$$
$$u''(x) = -100e^{-10x} = f(x) \quad \blacksquare$$

$-\frac{d^2u(x)}{dx^2} = f(x)$
$x \in [0, 1]$
$i = 0, 1, \ldots, n$
$h = \frac{x_{max} - x_{min}}{n}$
$x \to x_i$
$x_i = x_0 + ih$
We're using the three point formula to find the second derivative, $-\frac{u_{i-1} - 2u_i + u_{i+1}}{h^2} = f_i = f(x_i)$
$v_i \approx u_i \implies -\frac{v_{i-1} - 2v_i + v_{i+1}}{h^2} = \frac{-v_{i-1} + 2v_i - v_{i+1}}{h^2} = f_i, f_i = f(x_i)$, for $i \in [1, n-1]$
—
$v_i = f_i$ is a set of equations for every i. By multiplying every line with h^2 we get this

$$\begin{array}{ccccc} -v_0 & 2v_1 & -v_2 & & = h^2 f_i \\ & -v_1 & 2v_2 & -v_3 & = h^2 f_2 \end{array}$$

and so on and so forth. We know $v_0 = v_n = 0$, so we remove those. This can then be easily rewritten as a matrix equation $A\vec{v} = \vec{g}$

$$\begin{bmatrix} 2 & -1 & 0 & 0 & \ldots & 0 & 0 \\ -1 & 2 & -1 & 0 & \ldots & 0 & 0 \\ 0 & -1 & 2 & -1 & \ldots & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & -1 & 2 & -1 & 0 \\ 0 & 0 & \ldots & 0 & -1 & 2 & -1 \\ 0 & 0 & \ldots & 0 & 0 & -1 & 2 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ \vdots \\ v_{n-3} \\ v_{n-2} \\ v_{n-1} \end{bmatrix} = \begin{bmatrix} g_1 \\ g_2 \\ g_3 \\ \vdots \\ gn-3 \\ gn-2 \\ g_{n-1} \end{bmatrix}$$

g_i is $h^2 f_i$

Algorithm 1 Algorithm for solving general tridiagonal matrix

```
arrays a, b, c, u, f, temp of length n

btemp = b[1]
u[1] = f[1]/btemp

for  i = 2,3,...,n  do
    temp[i] = c[i-1] / btemp
    btemp = b[i] - a[i] * temp[i]
    u[i] = (f[i] - a[i] * u[i-1]) / btemp
for i = n-1, n-2, ..., 1 do
    u[i] -= temp[i+1] * u[i+1]
```

FLOPs: 1 + 6(n-1) + 2(n-1) = 1 + 8(n-1)

Finally, we can list algorithms by using the `algorithm` environment, as demonstrated here for algorithm 2.

Algorithm 2 Some algorithm

Some maths, e.g $f(x) = x^2$. ▷ Here's a comment
for $i = 0, 1, ..., n-1$ **do**
    Do something here
while Some condition **do**
    Do something more here
Maybe even some more math here, e.g $\int_0^1 f(x)\mathrm{d}x$