Project 1 - FYS3150

Dag Arne Lydvo
(Dated: September 13, 2021)

List a link to your github repository here!

PROBLEM 1

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

Solving for u:

$$-u = \int\int 100e^{-10x}dx^2$$

Using $w = -10x$ and $dx = \frac{dw}{-10}$

$$-u = 100\int\int e^w \frac{dw^2}{-10}$$

$$-u = -10\int e^{-10x} + C_1 dx$$

$$-10\int e^w + C_1\frac{dw}{-10} = e^w + C_1 w + C_2$$

$$-u = e^{-10x} + C_1(-10x) + C_2$$

Solving for constants using boundary conditions $u(0) = u(1) = 0$

$$-u(0) = 0 = e^0 + C_2 \rightarrow C_2 = -1$$

$$-u(1) = 0 = e^{-10} + C_1(-10) - 1 \rightarrow C_1 = \frac{1}{10}(e^{-10} - 1)$$

$$-u(x) = e^{-10x} + \frac{1}{10}(e^{-10} - 1)(-10x) - 1$$

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

PROBLEM 2

I. PROBLEM 3

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

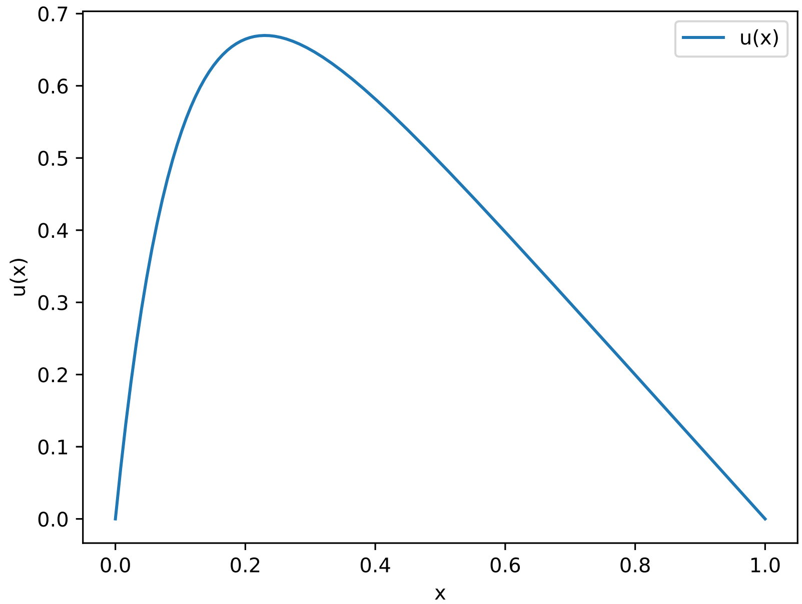

FIG. 1. Exact solution of u(x) over 1000 steps of x.

The continous variable x becomes discrete, $x \rightarrow x_1$
Where i = (0,1,...,n), where n is the number of steps.
$u(x)$ becomes $u(x_i)$, and $u(x_1) = u(x_0 + h)$, where h is the step size defined as $h = \frac{x_n - x_0}{n}$
The discrete version of $-\frac{d^2u}{dx^2} = f(x)$ now becomes

$$-\frac{d^2u}{dx^2}|_{x_i} = \frac{u_{i+1} + u_{i-1} - 2u_i}{h^2} + O(h^2)$$

and the approximation can be written as

$$v_i'' = \frac{-v_{i+1} - v_{i-1} + 2v_i}{h^2} = f_i$$

II. PROBLEM 4

The approximte discrete version of the Poisson equation from problem 3 can be rewritten as

$$-v_{i+1} - v_{i-1} + 2v_i = f_i h$$

Here we have n numbers of unknown variables v_i for $i = (0, 1, ..., n)$ making up the the vector $\vec{v} = (v_0, v_1, ..., v_n)$. We also have n numbers of known functions $g_i = f_i h^2$ making up the vector $\vec{g} = (f_0, f_1, ..., f_n)$.
The termns of $v'' h^2 = v_{i-1} - 2v_i + v_{i+1}$ makes up n sets of equations.

$$2v_1 - v_2 = g_1$$

$$-v_1 + 2v_2 - v_3 = g_2$$

$$...$$

...

...

$$-v_{n-1} + 2v_n = g_n$$

The terms of $-v_0 and - v_n + 1$ are zero because of the boundary conditions. The first and last equations therefore contains only two terms.

This set of equations can then be written as the matrix equation $\mathbf{A}\vec{v} = \vec{g}$.
Where $\mathbf{A}$ is:

$$\begin{bmatrix} 2 & -1 & 0 & \dots & 0 \\ -1 & 2 & -1 & \dots & 0 \\ \vdots & \vdots & \ddots & \dots & 0 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -1 & 2 \end{bmatrix}$$

and $\vec{v}$ and $\vec{g}$ is:

$$\begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_n \end{bmatrix}$$

III. PROBLEM 5

IV. PROBLEM 6

Solving a general tridiagonal matrix can be done by two general steps. First forward substitution then backward substitution on the subdiagonal a_i, main diagonal b_i and superdiagonal c_i.

Algorithm 1 Solving general tridiagonal matrix

Forward substitution
$\tilde{b_0} = b_0$ ▷ Here's a comment
$\tilde{g_0} = g_0$
for $i = 1, ..., n$ **do**
  $\tilde{b_i} = b_i - \frac{a_i}{\tilde{b_{i-1}}} c_{i-1}$ ▷ 3(n-1) FLOPs
  $\tilde{g_i} = g_i - \frac{a_i}{\tilde{b_{i-1}}} \tilde{g_{i-1}}$ ▷ 3(n-1) FLOPs
Backward substitution
$v_n = \frac{\tilde{g_n}}{\tilde{b_n}}$ ▷ 1 FLOP
for $i = n-1, n-2, ..., 1$ **do**
  $v_i = \frac{\tilde{g_i} - c_i v_{i+1}}{\tilde{b_i}}$ ▷ 3(n-1) FLOPs

With the forward subs. not counting the first element $i = 0$ and the backward substitution not counting the last element $i = n$ both for loops runs for $n - 1$ loops. The total FLOPs for the algorithm then becomes $9(n - 1) + 1$

V. PROBLEM 7

VI. PROBLEM 8

VII. PROBLEM 9

For the special case where the matrix $\mathbf{A}$ as a signature of (-1,2,-1), the diagonal vectors have constant values the general algorithm will be modified to possible reduce the number of FLOPs.

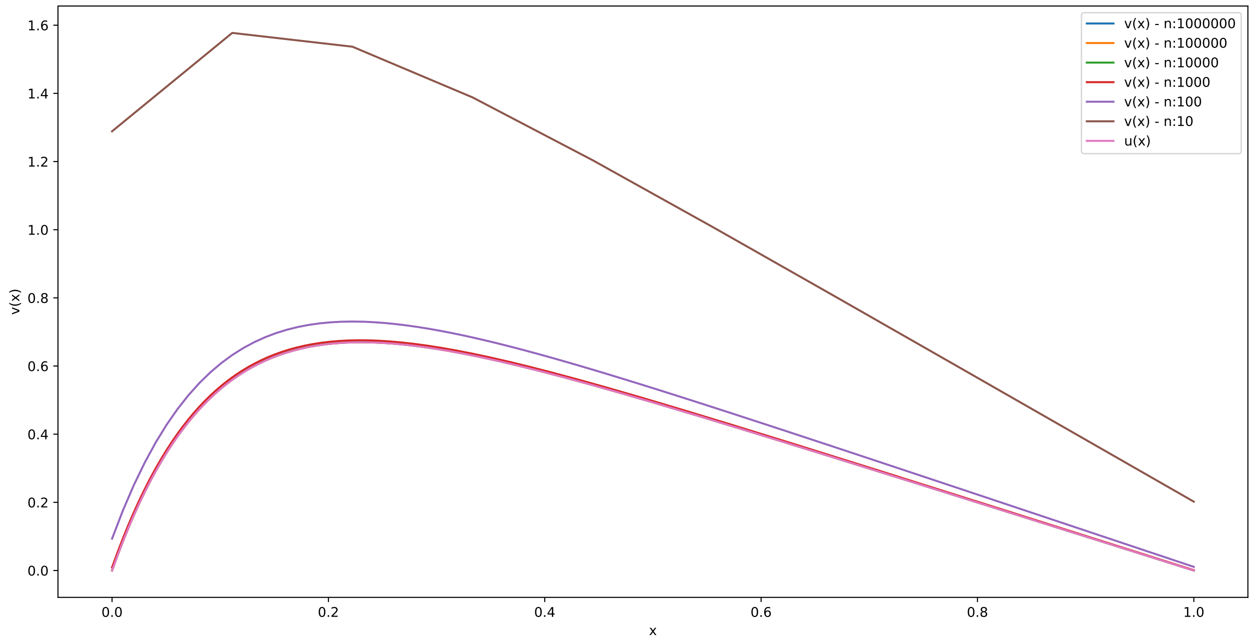

FIG. 2. Numerical solution using the general algorithm with varying values of n vs the exact solution u(x)

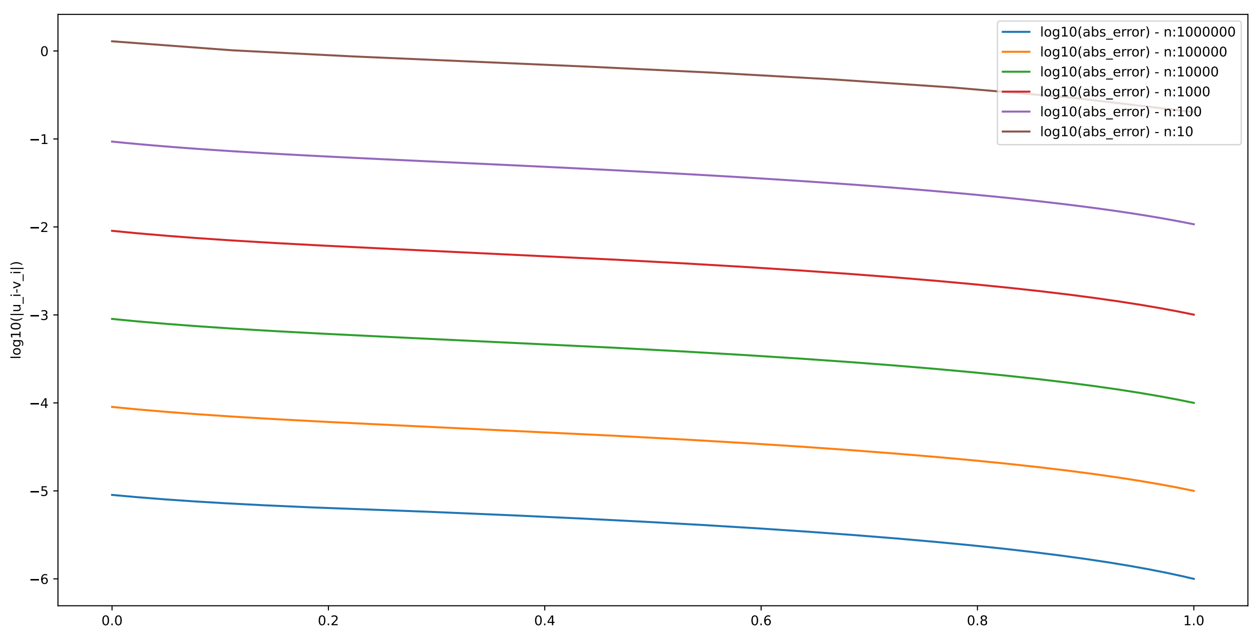

FIG. 3. Numerical solution using the general algorithm with varying values of n vs the exact solution u(x)

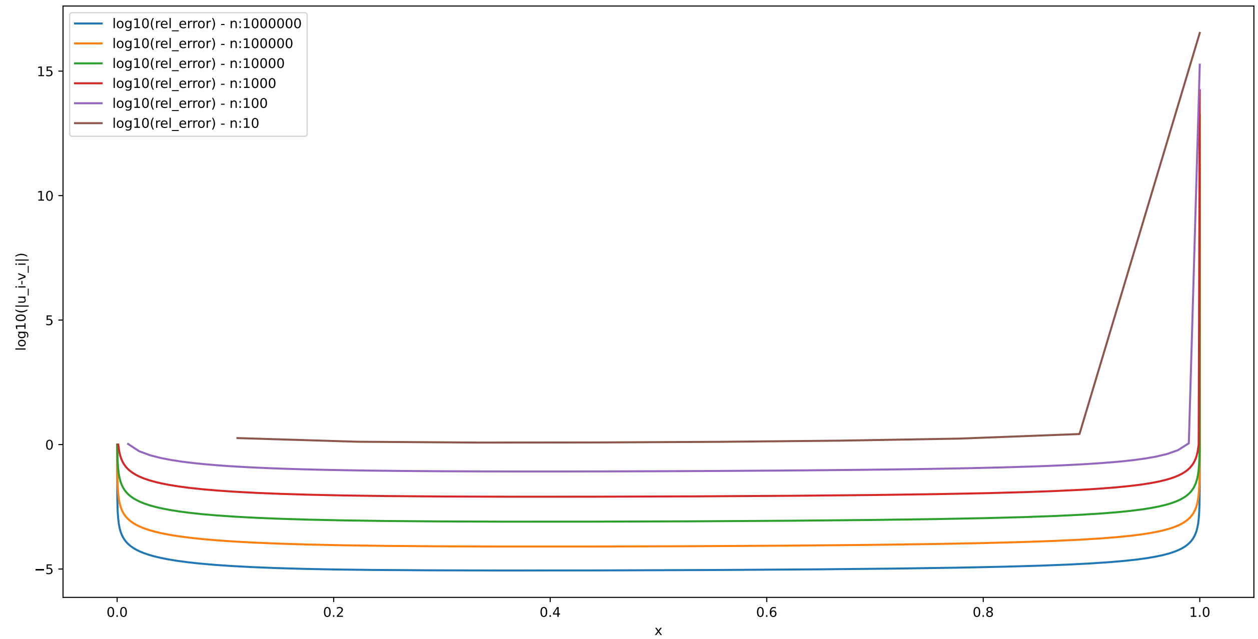

FIG. 4. Numerical solution using the general algorithm with varying values of n vs the exact solution u(x)

n	max	ex_inf	ex inf and x=1
1000000	inf	1.697370e+11	1.000014
100000	inf	1.697475e+12	1.000120
10000	inf	1.698546e+13	1.001202
1000	inf	1.709293e+14	1.012051
100	inf	1.820743e+15	1.123939
10	inf	3.430457e+16	2.638876

FIG. 5. Numerical solution using the general algorithm with varying values of n vs the exact solution u(x)

Since the values for a_i and c_i are constant the operations between these values can be removed. And since the value of a and c are -1 we can just switch the operator. This special algorithm reduces the number of FLOPs from $9(n-1)+1$ to $6(n-1)+1$.

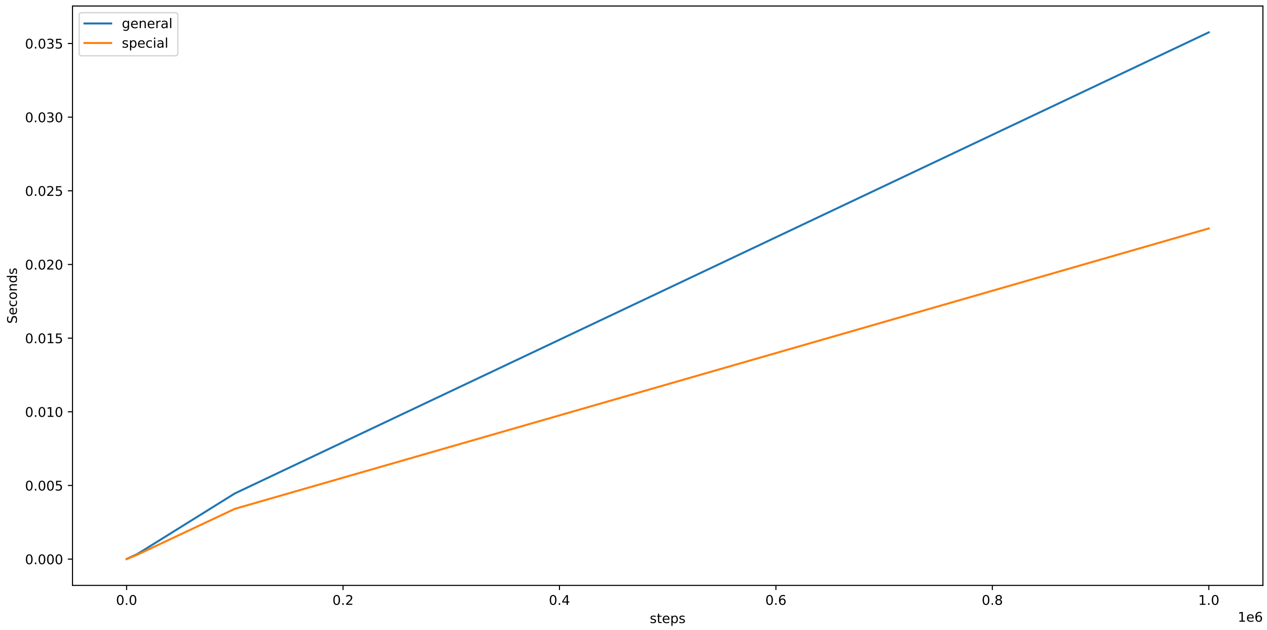

FIG. 6. Numerical solution using the general algorithm with varying values of n vs the exact solution u(x)

Algorithm 2 Solving special tridiagonal matrix

Forward substitution

$\tilde{b_0} = b_0$ ▷ Here's a comment

$\tilde{g_0} = g_0$

for $i = 1, ..., n$ **do**

$\tilde{b_i} = b_i - \frac{1}{\tilde{b_{i-1}}}$ ▷ 2(n-1) FLOPs

$\tilde{g_i} = g_i + \frac{\tilde{g_{i-1}}}{\tilde{b_{i-1}}}$ ▷ 2(n-1) FLOPs

Backward substitution

$v_n = \frac{\tilde{g_n}}{\tilde{b_n}}$ ▷ 1 FLOP

for $i = n-1, n-2, ..., 1$ **do**

$v_i = \frac{\tilde{g_i} + v_{i+1}}{\tilde{b_i}}$ ▷ 2(n-1) FLOPs

VIII. PROBLEM 10