FYS3150 - Project 1

Andrew Quan, Oskar Idland, Hishem Kløvnes, Håvard Skåli
(Dated: August 24, 2023)

GitHub Repository

PROBLEM 1

We have the one-dimensional Poisson equation

$$-\frac{\mathrm{d}^2u}{\mathrm{d}x^2} = 100e^{-10x}, \qquad x \in [0,1] \tag{1}$$

where the right hand side of the equation is the source term $f(x)$. We have the boundary conditions $u(0) = 0$ and $u(1) = 0$. Integrating both sides of (1) with respect to x twice we get

$$\begin{aligned}
\frac{\mathrm{d}^2u}{\mathrm{d}x^2} &= -100e^{-10x} \\
\iint \left(\frac{\mathrm{d}^2u}{\mathrm{d}x^2}\right)\mathrm{d}x^2 &= -100 \iint e^{-10x}\mathrm{d}x^2 \\
\int \left(\frac{\mathrm{d}u}{\mathrm{d}x}\right)\mathrm{d}x &= -100 \int \left(-\frac{1}{10}e^{-10x} + C\right)\mathrm{d}x \\
u(x) &= -100\left(\frac{1}{100}e^{-10x} + Cx + D\right) \\
u(x) &= Cx + D - e^{-10x}
\end{aligned} \tag{2}$$

Where C and D are some arbitrary constants. Invoking the boundary conditions we get

$$\begin{aligned}
u(0) = D - 1 = 0 \quad &\Rightarrow \quad D = 1 \\
u(1) = C + 1 - e^{-10} = 0 \quad &\Rightarrow \quad C = e^{-10} - 1
\end{aligned}$$

Thus, plugging these constants into the general solution and rearranging the terms, the final solution becomes

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

just like we wanted to show.

PROBLEM 2

After creating a program that defines a vector of x-values between 1 and 100 and evaluates the exact solution $u(x)$ defined as in (3), we wrote these numbers with four decimals into a text file and plotted them against each other in Python. The resulting plot is shown in FIG. 1.

PROBLEM 3

Our aim is to derive a discretized version of the Poisson equation which lets us calculate approximates values v of the exact values u. To create this program we need to discretize the double derivative, which we can do by recalling its definition:

$$\frac{\mathrm{d}^2u}{\mathrm{d}u^2} = \lim_{\mathrm{d}x\to 0} \frac{-u(x-\mathrm{d}x) + 2u(x) - u(x+\mathrm{d}x)}{\mathrm{d}x^2} \tag{4}$$

If we let the infinitesimal value $\mathrm{d}x$ rather become a finite, small value Δx, we'll still be able to calculate the double derivative with good precision, as long as we don't let this value grow too large. Since this is an approximation, a

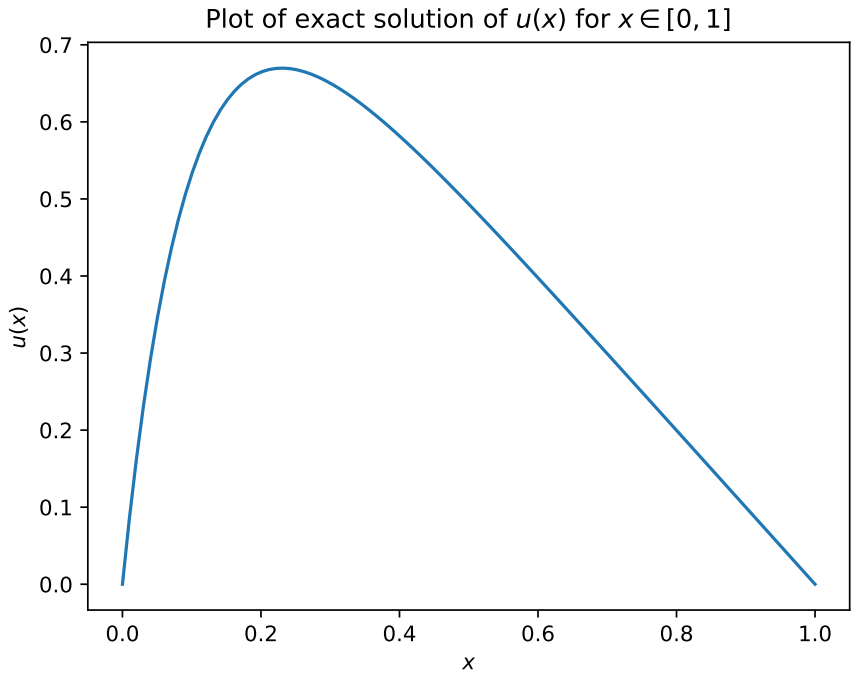

FIG. 1. Plot of the exact solution of $u(x)$ for $x \in [0, 1]$. We used 100 data points for the plot.

discretized version of (1), we will use $v(x)$ instead of $u(x)$ in the following procedures to make it clear that this is not an exact solution. The resulting equation becomes

$$\frac{-v(x - \Delta x) + 2v(x) - v(x + \Delta x)}{\Delta x^2} = -100e^{-10x} \tag{5}$$

Let's denote the i^{th} x-value in out set of data points with x_i, so that the corresponding v-value will be denoted with v_i. This implies that at the boundaries we have $x_0 = 0$, $v_0 = 0$, and $x_{N-1} = 1$, $v_{N-1} = 0$, where N are the number of data points. Thus, the final algorithm is

$$\frac{-v_{i-1} + 2v_i - v_{i+1}}{h^2} = f_i \tag{6}$$

where $h = \Delta x$ is defined as

$$h = \frac{x_N - x_0}{N - 1} \tag{7}$$

and we've rewritten the forcing term to include the minus sign that was originally on the left hand side of the equation, meaning that

$$f_i = f(x_i) = -100e^{-10x_i} \tag{8}$$

PROBLEM 4

Let us now multiply h^2 with both sides of (6), consequently rewriting the equation:

$$-v_{i-1} + 2v_i - v_{i+1} = g_i, \qquad g_i = h^2 f_i \tag{9}$$

Furthermore, we may define the vectors $\vec{v}$ and $\vec{g}$ such that

$$\vec{v} = \begin{bmatrix} v_0 \\ v_1 \\ \vdots \\ v_{N-1} \end{bmatrix}, \qquad \vec{g} = \begin{bmatrix} g_0 \\ g_1 \\ \vdots \\ g_{N-1} \end{bmatrix} \tag{10}$$

Using our knowledge of linear algebra, we may now define the matrix

$$\mathbf{A} = \begin{bmatrix} 2 & -1 & 0 & \dots & 0 \\ -1 & 2 & -1 & \dots & \vdots \\ 0 & -1 & 2 & \dots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & \dots & \dots & 2 \end{bmatrix} \tag{11}$$

which multiplied with $\vec{v}$ gives us

$$\mathbf{A}\vec{v} = \begin{bmatrix} 2 & -1 & \dots & 0 \\ -1 & 2 & \dots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & \dots & 2 \end{bmatrix} \begin{bmatrix} v_0 \\ v_1 \\ \vdots \\ v_{N-1} \end{bmatrix} = \begin{bmatrix} 2v_0 - v_1 \\ -v_0 + 2v_1 - v_2 \\ \vdots \\ -v_{N-2} + 2v_{N-1} \end{bmatrix} \tag{12}$$

We immediately see from (9) that this gives us the vector $\vec{g}$ containing the g_i-values with the same indexes as the corresponding v_i-values from $\vec{v}$. Thus, we can indeed rewrite the original discretized equation as the matrix equation

$$\mathbf{A}\vec{v} = \vec{g} \tag{13}$$

where $\mathbf{A}$ is the tridiagonal matrix with subdiagonal, main diagonal and superdiagonal specified by the signature $(-1, 2, -1)$, just like we wanted to show.

PROBLEM 5

Problem 5a

Problem 5b

PROBLEM 6

Problem 6a

Problem 6b

PROBLEM 7

Problem 7a

Problem 7b

PROBLEM 8

Problem 8a

Problem 8b

Problem 8c

PROBLEM 9

Problem 9a

Problem 9b

Problem 9c

PROBLEM 10

We write equations using the LaTeX `equation` (or `align`) environments. Here is an equation with numbering

$$\mathbf{F} = \frac{\mathrm{d}\mathbf{p}}{\mathrm{d}t}, \tag{14}$$

and here is one without numbering:

$$\oint_C \mathbf{F} \cdot \mathrm{d}\mathbf{r} = 0.$$

Sometimes it is useful to refer back to a previous equation, like we're demonstrating here for equation 14.

Also, note the LaTeX code we used to get correct quotation marks in the previous sentence. (Simply using the `"` key on your keyboard will give the wrong result.) Figures should preferably be vector graphics (e.g. a `.pdf` file) rather than raster graphics (e.g. a `.png` file).

By the way, don't worry too much about where LaTeX decides to place your figures and tables — LaTeX knows more than we do about proper document layout. As long as you label all your figures and tables and refer to them in the text, it's all good. Of course, in some cases it can be worth trying to force a specific placement, to avoid the figure/table appearing many pages away from the main text discussing it, but this isn't something you should spend time on until the very end of the writing process.

Next up is a table, created using the `table` and `tabular` environments. We refer to it by table I.

Finally, we can list algorithms by using the `algorithm` environment, as demonstrated here for algorithm 1.

TABLE I. Write a descriptive caption here, explaining the content of your table.

Number of points	Output
10	0.3086
100	0.2550

Algorithm 1 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 $\int_0^1 f(x)\mathrm{d}x$