Programming Assignment 1

Kyle

October 2024

Introduction

This assignment had two goals. The first goal was to demonstrate my proficiency of programming in C++. The second goal was to demonstrate my understanding of important mathematically and electrical concepts. Namely, I was tasked with solving several differential equations. Some equations were provided. Other equations were required to be derived from a given circuit through circuit analysis.

To properly program a continuous function on a discrete application (i.e. a computer), some assumptions and approximations had to be made. Mainly, it was assumed that $dt \approx \Delta t$. This assumption allowed for the value at a particular $t_a = t$ from the differential equation to be estimated using linear algebra. To calculate some $y(t_a)$ where $t_a \in \mathbf{R}$ and $t_a \geq 0$, the following equation was used:

$$\vec{y}(t_a) \approx \prod_{\substack{i=0 \\ \text{step } \Delta t}}^{t_a} (\vec{I} + \vec{A}\Delta t)\vec{y}(i) \tag{1}$$

Where $\vec{y}(t)$ is defined as a matrix with order n rows and a single column. Each row r where $0 \leq r \leq n$ is set to $\frac{d^r y}{dt^r}$. The vector $\vec{I}$ is simply the identity matrix with size n. Matrix $\vec{A}$ is constructed using the coefficients of each derivative term in the given differential equation using the pattern given in the assignment description.

Finally, it is clear that as Δt approaches zero, the more accurate this model becomes.

Program Overview

The source code of my project can be viewed from the accompanied zipped archive in my Canvas submission.

C++ Project

The C++ project was developed within *Visual Studio*. As such, no make file was used to compile and execute the program. All executions were done within the standard *Visual Studio* debugger.

The project includes the following header files:

Global.h	Global utility function, type declarations, imports, and namespace usings
Matrix.h	Declares Matrix structure type, data, and functions
DiffEq.h	Declares DiffEq class type, data, and functions

The project also includes the following C++ files:

Global.cpp	Implements global utility functions
Matrix.cpp	Defines Matrix data and behavior as well as constructor and functional operations
DiffEq.cpp	Defines DiffEq data and behavior
Main.cpp	Program's entry point and data gathering from the user.

Global The Global files serve as a simple utility center point for other header and C++ files to use. Utilites include gathering input, shorthand type definitions, and standard-library imports needed for the application.

Matrix The Matrix files defines my own matrix implementation. I am aware in the assignment I was to only need 1^{st} and 3^{rd} order differential equations to solve; however, I wanted to generalize the extent of my application. Also, I enjoyed this assignment enough to welcome the challenge. As such, I designed matrix operations (add, subtract, multiply, and scale) in accordance to standard arithmetic rules.

DiffEq The DiffEq files defines a class that stores all relevant information for a particular differential equation: information given by the operating user. Once constructed, the DiffEq class allows for the user to sample a particular value in time (greater than or equal to zero). If the user requests to sample $t = 0$, then the initial conditions matrix is simply returned. Otherwise, the DiffEq class will perform the equation described at (1).

The DiffEq class also allows for the export of data to a text file. The user provides the file name, t_{start}, t_{end}, and Δt where $t_{start}, t_{end}, \Delta t \geq 0$ and $t_{end} \geq t_{start}$. Once these values have been given, the DiffEq object will sample until $t = t_{start}$ and then begin mulitplying each iteration of Δt. Each iteration is written to the given text file.

Main The Main file is the entrypoint of the program. Here, standard text-style menu selection will prompt the user to provide information for a differential equation, then ask if they want to sample a value, export values, or quit.

MatLab Script

There is a single MatLab script in the *MatLab* directory of the project named **main.m**. This script reads *output.txt* and precisely parses information stored within the file. First, MatLab will parse the top two lines. The first line stores the coefficients of the differential equation that are separated by a space. The second line stores the initial conditions of each derivative of y at $t = 0$ that are also separeted by a space (excluding the highest order differential). Finally MatLab will treat the rest of the file as the sampled values exported from C++. Each line stores the value of t and $y(t)$ as a pair.

After parsing the file, MatLab then builds the differential equation using the provided data. Syms is a powerful tool in MatLab that allows programmers to create generalized differential equations and then solve the equation for you. It is guranteed that this found equation for $y(t)$ to be correct. The function that MatLab generates for $y(t)$ from the inputs provided by C++, and thus the user, is then graphed.

The final task for MatLab is to compare the graphs of the sampled values from the C++ code and the generated MatLab equation. To do this, the time of each function is limited between two values: the start and end values provided by the user in the C++ application. Then MatLab plots three graphs: an overlay of the sampled values from C++ and the MatLab function, the MatLab function, and the C++ approximation. The last two graphs are added in case the given Δt was so small that one plot overlaps the other. In other words, an overlap will occur if the C++ approximation is nearly equal to MatLab's generated equations.

Results

Section 2

Given the differential equation:

$$(D+3)y_0(t) = 0 \tag{2}$$

With $y_0(0) = 2$,

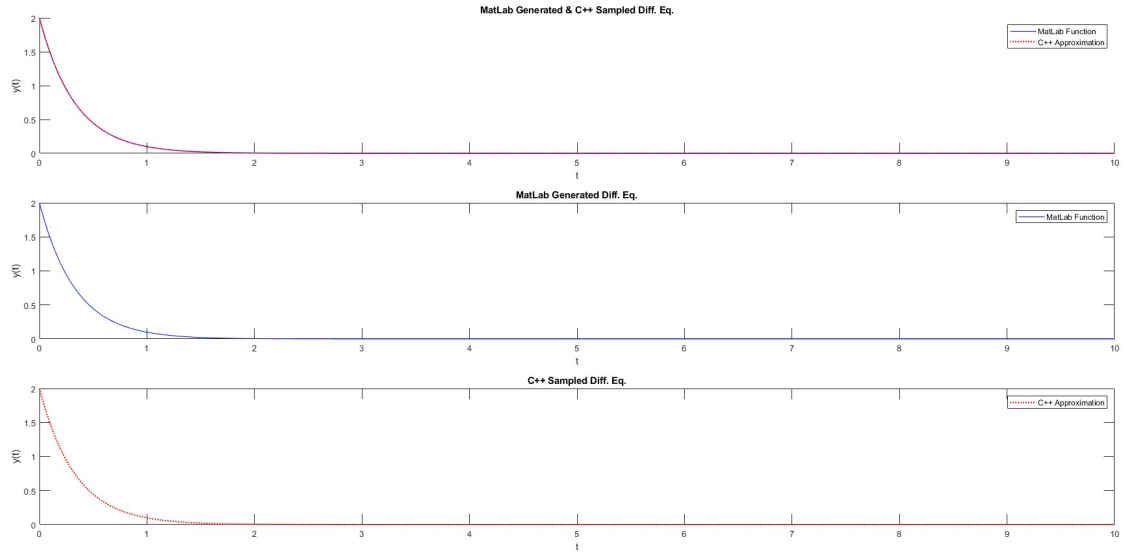
$$y(t) = 2e^{-3t} \tag{3}$$

This equation can easily be solved with the simple first-order, linear differential equation with initial conditions:

$$(D+a)y_0(t) = 0 \tag{4}$$

$$y_0(t) = c_1 e^{-at} \tag{5}$$

The three plots comparing the solution from MatLab and the sampled values from the C++ Application are shown:



As displayed, MatLab's solutions and the approximation from C++ are remarkably close. For this sample, $\Delta t = 0.0001$.

Section 3

Given the differential equation:

$$(D^3 + 0.45D^2 + 36.06D + 1.802)y_0(t) = 0 \tag{6}$$

This equation can be solved with solving a the characteristic equation and finding the roots. Observe:

$r^3 + 0.45r^2 + 36.06r + 1.802 = 0$	Standard characteristic equation from polynomial
$r = -0.05, -0.2 - 6j, -0.2 + 6j$	The roots of the equation

These roots can be inserted for a_1, a_2, a_3 following the same characteristic solution as described in Section 2:

$$y_0(t) = c_1 e^{-a_1 t} + c_2 e^{-a_2 t} + c_3 e^{-a_3 t} \tag{7}$$

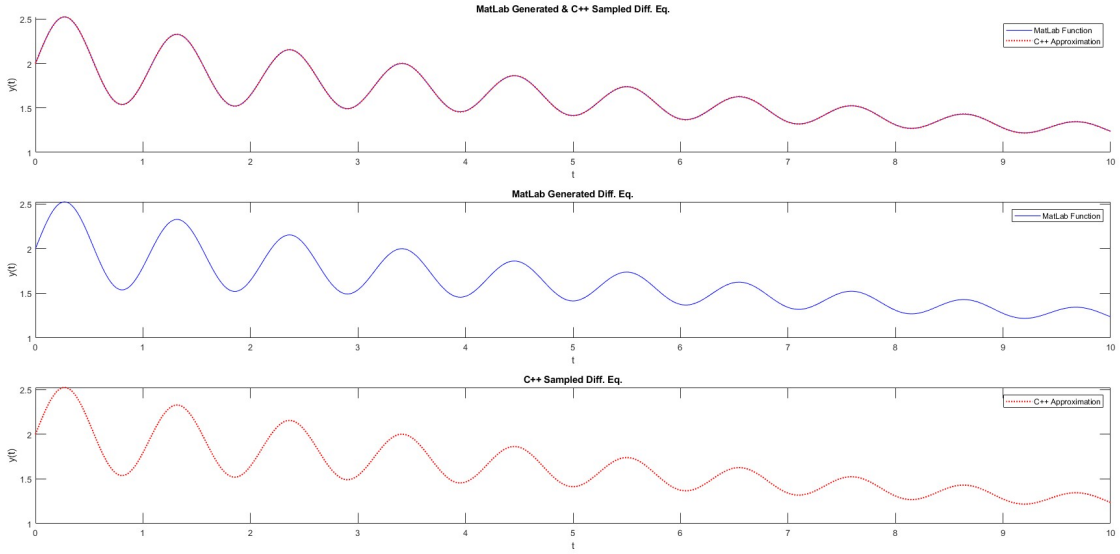
Then, Euler's formula is used to convert the complex roots into periodically decaying functions, resulting in the form:

$$y_0(t) = c_1 e^{\frac{-t}{20}} + c_2 e^{\frac{-t}{5}} sin(6t) - c_3 e^{\frac{-t}{5}} cos(6t) \tag{8}$$

With $y_0(0) = 2, \dot{y}_0(0) = 3, \ddot{y}_0(0) = 1$, the solution to the differential equation is approximately:

$$y_0(t) = 2.0620 e^{\frac{-t}{20}} + 0.51512 e^{\frac{-t}{5}} sin(6t) - 0.06620 e^{\frac{-t}{5}} cos(6t) \tag{9}$$

The three plots comparing the solution from MatLab and the sampled values from the C++ Application are shown:



Once again, MatLab's solutions and the approximation from C++ are very close. For this sample, $\Delta t = 0.0001$.

Finally, the state-space for this equation is as follows:

$$\begin{bmatrix} y_0(t+\Delta t) \\ \dot{y}_0(t+\Delta t) \\ \ddot{y}_0(t+\Delta t) \end{bmatrix} = \left(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1.802 & -36.06 & -0.45 \end{bmatrix} \Delta t \right) \begin{bmatrix} y_0(t) \\ \dot{y}_0(t) \\ \ddot{y}_0(t) \end{bmatrix} \tag{10}$$

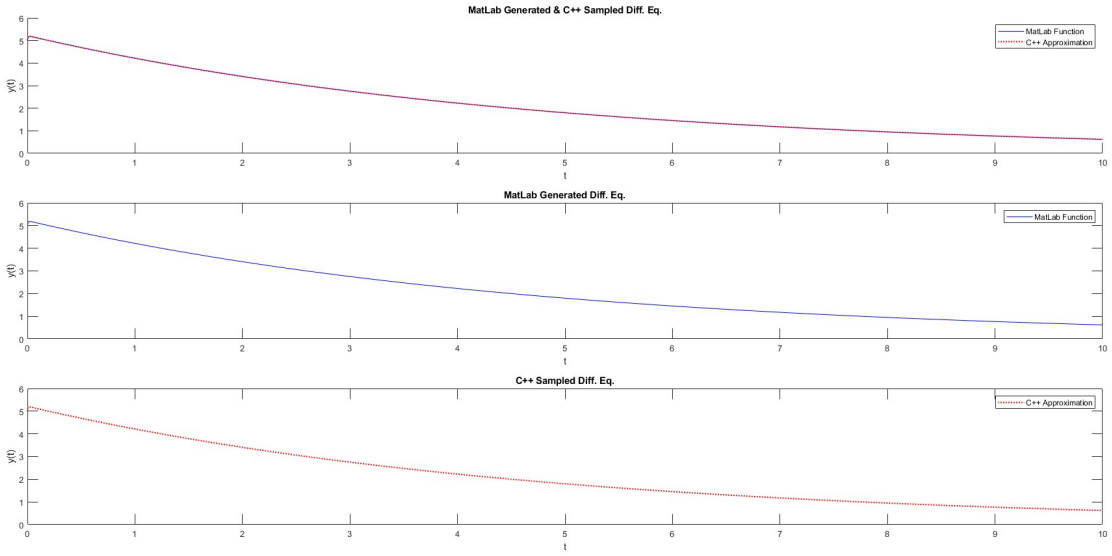
Section 4

For problems (a)-(d), refer to the handwritten work on the final few pages of the document.

Using both the C++ and the MatLab programs, the numerical solution of the differential equation to the zero input response is:

$$y_0(t) = 5.2131 e^{-0.2128t} - 0.2131 e^{-195.8292t} \tag{11}$$

Graphing the solution found by MatLab and the approximation from the C++ program, it can be observed that once again, the plots are nearly identical with $\Delta t = 0.0001$:



Finally, if a circuit has nonlinear elements, then the C++ approximation will not work. This is because the matrix math relies on the linear relationship between $y_0(t)$ and its derivatives. In more general terms, matrices can only be used to solve linear equations.

Analytically speaking, nonlinear differential equations are notoriously challenging to solve. Sometimes they simply cannot be solved.

Conclusion and Observations

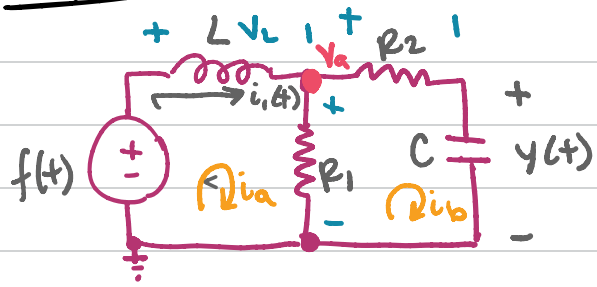
Ultimately, this assingment was a good way to push my knowledge of various topics:

- Programming in C++ (pointers, the heap vs. the stack, I/O, matrix math)
- Programming in MatLab (reading from a file, dynamic differential equations, plots)
- Circuit Analysis in the time domain
- Expanding my knowledge of Latex formatting and tools

Programming in C++ was among the easiest for me on this assignment. I had a little confusion when to add objects on the stack vs. the heap, and I ran into a few issues with proper pointers. Working with MatLab to read files and parse information properly took a lot of research, yet I am happy with the end result.

Section 4 Programming Assignment 1

Section 4



$R_1 = 1\ k\Omega$ = A

$R_2 = 47k\Omega$ = B

$C = 100\ \mu F$

$L = 5H$

$v_L(t) = L\frac{di_L}{dt}$

$i_c(t) = C\frac{dv_c}{dt} = CDy(t)$

$f(t) = ?$

$y(t) = ?$

$\frac{1}{C}i_c(t) = \frac{dv_c}{dt}$

$\frac{1}{C}\int i_c(t)dt = v_c(t)$

$\frac{1}{CD}i_c(t) = v_c(t)$

(a)

i_a: $f(t) = L\frac{di_1}{dt} + R_1(i_a - i_b)$

$i_a = i_1(t)$

i_b: $0 = R_1(i_b - i_a) + R_2 i_b + \frac{1}{CD}i_b$

$i_b = CDy(t)$

$f(t) = LDi_a + R_1(i_a - i_b)$

$0 = R_1(i_b - i_a) + R_2 i_b + \frac{1}{CD}i_b$

$f(t) = LD\left(i_b + \frac{R_2}{R_1}i_b + \frac{1}{R_1CD}i_b\right)$

$+ R_1\left(i_b + \frac{R_2}{R_1}i_b + \frac{1}{R_1CD}i_b\right) - R_1 i_b$

$i_b = CDy(t)$

$f(t) = LDi_b + D\frac{R_2}{R_1}Li_b + \frac{L}{R_1C}i_b$

$+ R_1 i_b + R_2 i_b + \frac{1}{CD}i_b - R_1 i_b$

$0 = R_1 i_b - R_1 i_a + R_2 i_b + \frac{1}{CD}i_b$

$R_1 i_a = (R_1 + R_2)i_b + \frac{1}{CD}i_b$

$i_a = \frac{(R_1 + R_2)i_b + \frac{1}{CD}i_b}{R_1}$

$i_a = \frac{(R_1 + R_2)i_b}{R_1} + \frac{1}{R_1CD}i_b$

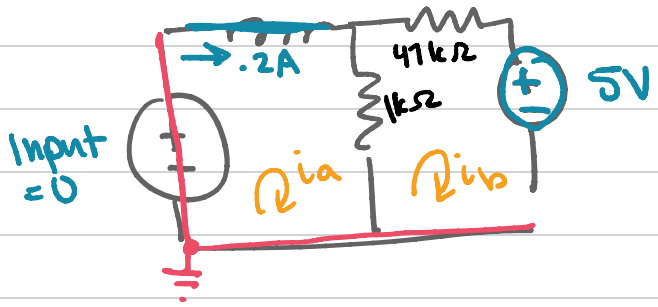
$= i_b + \frac{R_2}{R_1}i_b + \frac{1}{R_1CD}i_b$

$f(t) = LCD^2y(t) + CD^2\frac{R_2}{R_1}Ly(t) + \frac{LD}{R_1}y(t) + R_2CDy(t) + y(t)$

$= \left(LC + C\frac{R_2}{R_1}L\right)D^2y(t) + \left(\frac{L}{R_1} + R_2C\right)Dy(t) + y(t) = f(t)$

$$\left(\frac{3}{125}D^2 + \frac{941}{200}D + 1\right)y(t) = f(t)$$

(b) $\dot{y}(t) = ?$ $\quad i_L(0) = .2A$ $\quad y(0) = 5$



$i_a = .2A$

$-5V = 1k(i_b - .2A) + 47k(i_b)$

$i_b = 4.063mA$

$i_b = C D y(0)$

$\frac{4.063mA}{C} = Dy(0) = \boxed{40.625}$

(c) $\frac{3}{125}\lambda^2 + \frac{941}{200}\lambda + 1 = 0$

$\boxed{\lambda = -195.829, \quad \lambda = -0.212771}$

(d) $x(t + \Delta t) = (I + A\Delta t)\,x(t)$

$\left(\frac{3}{125}D^2 + \frac{941}{200}D + 1\right) y(t) = f(t)$

$\left(D^2 + {}^{4705}/_{24}\, D + {}^{125}/_{3}\right) y(t) = f(t)$

$A = \begin{bmatrix} 0 & 1 \\ -4705/24 & -125/3 \end{bmatrix}$

$x = \begin{bmatrix} y(t) \\ \dot{y}(t) \end{bmatrix}$

$$\boxed{\begin{bmatrix} y(t+\Delta t) \\ \dot{y}(t+\Delta t) \end{bmatrix} = \left(\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ \frac{-4705}{24} & \frac{-125}{3} \end{bmatrix} \cdot \Delta t \right) \begin{bmatrix} y(t) \\ \dot{y}(t) \end{bmatrix}}$$