Report for Assignment4

Overview

In this project, we compose the transient simulation of circuits into ODE and try to construct functions to solve the generic ODE. We have implemented three methods: Forward Euler method, RK4 method and RK34 method.

Modular Design & Code Reuse :

We wrap all methods in ODEModel class.

Then the user can get new value for the next timestamp using *ODESolver* method. In this method, we construct *increFunc* to keep general time stepping scheme: $X_{i+1} = X_i + ¢*h$. Besides, in *increFunc*, different method can be called through "switch case", for example, when case == 1, the user is able to get new value using one-step method, and when case ==2, the user can use forward euler method. In this way, we can simply add more solutions just by adding more "switch case", which implement good code reusability. Also, for different problems, we can just edit *fxSolver* to change differential equation, for task 3 &4 &5, the only thing we need to change are the step size h, initial guess X as well as differential equation in *fxSolver*.

On one hand, we reuse functions we implemented in previous assignment to be the helper functions in this project; on the other hand, we define our functions to be as general as possible to make them reusable in later projects.

Analysis of output

Task 2:

Revised:

1. As vectors in previous problems are all single vectors, I now edit the original functions to allow them to handle multi-vector. Besides, I usually wrap the function before, for example, I made a function called oneStep which can output new X directly. However, for generalization in the following work, I create ***increment***

function individually so that different increment value could be returned depending on our choice.

2. For different ordinary differential equations corresponding to different circuits, we implement different methods to calculate derivatives which make our solver more generic.

Inherited:

1. The issue-resolution steps are still the same as before. That is, we are still supposed to calculate increment function, and then update x accordingly. Therefore, the basic logic has been inherited.

2. The way to calculate the parameters (k1, k2 …) that are required during increment function calculation is the same with previous practice. Also, the method of calculating $I_{D,EKV}$ is the same with EKV model in assignment3.

Task 3:

In the validation part, we tested ***ODESolver*** in three different methods, but this time, the input vector now are multi-vectors. Knowing the ground truth, we then calculated relative error for three methods and we observed that the relative error is accumulating. However, RK34 with time adaptation has better performance than other two methods.

Task4

In task4, we obtain V1 and V2 with given parameters. The output V1(t) and V2(t) are periodic voltages which change in value, reflecting that the capacitor variably acts between two extremes “on/off” as transient i(t) changes. Therefore, the RC circuits are like a low-pass filter.

As for the comparison of the results from all three one-step method, we haven’t noticed obvious difference.

Task5

In this task, we also obtain V1 and V2 with given parameters. According to the figures we output, we can observe that the CS amplifier amplifies $\Delta V1$ to $\Delta V2$ and V2(t) will be bounded between 0 and 5V which are correspond to expected results.

As for the comparison of the results from all three one-step method, we haven't noticed obvious difference.