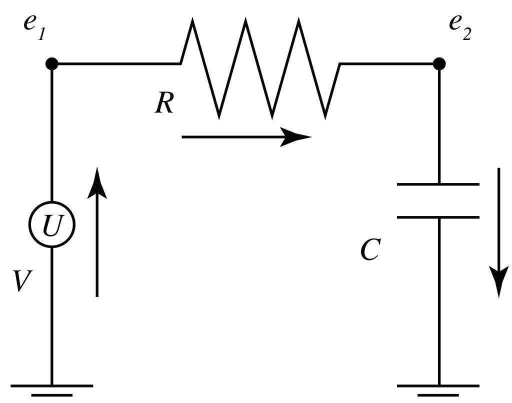
Homework #3

ES_APPM 346-0

Due Feb. 19, 2020

Introduction

Differential algebraic equations come up in various places. One place they frequently arise is in models of electrical circuits derived using a modified nodal analysis method. For this project, we'll look at the circuit illustrated below:



In this example, it consists of a variable voltage source, $V(t)$, on the left, a resistor with resistance R, and a capacitor with capacitance C. Let I_V, I_R, and I_C represent the currents through the voltage source, the resistor, and the capacitor, respectively. Let e_1 and e_2 be the voltages at the nodes relative to the ground voltage.

Kirchoff's Current Law states that sum of all the currents through a node is zero. Checking the currents at the nodes marked e_1 and e_2, we get the equations

$$-I_V + I_R = 0, \qquad -I_R + I_C = 0.$$

Kirchoff's Voltage Law states that the sum of the voltages around a loop sum to zero. In this case, we can measure the differences in voltage between the ground, and the nodes e_1, e_2. We assume the ground state has zero voltage, and the voltage source has to match the difference in voltage between the ground and the node e_1, hence $V_V = 0 - e_1$. Similarly, the drop in voltage across the resistor is $V_R = e_1 - e_2$, and the drop in voltage across the capacitor is $V_C = e_2 - 0$.

For an ideal resistor, the voltage is related to the current by $V_R = RI_R$. For a capacitor, the current is related to the change in voltage, hence $i_C = C\frac{dV_C}{dt}$.

Collecting all the equations, we get

$$
\begin{aligned}
-I_V + I_R &= 0 \\
-I_R + I_C &= 0 \\
V_V &= -e_1 \\
V_R &= e_1 - e_2 \\
V_C &= e_2 \\
V_R &= R I_R \\
I_C &= C \frac{dV_C}{dt}
\end{aligned}
$$

We can simplify this system by doing some substitutions, for example, since $I_R = \frac{1}{R} V_R$ and $V_R = e_1 - e_2$, then we reduce the system to

$$
\begin{aligned}
-I_V + \frac{1}{R}(e_1 - e_2) &= 0 \\
-\frac{1}{R}(e_1 - e_2) + I_C &= 0 \\
V_V &= -e_1 \\
V_C &= e_2 \\
I_C &= C \frac{dV_C}{dt}
\end{aligned}
$$

Next, note that $I_C = C \frac{dV_C}{dt} = C \frac{de_2}{dt}$ and the voltage source will be a given time dependent function $V_V = V(t)$, so that we have, after rearranging the equations a bit, the final version of the system:

$$
\begin{aligned}
\frac{de_2}{dt} &= \frac{1}{RC}(e_1 - e_2) && (1) \\
0 &= -I_V + \frac{1}{R}(e_1 - e_2) && (2) \\
0 &= V(t) + e_1 && (3)
\end{aligned}
$$

where here the dependent variables are e_1, e_2, and I_V, and the independent variable is t. We will solve this system in the current project.

Written Assignment

1. Identify which variables are differential variables, and which variables are algebraic variables.

2. Determine the index of this differential algebraic system.

3. This system is simple enough that we can actually compute an exact solution. Suppose $V(t) = A\sin(\omega t)$. Follow these steps to compute the general solution:

   (a) Use (3) to determine e_1.

   (b) Use (1) to determine e_2.

   (c) Use (2) to determine I_V.

   Describe all the possible initial conditions at $t = 0$ such that $e_2(0) = E_2$ given that R, L, C, A, and ω and known constants. Find the unique solution that has the initial condition $e_1(0) = e_2(0) = I_V(0) = 0$.

4. If the system were written in the standard form

$$\mathbf{y}' = \mathbf{F}(t, \mathbf{y}, \mathbf{z})$$
$$0 = \mathbf{G}(t, \mathbf{y}, \mathbf{z})$$

   Write the following expressions: $\mathbf{F}$, $\mathbf{G}$, $\partial\mathbf{F}/\partial\mathbf{y}$, $\partial\mathbf{F}/\partial\mathbf{z}$, $\partial\mathbf{G}/\partial\mathbf{y}$, $\partial\mathbf{G}/\partial\mathbf{z}$, and the residual and the Jacobian matrix you will need for the Quasi-Newton iteration solver that solves the system:

$$\mathbf{y}_{n+1} = \gamma\Delta t\mathbf{F}(t_{n+1}, \mathbf{y}_{n+1}, \mathbf{z}_{n+1}) + B$$
$$0 = \mathbf{G}(t_{n+1}, \mathbf{y}_{n+1}, \mathbf{z}_{n+1})$$

   where γ and B are given constants determined by whether BDF-1 or BDF-2 implicit method is being used.

Programming Assignment

For the programming project, you may use your program from Project #2 as a starting point for this project.

1. Modify your `solver` method from the previous project to be called

```
function [y,z] = daesolver(k, t, gam, dt, B, yn, zn, tol)
```

   Note first that to reduce (or perhaps add to) confusion, the variable C from the BDF method is changed to B here so that it is not confused with the constant C that appears in the equations to be solved. In addition to that change of variable there are four other changes: The parameter variable `k` should contain for values, R, C, ω, and A, (2) the function takes an extra argument, `t`, that is the current time plus Δt, in other words t_{n+1} when solving for y_{n+1}, (3) the function takes an extra argument, `zn`, that is the value of the algebraic variables from the previous time step, similar to the `yn` argument that was there before, and (4) the function returns two values, $\mathbf{y}_{n+1}$ and $\mathbf{z}_{n+1}$. You will need to modify your residual and your Jacobian so that it incorporates the algebraic equations. Test your code with the following line:

```
[y,z] = daesolver([1,1,pi,0.01],1e-3,1,1e-3,0,0,[0;0],1e-12)
```

   The answer should be:

```
y =

   -3.1384e-08

z =

   1.0e-04 *

   -0.3142
   -0.3138
```

2. Write a solver that uses BDF 1 with fixed time steps to solve the system (1–3) with parameter values $R = C = 1$, $A = 10^{-2}$, $\omega = \pi$, and with initial conditions $e_1 = e_2 = I_V = 0$. Solve it to time $T = 6$ using $N = 1000$ steps. In this case, you may compare your results at time $T = 6$ with the exact solution derived in the written assignment to get the error. By taking $N = 2000$, and $N = 4000$, confirm that you again achieve first order accuracy. Submit the error data with your written assignment.

3. Repeat the previous problem, but with BDF 2. Again, check your results to confirm you have a second order accurate solution.

4. Modify your adaptive time stepping code from project 2 to solve this system. Set the tolerance to be 10^{-5}. Compare the final solution with the exact solution and determine whether the tolerance is actually met. Submit a plot of all three variables, e_1, e_2, and I_V for your solution with your written assignment. Name your function

```
function [t,y,z] = dae(T, idt, k, y0, z0, tol, itol)
```

   where again the arguments are the same as for the BDF method of Project 2 except for the addition of the initial algebraic variable values `z0`. See Matlab Grader for further code formatting instructions and to submit your code.

5. **[EXTRA CREDIT]** Implement a fixed time step solver using the Radau 5 implicit Runge-Kutta method. If you choose to do this, you should submit your code as a function

```
function [t,y,z] = radau5(T, N, k, y0, z0, itol)
```

   Note that the global error tolerance is not required because you will do a fixed number N time steps. Compare the accuracy of the Radau 5 method using $N = 20$ with the fixed time step BDF-2 method.