Exercise 16.1 freud

In Exercise 14.1 nlin., you derived a nonlinear state-space model for the RLC circuit of Fig. nlin.1, which includes a nonlinear capacitor, and linearized the state equation about an operating point. Use these results to perform the following analysis.

a. Write a program to simulate the nonlinear state-space model for initial condition $\mathbf{x}(0) = \begin{bmatrix} 1 & 0 \end{bmatrix}^T$ and step input $\mathbf{u}(t) = 5u_s(t)$. Let $R = 10\ \Omega$, $L = 1$ mH, and $k = 10^{-6}$. Try simulating for 1 ms.
b. Add to the program the simulation of the *linearized* system for the same initial condition and input.
c. Compare (by graphing) the nonlinear and linearized step responses. (Don't forget that $\mathbf{x}^* \neq \mathbf{x}$!)

Solution for exercise 16.1 sim.

```
clear; close
save_figures = 1;
```

Part a: nonlinear solution

We begin by defining the nonlinear state equation in the following function file (could be a local function).

```
type ode_15_1a.m
```

```
function f = ode_nonlin(t,x)
    k = 1e-6;
    R = 10;
    L = 1e-3;
    VS = 5;
    f = [ ...
      2/(3*k*sqrt(x(1)))*x(2); ...
      1/L*(-R*x(2) + VS - x(1)) ...
    ];
end
```

Now let's use ode45 for the simulation.

```
x0 = [1;0]; % initial condition
t_a = linspace(0,1e-3,200);
[~,x_a] = ode45( ...
    @ode_15_1a, ... % ODE function handle
    t_a, ... % time array
    x0 ... % initial state
);
```

Now we plot. Two vertical axes are used so the full range of each state variable is usefully displayed.

```
figure
yyaxis left
plot(1e3*t_a,x_a(:,1),'linewidth',1.5)
xlabel('time (ms)')
ylabel('v_C (V)')

yyaxis right
plot(1e3*t_a,1e3*x_a(:,2),'linewidth',1.5)
ylabel('i_L (mA)')
```

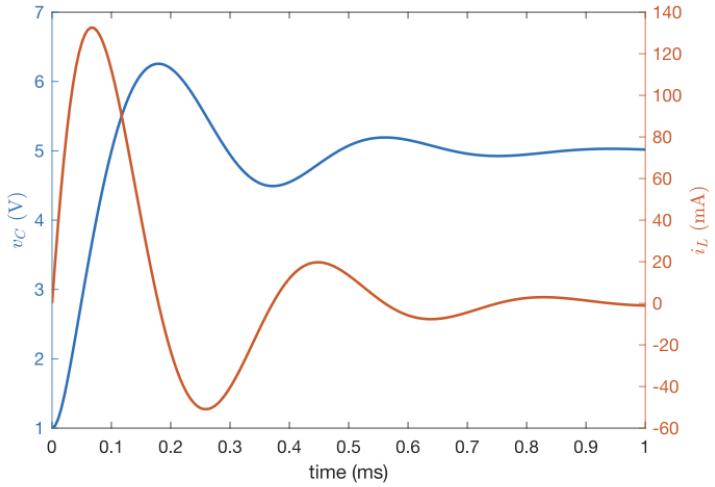


Figure sim.1: nonlinear step response.

Part b: linear solution

Instead of using the usual linear simulatino methods, for parallelism with the nonlinear solution, we begin by defining the linear state equation in the following function file (could be a local function).

```
type ode_15_1b.m
```

```
function f = ode_nonlin(t,x)
    % not performant really
    k = 1e-6;
    R = 10;
    L = 1e-3;
    VS = 5;
    u = [VS];
    A = [ ...
      0, 2/(3*k*sqrt(5)); ...
```

```
      -1/L, -R/L ...
    ];
    B = [ ...
      0; 1/L ...
    ];
    f = A*x + B*u; % linear state eq!
end
```

Now let's use `ode45` for the simulation.

```
x0 = [1;0]; % initial condition
t_a = linspace(0,1e-3,200);
[~,x_b] = ode45( ...
    @ode_15_1b, ... % ODE function handle
    t_a, ... % time array
    x0 ... % initial state
);
```

We could plot the linear solution alone, but let's just plot them together, which is the next part.

Part c: plotting the linear and nonlinear solutions together

We use a similar approach.

```
figure
yyaxis left
hold on
plot(1e3*t_a,x_a(:,1),'linewidth',1.5)
plot(1e3*t_a,x_b(:,1),'linewidth',1.5)
xlabel('time (ms)')
ylabel('v_C (V)')

yyaxis right
plot(1e3*t_a,1e3*x_a(:,2),'linewidth',1.5)
plot(1e3*t_a,1e3*x_b(:,2),'linewidth',1.5)
hold off
ylabel('i_L (mA)')
legend('nonlinear v_C', ...
    'linear v_C', ...
```

```
    'nonlinear i_L', ...
    'linear i_L' ...
)
```

These plots show that the linear solution is pretty good, but would not be sufficient for precise analysis.

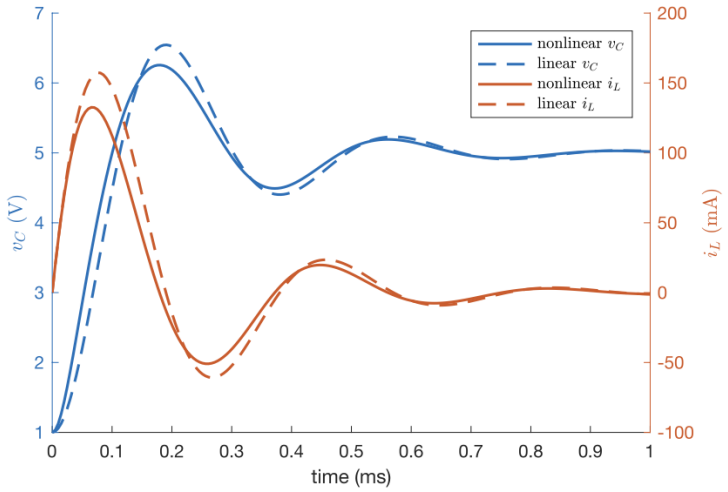


Figure sim.2: comparison of nonlinear and linear step responses.

Exercise 16.2 kafka

Let the nonlinear state equation of a circuit like Fig. nlin.2, including a diode, be

$$\begin{aligned}\frac{d\mathbf{x}}{dt} &= \mathbf{f}(\mathbf{x},\mathbf{u}) \\ &= \begin{bmatrix} \frac{1}{C} i_L \\ \frac{1}{L}(-V_{TH}\ln(i_L/I_s+1) - Ri_L + V_S - v_C) \end{bmatrix}.\end{aligned}$$

a. Write a program to simulate the nonlinear state-space model for initial condition $\mathbf{x}(0) = \begin{bmatrix} 0 & 0 \end{bmatrix}^\top$ and input $\mathbf{u}(t) = 1 + 0.1\cos(8000\pi t)$. Let $I_s = 10^{-12}$ A, $V_{TH} = 25$ mV, $R = 10\ \Omega$, $L = 1$ mH, and $C = 10\mu$F. Try simulating for 1 ms. *Hint*: the ode is stiff, so simulate with `ode23s`.
b. Add to the program the simulation of the *linearized* system (with operating point $\mathbf{x}_o = \begin{bmatrix} 0 & I_s \end{bmatrix}^\top$, $u_o = 0$) with A and B matrices

$$A = \begin{bmatrix} 0 & 1/C \\ -1/L & -(V_{TH}/(2I_s) + R)/L \end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix} 0 \\ 1/L \end{bmatrix}$$

for the same initial condition and input.
c. Compare (by graphing) the nonlinear and linearized step responses. (Don't forget that $\mathbf{x}^* \neq \mathbf{x}$!)

Solution for exercise 16.2 sim.

Begin by defining parameters globally.

```
global R L C VTH IS
R = 10; % Ohms
L = 1e-3; % H
C = 10e-6; % F
VTH = 25e-3; % V
IS = 1e-12; % A
```

Now define the initial condition, input, and a time array for simulation.

```
x0 = [0;0]; % initial state
u = @(t) 1 + .1*cos(4*2*pi*1e3*t);
t_a = linspace(0,1e-3,700);
```

Part a: nonlinear solution

We begin by defining the nonlinear state equation in the following function file (could be a local function).

```
type kafka_ode.m
```

```
function dxdt = kafka_ode(t,x,u)
    global R L C VTH IS
    dxdt = [ 1/C*x(2);
        1/L*(u(1) - R*x(2) - VTH*log(x(2)/IS+1) - x(1)) ];
end
```

Let's first try `ode45` for the simulation.

```
[~,x] = ode45( ...
    @(t,x) kafka_ode(t,x,u(t)), ... % ODE ...
    t_a, ... % time array or span
    x0 ... % initial state
);
```

We now have our initial solution array `x`, which represents our state solution $x(t)$. Let's plot the solution.

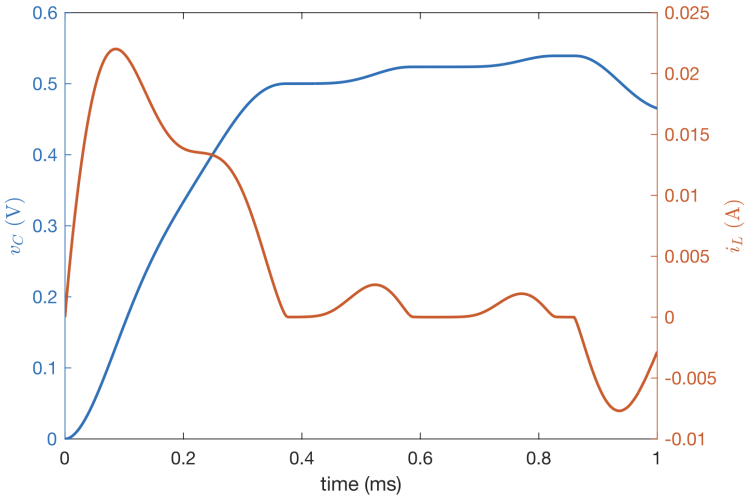
```
figure
yyaxis left
plot(1e3*t_a,x(:,1),'linewidth',1.5)
ylabel('v_C (V)')
yyaxis right
plot(1e3*t_a,x(:,2),'linewidth',1.5)
xlabel('time (ms)')
ylabel('i_L (A)')
```

Figure sim.3: nonlinear system response via `ode45`.

```
Warning: Imaginary parts of complex X and/or Y arguments ignored.
Warning: Imaginary parts of complex X and/or Y arguments ignored.
```

There's something rotten in the state of Denmark. First of all, there are imaginary parts being ignored in the plots (see the warnings above), which is a bad sign (we could check them out to see how significant they are relative to the real parts). Even worse, as we can see in Fig. sim.3, a *negative* current $i_L = i_D$ is predicted at certain times. This is literally impossible with a proper solution of this model because our diode model will not allow for negative current.

The most likely culprit here is the numerical solution. Our ODE is actually "stiff," and would be better solved with a stiff solver like `ode23s`. Let's try that.


0.6
0.5
0.4
0.3
0.2
0.1
0
v_C (V)
0.025
0.02
0.015
0.01
0.005
0
-0.005
-0.01
i_L (A)
0
0.2
0.4
0.6
0.8
1
time (ms)

```
[~,x] = ode23s( ...
    @(t,x) kafka_ode(t,x,u(t)), ... % ODE ...
    t_a, ... % time array or span
    x0 ... % initial state
);
```

We now have our initial solution array `x`, which represents our state solution $x(t)$. Let's plot the solution.

```
figure
yyaxis left
plot(1e3*t_a,x(:,1),'linewidth',1.5)
ylabel('v_C (V)')
yyaxis right
plot(1e3*t_a,x(:,2),'linewidth',1.5)
xlabel('time (ms)')
ylabel('i_L (A)')
```

Figure sim.4: nonlinear system response via ode23s.

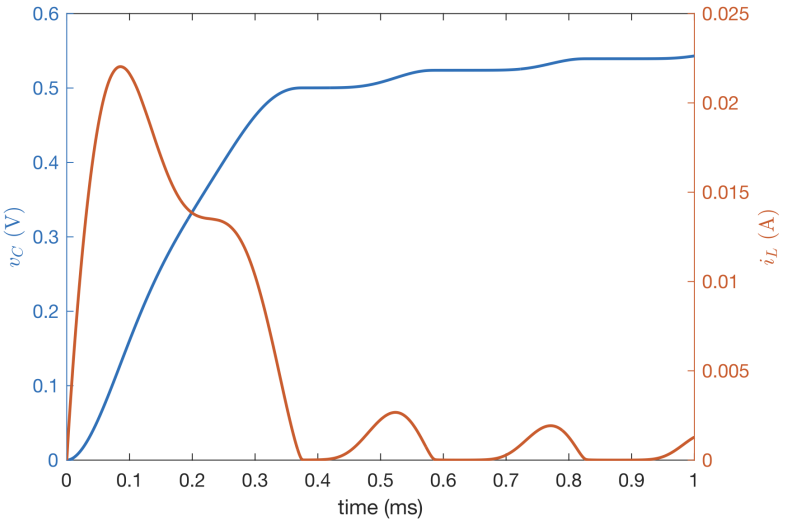
The result, shown in Fig. sim.4, had no "imaginary part" warnings and satisfies our knowledge that $i_L(t) > 0$. And the response makes sense: the capacitor charges, the current stays non-negative, oscillations appear similar to the input frequency, and even three "cutoffs" of current when the system is tending toward negative current.

Part b: linear solution

Instead of using the usual linear simulation methods, for parallelism with the nonlinear solution, we begin by defining the linear state equation in the following function file (could be a local function).

```
type kafka_ode_lin.m
```

```
function dxdt = kafka_ode_lin(t,x,u)
    global R L C VTH IS
    A = [ ...
      0, 1/C; ...
      -1/L, -(VTH/(2*IS)+R)/L ...
```


0.6
0.5
0.4
0.3
0.2
0.1
0
v_C (V)
0.025
0.02
0.015
0.01
0.005
0
i_L (A)
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
time (ms)

```
    ];
    B = [ ...
      0; 1/L ...
    ];
    dxdt = A*x + B*u; % linear state eq!
end
```

Out of curiousity, what are the eigenvalues?

```
A = [ ...
      0, 1/C; ...
      -1/L, -(VTH/(2*IS)+R)/L ...
    ];
evals = eig(A);
disp(evals(1))
disp(evals(2))
```

```
     0

  -1.2500e+13
```

Interesting: a zero-eigenvalue and a large, negative, real one. The zero-eigenvalue corresponds to a capacitor voltage that grows without bound (in the model).
Now let's use `ode23` for the simulation.

```
options = odeset( ... % solver options
  'AbsTol',1e-13, ...
  'RelTol',1e-13 ...
);
[~,x_lin] = ode23s( ...
    @(t,x) kafka_ode_lin(t,x,u(t)), ... % ODE
    t_a, ... % time array
    x0, ... % initial state
    options ...
);
```

Now let's remember that this is the solution for $\mathbf{x}^*(t) = \mathbf{x}(t) - \mathbf{x}_o(t)$. Let's "correct" for the operating point in the solution, even though it's quite small.

```
xo = [0;IS]; % operating point
x_lin = x_lin + xo.';
```

We could plot the nonlinear and linear solutions together, but they're so different we choose to plot the separately.

```
figure
yyaxis left
hold on
plot(1e3*t_a,x_lin(:,1),'linewidth',1.5)
xlabel('time (ms)')
ylabel('v_C (V)')
yyaxis right
plot(1e3*t_a,1e3*x_lin(:,2),'linewidth',1.5)
hold off
ylabel('i_L (mA)')
legend( ...
    'linear v_C', ...
    'linear i_L', ...
    'location','southeast' ...
)
```

Figure sim.5: linear system response.

part c: comparison

The plots of Fig. sim.4 and Fig. sim.5 show that the linear solution is absolutely *terrible*. This is to be expected with such strong nonlinearities and a state that varies significantly from the "operating point" assumed by the linearized model.