

Lecture I

1D FDTD - how to solve a transient circuit?

In this lecture, we start to study the so-called Finite-Difference Time-Domain (FDTD) method (FDTD). We intend to solve a (linear or not) first- and second-order transient circuit.

1. Transient circuits

All three basic circuit elements - resistor, inductor, and capacitor - are described by their own "Ohm's law" that relates the voltage and the current in the passive reference configuration as shown in Fig. 1.

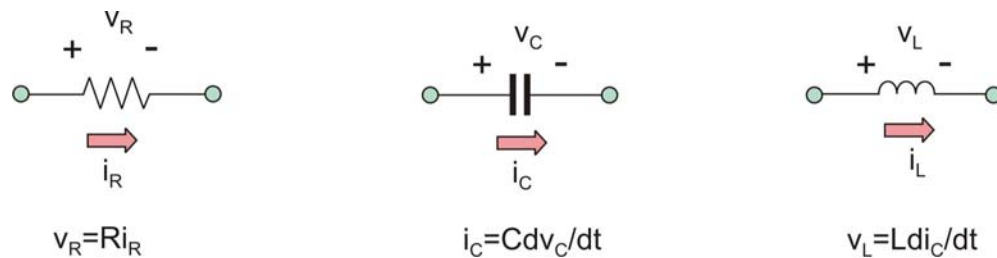


Fig. 1. Three linear circuit elements - building blocks of any electric circuit. All elements use the passive reference configuration.

When these differential relations are substituted into the KVL and the KCL - a differential circuit equation may be obtained. As long as resistance R , capacitance C , and inductance L are constants (do not depend on circuit parameters), this equation is linear.

2. Charging a capacitor

The corresponding circuit is shown in Fig. 2. The switch closes at $t = 0$. The resistor R in Fig. 2 can be either the series parasitic resistance of the capacitor itself or the load resistance. In fact, it can even be a combination of source, load, and parasitic resistances.

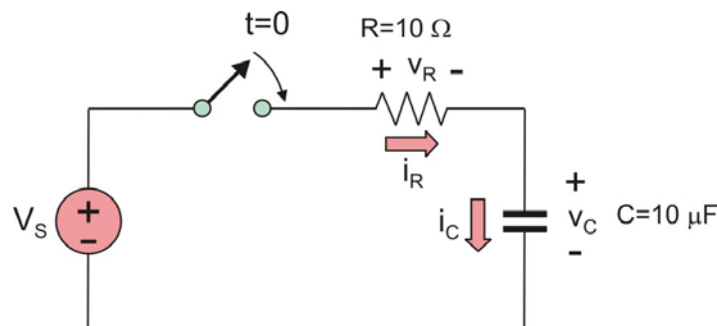


Fig. 2. Charging the capacitor with the voltage power supply.

The differential equation to this dynamic circuit is based on the use of the KVL and the KCL. Specifically, the KCL yields

$$i_C = i_R \quad (1)$$

at any time instance t . Since both circuit elements in Fig. 2 are in a passive reference configuration, we could apply the constitutive relations between currents and voltages without changing the sign:

$$C \frac{dv_C}{dt} = \frac{v_R}{R} = \frac{V_s - v_C}{R} \quad (2)$$

This is because the KVL states

$$v_R = V_s - v_C \quad (3)$$

at any positive time instance t . Eq. (2) yields

$$C \frac{dv_C}{dt} + \frac{v_C}{R} = \frac{V_s}{R} \Rightarrow \frac{dv_C}{dt} + \frac{v_C}{\tau} = \frac{V_s}{\tau}, \quad \tau = RC \quad (4)$$

3. Analytical solution

Eq. (4) has an excitation term (the power supply voltage) on its right-hand side. Therefore, the solution to this equation is called a *forced response*¹. However, Eq. (4) still remains a first-order transient equation with the same time constant τ . The solution to any first-order ordinary differential equation of the type (4), with a constant right-hand side, has the form

$$v_C(t) = K_1 \exp\left(-\frac{t}{\tau}\right) + K_2 \quad (5a)$$

This fact can be checked by direct substitution. The two terms containing the exponential factor will cancel out after differentiation. The remaining terms in Eq. (4) yield

$$\frac{K_2}{\tau} = \frac{V_s}{\tau} \Rightarrow K_2 = V_s \quad (5b)$$

The constant parameter K_1 can be determined from the initial condition, $v_C(t=0) = 0$. This yields

¹ From a mathematical point of view, Eq. (4) is known as an inhomogeneous first-order differential equation. This is in contrast to the homogenous differential equation.

$$K_1 + K_2 = 0 \Rightarrow K_1 = -V_s \quad (5c)$$

Thus, the circuit voltages have the form

$$v_C(t) = V_s \left[1 - \exp\left(-\frac{t}{\tau}\right) \right], \quad v_R(t) = V_s \exp\left(-\frac{t}{\tau}\right) \quad (5d)$$

Example 1: A 10 μF capacitor in Fig. 2 is charged by a 10V voltage power supply. The switch closes at $t = 0$, and the system resistance is 10Ω . Plot the capacitor voltage v_C and the capacitor current i_C to scale versus time over the time interval from -0.2 to 0.5 ms.

Solution: First, we find the time constant τ . According to Eq. (4), or Eq. (5), $\tau = RC = 10^{-5} \text{F} \times 10\Omega = 10^{-4} \text{s} = 0.1 \text{ms}$. The solution to the example is given by Eqs. (5d) and with $V_s = 10\text{V}$. The MATLAB script given below plots the corresponding solution. The output of the script is depicted in Fig. 3. The vertical line in both plots is the time constant τ .

One can see that the time constant determines how fast the capacitor charges. At $\tau = 1$, the capacitor is charged to $(1 - 1/e)V_s$ or to 63.2% of the supply voltage V_s .

```
clear all
C      = 10e-6;           % capacitance, F
R      = 10;              % load resistance, Ohm
tau    = R*C              % time constant, sec
Vs     = 10;              % power supply voltage, Vs
dt     = 0.01*tau;        % discretization step
t      = [-2*tau:dt:10*tau]; % time vector, sec

% Analytical solution
vC     = Vs*(1 -exp(-t/tau)); % capacitor voltage
vC(find(t<0)) = 0;

plot(t*1e3, vC, 'Color', 'm', 'Linewidth', 8); grid on;
xlabel('time, ms'); ylabel('capacitor voltage v_C, V');
axis([-0.2 1 0 15])
```

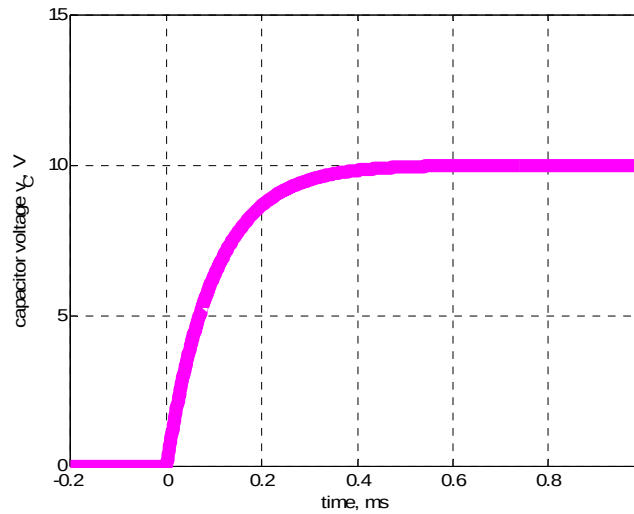


Fig. 3. Voltage plot for charging the $10\mu\text{F}$ capacitor in series with the 10Ω resistor. The thick curve corresponds to the capacitor voltage. The time constant, $\tau=0.1\text{ms}$, determines the scale at which the voltage reaches 0.632 or 63.2% of the power supply voltage $V_s = 10\text{V}$.

4. Single-step numerical method - the first-order Euler method

Eq. (4) is to be written in a more general form

$$\frac{df}{dt} = F(f, t), \quad f = v_C, \quad F = -\frac{v_C}{\tau} + \frac{V_s}{\tau} \quad (6)$$

The time-domain discretization of Eq. (6) is introduced by subdividing the time domain into N intervals of size Δt in the form

$$f_n = f(t_n), \quad t_n = (n-1)\Delta t, \quad n = 1, 2, \dots, N \quad (7)$$

We use a numerical approximation scheme which is *first-order accurate*; this means that the error in a *finite-difference* approximation of Eq. (6) at each interval Δt is proportional to Δt^2 . This is the *Euler method*, which has the finite-difference approximation in the form

$$\frac{f_{n+1} - f_n}{\Delta t} \approx F(f_n, t_n) \Rightarrow f_{n+1} = f_n + \Delta t F(f_n, t_n) \quad (8)$$

Eq. (8) implies that we perform *marching on in time*, i.e. find voltage f_{n+1} at step $n+1$ according to Eq. 8 once voltage f_n at step n is known. We indeed start with zero voltage of the capacitor at the first step, $f_1 = 0, t_1 = 0$.

Example 2: Solve Eq. (4) for charging the capacitor numerically and compare your numerical solution with the analytical solution from Fig. 3. The $10\mu\text{F}$ capacitor in series with the 10Ω resistor is charged by a DC voltage source of $V_s = 10\text{V}$.

Solution: The following MATLAB script implements the Euler method described by Eqs. (8) for the differential voltage equation (4). The result of the numerical simulation is reported in Fig. 4a. It is not distinguishable from the corresponding analytical solution. We note that the numerical method itself is described by one (!) line of code in the *for* loop.

```
clear all
C      = 10e-6;           % capacitance, F
R      = 10;              % load resistance, Ohm
tau    = R*C              % time constant, sec
Vs     = 10;              % power supply voltage, Vs
dt     = 0.01*tau;        % discretization step
t      = [-2*tau:dt:10*tau]; % time vector, sec

% Analytical solution
vC = Vs*(1 - exp(-t/tau)); % capacitor voltage
vC(find(t<0)) = 0;

plot(t*1e3, vC, 'Color', 'm', 'LineWidth', 8); grid on;
xlabel('time, ms'); ylabel('capacitor voltage v_C, V');
axis([-0.2 1 0 15])

% Numerical solution (first-order Euler)
vCn = zeros(size(t));
N1 = min(find(t>eps));
N2 = length(t);
for n = N1-1:N2-1;
    vCn(n+1) = vCn(n) + dt/tau*(Vs - vCn(n));
end

hold on;
plot(t*1e3, vCn, 'Color', 'k', 'LineWidth', 2); grid on;
xlabel('time, ms'); ylabel('capacitor voltage v_C, V');
```

It might appear at first sight that the Euler method gives a sufficiently accurate solution and does not have to be improved. However, there is a significant objection. The reason for obtaining an acceptable solution is a very small discretization step in Eq. (8),

$$\Delta t = 0.01\tau \quad (9)$$

This small discretization step implies that we need to calculate the right-hand side of Eq. (6) 1200 times in order to obtain the solution in Fig. 4a. Such a large number is legitimate

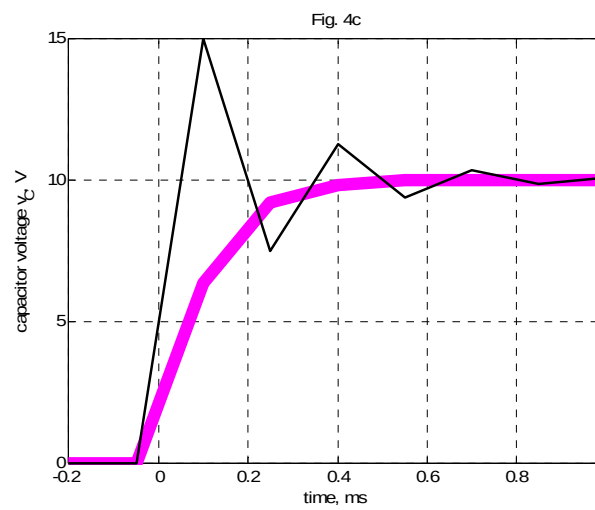
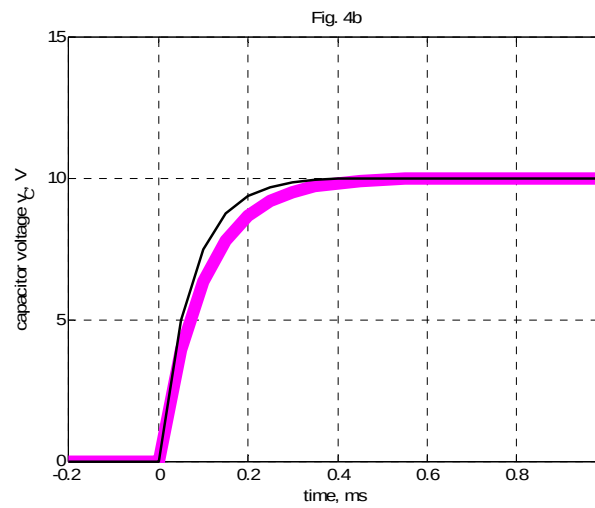
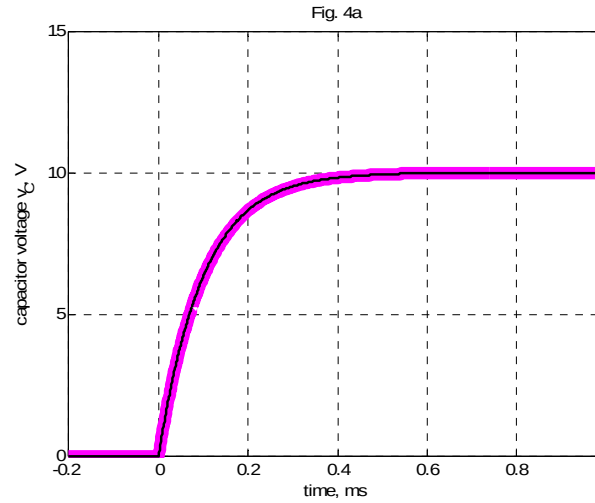


Fig. 4. Euler method for $\Delta t = 0.01\tau$ - Fig. 4a, $\Delta t = 0.5\tau$ - Fig. 4b, and $\Delta t = 1.5\tau$ - Fig. 4c. The analytical solution is shown by a thick curve.

for a simple equation, but may become a great obstacle for a large system of linear equations, which models a distributed high-frequency circuit. What happens if we increase the step size? The answer is given in Fig. 4b and Fig. 4c for $\Delta t = 0.5\tau$ and $\Delta t = 1.5\tau$ respectively. One can see that the solution initially becomes less accurate, as in Fig. 4b for $\Delta t = 0.5\tau$. Then, it falls apart (starts to become *divergent*) as in Fig. 4c for $\Delta t = 1.5\tau$.

Ideally, however, a good numerical method should follow the solution behavior at the discretization intervals on the order of τ , such as those in Fig. 4b and Fig. 4c.

5. Stability

To evaluate the meaning of *stability* of a finite-difference numerical method, we consider a simplified version of Eq. (6), without the DC voltage term:

$$\frac{df}{dt} = F(f, t), \quad F = -\frac{f}{\tau} \quad (10)$$

This equation corresponds to a discharge of a capacitor. The Euler method has the form,

$$f_{n+1} = f_n + \Delta t \left(-\frac{f_n}{\tau} \right) \Rightarrow f_{n+1} = \left(1 - \frac{\Delta t}{\tau} \right) f_n \quad (11)$$

When

$$1 - \frac{\Delta t}{\tau} > -1 \Leftrightarrow \Delta t > 2\tau \quad (12)$$

the numerical solution starts to increase indefinitely in magnitude (and simultaneously oscillate in sign) since

$$|f_{n+1}| > |f_n| \quad (13)$$

for any n . Eq. (12) thus introduces the *stability criterion*,

$$\Delta t < 2\tau \quad (14)$$

for the present numerical method.

Fig. 5 shows what happens exactly at $\Delta t = 2\tau$, i.e. at the border of the stability domain. Large oscillations occur, which still do not rise in time.

Note that the same stability estimate (Eq. (14)) remains valid when any supply voltage V_s is included into Eq. (10), either DC or AC.

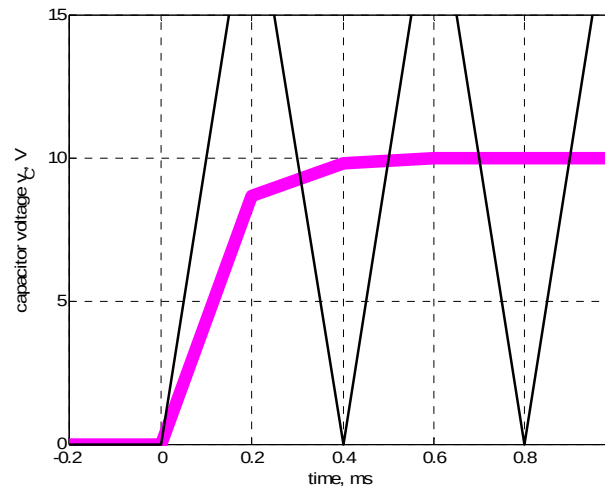


Fig. 5. The border of the stability domain, $\Delta t = 2\tau$. The analytical solution is shown by a thick curve. At any larger value of the time step the solution not only oscillates, but also increases indefinitely.

6. Single-step numerical method - the second-order Runge-Kutta method

How could we improve the results at large time steps? To do so we note that the major source of inaccuracy in the Euler method is related to the calculation of the time derivative (slope) at node n as Fig. 6a shows. If, for example, the time derivative were calculated at the midpoint between nodes n and $n+1$, much better results would be obtained - see a hypothetical Fig. 6b.

Unfortunately, we do not yet know the function value, $f_{n+1/2}$, at that midpoint $t_{n+1/2}$; we only know f_n . But couldn't we find this value using the Euler method for a half time interval Δt ?

Yes, we can! This is by the way exactly the idea of the *second-order Runge-Kutta* method - see Fig. 6c. Mathematically, then

$$\begin{aligned} f_{n+1/2} &= f_n + 0.5\Delta t F(f_n, t_n), \\ f_{n+1} &= f_n + \Delta t F(f_{n+1/2}, t_{n+1/2}) \end{aligned} \tag{15}$$

The error of the approximation given by Eq. (15) at each interval Δt is proportional to Δt^3 . The right-hand side of Eq. (6) is thus calculated twice; the information about the intermediate point is lost when we proceed to the next step.

Example 3: Solve Eq. (4) for charging the capacitor numerically using the second-order Runge-Kutta method described by Eqs. (15).

Solution: The following MATLAB script implements the second-order Runge-Kutta method described by Eqs. (15) for the differential voltage equation (4). The result of the numerical simulation is reported in Fig. 7.

```
clear all
C      = 10e-6;           % capacitance, F
R      = 10;              % load resistance, Ohm
tau    = R*C              % time constant, sec
Vs     = 10;              % power supply voltage, Vs
dt     = 0.01*tau;        % discretization step
t      = [-2*tau:dt:10*tau]; % time vector, sec

% Analytical solution
vC = Vs*(1 -exp(-t/tau)); % capacitor voltage
vC(find(t<0)) = 0;

plot(t*1e3, vC, 'Color', 'm', 'LineWidth', 8); grid on;
xlabel('time, ms'); ylabel('capacitor voltage v_C, V');
axis([-0.2 1 0 15])

% Numerical solution (second-order Runge-Kutta method)
vCn = zeros(size(t));
N1 = min(find(t>eps));
N2 = length(t);
for n = N1-1:N2-1;
    temp = vCn(n) + 0.5*dt/tau*(Vs - vCn(n));
    vCn(n+1) = vCn(n) + 1.0*dt/tau*(Vs - temp);
end

hold on;
plot(t*1e3, vCn, 'Color', 'k', 'LineWidth', 2); grid on;
xlabel('time, ms'); ylabel('capacitor voltage v_C, V');
```

Indeed, Fig. 7 may be compared to Fig. 4 which presents the same results, but obtained with the Euler method. The second-order Runge-Kutta method is clearly superior: it gives a much better accuracy for $\Delta t = 0.5\tau$ and $\Delta t = 1.5\tau$. However, it requires twice as much work as the Euler method: the right-hand side is to be calculated at the midpoint.

Therefore, for the real comparison, we should use the Euler method with the time step being *twice as small*. Now, Fig. 8 shows the result obtained with the Euler method and with the time step of $\Delta t = 0.25\tau$ versus the Runge-Kutta method with $\Delta t = 0.5\tau$. The Runge-Kutta method is still superior. In that way, its advantage has been established.

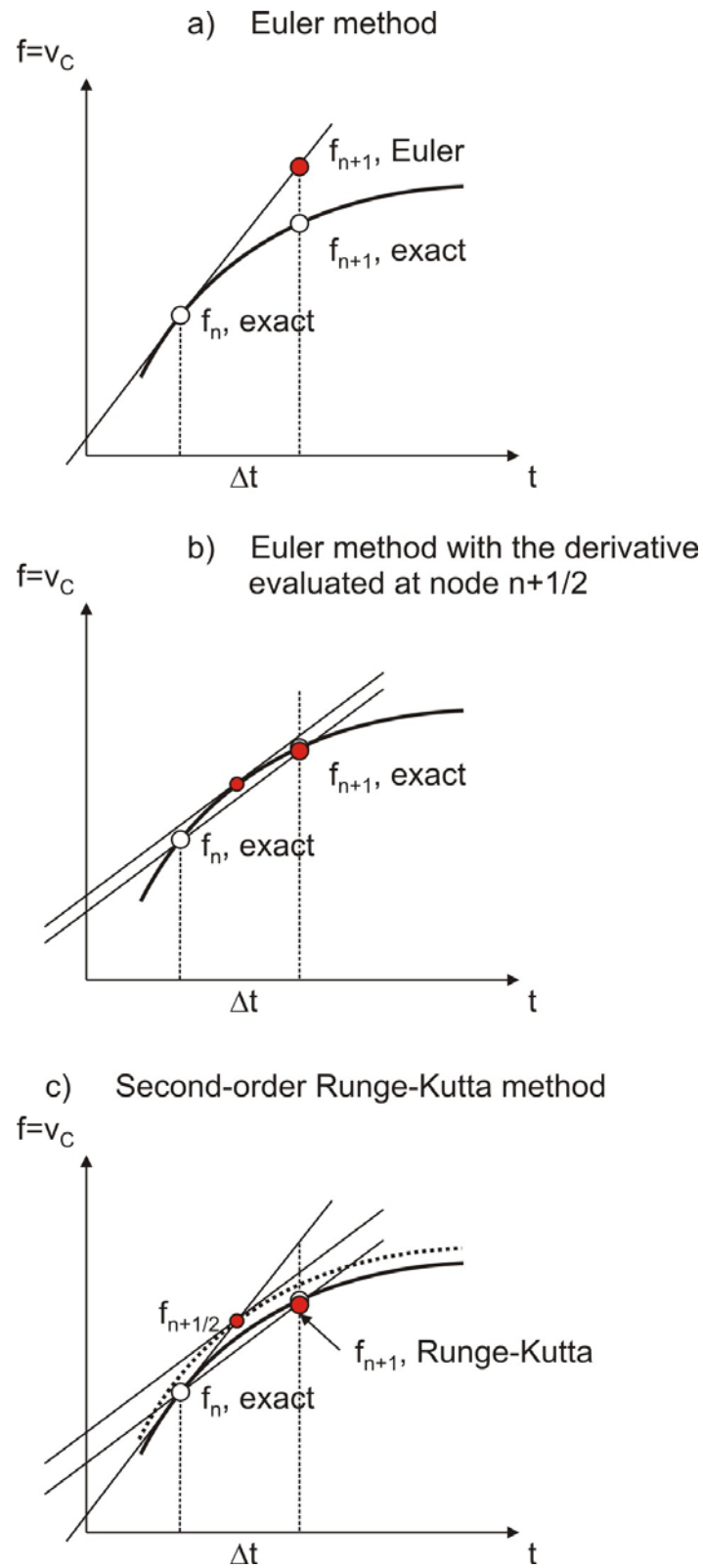


Fig. 6. An explanation of the difference between the Euler method and the second-order Runge-Kutta method. The Runge-Kutta method tries to approximately replicate the hypothetical situation shown in Fig. 6c.

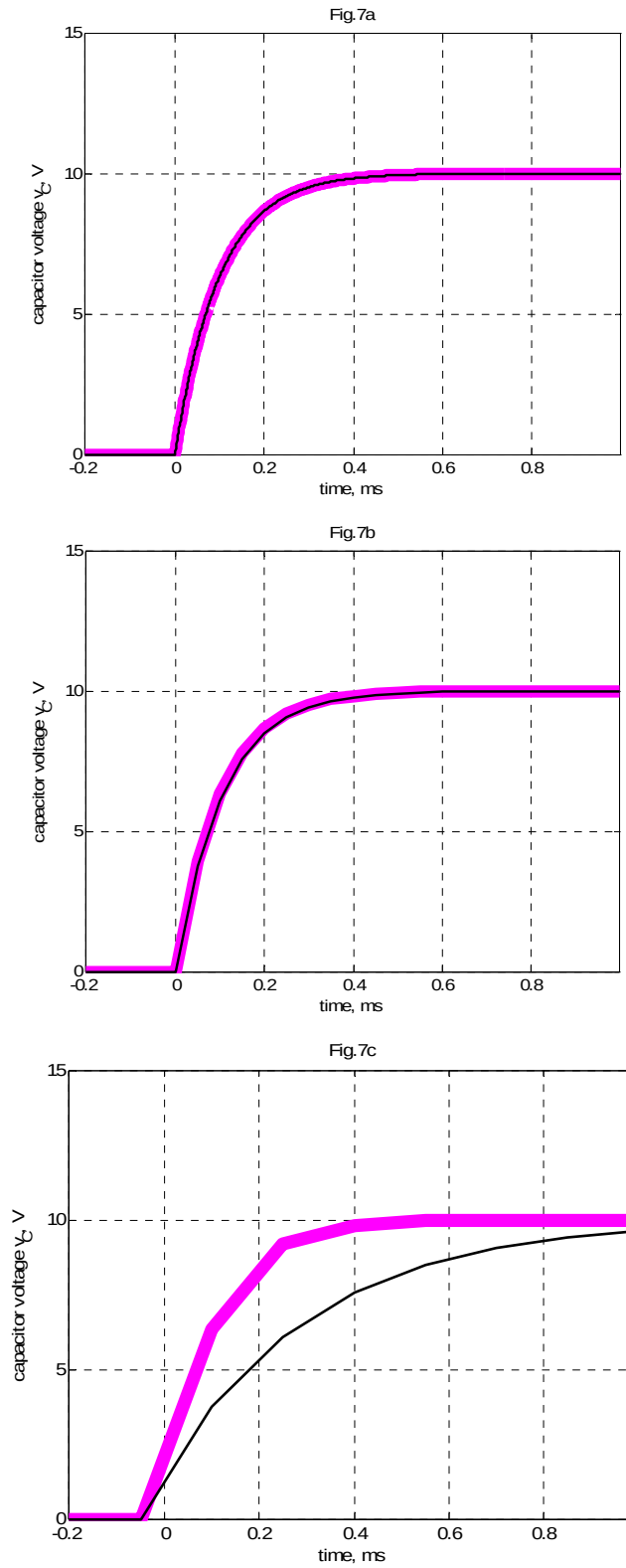


Fig. 7. Second-order Runge-Kutta method for $\Delta t = 0.01\tau$, $\Delta t = 0.5\tau$, and $\Delta t = 1.5\tau$. The analytical solution is shown by a thick curve. It is instructional to compare this solution to the same result obtained with the Euler method - see Fig. 4 above.

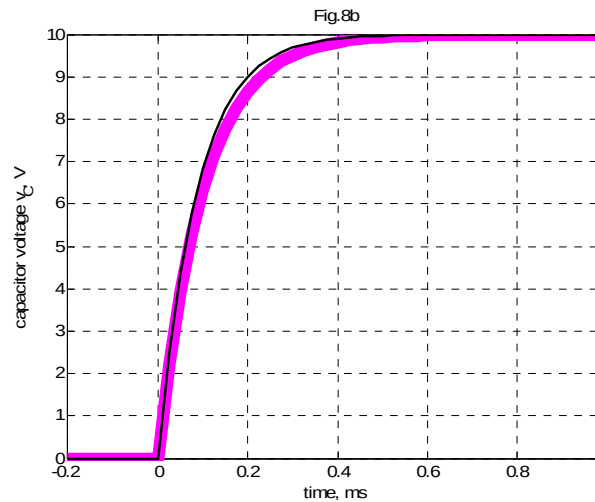
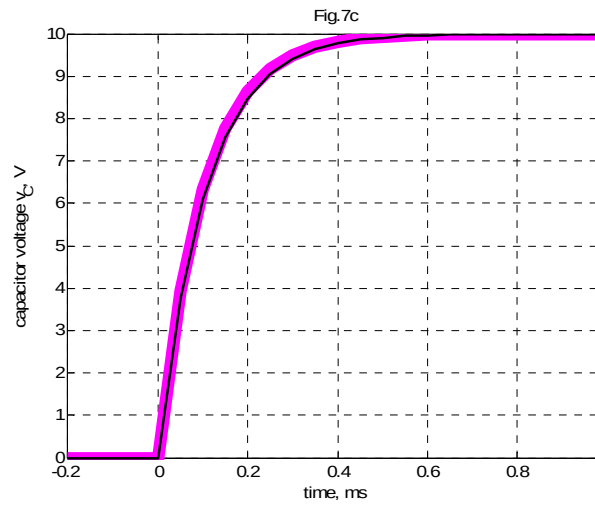


Fig. 8. Comparison between the second-order Runge-Kutta method with $\Delta t = 0.5\tau$ (Fig. 8a) versus the Euler method with $\Delta t = 0.25\tau$ (Fig. 8b). Both methods require nearly the same CPU execution time. However, the Runge-Kutta method is still superior. The analytical solution is shown by a thick curve.

It follows from Fig. 8 that the increase of the accuracy of the finite-difference method is an advantage. Why don't we increase the accuracy further? This way of thinking leads us to a fourth-order Runge-Kutta method, which is the standard method for solving the ordinary differential equations.

7. Single-step numerical method - the fourth-order Runge-Kutta method

To realize the advantage of a higher-order method we could extend the second-order Runge-Kutta method in a number of ways. For example, we could start with using Eqs. (15) for the second-order method

$$f_{n+1/2} = f_n + 0.5\Delta t F(f_n, t_n),$$

$$f_{n+1} = f_n + \Delta t F(f_{n+1/2}, t_{n+1/2})$$

After that we might want to go back and refine the midpoint approximation $f_{n+1/2}$ as $(f_{n+1} + f_n)/2$. Then, we again refine the value f_{n+1} according to the new value of $f_{n+1/2}$.

A similar (but not identical) procedure is the heart of the fourth-order Runge-Kutta method. Adding up a right combination of the right-hand sides allows us to eliminate all error terms on the order of Δt^4 or lower. This is the *fourth-order Runge-Kutta method*, which may be written in the concise form

$$\begin{aligned} k_1 &= \Delta t F(f_n, t_n), \\ k_2 &= \Delta t F\left(f_n + \frac{k_1}{2}, t_n + \frac{\Delta t}{2}\right) \\ k_3 &= \Delta t F\left(f_n + \frac{k_2}{2}, t_n + \frac{\Delta t}{2}\right) \\ k_4 &= \Delta t F(f_n + k_3, t_n + \Delta t) \\ f_{n+1} &= f_n + \frac{k_1}{6} + \frac{k_2}{3} + \frac{k_3}{3} + \frac{k_4}{6} \end{aligned} \tag{16}$$

Four evaluations of the right-hand side per time step Δt are required: once at the initial point, twice at the midpoint, and once at the endpoint.

Example 4: Solve Eq. (4) for charging the capacitor numerically using the fourth-order Runge-Kutta method described by Eqs. (16).

Solution: The following MATLAB script implements the second-order Runge-Kutta method described by Eqs. (16) for the differential voltage equation (4). The result of the numerical simulation is reported in Fig. 9c.

```
clear all
C      = 10e-6;           % capacitance, F
R      = 10;              % load resistance, Ohm
tau    = R*C              % time constant, sec
Vs     = 10;              % power supply voltage, Vs
dt     = 1.5*tau;         % discretization step
t      = [-2*tau:dt:10*tau]; % time vector, sec

% Analytical solution
vC = Vs*(1 - exp(-t/tau)); % capacitor voltage
vC(find(t<0)) = 0;

plot(t*1e3, vC, 'Color', 'm', 'LineWidth', 8); grid on;
xlabel('time, ms'); ylabel('capacitor voltage v_C, V');
axis([-0.2 1 0 15])

% Numerical solution (second-order Runge-Kutta method)
vCn = zeros(size(t));
N1 = min(find(t>eps));
N2 = length(t);
for n = N1-1:N2-1;
    k1 = dt/tau*(Vs - vCn(n));
    k2 = dt/tau*(Vs - (vCn(n) + k1/2));
    k3 = dt/tau*(Vs - (vCn(n) + k2/2));
    k4 = dt/tau*(Vs - (vCn(n) + k3/1));
    vCn(n+1) = vCn(n) + k1/6 + k2/3 + k3/3 + k4/6;
end

hold on;
plot(t*1e3, vCn, 'Color', 'k', 'LineWidth', 2); grid on;
xlabel('time, ms'); ylabel('capacitor voltage v_C, V');
```

It is now instructive to compare the performance of all three methods introduced in this lecture. Fig. 9 shows the corresponding results for charging the capacitor at $\Delta t = 1.5\tau$. No comments seem to be necessary.

In a similar way one could show that the fourth-order Runge-Kutta method is still more accurate when nearly the same CPU time is requested for all three methods.

The higher-order Runge-Kutta schemes are rarely used. MATLAB realizes the second-order Runge-Kutta method as `ode25` and the fourth-order Runge-Kutta method as `ode45`. An automatic time step selection and an automatic error control are foreseen.

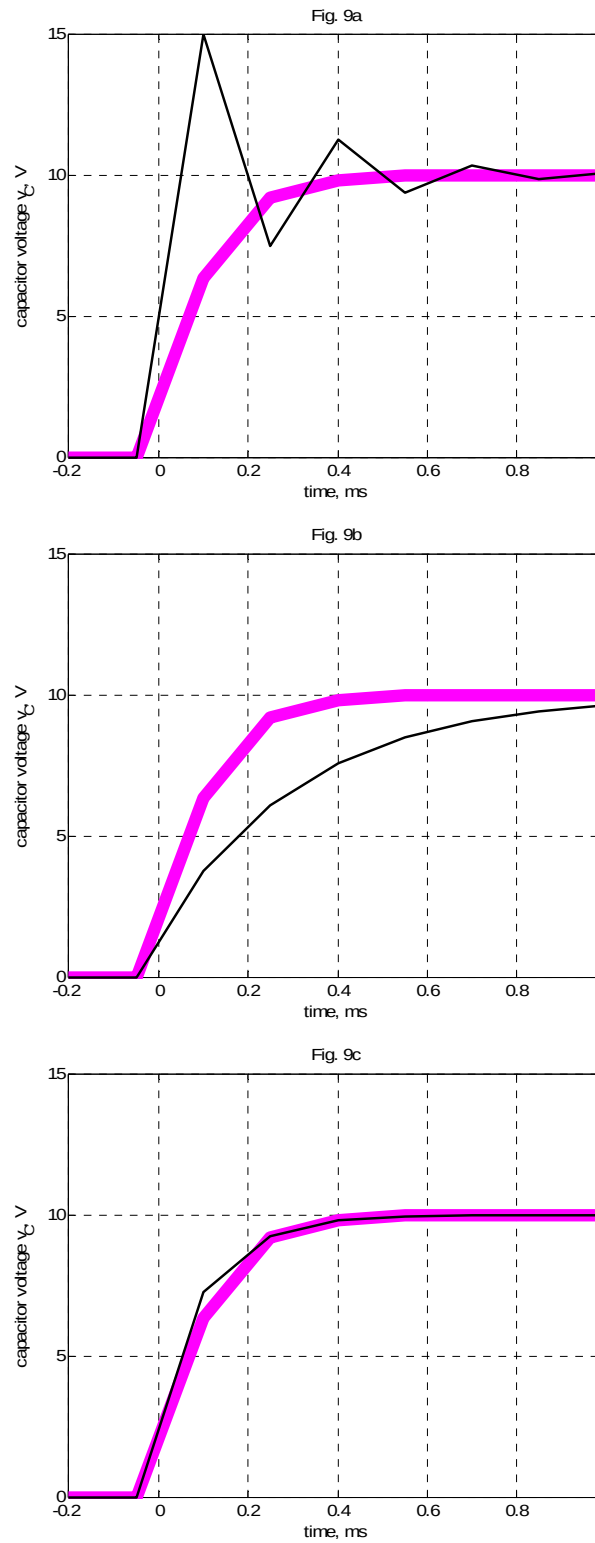


Fig. 9. First order Euler method (Fig. 9a), second-order Runge-Kutta method (Fig. 9b), and fourth-order Runge-Kutta method (Fig. 9c); all method use the same time step $\Delta t = 1.5\tau$. The analytical solution is shown by a thick curve.

8. Principle of inductor choke

We will apply the developed theory to the solution of a practical circuit - the inductor choke. Often it is necessary to supply two distinct voltages - one is a very weak AC signal, the other is a strong DC signal - to the same load. This problem frequently appears in RF (or Radio Frequency) circuits.

The weak AC voltage is typically the voltage received by an antenna used in a wireless device. The DC voltage may be a bias voltage applied to an RF device such as a mixer². The modeling circuit is shown in Fig. 10. Two power supplies are represented by their Thévenin equivalents, respectively.

We must *isolate* the weak AC signal from the DC power supply, i.e. prevent the AC current from flowing through the (low-resistance!) DC power supply and force it to flow through the load with a higher resistance.

The goal is thus not to lose the very weak AC power. For this purpose, the inductor (or *inductor choke*) shown in Fig. 10 is used. At the same time, we do not worry too much about the DC flowing through the AC power supply. Since the AC *voltage* source (such as an antenna feed) is usually an open circuit for DC, no DC current in reality flows through it.

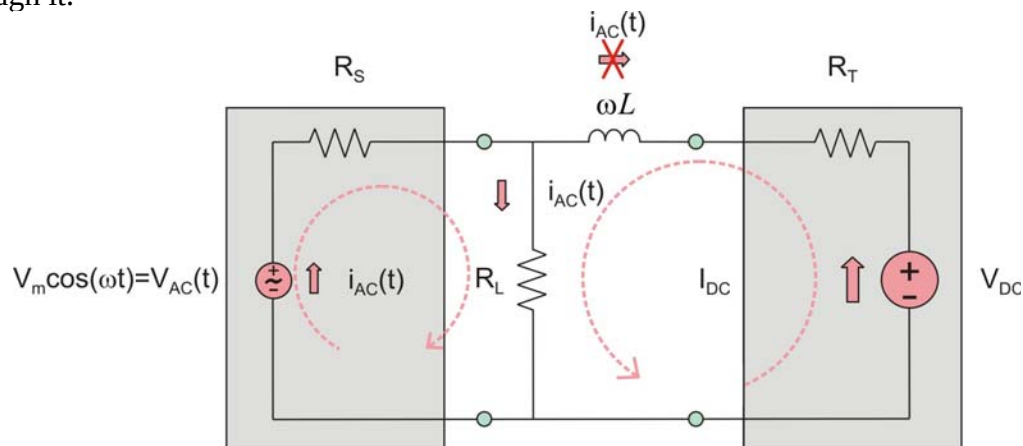


Fig. 10. Model of two power supplies (AC and DC) connected to the same load and the use of the isolating inductor choke. The power supplies are represented by their Thévenin equivalents.

Fig. 11 shows the hardware realization of the circuit in Fig. 10 and its operation at different values of the blocking inductance.

² The RF mixer "mixes" two or more signals and outputs their product. The mixer uses fast electronic diodes. The mixer circuits themselves are studied in advanced ECE classes.

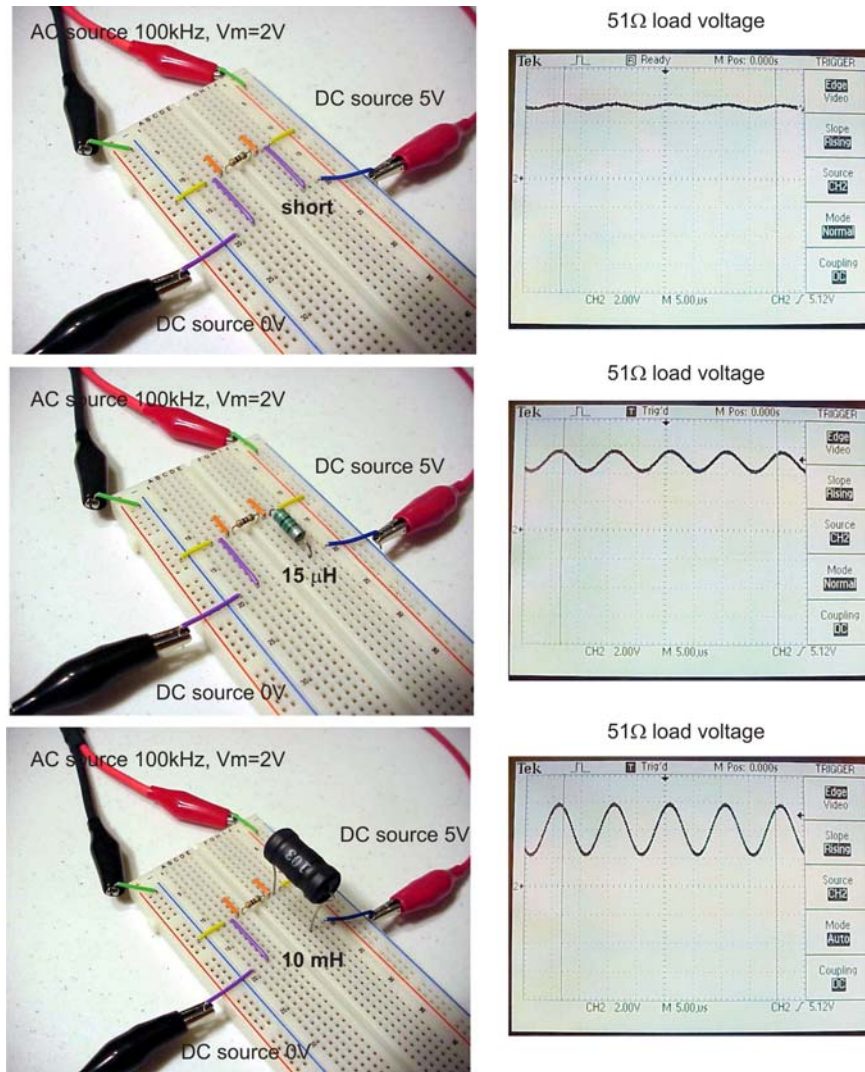


Fig. 11. A hardware realization of the circuit from Fig. 10. A 100 kHz AC source with the voltage amplitude of 2V (50 Ω function generator) is connected to a 51 Ω load resistor along with a 5V DC laboratory supply. The load voltage (a combination of AC and DC) is shown on the right. There is no inductor choke in the circuit on top. As a result, the AC signal is almost entirely shorted out by the DC power supply. In the center circuit, an inductor choke of 15 μ H used; the AC signal across the load becomes apparent. In the bottom circuit, an inductor choke of 10 mH is used. The AC signal amplitude is close to the desired 2V.

9. Simplified circuit model for the homework

Although the circuit in Fig. 10 may be solved directly, we prefer to use a very simplified transient circuit model shown in Fig. 12 that follows - the first-order low-pass filter. This model still preserves the effect of the inductor choke. The DC source is assumed to be imperfect. Its voltage has a small but visible high-frequency component of the form

$$V_s(t) = 5V + 1V \sin(10^7 t) \quad (17)$$

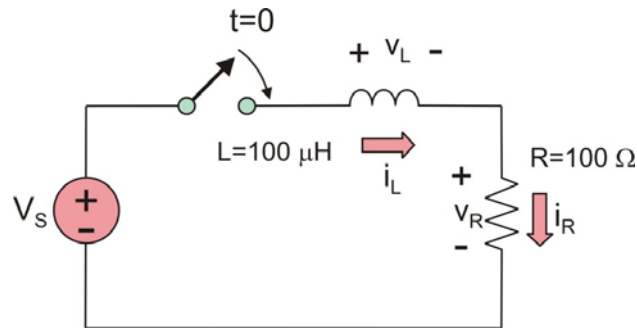


Fig. 12. A simplified choke circuit: the inductor is a choke in series with a resistor - the Thévenin resistance of the DC power supply in Fig. 10.

The resistor R plays the role of the (small) Thévenin resistance of the DC supply in Fig. 10. The source is turned on at the initial time instant $t = 0$.

If there were no inductor in the circuit, the voltage across resistor R would be exactly equal to the power supply voltage. It is now claimed that the inductor in Fig. 12 plays the same blocking role: it isolates resistance R from the high-frequency signal. In other words, the voltage across resistance R must have a much weaker high-frequency AC component when a (sufficient) inductance is present.

HW#1 exercise

Using MATLAB please compare the voltage across resistance R with and without the inductor choke in Fig. 12. To do so plot the load voltage over the time interval from -2τ to 12τ for both circuit designs. The choke inductance is $100\mu\text{H}$, the load resistance is 100Ω .

Obtain the solution with the inductor choke using

- The Euler method;
- The second-order Runge-Kutta method;
- The fourth-order Runge-Kutta method.

In all three cases use $\Delta t = 0.1\tau$. Your HW report should include:

- Three separate MATLAB scripts that realize three separate numerical methods for the circuit in Fig. 12.
- Three corresponding figures - script outputs.
- Two error plots for the Euler method and the second-order Runge-Kutta method as compared to the fourth-order Runge-Kutta method.

Please put the MATLAB comments wherever is necessary.