# Resposta a Degrau Circuito RLC série

Jupyter Notebook desenvolvido por [Gustavo S.S. (https://github.com/GSimas)](https://github.com/GSimas)

a resposta a um degrau é obtida por uma aplicação repentina de uma fonte CC. Consideremos o circuito RLC em série, mostrado na Figura 8.18. Aplicando a LKT no circuito para t > 0:

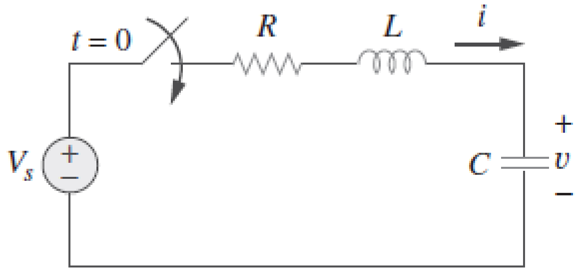


**Figura 8.18** Tensão em degrau aplicada a um circuito *RLC* em série.

$$L\frac{di}{dt} + Ri + v = V_s$$

Porém:

$$i = C\frac{dv}{dt}$$

Substituindo i e reorganizando os termos:

$$\frac{d^2v}{dt^2} + \frac{R}{L}\frac{dv}{dt} + \frac{v}{LC} = \frac{V_s}{LC}$$

que tem a mesma forma da Equação para circuitos RLC sem fonte. Logo, a equação característica para o circuito RLC em série não é afetada pela presença da fonte CC. A solução para a Equação (8.40) possui duas componentes: resposta transiente vt(t) e resposta de estado estável vss(t); ou seja:

$$v(t) = v_t(t) + v_{ss}(t)$$

O valor final da tensão no capacitor é o mesmo da fonte de tensão vs.

$$v_{ss}(t) = v(\infty) = V_s$$

Consequentemente, junto com a resposta transiente vt(t) para os casos de amortecimento supercrítico, subamortecimento e amortecimento crítico são:

$$v(t) = V_s + A_1 e^{s_1 t} + A_2 e^{s_2 t} \quad Superamortecido$$

$$v(t) = V_s + (A_1 + A_2 t)e^{-\alpha t} \quad Amortecimento \ Critico$$

$$v(t) = V_s + (A_1 cos(\omega_d t) + A_2 sin(\omega_d t))e^{-\alpha t} \ \ Subamortecido$$

Relembra-se que:

$$\alpha = \frac{R}{2L}$$

$$\omega_0 = \frac{1}{\sqrt{LC}}$$

$$\omega_d = \sqrt{\omega_0^2 - \alpha^2}$$

$$s_{1,2} = -\alpha \pm \sqrt{\alpha^2 - \omega_0^2}$$

**Exemplo 8.7**

Para o circuito da Figura 8.19, encontre v(t) e i(t) para t > 0. Considere os seguintes casos: R = 5 Ω, R = 4 Ω e R = 1 Ω.

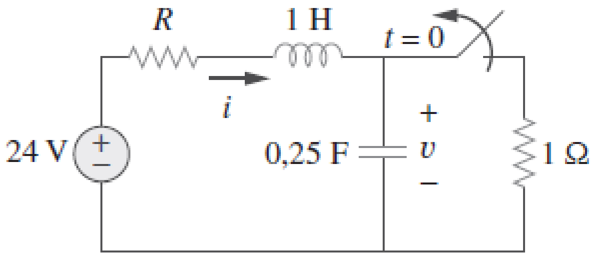


**Figura 8.19** Esquema para o Exemplo 8.7.

In [37]:

```
print("Exemplo 8.7")

from sympy import *

L = 1
C = 0.25
Vs = 24

t = symbols('t')
A1 = symbols('A1')
A2 = symbols('A2')

def sqrt(x,root=2): #definicao funcao raiz
    y = x**(1/root)
    return y

#Para R = 5
R = 5
print("\nPara R = 5\n")
#Para t < 0
v0 = Vs*1/(1 + R)
i0 = Vs/(1 + R)
```

```
print("v(0):",v0,"V")
print("i(0):",i0,"A")
print("dv(0)/dt:",i0/C,"V/s")

#Para t > 0

def rlc_serie(R,L,C):
    alpha = R/(2*L)
    omega0 = 1/sqrt(L*C)

    print("Alpha:",alpha,"Np/s")
    print("Omega0:",omega0,"rad/s")

    s1 = -alpha + sqrt(alpha**2 - omega0**2)
    s2 = -alpha - sqrt(alpha**2 - omega0**2)
    omegad = sqrt(omega0**2 - alpha**2)

    if alpha > omega0:
        resposta = "Superamortecido"
        v = Vs + A1*exp(s1*t) + A2*exp(s2*t)

    elif alpha == omega0:
        resposta = "Amortecimento Crítico"
        v = Vs + (A1 + A2*t)*exp(-alpha*t)

    else:
        resposta = "Subamortecido"
        v = Vs + (A1*cos(omegad*t) + A2*sin(omegad*t))*exp(-alpha*t)

    print("Tipo de resposta:",resposta)
    print("v(t):",v,"V")
    print("v(0):",v.subs(t,0),"V")
    print("dv(0)/dt:",diff(v,t).subs(t,0))

    return alpha, omega0, omegad, resposta, s1, s2, v

alpha, omega0, omegad, resposta, s1, s2, v = rlc_serie(R,L,C)

#v0 = A1 + A2 + 24 = 4
    #A1 = -20 - A2
#dv0/dt = ic0/C
    #-A1 - 4A2 = 4/0.25 = 16
    #20 + A2 - 4A2 = 16
    #-3A2 = -4
A_2 = -4/-3
A_1 = -20 - A_2

print("Constante A1:",A_1)
print("Constante A2:",A_2)
v = v.subs(A1,A_1).subs(A2,A_2)
print("v(t):",v,"V")
i = C*diff(v,t)
print("i(t):",i,"A")

print("\n\n-------------\n\n")

#Para R = 4
R = 4
print("\nPara R = 4\n")
v0 = Vs*1/(1 + R)
i0 = Vs/(1 + R)
```

```
print("v(0):",v0,"V")
print("i(0):",i0,"A")
print("dv(0)/dt:",i0/C,"V/s")

alpha, omega0, omegad, resposta, s1, s2, v = rlc_serie(R,L,C)

#dv0/dt = -2A1 + A2 = 19.2
    #A2 = 19.2 + 2A1
#v0 = A1 + 24 = 4.8
A_1 = 4.8 - 24
A_2 = 19.2 + 2*A_1

print("Constante A1:",A_1)
print("Constante A2:",A_2)
v = v.subs(A1,A_1).subs(A2,A_2)
print("v(t):",v,"V")
i = C*diff(v,t)
print("i(t):",i,"A")

print("\n\n-------------\n\n")

#Para R = 1
R = 1
print("\nPara R = 1\n")
v0 = Vs*1/(1 + R)
i0 = Vs/(1 + R)

print("v(0):",v0,"V")
print("i(0):",i0,"A")
print("dv(0)/dt:",i0/C,"V/s")

alpha, omega0, omegad, resposta, s1, s2, v = rlc_serie(R,L,C)

#v0 = A1 + 24 = 12
A_1 = 12 - 24
#dv0/dt = -0.5A1 + 1.94A2 = 48
A_2 = (48 + 0.5*A_1)/1.94

print("Constante A1:",A_1)
print("Constante A2:",A_2)
v = v.subs(A1,A_1).subs(A2,A_2)
print("v(t):",v,"V")
i = C*diff(v,t)
print("i(t):",i,"A")
```

```
Exemplo 8.7

Para R = 5

v(0): 4.0 V
i(0): 4.0 A
dv(0)/dt: 16.0 V/s
Alpha: 2.5 Np/s
Omega0: 2.0 rad/s
Tipo de resposta: Superamortecido
v(t): A1*exp(-1.0*t) + A2*exp(-4.0*t) + 24 V
v(0): A1 + A2 + 24 V
dv(0)/dt: -1.0*A1 - 4.0*A2
Constante A1: -21.333333333333332
Constante A2: 1.3333333333333333
v(t): 24 + 1.33333333333333*exp(-4.0*t) - 21.3333333333333*exp(-1.0*t) V
i(t): -1.33333333333333*exp(-4.0*t) + 5.33333333333333*exp(-1.0*t) A


-------------



Para R = 4

v(0): 4.8 V
i(0): 4.8 A
dv(0)/dt: 19.2 V/s
Alpha: 2.0 Np/s
Omega0: 2.0 rad/s
Tipo de resposta: Amortecimento Crítico
v(t): (A1 + A2*t)*exp(-2.0*t) + 24 V
v(0): A1 + 24 V
dv(0)/dt: -2.0*A1 + A2
Constante A1: -19.2
Constante A2: -19.2
v(t): (-19.2*t - 19.2)*exp(-2.0*t) + 24 V
i(t): -0.5*(-19.2*t - 19.2)*exp(-2.0*t) - 4.8*exp(-2.0*t) A


-------------



Para R = 1

v(0): 12.0 V
i(0): 12.0 A
dv(0)/dt: 48.0 V/s
Alpha: 0.5 Np/s
Omega0: 2.0 rad/s
Tipo de resposta: Subamortecido
v(t): (A1*cos(1.93649167310371*t) + A2*sin(1.93649167310371*t))*exp(-0.5*
t) + 24 V
v(0): A1 + 24 V
dv(0)/dt: -0.5*A1 + 1.93649167310371*A2
Constante A1: -12
Constante A2: 21.649484536082475
v(t): (21.6494845360825*sin(1.93649167310371*t) - 12*cos(1.93649167310371*
t))*exp(-0.5*t) + 24 V
i(t): -0.125*(21.6494845360825*sin(1.93649167310371*t) - 12*cos(1.93649167
```

```
310371*t))*exp(-0.5*t) + 0.25*(23.2379000772445*sin(1.93649167310371*t) +
 41.9240465311112*cos(1.93649167310371*t))*exp(-0.5*t) A
```

**Problema Prático 8.7**

Já na posição a há muito tempo, a chave na Figura 8.21 é mudada para a posição b em t = 0. Determine v(t) e vR(t) para t > 0.

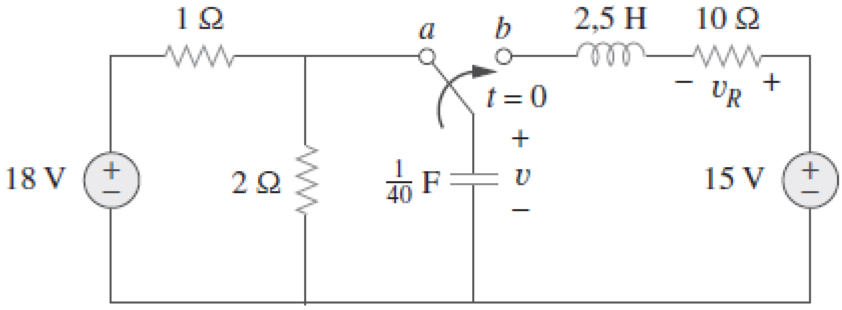


**Figura 8.21** Esquema para o Problema prático 8.7.

In [46]:

```
print("Problema Prático 8.7")

Vs1 = 18
C = 1/40
L = 2.5

#Para t < 0 e 0+
v0 = Vs1*2/(2 + 1)
i0 = 0

print("v0:",v0,"V")
print("i0:",i0,"A")
print("dv0/dt:",i0/C)

#Para t > 0
R = 10
Vs = 15
alpha, omega0, omegad, resposta, s1, s2, v = rlc_serie(R,L,C)

#v0 = A1 + 15 = 12
A_1 = 12 - 15
#dv0/dt = -2A1 + 3.46A2 = 0
A_2 = 2*A_1/3.46

print("Constante A1:",A_1)
print("Constante A2:",A_2)
v = v.subs(A1,A_1).subs(A2,A_2)
print("v(t):",v,"V")
i = C*diff(v,t)
vr = R*i
print("vr(t):",vr,"V")
```

```
Problema Prático 8.7
v0: 12.0 V
i0: 0 A
dv0/dt: 0.0
Alpha: 2.0 Np/s
Omega0: 4.0 rad/s
Tipo de resposta: Subamortecido
v(t): (0.578034682080925*A1*sin(3.46410161513775*t) + A1*cos(3.46410161513
775*t))*exp(-2.0*t) + 15 V
v(0): A1 + 15 V
dv(0)/dt: 0.00237087580217032*A1
Constante A1: -3
Constante A2: -1.7341040462427746
v(t): (-1.73410404624277*sin(3.46410161513775*t) - 3*cos(3.46410161513775*
t))*exp(-2.0*t) + 15 V
vr(t): -0.5*(-1.73410404624277*sin(3.46410161513775*t) - 3*cos(3.464101615
13775*t))*exp(-2.0*t) + 0.25*(10.3923048454133*sin(3.46410161513775*t) -
 6.00711262740651*cos(3.46410161513775*t))*exp(-2.0*t) V
```