Laboratorium z podstaw fizyki Wydziału EliT AGH

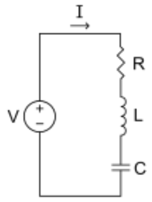
Przykłady obliczeń

© Michał Kołodziej 2016, kolodziej.michal@gmail.com

Laboratorium 10 - Drgania tłumione w obwodzie RLC

Opis obwodu RLC

http://pl.wikipedia.org/wiki/Drgania_t%C5%82umione (http://pl.wikipedia.org/wiki/Drgania_t%C5%82umione)
http://pl.wikipedia.org/wiki/Obw%C3%B3d_RLC (http://pl.wikipedia.org/wiki/Obw%C3%B3d_RLC)



Powyższy rysunek można opisać korzystając z napięciowego prawa Kirhoff'a:

$$v_R + v_L + v_C = v(t)$$

Podstawiając spadki napięć na elementach otrzymujemy równanie:

$$Ri(t) + L\frac{di}{dt} + \frac{1}{C}\int_{-\infty}^{\tau=t} i(\tau)\,d\tau = v(t)$$

Jeżeli źródło jest źródłem napięcia stałego, zróżniczkowanie po czasie obu stron równania i podzielenie przez L doprowadzi do równania różniczkowego drugiego rzędu:

$$\frac{d^2 i(t)}{dt^2} + \frac{R}{L}\frac{di(t)}{dt} + \frac{1}{LC}i(t) = 0$$

Równanie to można wyrazić w bardziej przyjaznej postaci:

$$\frac{d^2 i(t)}{dt^2} + 2\beta\frac{di(t)}{dt} + {\omega_0}^2 i(t) = 0$$

Tłumienie β oraz częstotliwość drgań ω_0, są w jednostkach częstotliwości kątowej. β mierzy jak szybko odpowiedź chwilowa układu zaniknie po usunięciu wymuszenia (napięcia v). ω_0, jest częstotliwością kołową rezonansu.

Parametr tłumienia β oraz częstotliwość drgań ω dane są następująco

$$\beta = \frac{R}{2L}$$
$$\omega_0 = \frac{1}{\sqrt{LC}}$$

W ćwiczeniu obserwujemy napięcie na kondensatorze w czasie, które odpowiada ładunkowi (U=Q/C) na kondensatorze w czasie, a to z kolei odpowiada prądowi w czasie.

$$\frac{d^2 Q(t)}{dt^2} + \frac{L}{R}\frac{dQ(t)}{dt} + \frac{1}{LC}Q(t) = 0$$

Napięcie na kondensatorze

Rozwiązanie powyższego równania (przy małym tłumieniu):

$$Q = Q_m \, e^{-\beta t} \cos(\omega t + \varphi)$$

Zmiana napięcia na kondensatorze:

$$U = \frac{Q_m}{C} \, e^{-\beta t} \cos(\omega t + \varphi) = U_m e^{-\beta t} \cos(\omega t + \varphi)$$

Przy czym:

$$\omega = \sqrt{\omega_0^2 - \beta^2} = \sqrt{\frac{1}{LC} - \frac{R^2}{4L^2}}$$

A warunek małego tłumienia jest następujący (pierwiastek ma być rzeczywisty):

$$\beta^2 < \omega_0^2 \Rightarrow \frac{R^2}{4L^2} < \frac{1}{LC}$$

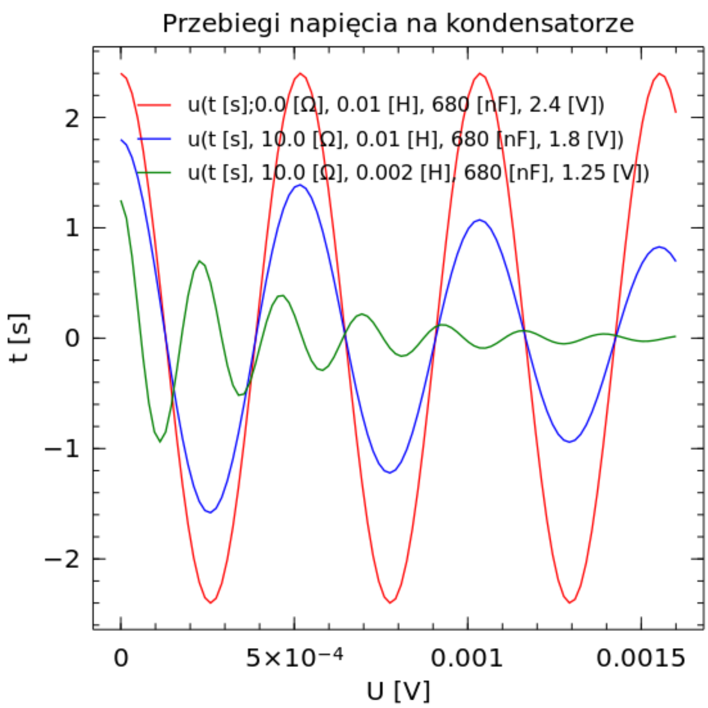
In [1]:

```
nano = 1e-9

function genU(R,L,C,Um)
    β = R/2/L
    ω_0_sq = 1/L/C
    ω = sqrt(ω_0_sq - β^2)
    return t -> Um*e^(-β*t)*cos(ω*t)
end
```

Out[1]: genU (generic function with 1 method)

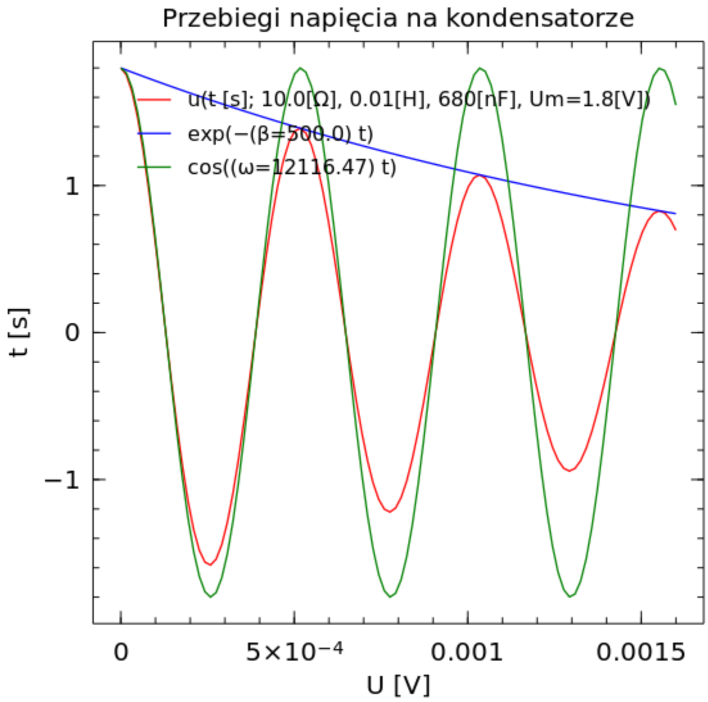
In [164]:
```
L0 = 0.01 # [H]
R0 = 0.0
C0_teor = 680 # [nF]
C0_teor_nano = C0_teor * nano
Um0 = 2.4 # [V]

u0 = genU(R0,L0,C0_teor_nano,Um0)


L1 = 0.01 # [H]
R1 = 10.0
C1_teor = 680 # [nF]
C1_teor_nano = C1_teor * nano
Um1 = 1.8 # [V]

u1 = genU(R1,L1,C1_teor_nano,Um1)


L2 = 0.002 # [H]
R2 = 10.0
C2_teor = 680 # [nF]
C2_teor_nano = C2_teor * nano
Um2 = 1.25 # [V]

u2 = genU(R2,L2,C2_teor_nano,Um2)



ts = linspace(0.0, 8*0.0002, 100)
us0 = map(u0, ts)
us1 = map(u1, ts)
us2 = map(u2, ts)

import Winston
p = Winston.FramedPlot(title = "Przebiegi napięcia na kondensatorze", xlabel="U [V]", ylab
a = Winston.Curve(ts, us0, color=parse(Winston.Colorant, "red"))
Winston.setattr(a, label="u(t [s];$R0 [Ω], $L0 [H], $C0_teor [nF], $Um0 [V])")
# Winston.style(a, color=parse(Winston.Colorant, "red"))
b = Winston.Curve(ts, us1, color=parse(Winston.Colorant, "blue"))
Winston.setattr(b, label="u(t [s], $R1 [Ω], $L1 [H], $C1_teor [nF], $Um1 [V])")
c = Winston.Curve(ts, us2, color=parse(Winston.Colorant, "green"))
Winston.setattr(c, label="u(t [s], $R2 [Ω], $L2 [H], $C2_teor [nF], $Um2 [V])")
l = Winston.Legend(.1, .9, Any[a,b,c])
Winston.add(p, a, b, c, l)
Winston.savepng(p, "Lab_10_drgania_u0_u1_u2.png", 600, 600)
HTML("""<img src="/files/Lab_10_drgania_u0_u1_u2.png?$(datetime2unix(now()))" alt="Test" w
```

Out[164]:



In [161]:
```
L1 = 0.01 # [H]
R1 = 10.0
C1_teor = 680 # [nF]
C1_teor_nano = C1_teor * nano # [F]
Um1 = 1.8 # [V]

u1 = genU(R1,L1,C1_teor_nano,Um1)

ts = linspace(0.0, 8*0.0002, 100)
us1 = map(u1, ts)
β1 = R1/2/L1
ω(r,l,c) = sqrt(1/l/c - (r/2/l)^2)
ω1 = round(ω(R1,L1,C1_teor_nano), 2)

import Winston
p = Winston.FramedPlot(title = "Przebiegi napięcia na kondensatorze", xlabel="U [V]", ylab
Winston.setattr(p, height=600)
Winston.ylim([-4,4])
a = Winston.Curve(ts, us1, color=parse(Winston.Colorant, "red"))
Winston.setattr(a, label="u(t [s]; $R1[Ω], $L1[H], $C1_teor[nF], Um=$Um1[V])")
# Winston.style(a, color=parse(Winston.Colorant, "red"))
b = Winston.Curve(ts, map(t->Um1*exp(-β1*t), ts), color=parse(Winston.Colorant, "blue"))
Winston.setattr(b, label="exp(-(β=$β1) t)")
c = Winston.Curve(ts, map(t->Um1*cos(ω1*t), ts), color=parse(Winston.Colorant, "green"))
Winston.setattr(c, label="cos((ω=$ω1) t)")
l = Winston.Legend(.1, .9, Any[a,b,c])

Winston.add(p, a, b, c, l)
Winston.savepng(p, "Lab_10_drgania_tlumione.png", 600, 600)
HTML("""<img src="/files/Lab_10_drgania_tlumione.png?$(datetime2unix(now()))" alt="Test" w
```

Out[161]:



Graficzne wyznaczanie parametrów funkcji $U_m e^{-\beta t}\cos(\omega t + \varphi)$

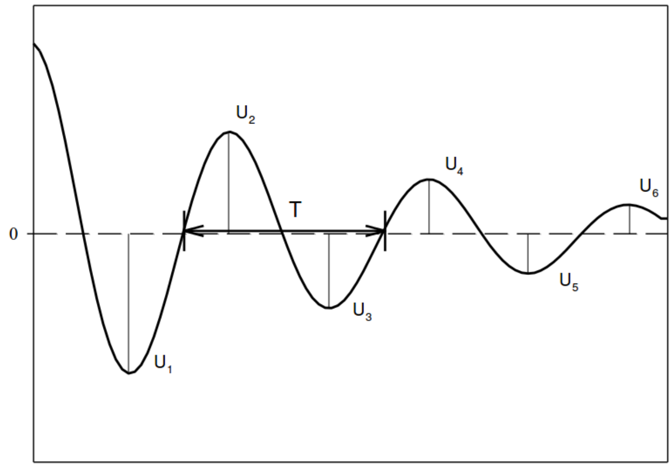
Mając do dyspozycji przebiegi napięcia na kondensatorze, ponieważ są one wyświetlane na oscyloskopie, oraz mając możliwość mierzenia ich na oscyloskopie, możemy spróbować wyznaczyć parametry układu RLC w sposób graficzny.

Dzieląc równania:

$$U_m e^{-\beta t} = U_1$$
$$U_m e^{-\beta(t+T)} = U_3$$

Możemy wyznaczyć β na podstawie:

$$\beta = \frac{1}{T} ln\left(\frac{U_i}{U_{i+2}}\right)$$



Pomiar ekstremów napięcia i okresów drgań

Zmierzone ekstrema napięcia U

In [55]:
```
mikro(x) = x / 1e6

Us_0R_001L = [1.6, 1.2, 0.9]
Us_20R_001L = [3.7, 0.6, 0.2]
Us_0R_01L = [3.4, 2.4, 1.6]
Us_20R_01L = [6.2, 2.6, 1]

[
    ("Us_0R_001L [V]", Us_0R_001L),
    ("Us_0R_01L [V]", Us_0R_01L),
    ("Us_20R_001L [V]", Us_20R_001L),
    ("Us_20R_01L [V]", Us_20R_01L)
]
```

Out[55]:
```
Dict{Any,Any} with 4 entries:
  "Us_0R_01L [V]"   => [3.4,2.4,1.6]
  "Us_0R_001L [V]"  => [1.6,1.2,0.9]
  "Us_20R_01L [V]"  => [6.2,2.6,1.0]
  "Us_20R_001L [V]" => [3.7,0.6,0.2]
```

Zmierzone okresy drgań T

In [165]:
```
T_0R_001L = mikro(55)
T_20R_001L = mikro(60)
T_0R_01L = mikro(185)
T_20R_01L = mikro(180)

[
    ("T_0R_001L [s]", T_0R_001L),
    ("T_0R_01L [s]", T_0R_01L),
    ("T_20R_001L [s]", T_20R_001L),
    ("T_20R_01L [s]", T_20R_01L)
]
```

Out[165]:
```
4-element Array{Tuple{ASCIIString,Float64},1}:
 ("T_0R_001L [s]",5.5e-5)
 ("T_0R_01L [s]",0.000185)
 ("T_20R_001L [s]",6.0e-5)
 ("T_20R_01L [s]",0.00018)
```

Błąd pomiaru napięcia ΔU

In [178]:
```
ΔU = 0.2
[("Błąd pomiaru napięcia ΔU", ΔU)]
```

Out[178]:
```
1-element Array{Tuple{UTF8String,Float64},1}:
 ("Błąd pomiaru napięcia ΔU",0.2)
```

Błąd pomiaru okresu ΔT

In [167]:
```
ΔT = mikro(10)
[("Błąd pomiaru okresu ΔT", ΔT)]
```

Out[167]:
```
1-element Array{Tuple{UTF8String,Float64},1}:
 ("Błąd pomiaru okresu ΔT",1.0e-5)
```

Wyznaczenie współcznynnika tłumienia β

$$\beta(T, u_i) = \frac{1}{T} log\left(\frac{u_i}{u_{i+2}}\right)$$

u_i, u_i+2 to kolejne maksima napięcia obserwowane na oscyloskopie, T to okres drgań obserwowany na oscyloskopie.

Z drugiej strony wiemy, że

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

gdzie R to całkowity opór obwodu, a L całkowita indukcyjność. Wynika z tego, że dla różnych R i L, współczynniki tłumienia β powinny być różne:

In [168]:
```
tlumienie(u1, u2, T) = round(1/T * log(u1/u2))
tlumienia(us, T) = map((u1, u2) -> tlumienie(u1, u2, T), us[1:end-1], us[2:end])


βs_0R_001L = tlumienia(Us_0R_001L, T_0R_001L)
βs_20R_001L = tlumienia(Us_20R_001L, T_20R_001L)
βs_0R_01L = tlumienia(Us_0R_01L, T_0R_01L)
βs_20R_01L = tlumienia(Us_20R_01L, T_20R_01L)

[
    ("βs_0R_001L [rad/s]", βs_0R_001L),
    ("βs_0R_01L [rad/s]", βs_0R_01L),
    ("βs_20R_001L [rad/s]", βs_20R_001L),
    ("βs_20R_01L [rad/s]", βs_20R_01L)
]
```

Out[168]:
```
4-element Array{Tuple{UTF8String,Array{Float64,1}},1}:
 ("βs_0R_001L [rad/s]",[5231.0,5231.0])
 ("βs_0R_01L [rad/s]",[1883.0,2192.0])
 ("βs_20R_001L [rad/s]",[30319.0,18310.0])
 ("βs_20R_01L [rad/s]",[4828.0,5308.0])
```

Błąd wyznaczania współczynnika tłumienia Δβ

Do wyznaczenia współczynnika tłumienia możemy się posłużyć aproksymacją liniową (metodą różniczki zupełnej) albo wprost przeliczyć odchylenie dla odchyleń argumentów. Załóżmy że mamy jakąś funkcję g i chcemy oszacować jej błąd na podstawie błędów jej parametrów, możemy do tego problemu podejść wprost:

$$\Delta\hat{g} \quad = \quad \hat{g}\,(L + \Delta L,\ \ T + \Delta T,\ \ \theta + \Delta\theta) \ \ - \ \ \hat{g}\,(L,\ T,\ \theta)$$

Błąd względny g możemy zapisać następująco:

$$\frac{\Delta\hat{g}}{\hat{g}} \quad = \quad \frac{\hat{g}\,(L + \Delta L,\ \ T + \Delta T,\ \ \theta + \Delta\theta) \ \ - \ \ \hat{g}\,(L,\ T,\ \theta)}{\hat{g}\,(L,\ T,\ \theta)}$$

Dla różnych R i L, błędy współczynników tłumienia β (mamy dwa pomiary dla każdego β odpowiadające parom maksimów napięcia, z których wyznaczamy β) są następujące:

In [179]:

```
βs_0R_001L_bledy = tlumienia(Us_0R_001L + ΔU, T_0R_001L + ΔT) - tlumienia(Us_0R_001L, T_0R
βs_20R_001L_bledy = tlumienia(Us_20R_001L + ΔU, T_20R_001L + ΔT) - tlumienia(Us_20R_001L,
βs_0R_01L_bledy = tlumienia(Us_0R_01L + ΔU, T_0R_01L + ΔT) - tlumienia(Us_0R_01L, T_0R_01L
βs_20R_01L_bledy = tlumienia(Us_20R_01L + ΔU, T_20R_01L + ΔT) - tlumienia(Us_20R_01L, T_20

[
    ("βs_0R_001L_bledy [rad/s]", βs_0R_001L_bledy),
    ("βs_20R_001L_bledy [rad/s]", βs_20R_001L_bledy),
    ("βs_0R_01L_bledy [rad/s]", βs_0R_01L_bledy),
    ("βs_20R_01L_bledy [rad/s]", βs_20R_01L_bledy)
]
```

Out[179]:

```
4-element Array{Tuple{UTF8String,Array{Float64,1}},1}:
 ("βs_0R_001L_bledy [rad/s]",[1365.0,1521.0])
 ("βs_20R_001L_bledy [rad/s]",[7689.0,8408.0])
 ("βs_0R_01L_bledy [rad/s]",[214.0,306.0])
 ("βs_20R_01L_bledy [rad/s]",[477.0,849.0])
```

Średnie współczynniki tłumienia $\bar{\beta}$

Średnia pomiarów z błędami:

$$\bar{x} = \frac{\sum x_i/\sigma_i^2}{\sum 1/\sigma_i^2}$$

czyli w naszym przypadku:

$$\bar{\beta} = \frac{\sum \beta_i/\Delta\beta_i}{\sum 1/\Delta\beta_i}$$

Dla różnych R i L wyznaczamy odpowiednie średnie współczynniki tłumienia $\bar{\beta}$

In [169]:

```
wmean(xs, es) = sum(map(/, xs, es)) / sum(1./es)

βs_mean_0R_001L = round(wmean(βs_0R_001L, βs_0R_001L_bledy), 2)
βs_mean_20R_001L = round(wmean(βs_20R_001L, βs_20R_001L_bledy), 2)
βs_mean_0R_01L = round(wmean(βs_0R_01L, βs_0R_01L_bledy), 2)
βs_mean_20R_01L = round(wmean(βs_20R_01L, βs_20R_01L_bledy), 2)


[
    ("βs_mean_0R_001L [rad/s]", βs_mean_0R_001L),
    ("βs_mean_20R_001L [rad/s]", βs_mean_20R_001L),
    ("βs_mean_0R_01L [rad/s]", βs_mean_0R_01L),
    ("βs_mean_20R_01L [rad/s]", βs_mean_20R_01L)
]
```

Out[169]:

```
4-element Array{Tuple{UTF8String,Float64},1}:
 ("βs_mean_0R_001L [rad/s]",5231.0)
 ("βs_mean_20R_001L [rad/s]",24582.7)
 ("βs_mean_0R_01L [rad/s]",2010.17)
 ("βs_mean_20R_01L [rad/s]",5000.67)
```

Błędy średnich współczynników tłumienia $\Delta\bar{\beta}$

Błąd średniej ważonej można przedstawić jako:

$$\Delta\bar{\beta} = \frac{\sum \Delta\beta_i}{n}$$

Dla różnych R i L wyznaczamy odpowiednie błędy średnich współczynników tłumienia $\Delta\bar{\beta}$

In [84]:
```
werror(Δs) = sum(Δs)/length(Δs)


βs_mean_0R_001L_blad = round(werror(βs_0R_001L_bledy), 2)
βs_mean_20R_001L_blad = round(werror(βs_20R_001L_bledy), 2)
βs_mean_0R_01L_blad = round(werror(βs_0R_01L_bledy), 2)
βs_mean_20R_01L_blad = round(werror(βs_20R_01L_bledy), 2)

[
    ("βs_mean_0R_001L_blad [rad/s]",βs_mean_0R_001L_blad),
    ("βs_mean_20R_001L_blad [rad/s]",βs_mean_20R_001L_blad),
    ("βs_mean_0R_01L_blad [rad/s]",βs_mean_0R_01L_blad),
    ("βs_mean_20R_01L_blad [rad/s]",βs_mean_20R_01L_blad)
]
```

Out[84]:
```
4-element Array{Tuple{UTF8String,Float64},1}:
 ("βs_mean_0R_001L_blad [rad/s]",1443.0)
 ("βs_mean_20R_001L_blad [rad/s]",8048.5)
 ("βs_mean_0R_01L_blad [rad/s]",260.0)
 ("βs_mean_20R_01L_blad [rad/s]",663.0)
```

Wyznaczanie oporu pasożytniczego R_L obwodu

$$R_L = 2\beta L$$

Opór pasożytniczy obwodu powinien być taki sam niezależnie od ustawień opornika dekadowego R i indukcyjności dekadowej L, jednak możemy zauważyć, że opór pasożytniczy zmienia się wraz ze zmianą L!

Gdyby tak się nie działo, policzylibyśmy średni opór pasożytniczy, ponieważ jednak tak się nie dzieje, do dalszych obliczeń będziemy się posługiwali oporami pasożytniczymi w zależności od średnich współczynników tłumienia $R_L(\bar{\beta})$:

In [172]:
```
opor_pasozytniczy(β, L) = β * 2 * L
R_L_0R_001L = round(opor_pasozytniczy(βs_mean_0R_001L, 0.001), 2)
R_L_20R_001L = round(opor_pasozytniczy(βs_mean_20R_001L, 0.001), 2)
R_L_0R_01L = round(opor_pasozytniczy(βs_mean_0R_01L, 0.01), 2)
R_L_20R_01L = round(opor_pasozytniczy(βs_mean_20R_01L, 0.01), 2)

[
    ("R_L_0R_001L [Ω]", R_L_0R_001L),
    ("R_L_0R_01L [Ω]", R_L_0R_01L),
    ("R_L_20R_001L [Ω]", R_L_20R_001L),
    ("R_L_20R_01L [Ω]", R_L_20R_01L)
]
```

Out[172]:
```
4-element Array{Tuple{UTF8String,Float64},1}:
 ("R_L_0R_001L [Ω]",10.46)
 ("R_L_0R_01L [Ω]",40.2)
 ("R_L_20R_001L [Ω]",49.17)
 ("R_L_20R_01L [Ω]",100.01)
```

Wyznaczanie błędu oporu pasożytniczego ΔR_L obwodu

$$\Delta R_L = R_L(\beta + \Delta\beta, L + \Delta L) - R_L(\beta, L) = 2((\beta + \Delta\beta)(L + \Delta L) - \beta L)$$

Konsekwentnie wyznaczamy błędy poszczególnych oporów pasożytniczych:

In [173]:
```
opor_pasozytniczy_blad(β, L, Δβ) = opor_pasozytniczy(β+Δβ, L) - opor_pasozytniczy(β, L)

R_L_0R_001L_blad = round(opor_pasozytniczy_blad(βs_mean_0R_001L, 0.001, βs_mean_0R_001L_bl
R_L_20R_001L_blad = round(opor_pasozytniczy_blad(βs_mean_20R_001L, 0.001, βs_mean_20R_001L
R_L_0R_01L_blad = round(opor_pasozytniczy_blad(βs_mean_0R_01L, 0.01, βs_mean_0R_01L_blad),
R_L_20R_01L_blad = round(opor_pasozytniczy_blad(βs_mean_20R_01L, 0.01, βs_mean_20R_01L_bla

[
    ("R_L_0R_001L_blad [Ω]", R_L_0R_001L_blad),
    ("R_L_20R_001L_blad [Ω]", R_L_20R_001L_blad),
    ("R_L_0R_01L_blad [Ω]", R_L_0R_01L_blad),
    ("R_L_20R_01L_blad [Ω]", R_L_20R_01L_blad)
]
```

Out[173]:
```
4-element Array{Tuple{UTF8String,Float64},1}:
 ("R_L_0R_001L_blad [Ω]",2.89)
 ("R_L_20R_001L_blad [Ω]",16.1)
 ("R_L_0R_01L_blad [Ω]",5.2)
 ("R_L_20R_01L_blad [Ω]",13.26)
```

Wyznaczenie pojemności kondensatora C z przebiegów na oscyloskopie

Równanie zawierające pojemność kondensatora:

$$\omega = \sqrt{\omega_0^2 - \beta^2} = \sqrt{\frac{1}{LC} - \frac{R^2}{4L^2}}$$

Przekształcamy:

$$\omega^2 + \frac{R^2}{4L^2} = \frac{1}{LC}$$

Otrzymując równanie na pojemność kondensatora:

$$C = \frac{1}{L\left(\omega^2 + \frac{R^2}{4L^2}\right)}$$

Korzystając ze zmierzonego okresu drgań na oscyloskopie T, zadanych oporności R oraz odpowiednich oporności pasożytniczych $R_L(\bar{\beta})$ wyznaczamy pojemności kondensatora:

In [183]:
```
pojemnosc(R, L, T) = 1/L/(2*pi/T^2 + R^2 / 4*L^2) # [F]

C_0R_001L = round(pojemnosc(R_L_0R_001L, 0.001, T_0R_001L), 8)
C_0R_01L = round(pojemnosc(R_L_0R_01L, 0.01, T_0R_01L), 8)
C_20R_001L = round(pojemnosc(R_L_0R_001L + 20, 0.001, T_20R_001L), 8)
C_20R_01L = round(pojemnosc(R_L_0R_01L + 20, 0.01, T_20R_01L), 8)
C_20R_001L_2 = round(pojemnosc(R_L_20R_001L, 0.001, T_20R_001L), 8)
C_20R_01L_2 = round(pojemnosc(R_L_20R_01L, 0.01, T_20R_01L), 8)
nano = 1e-9

[
    ("C_0R_001L [nF]", round(C_0R_001L/nano,2)),
    ("C_0R_01L [nF]", round(C_0R_01L/nano, 2)),
    ("C_20R_001L [nF]", round(C_20R_001L/nano, 2)),
    ("C_20R_01L [nF]", round(C_20R_01L/nano, 2)),
    ("C_20R_001L_2 [nF]", round(C_20R_001L_2/nano, 2)),
    ("C_20R_01L_2 [nF]", round(C_20R_01L_2/nano, 2))
]
```

Out[183]:
```
6-element Array{Tuple{ASCIIString,Float64},1}:
 ("C_0R_001L [nF]",480.0)
 ("C_0R_01L [nF]",540.0)
 ("C_20R_001L [nF]",570.0)
 ("C_20R_01L [nF]",520.0)
 ("C_20R_001L_2 [nF]",570.0)
 ("C_20R_01L_2 [nF]",520.0)
```

Wyznaczenie błędu pojemności kondensatora ΔC z przebiegów na oscyloskopie

Równanie zawierające pojemność kondensatora:

$$\omega = \sqrt{\omega_0^2 - \beta^2} = \sqrt{\frac{1}{LC} - \frac{R^2}{4L^2}}$$

Równanie na pojemność kondensatora:

$$C = \frac{1}{L\left(\omega^2 + \frac{R^2}{4L^2}\right)}$$

Błąd możemy wyznaczyć następująco:

$$\Delta C = C(L + \Delta L, R + \Delta R, T + \Delta T) - C(L, R, T)$$

Dla odpowiednich ustawień na oporniku dekadowym R i indukcji dekadowej L błędy pojemności kondensatora wynoszą:

In [182]:
```
pojemnosc_blad(R, L, T, ΔR, ΔT) = pojemnosc(R + ΔR, L, T + ΔT) - pojemnosc(R, L, T) # [F]

C_0R_001L_blad = round(pojemnosc_blad(R_L_0R_001L, 0.001, T_0R_001L, R_L_0R_001L_blad, ΔT)
C_0R_01L_blad = round(pojemnosc_blad(R_L_0R_01L, 0.01, T_0R_01L, R_L_0R_01L_blad, ΔT), 8)
C_20R_001L_blad = round(pojemnosc_blad(R_L_0R_001L + 20, 0.001, T_20R_001L, R_L_0R_001L_bl
C_20R_01L_blad = round(pojemnosc_blad(R_L_0R_01L + 20, 0.01, T_20R_01L, R_L_0R_01L_blad, Δ
C_20R_001L_blad_2 = round(pojemnosc_blad(R_L_20R_001L, 0.001, T_20R_001L, R_L_20R_001L_bla
C_20R_01L_blad_2 = round(pojemnosc_blad(R_L_20R_01L, 0.01, T_20R_01L, R_L_20R_01L_blad, ΔT

[
    ("C_0R_001L_blad [nF]", round(C_0R_001L_blad/nano,2)),
    ("C_0R_01L_blad [nF]", round(C_0R_01L_blad/nano, 2)),
    ("C_20R_001L_blad [nF]", round(C_20R_001L_blad/nano, 2)),
    ("C_20R_01L_blad [nF]", round(C_20R_01L_blad/nano, 2)),
    ("C_20R_001L_blad_2 [nF]", round(C_20R_001L_blad_2/nano, 2)),
    ("C_20R_01L_blad_2 [nF]", round(C_20R_01L_blad_2/nano, 2))
]
```

Out[182]:
```
6-element Array{Tuple{ASCIIString,Float64},1}:
 ("C_0R_001L_blad [nF]",190.0)
 ("C_0R_01L_blad [nF]",60.0)
 ("C_20R_001L_blad [nF]",210.0)
 ("C_20R_01L_blad [nF]",60.0)
 ("C_20R_001L_blad_2 [nF]",210.0)
 ("C_20R_01L_blad_2 [nF]",60.0)
```

Średnia pojemność kondensatora $\bar{C}$

$$\bar{C} = \frac{\sum C_i/\Delta C_i}{\sum 1/\Delta C_i}$$

Pojemność nie zmienia się znacznie przy zmianach ustawień na oporniku dekadowym i indukcji dekadowej (teoretycznie nie powinna), w związku z tym liczymy średnią pojemność $\bar{C}$:

In [184]:
```
C_mean = wmean(
    [C_0R_001L, C_0R_01L, C_20R_001L, C_20R_01L],
    [C_0R_001L_blad, C_0R_01L_blad, C_20R_001L_blad, C_20R_01L_blad]
)
("C_mean [nF]", round(C_mean/nano,2))
```

Out[184]:
```
("C_mean [nF]",528.32)
```

Błąd średniej pojemności kondensatora $\Delta\bar{C}$

Błąd średniej ważonej można przedstawić jako:

$$\Delta\bar{C} = \frac{\sum \Delta C_i}{n}$$

In [186]:
```
C_mean_blad = werror([C_0R_001L_blad, C_0R_01L_blad, C_20R_001L_blad, C_20R_01L_blad])
("C_mean_blad [nF]", round(C_mean_blad/nano,2))
```

Out[186]:
```
("C_mean_blad [nF]",130.0)
```

Wyznaczanie rezystancji krytycznych R_c

Czyli dla danych L i C, jeżeli opór układu (zadany+pasożytniczy) jest większy od R_c to na oscyloskopie powinniśmy zaobserwować przebiegi aperiodyczne (rozwiązaniem równania różniczkowego układu RLC jest funkcja wykładnicza bez członu oscylacyjnego cos)

$$R_c = 2\sqrt{\frac{L}{C}}$$

Posłużymy się pojemnością średnią, opory krytycze R_c dla odpowiednich indukcyjności L wynoszą:

In [188]:
```
rezystancja_kryt(L, C) = 2*sqrt(L/C)

R_C_001L = rezystancja_kryt(0.001, C_mean)
R_C_01L = rezystancja_kryt(0.01, C_mean)


[
    ("R_C_001L [Ω]", round(R_C_001L,2)),
    ("R_C_01L [Ω]", round(R_C_01L, 2))
]
```

Out[188]:
```
2-element Array{Tuple{UTF8String,Float64},1}:
 ("R_C_001L [Ω]",87.01)
 ("R_C_01L [Ω]",275.16)
```

Wyznaczanie błędu rezystancji krytycznych ΔR_c

Błąd możemy wyznaczyć następująco:

$$\Delta R_c = R_c(L + \Delta L, C + \Delta C) - R_c(L, C)$$

Dla odpowiednich ustawień na indukcji dekadowej L błędy rezystancji krytycznych ΔR_c wynoszą:

In [190]:
```
rezystancja_kryt_blad(L, C, ΔC) = rezystancja_kryt(L, C + ΔC) - rezystancja_kryt(L, C)

R_C_001L_blad = rezystancja_kryt_blad(0.001, C_mean, C_mean_blad)
R_C_01L_blad = rezystancja_kryt_blad(0.01, C_mean, C_mean_blad)


[
    ("R_C_001L_blad [Ω]", round(R_C_001L_blad,2)),
    ("R_C_01L_blad [Ω]", round(R_C_01L_blad, 2))
]
```

Out[190]:
```
2-element Array{Tuple{UTF8String,Float64},1}:
 ("R_C_001L_blad [Ω]",-9.06)
 ("R_C_01L_blad [Ω]",-28.66)
```

In [ ]: