Projeto Usando Lugar Geométrico das Raizes Parte 2

Prof. Fernando Passold
Engenharia Elétrica / UPF
Out/2009 – Maio/2020

Resumo

- Parte I:
  - Propostas de “novos” controladores:
    - **PI + ceros:**

$$C(s) = K_1 + \frac{K_2}{s} = \frac{K_1\left(s + \frac{K_2}{K_1}\right)}{s}$$

← Zero próximo do polo

← Pólo na origem

Vantagem: e(∞)=0, Desvantagem: resposta + lenta

    - **Por atraso de fase (Lag Compensator):**

$$C(s) = \frac{K(s + z_c)}{(s + p_c)}$$

← par polo-zero próximo da origem

Vantagem: Resposta + rápida, Desvantagem: e(∞)≠0

Contenido Parte II

- Controlador PD
  - Melhorar respostas transitória
  - Controlador D ideal
  - Vantagens
  - Desvantagens

- Controlador por Avanço de Fase (*Lead Compensator*)

  - Parte III…

Ideias para melhorar Resposta Transitória

Formas de melhorar:

1. **Compensador PD** (*Proportional-plus-Derivative Controller*)
   - Acrescentar um diferenciador puro na malha direta para compensação derivativa ideal (rede ativa)
   - Projetar uma resposta que respeita um valor desejável de sobressinal, com menor tempo de assentamento ($\downarrow$ ts = *settling time*)

2. **Controlador por Avanço de Fase** (*Lead Controller*)
   - Realiza diferenciação aproximada usando rede passiva (acrescenta um zero e um polo distante na malha direta)

Compensação Derivativa Ideal (PD)

$$C(s) = s + z_c$$

- Seleção adequada da posição (do zero) para garantir resposta + rápida
- Modifica RL!
- Exemplo:

Planta → $G(s) = \dfrac{K}{(s+1)(s+2)(s+5)}$

Propostas de Controladores → PD

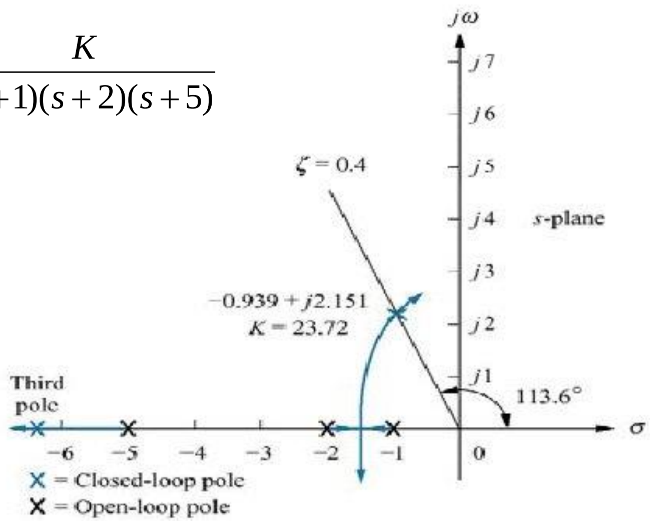
$$C(s)G(s) = \frac{K(s+2)}{(s+1)(s+2)(s+5)}$$ ← Zero em z_c= -2

$$C(s)G(s) = \frac{K(s+3)}{(s+1)(s+2)(s+5)}$$ ← Zero em z_c= -3

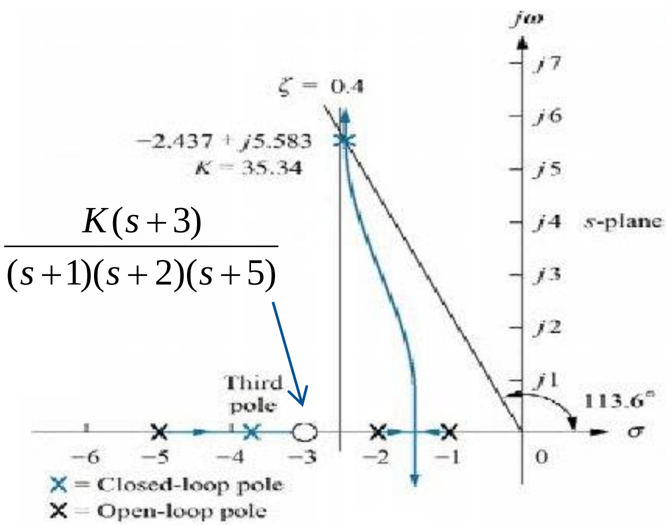
$$C(s)G(s) = \frac{K(s+4)}{(s+1)(s+2)(s+5)}$$ ← Zero em z_c= -4

Compensação Derivativa Ideal (PD)

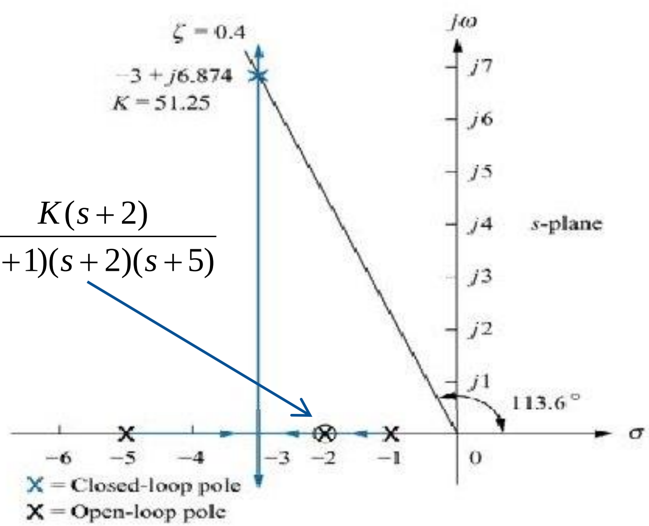
$$G(s)=\frac{K}{(s+1)(s+2)(s+5)}$$



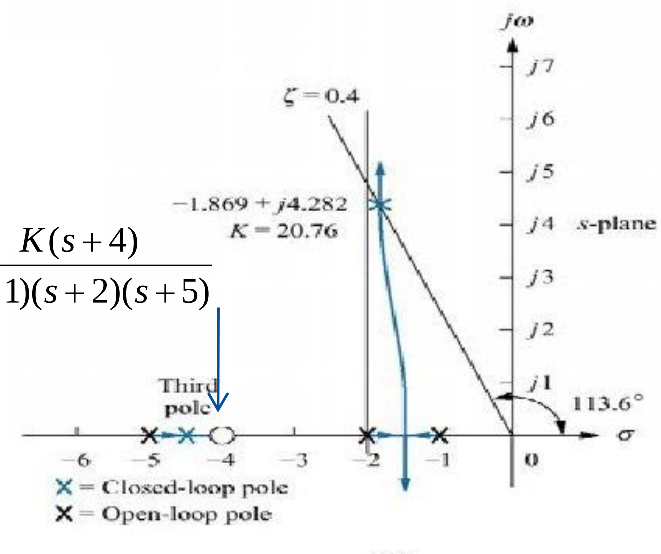
$$C(s)G(s)=\frac{K(s+3)}{(s+1)(s+2)(s+5)}$$



$$C(s)G(s)=\frac{K(s+2)}{(s+1)(s+2)(s+5)}$$

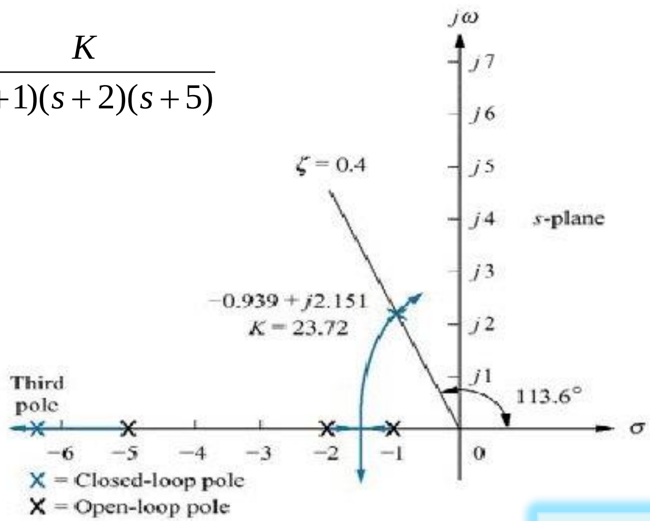


$$C(s)G(s)=\frac{K(s+4)}{(s+1)(s+2)(s+5)}$$

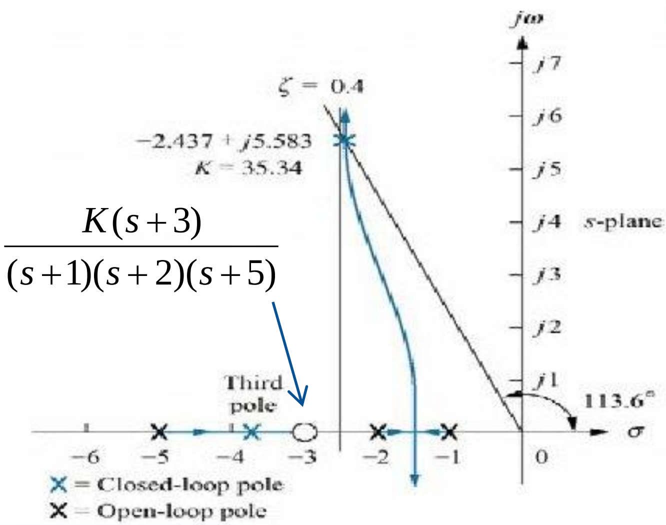


Compensação Derivativa Ideal (PD)

$$G(s)=\frac{K}{(s+1)(s+2)(s+5)}$$



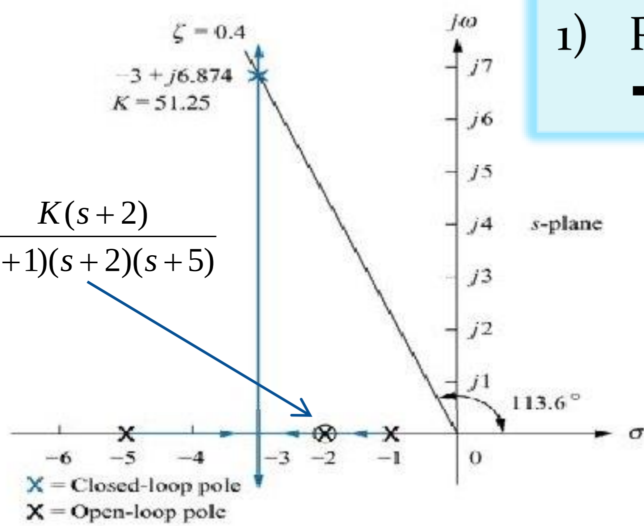
$$C(s)G(s)=\frac{K(s+3)}{(s+1)(s+2)(s+5)}$$



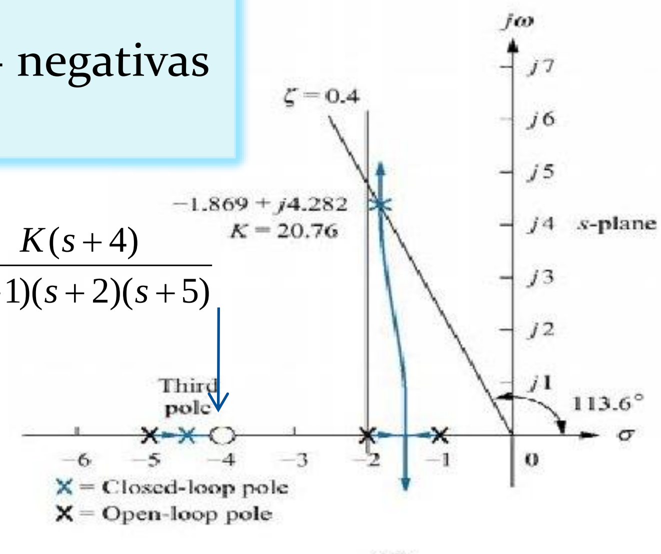
Conclusões:

1) Partes reales + negativas $\rightarrow \downarrow t_s$

$$C(s)G(s)=\frac{K(s+2)}{(s+1)(s+2)(s+5)}$$

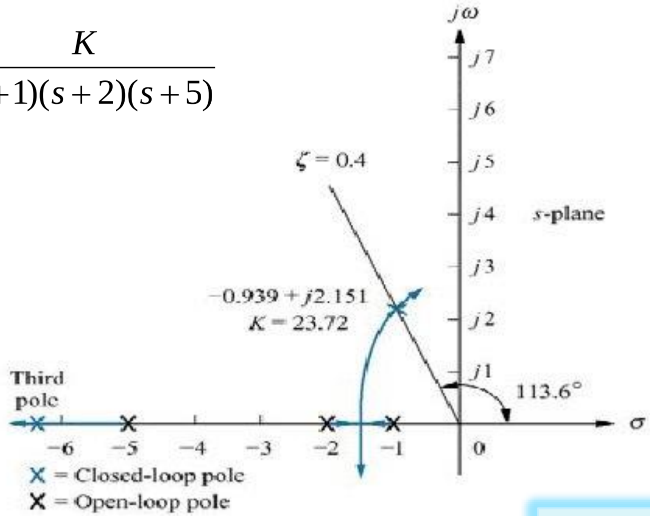


$$C(s)G(s)=\frac{K(s+4)}{(s+1)(s+2)(s+5)}$$

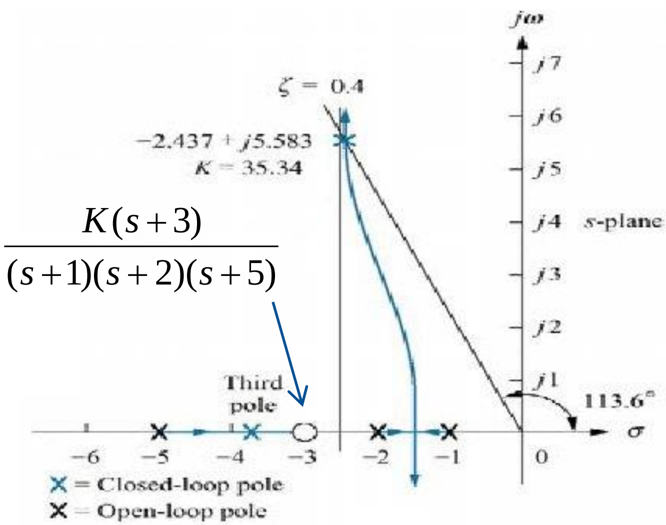


Compensação Derivativa Ideal (PD)

$$G(s) = \frac{K}{(s+1)(s+2)(s+5)}$$

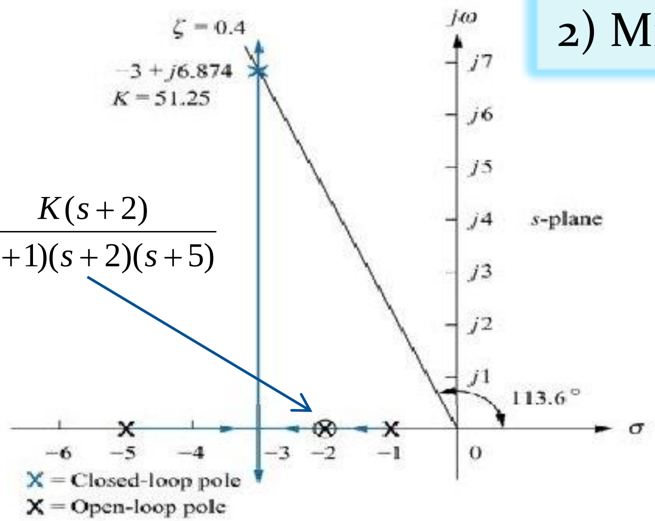


$$C(s)G(s) = \frac{K(s+3)}{(s+1)(s+2)(s+5)}$$

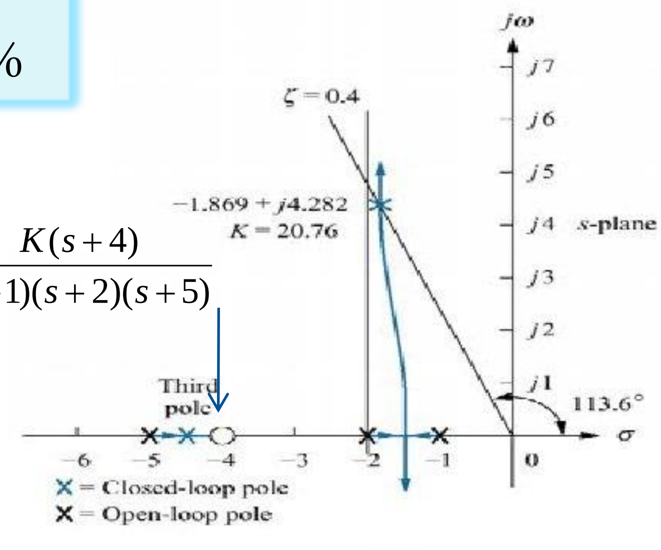


Conclusões:
2) Mismo ζ ≅ OS%

$$C(s)G(s) = \frac{K(s+2)}{(s+1)(s+2)(s+5)}$$

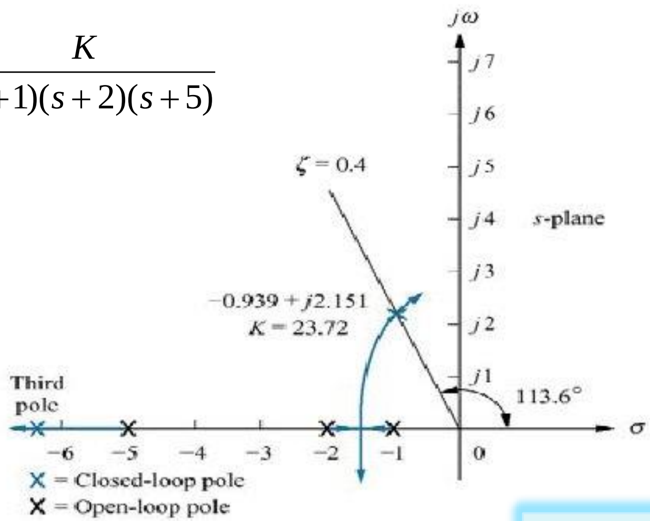


$$C(s)G(s) = \frac{K(s+4)}{(s+1)(s+2)(s+5)}$$

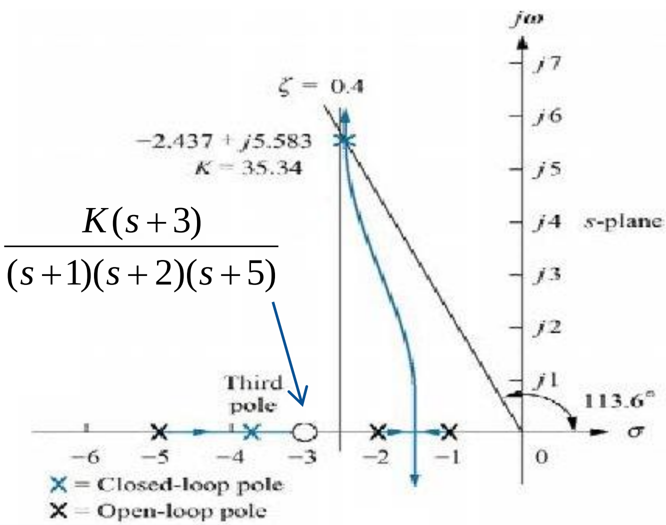


Compensação Derivativa Ideal (PD)

$$G(s) = \frac{K}{(s+1)(s+2)(s+5)}$$



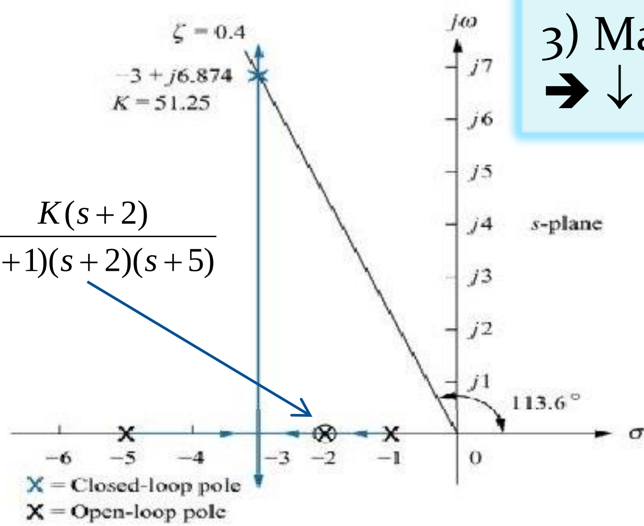
$$C(s)G(s) = \frac{K(s+3)}{(s+1)(s+2)(s+5)}$$



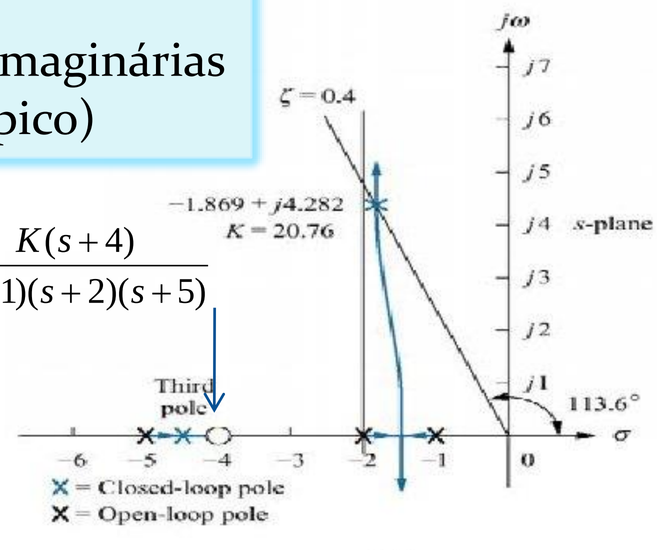
Conclusões:

3) Maiores partes imaginárias ➔ $\downarrow t_p$ (tempo de pico)

$$C(s)G(s) = \frac{K(s+2)}{(s+1)(s+2)(s+5)}$$

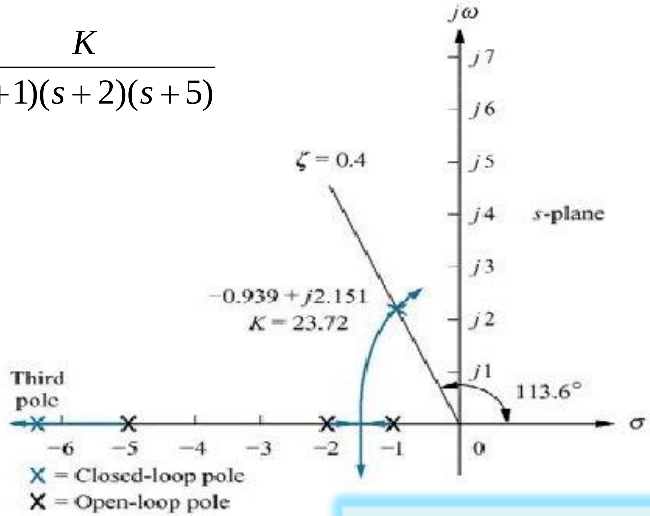


$$C(s)G(s) = \frac{K(s+4)}{(s+1)(s+2)(s+5)}$$

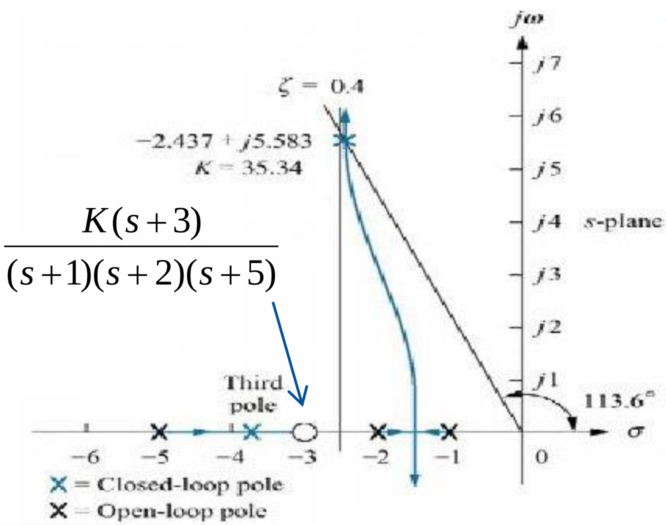


Compensação Derivativa Ideal (PD)

$$G(s) = \frac{K}{(s+1)(s+2)(s+5)}$$



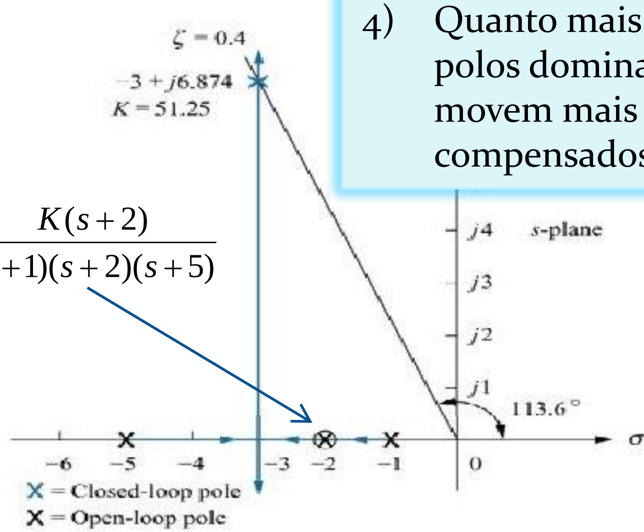
$$C(s)G(s) = \frac{K(s+3)}{(s+1)(s+2)(s+5)}$$



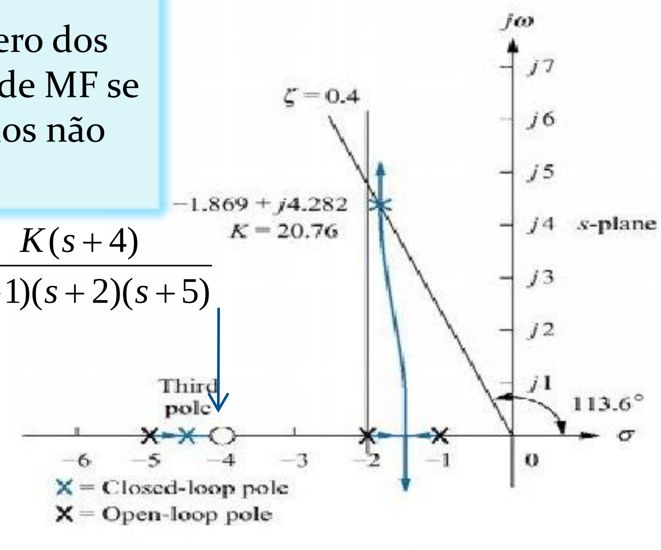
Conclusões:

4) Quanto mais afastado está o zero dos polos dominantes ➔ os polos de MF se movem mais próximos dos polos não compensados (de MA).

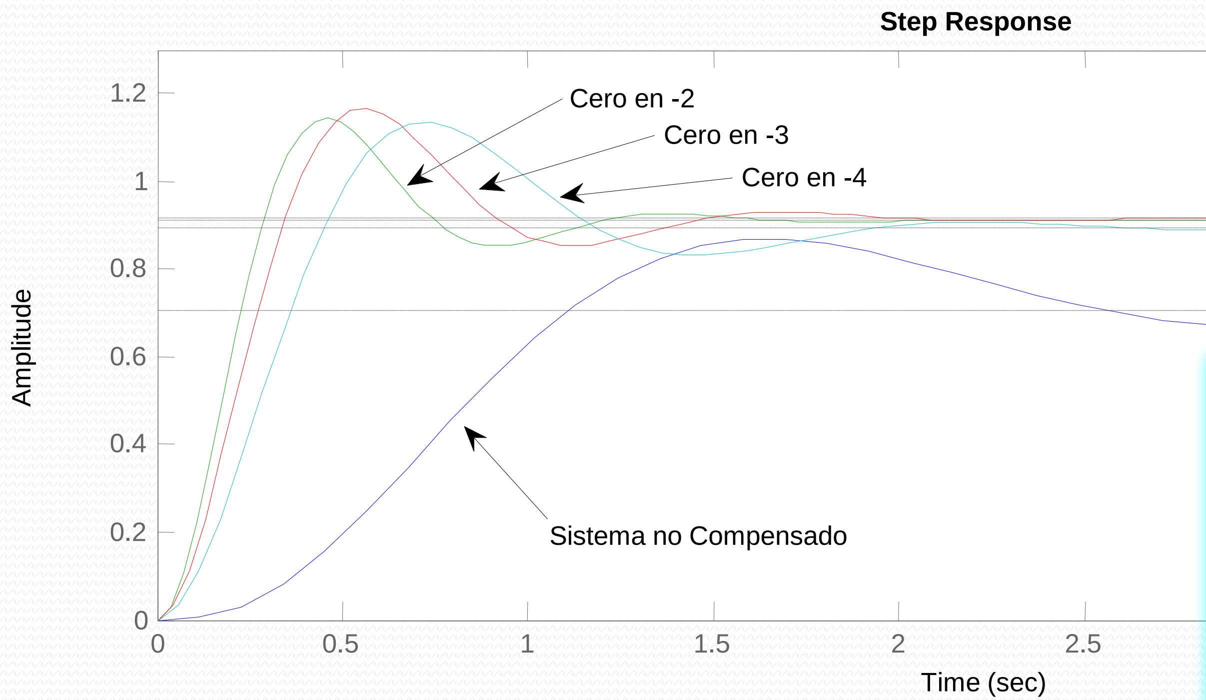
$$C(s)G(s) = \frac{K(s+2)}{(s+1)(s+2)(s+5)}$$



$$C(s)G(s) = \frac{K(s+4)}{(s+1)(s+2)(s+5)}$$



Compensação Derivativa Ideal (PD)



Conclusões:

1. Partes reais + negativas ➔ ↓ t_s;
2. Mesmo ζ ➔ ≅ OS%;
3. Maiores partes imaginarias ➔ ↓ t_p (tempos de pico)
4. Quanto mais afastado está o zero dos polos dominantes ➔ os polos de MF se movem mais próximos dos polos não compensados.

Planta → $$G(s) = \frac{K}{(s+1)(s+2)(s+5)}$$

Propostas de Controladores → PD

$$C(s)G(s) = \frac{K(s+2)}{(s+1)(s+2)(s+5)}$$ ← Zero em z_c= -2

$$C(s)G(s) = \frac{K(s+3)}{(s+1)(s+2)(s+5)}$$ ← Zero em z_c= -3

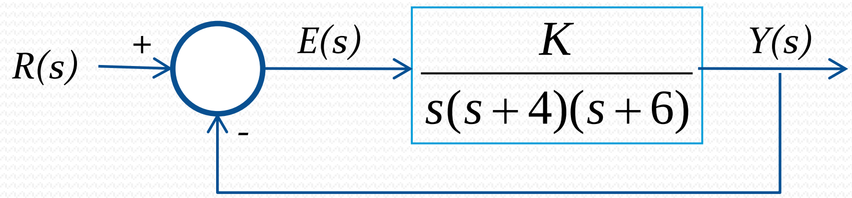
$$C(s)G(s) = \frac{K(s+4)}{(s+1)(s+2)(s+5)}$$ ← Zero em z_c= -4

Vantagens principais:

- Menores t_s,
- Menores OS%.

Compensação Derivativa Ideal (PD)

- Outro exemplo:



- Requisitos: $\%OS < 16\%,\ 3 \times \downarrow t_s$
- *Solução:*

1. Descobrindo ζ desejado:

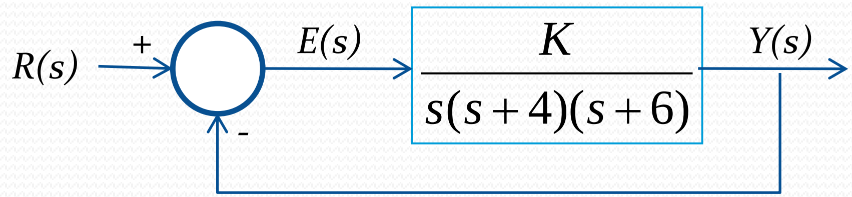
$$\zeta = \frac{-\ln(\%OS/100)}{\sqrt{\pi^2 + \ln^2(\%OS/100)}} = 0{,}504$$

Matlab:

```
>> num=1;
>> zeta=(-log(16/100))/(sqrt(pi*pi+(log(16/100))^2))
zeta =
   0.5039
>> den=poly([0 -4 -6]);
>> g=tf(num,den);
>> zpk(g)
Zero/pole/gain:
      1
-------------
s (s+6) (s+4)
>>
```

Compensação Derivativa Ideal (PD)

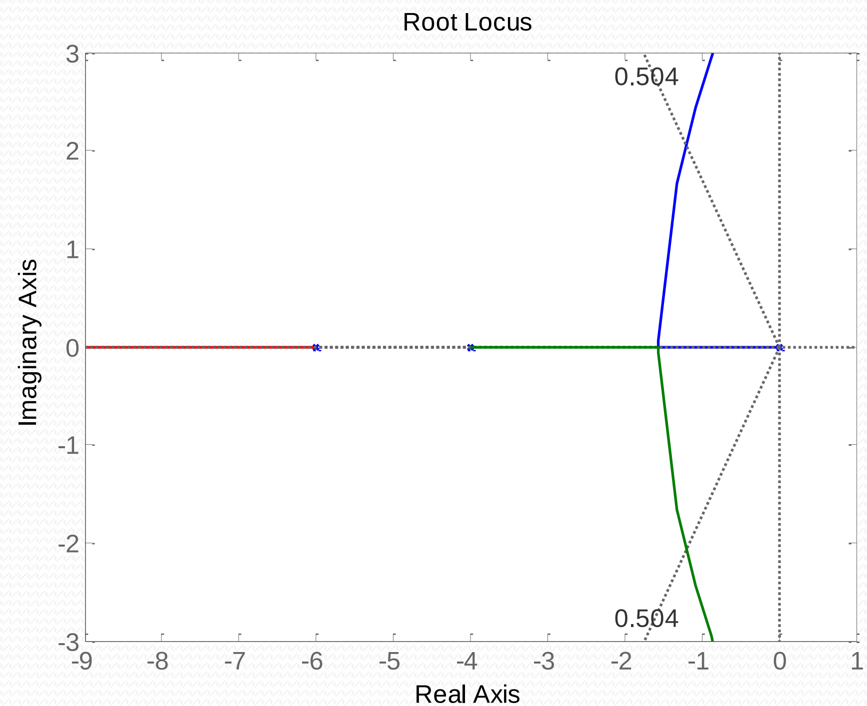
- Outro exemplo:



  - Requisitos: $\%OS < 16\%,\ 3 \times \downarrow t_s$
  - *Solução:*

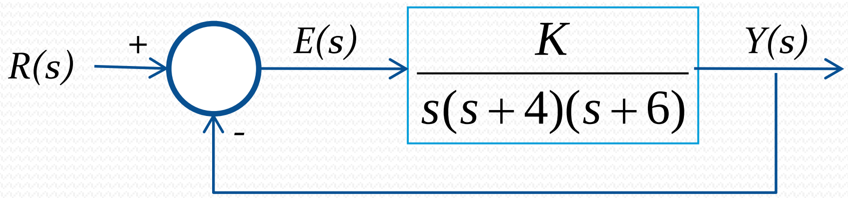
2. Verificando RL original...

```
>> zpk(g)
Zero/pole/gain:
      1
-------------
s (s+6) (s+4)
>> rlocus(g)
>> sgrid(zeta,0)
>> axis([-9 1 -3 3])
```



Compensação Derivativa Ideal (PD)

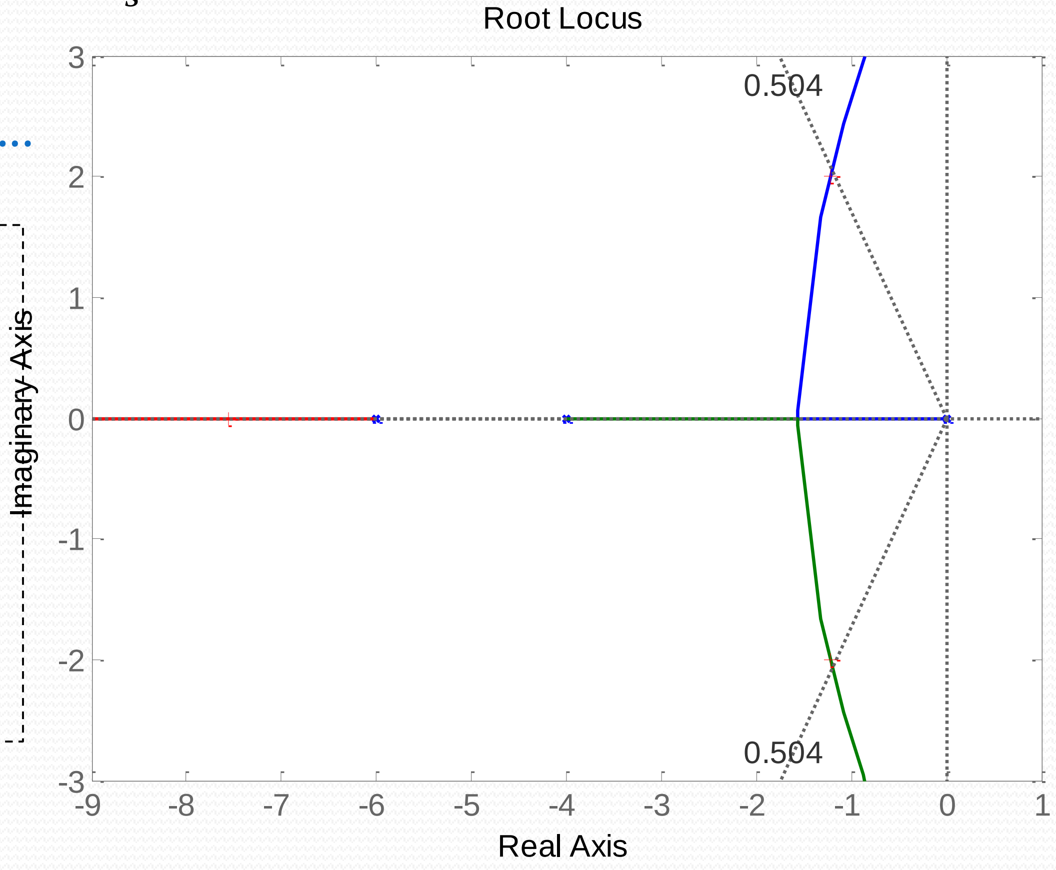
- Outro exemplo:



  - Requisitos: $\%OS < 16\%,\ 3 \times \downarrow t_s$
  - *Solução:*

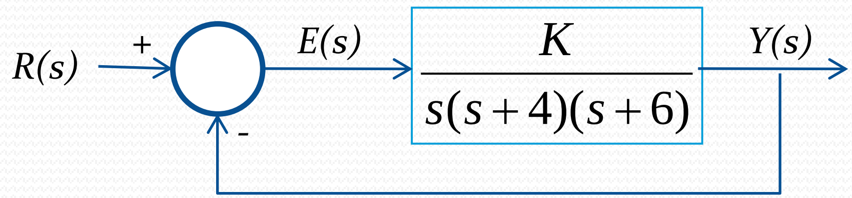
3. Descobrindo K necessário...

```
>> [k,poles]=rlocfind(g)
Select a point in the graphics window
selected_point =
   -1.2156 + 2.0031i
k =
    41.6859
poles =
   -7.5532
   -1.2234 + 2.0056i
   -1.2234 - 2.0056i
>>
```



Compensação Derivativa Ideal (PD)

- Outro exemplo:



  - Requisitos: $\%OS < 16\%$, $3 \times \downarrow t_s$
  - *Solução:*

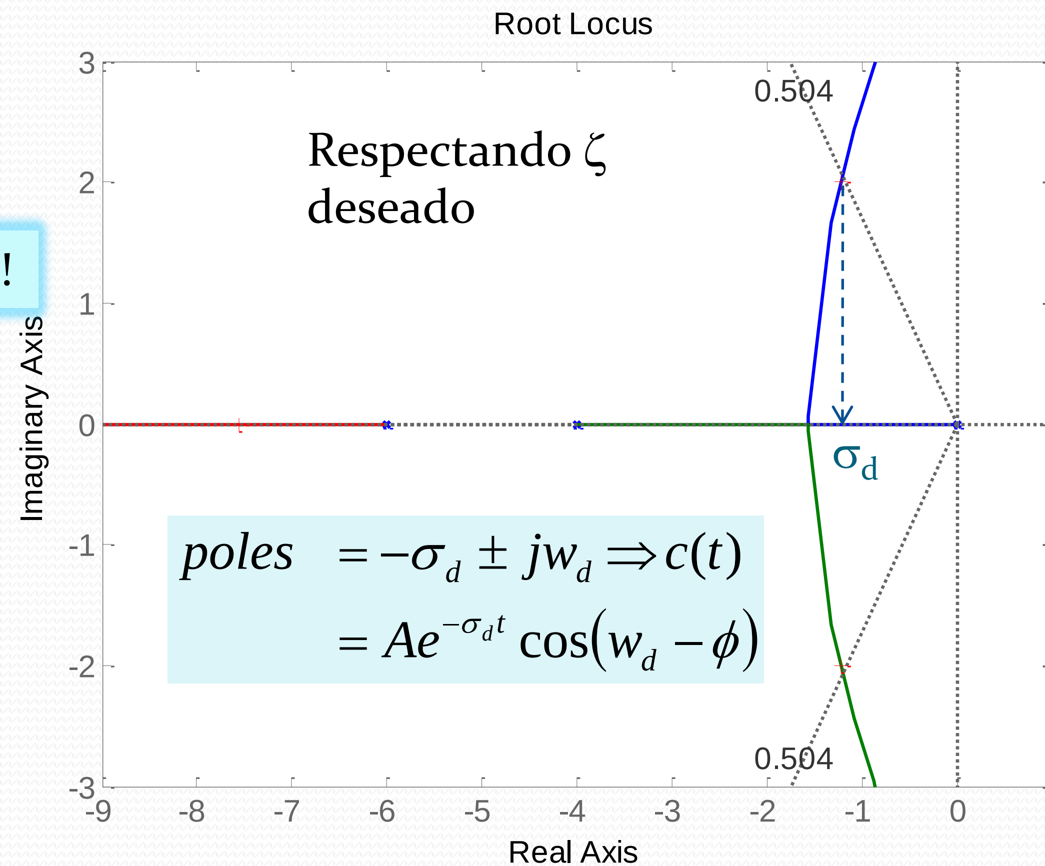
4. Acelerando o sistema: $\downarrow t_s$

```
poles =
  -7.5532
  -1.2234 + 2.0056i
  -1.2234 - 2.0056i
>> Ts=4/real(-poles(2))
Ts =
   3.2696
>> Ts=4/real(-poles(2))
Ts =
   3.2696
> newTs=Ts/3
newTs =
   1.0899
>>
```

t_s original!

Novo t_s!

$$Ts = \frac{4}{\sigma_d}$$

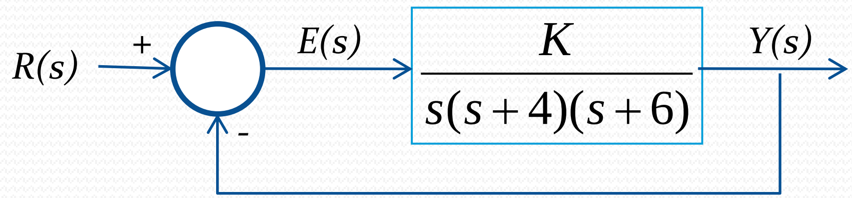


Respectando ζ desejado

$$poles = -\sigma_d \pm jw_d \Rightarrow c(t) = Ae^{-\sigma_d t}\cos(w_d - \phi)$$

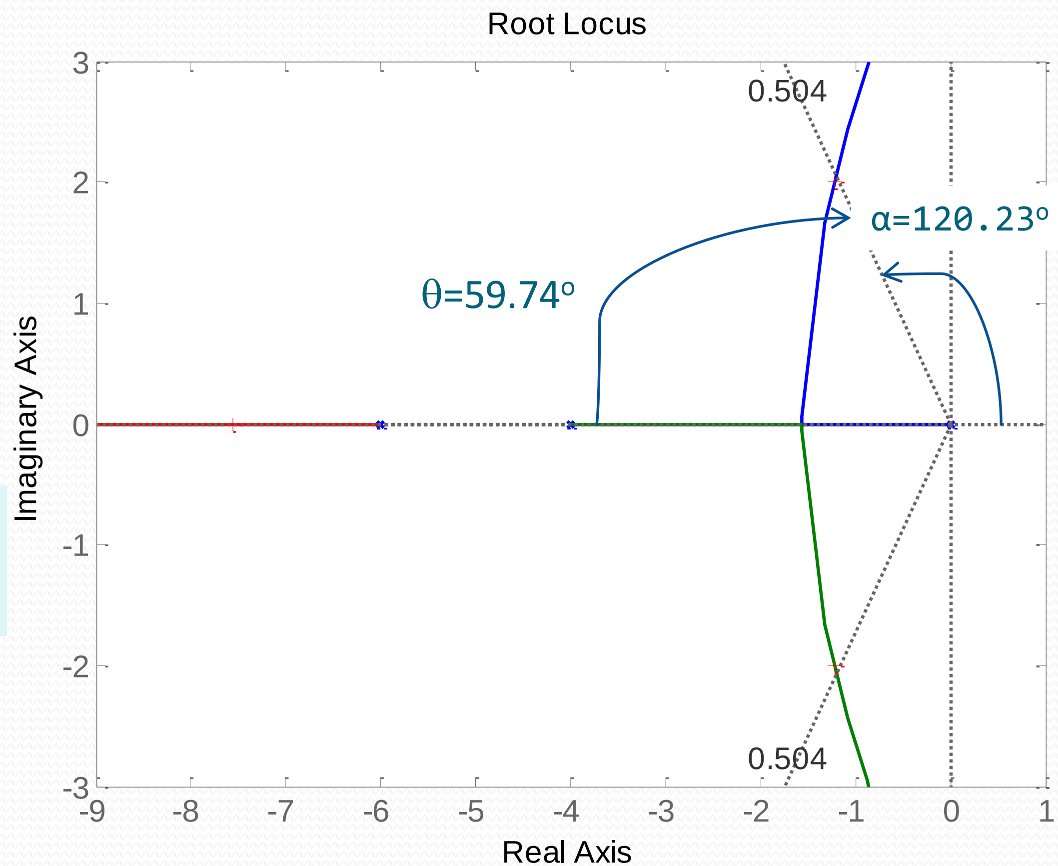
Compensação Derivativa Ideal (PD)

- Outro exemplo:



  - Requisitos: $\%OS < 16\%,\ 3 \times \downarrow t_s$
  - *Solução:*

5. Descobrindo a nova posição do polo de malha fechada para o novo t_s




```
newTs =
    1.0899
>> newsigma=4/newTs
newsigma =
    3.6702
>> theta=acos(zeta)
theta =
     1.0427
>> theta*180/pi
ans =
    59.7438
>>
```

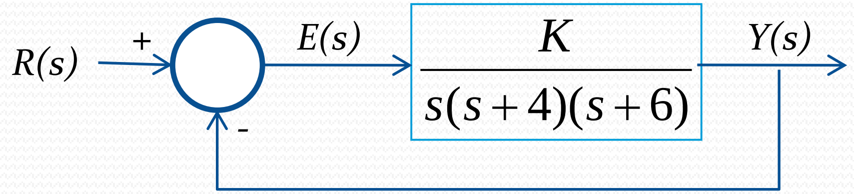
Novo σ para o Nuevo t_s !

$$T_s = \frac{4}{\sigma_d} = \frac{4}{\zeta\, w_n}$$

$$\zeta = \cos\theta$$

Compensação Derivativa Ideal (PD)

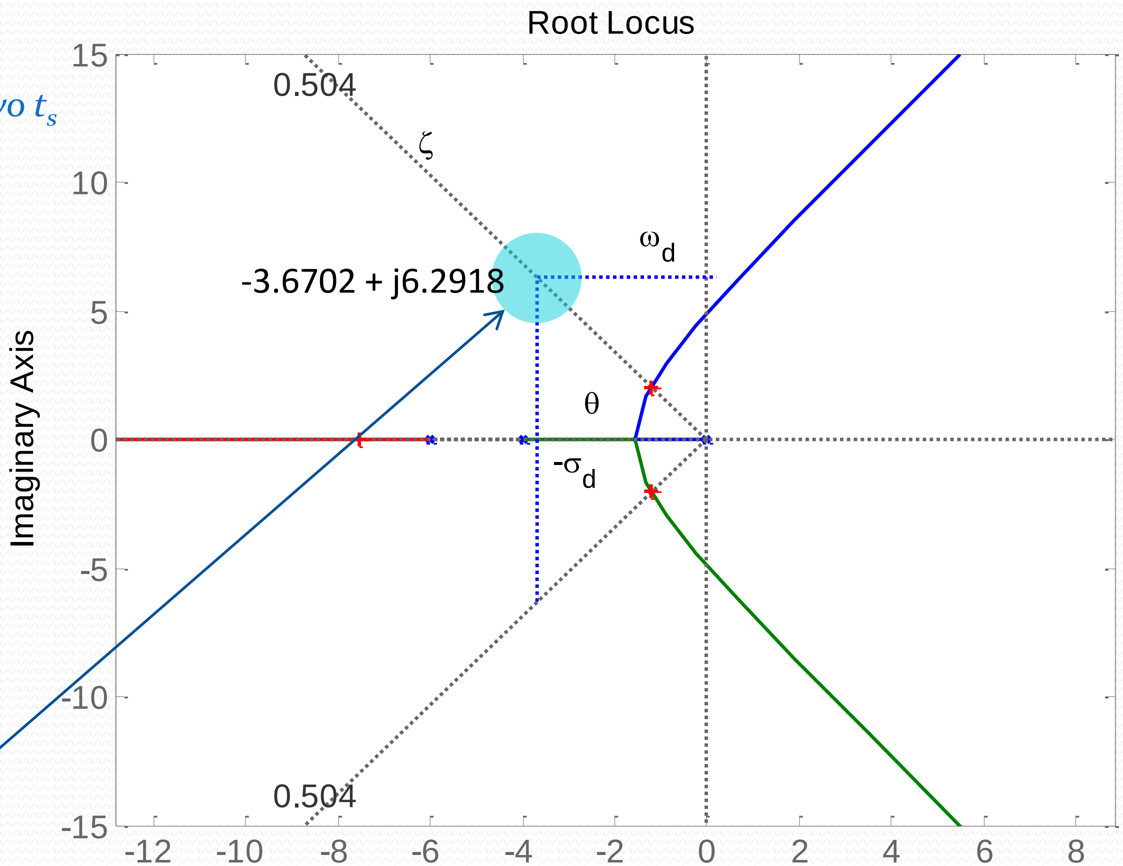
- Outro exemplo:



  - Requisitos: $\%OS < 16\%,\ 3 \times \downarrow t_s$
  - *Solução:*

5. Descobrindo a nova posição do polo de malha fechada para o novo t_s

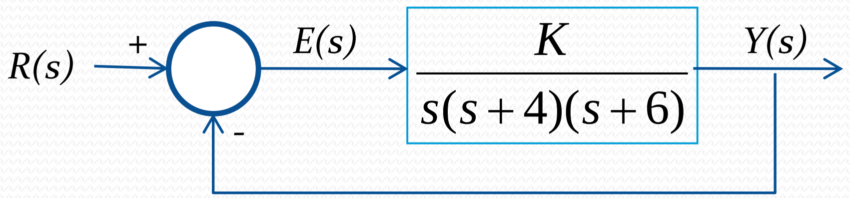
```
>> newomega=newsigma*tan(theta)
newomega =
    6.2918
>> hold on;
>> plot([-newsigma 0.2],[newomega newomega],'b:')
>> plot([-newsigma -newsigma],[-newomega newomega],'b:')
```



Ponto desejado no RL!
Mas este lugar está fora do RL...

Compensação Derivativa Ideal (PD)

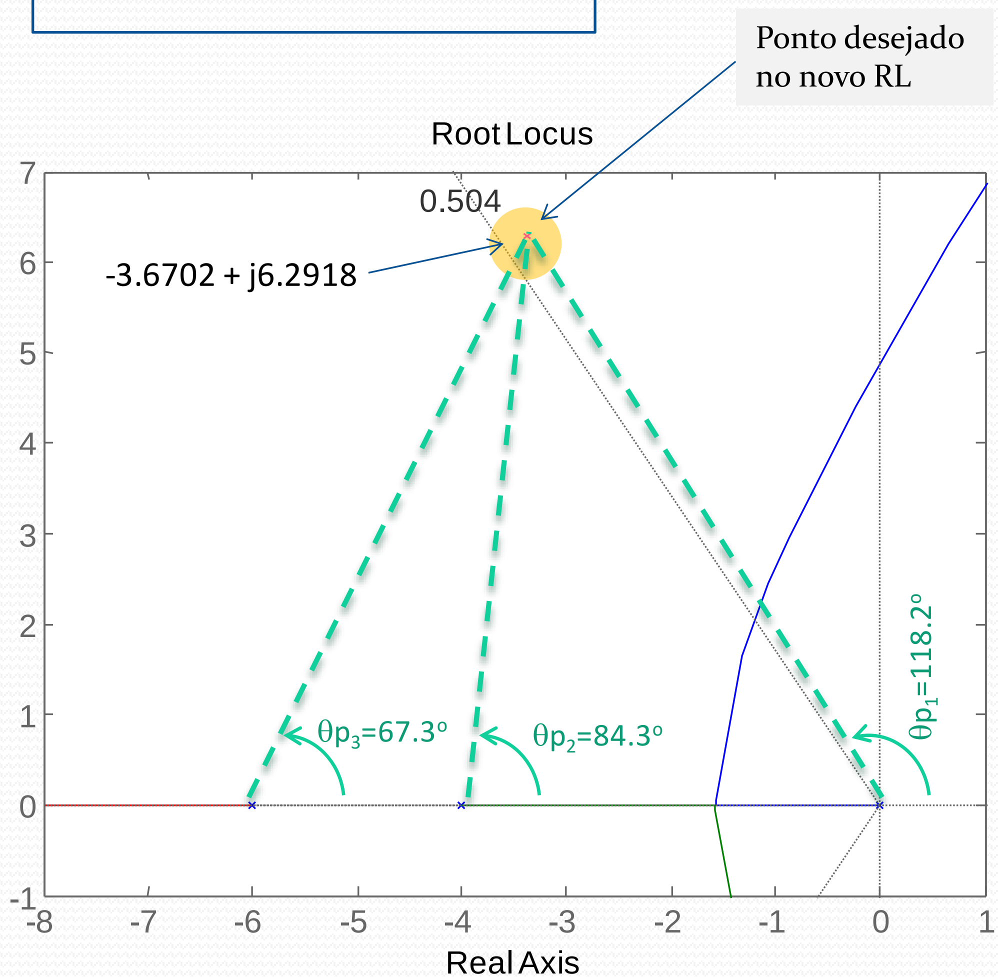
- Outro exemplo:



  - Requisitos: $\%OS < 16\%,\ 3 \times \downarrow t_s$
  - *Solução:*

6. Determinando a posição desejada para o zero do PD:

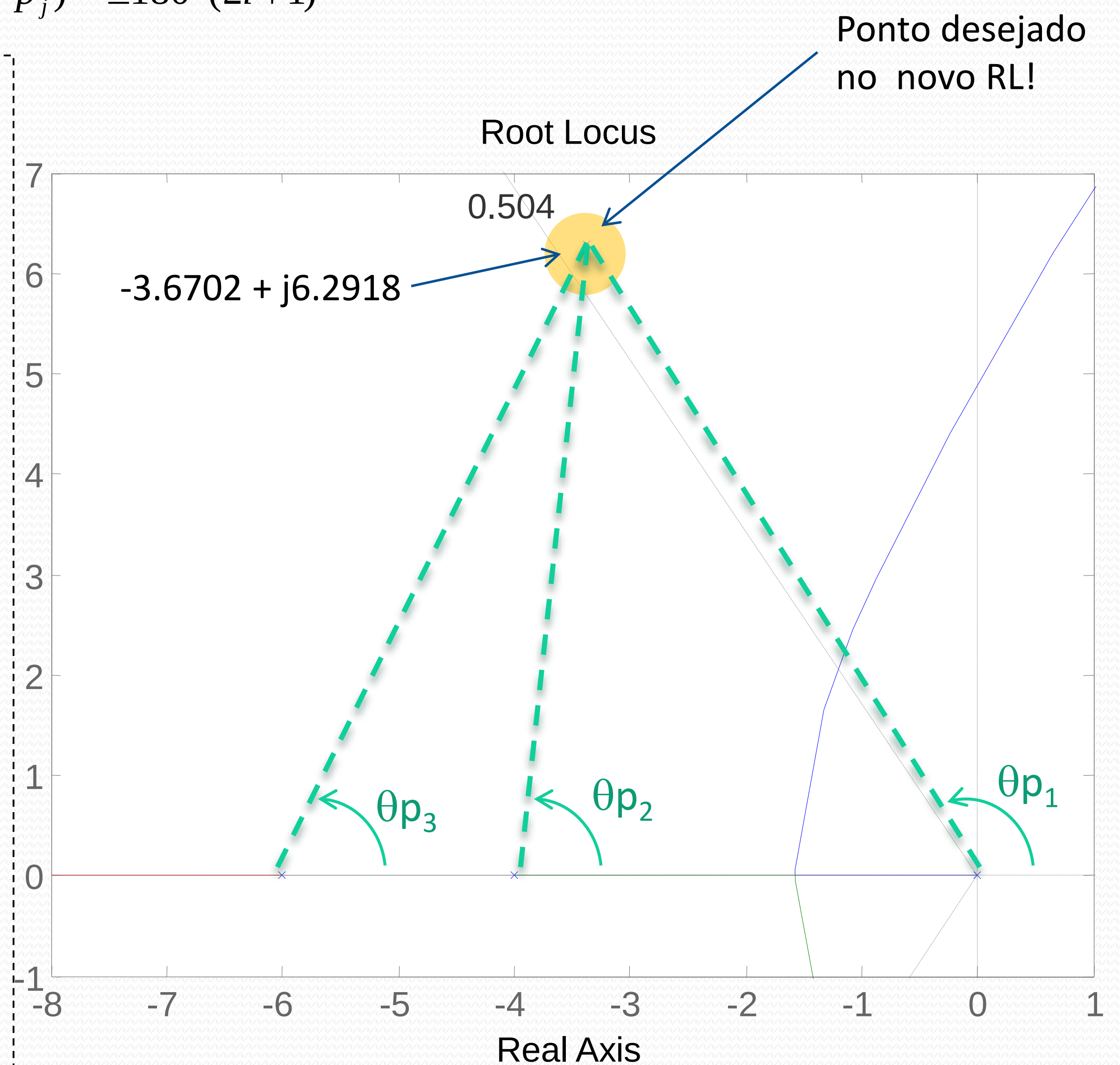
$$\sum_m \angle(s - c_i) - \sum_n \angle(s - p_j) = \pm 180^o (2i+1)$$



6. Determinando a posição desejada para o zero do PD

$$\sum_m \angle(s-c_i) - \sum_n \angle(s-p_j) = \pm 180^o(2i+1)$$

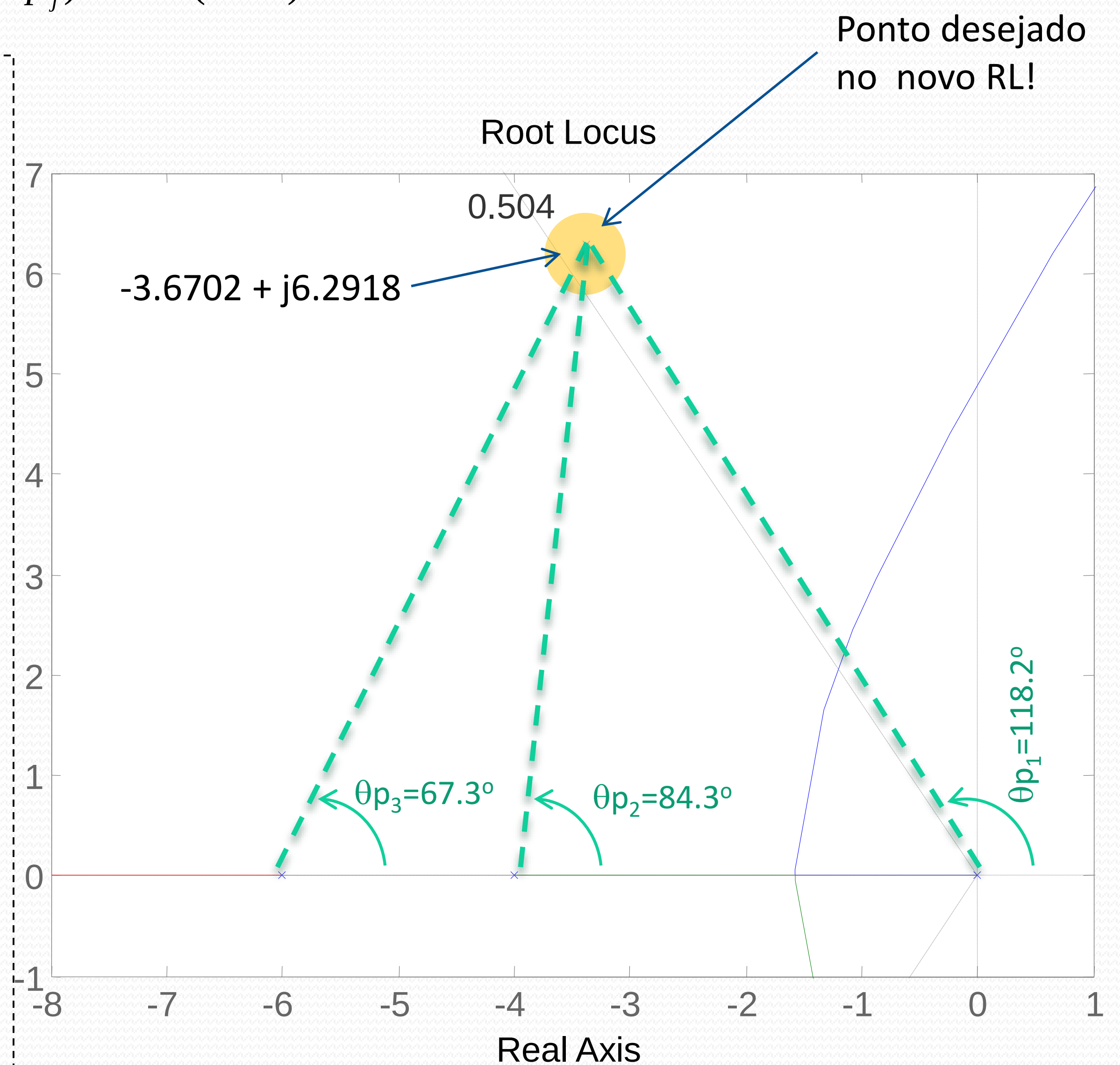
```
>> th_p1=atan2(newomega,-newsigma)
th_p1 =
   2.0626
>> th_p1*180/pi
ans =
  118.1757
>> th_p2=atan2(newomega,4-newsigma)
th_p2 =
   1.4710
>> th_p2*180/pi
ans =
   84.2838
>> th_p3=atan2(newomega,6-newsigma)
th_p3 =
   1.1749
>> th_p3*180/pi
ans =
   67.3164
>> sum_th_p=th_p1+th_p2+th_p3
sum_th_p =
   4.7085
>> sum_th_p*180/pi
ans =
  269.7759
>>
```



6. Determinando a posição desejada para o zero do PD

$$\sum_m \angle(s-c_i)-\sum_n \angle(s-p_j)=180^o(2i+1)$$

```
>> th_p1=atan2(newomega,-newsigma)
th_p1 =
   2.0626
>> th_p1*180/pi
ans =
 118.1757
>> th_p2=atan2(newomega,4-newsigma)
th_p2 =
   1.4710
>> th_p2*180/pi
ans =
  84.2838
>> th_p3=atan2(newomega,6-newsigma)
th_p3 =
   1.1749
>> th_p3*180/pi
ans =
  67.3164
>> sum_th_p=th_p1+th_p2+th_p3
sum_th_p =
   4.7085
>> sum_th_p*180/pi
ans =
 269.7759
>>
```

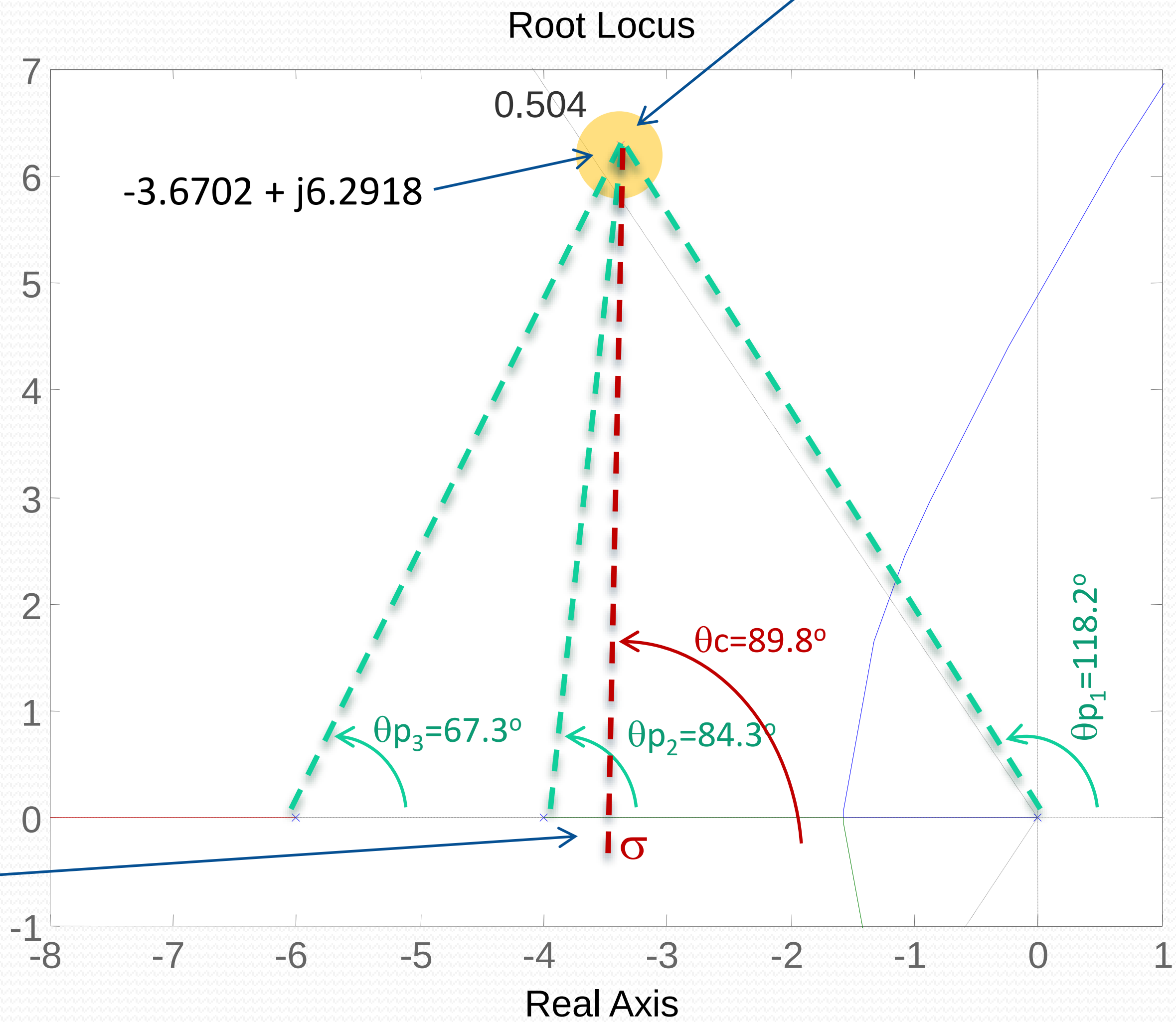


6. Determinando a posição desejada para o zero do PD

$$\sum_m \angle(s-c_i) - \sum_n \angle(s-p_j) = 180^o(2i+1)$$

```
>> sum_th_p=th_p1+th_p2+th_p3
sum_th_p =
    4.7085
>> sum_th_p*180/pi
ans =
  269.7759
>>
>> th_c=sum_th_p-pi
th_c =
    1.5669
>> th_c*180/pi
ans =
   89.7759
>>
```

Ponto desejado no novo RL!



Determinado o ponto σ para o zero do PD!

6. Determinando a posição desejada para o zero do PD

$$\frac{6.2918}{3.3702 - sigma} = \tan\left(180^o - 89.7759^o\right)$$

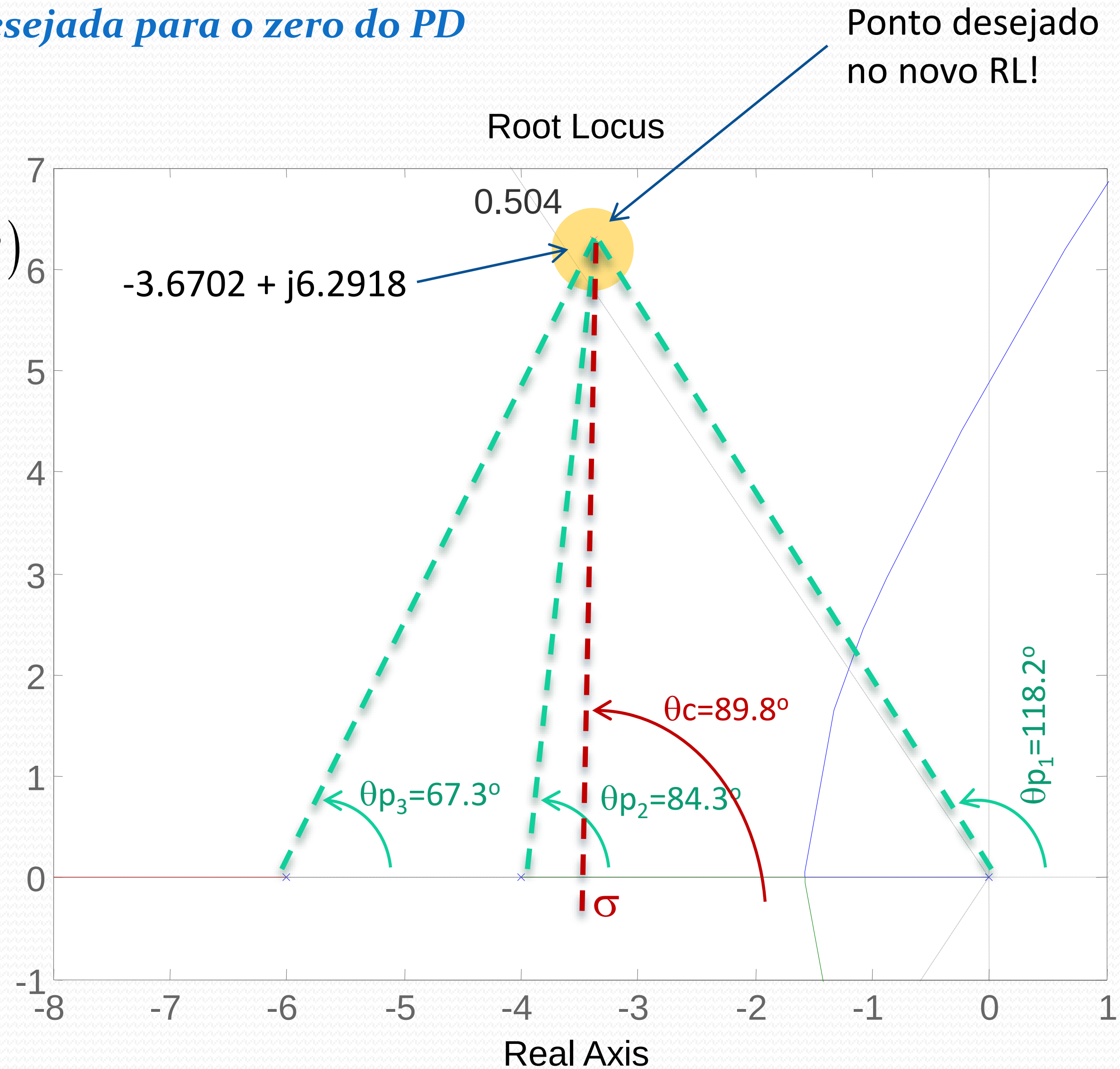
```
>> sigma = newsigma - ( newomega /
tan(pi -th_c) )
sigma =
   3.3948
>>
```

El PD se queda:

$$C(s) = K(s + 3.3948)$$

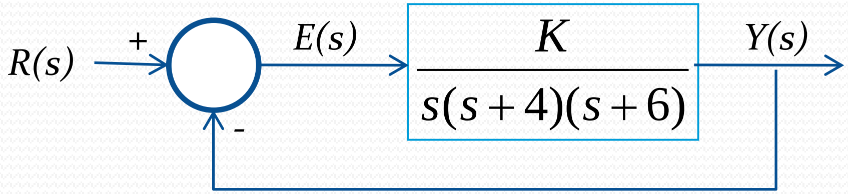
y:

$$C(s)G(s) = \frac{K(s+3.3948)}{s(s+4)(s+6)}$$



Compensação Derivativa Ideal (PD)

- Outro exemplo:

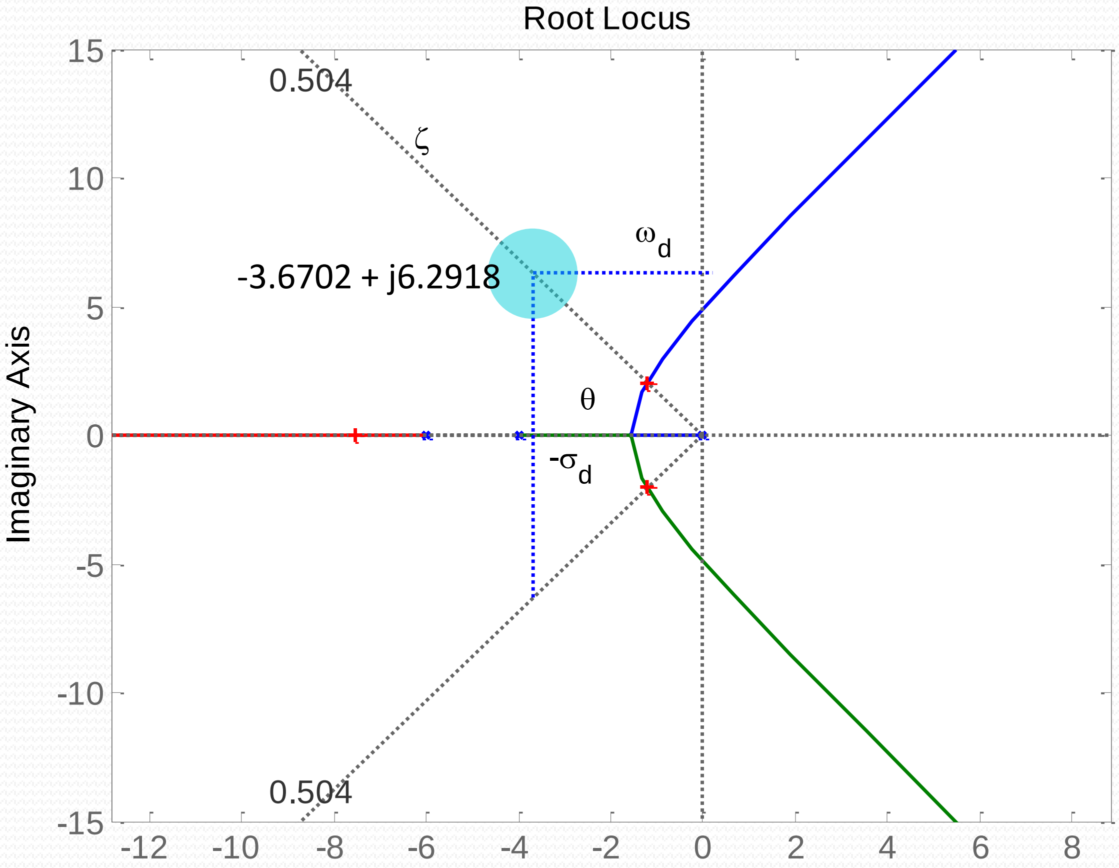


  - Requisitos: $\%OS < 16\%,\ 3 \times \downarrow t_s$
  - *Solução:*

$$C(s) = K(s+3.3948)$$

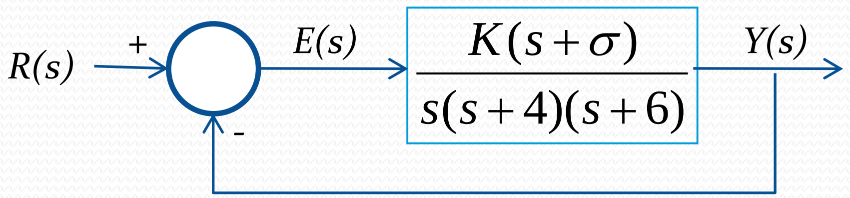
$$C(s)G(s) = \frac{K(s+3.3948)}{s(s+4)(s+6)}$$

7. Verificando o RL final...



Compensação Derivativa Ideal (PD)

- Outro exemplo:



  - Requisitos: $\%OS < 16\%,\ 3 \times \downarrow t_s$
  - *Solução:*

$$C(s) = K(s + 3.3948)$$

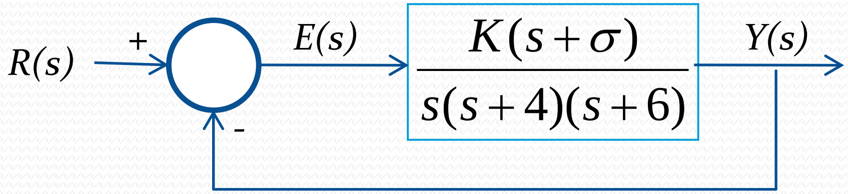
$$C(s)G(s) = \frac{K(s + 3.3948)}{s(s+4)(s+6)}$$

7. Verificando o RL final e K necessário...

```
>> num2=[1 sigma];
>> den2=den;
>> cg=tf(num2,den2);
>> zpk(cg)

Zero/pole/gain:
  (s+3.395)
-------------
s (s+6) (s+4)


>>
>> figure(3);rlocus(cg)
```

Compensação Derivativa Ideal (PD)

- Outro exemplo:

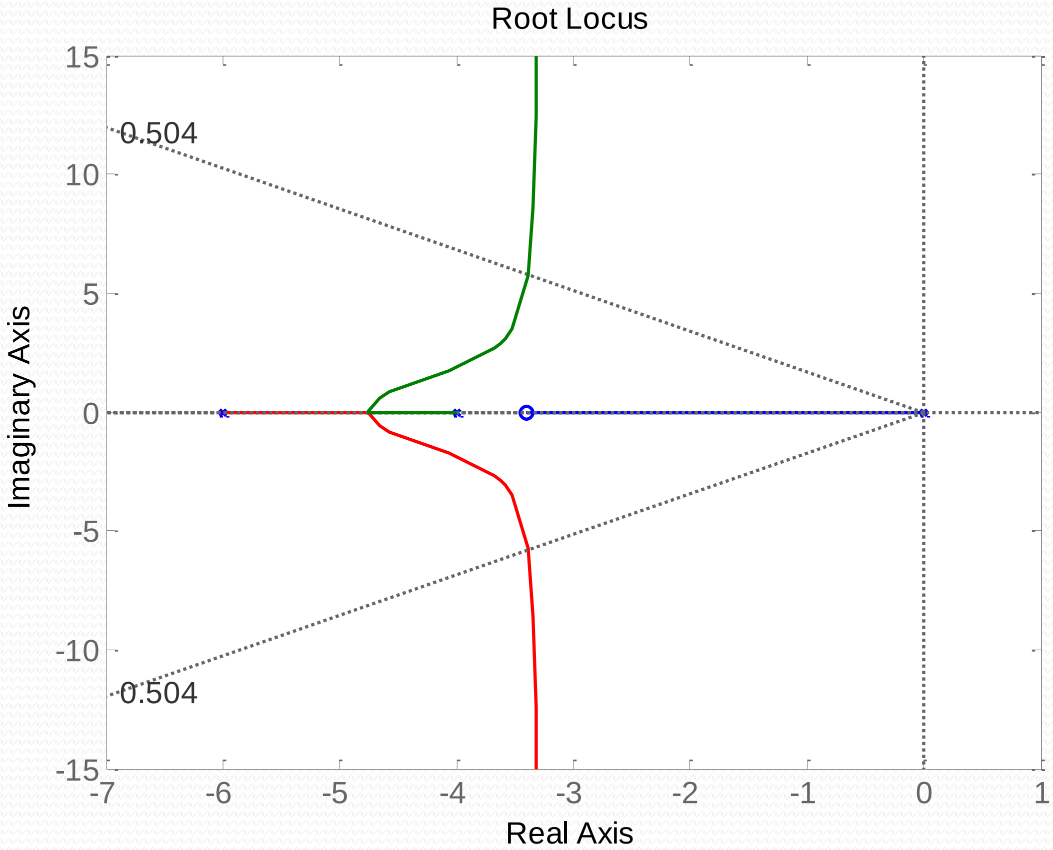


  - Requisitos: $\%OS < 16\%$, $3 \times \downarrow t_s$
  - *Solução:*

$$C(s) = K(s + 3.3948)$$

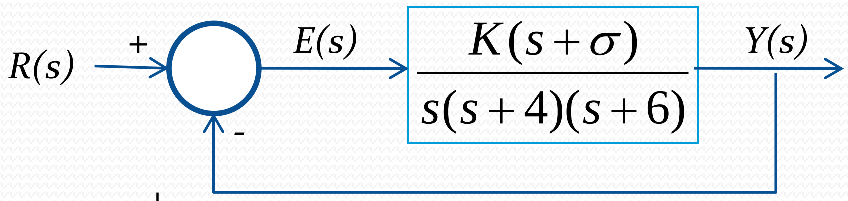
$$C(s)G(s) = \frac{K(s+3.3948)}{s(s+4)(s+6)}$$

7. Verificando o RL final e K necessário...



Compensação Derivativa Ideal (PD)

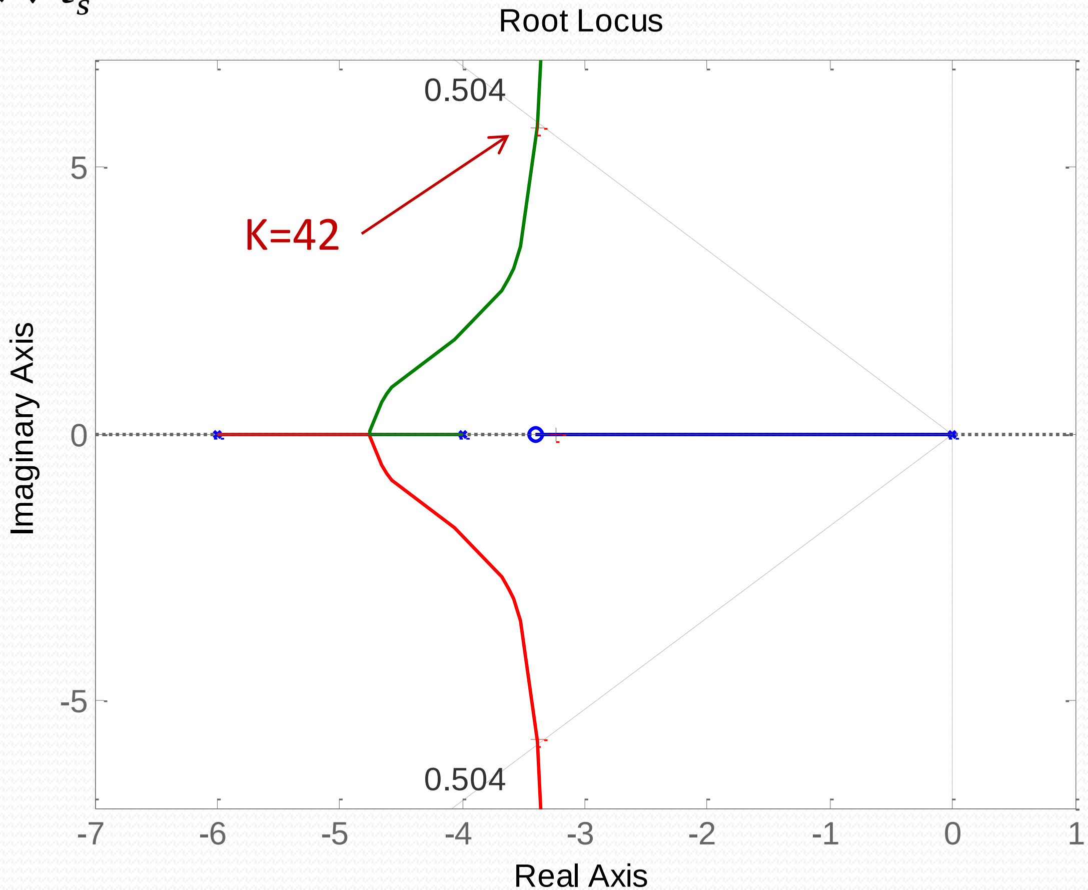
- Outro exemplo:



  - Requisitos: $\%OS < 16\%,\ 3 \times \downarrow t_s$
  - *Solução:*

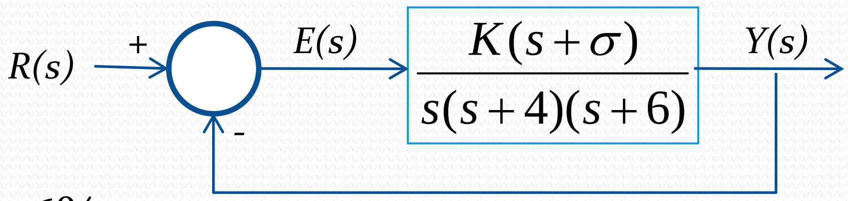
7. Verificando o RL final e K necessário...

```
>> figure(3);rlocus(cg)
>> sgrid(zeta,0)
>> axis([-7 1 -7 7])
>> rlocfind(cg)
Select a point in the graphics window
selected_point =
  -3.4076 + 5.7174i
ans =
  42.0068
>>
```



Compensação Derivativa Ideal (PD)

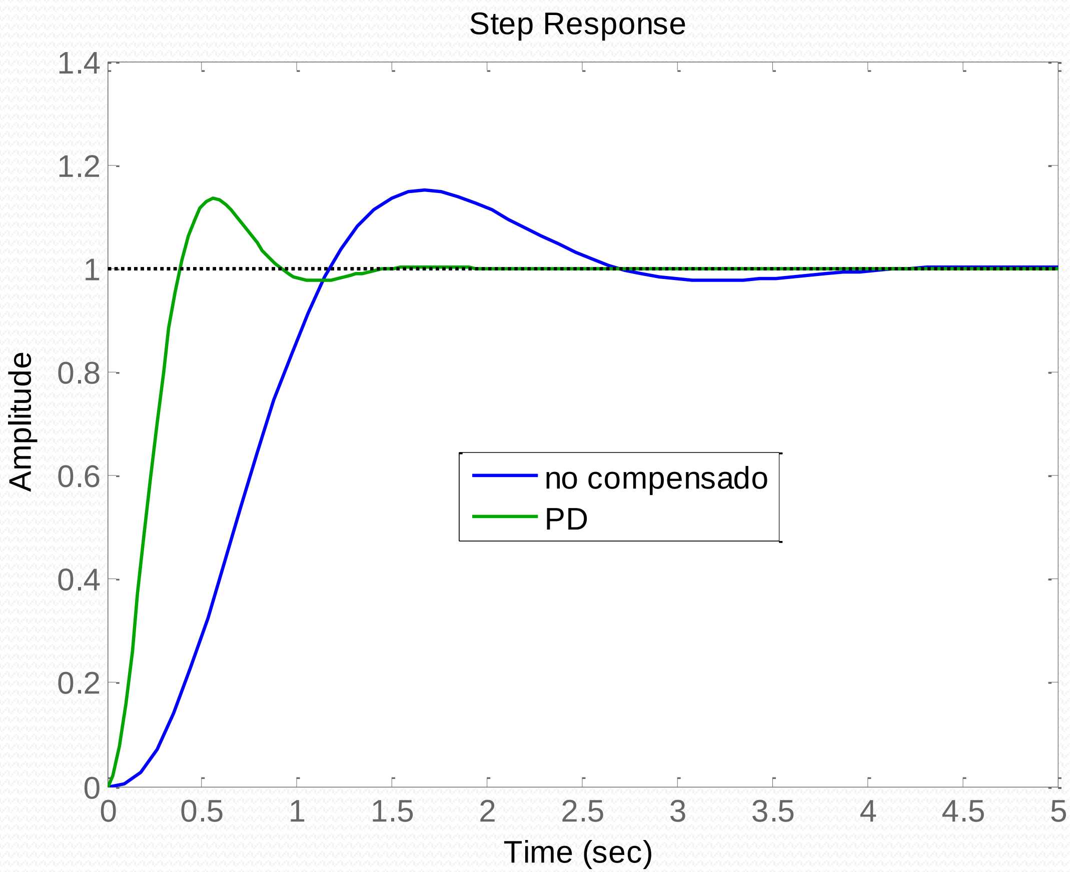
- Outro exemplo:



  - Requisitos: $\%OS < 16\%$, $3 \times \downarrow t_s$
  - *Solução:*
  - *Comparando respostas...*

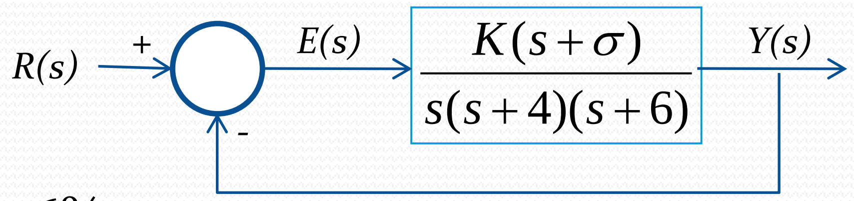
K=42

```
>> tf1=feedback(43.35*g,1);
>> tf2=feedback(42*cg,1);
>> figure(4);step(tf1,tf2)
>> legend('no compensado','PD')
>>
```



Compensação Derivativa Ideal (PD)

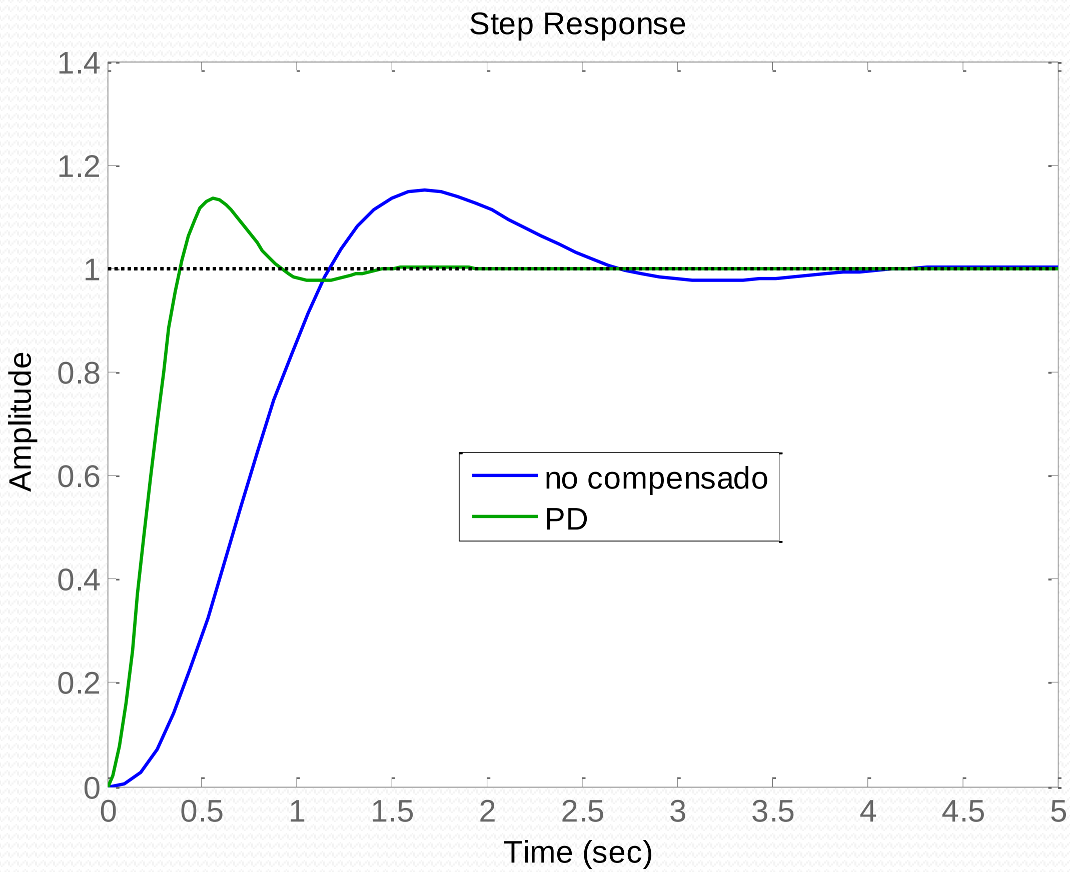
- Outro exemplo:



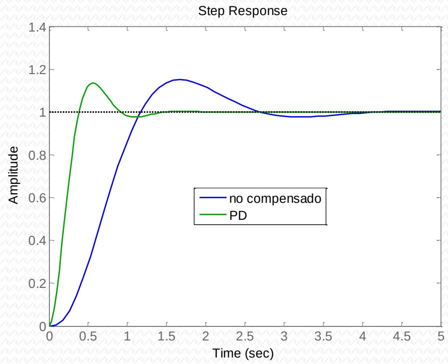
  - Requisitos: $\%OS < 16\%$, $3 \times \downarrow t_s$
  - *Solução:*
  - *Comparando respostas...*

K=42

```
>> tf1=feedback(43.35*g,1);
>> tf2=feedback(42*cg,1);
>> figure(5);ltiview(tf1,tf2)
>>
```



Compensação Derivativa Ideal (PD)



- *Ideia original:*
  - *Melhorar (acelerar) a resposta transitória*

- *Realização mediante Controlador derivativo (PD):*
  - *Desvantagens:*
    1. *Requer circuito ativo para realizar a diferenciação;*
    2. *Diferenciação pode gerar maus resultados no caso de processos ruidosos*

  - Por exemplo, suponha que temos o seguinte sinal:

$$y(t) = \text{sen}(t) + \underbrace{a_n \cdot \text{sen}(wt)}_{ruído}$$

  - donde:
    - sin(t) = sinal original de frequência = 1 rad/s y amplitude = 1;
    - a_n = amplitude do ruído, de frequência = 100 rad/s.

Compensação Derivativa Ideal (PD)

2. *Diferenciação pode gerar maus resultados no caso de processos ruidosos*

- Por exemplo, suponha que temos o seguinte sinal:

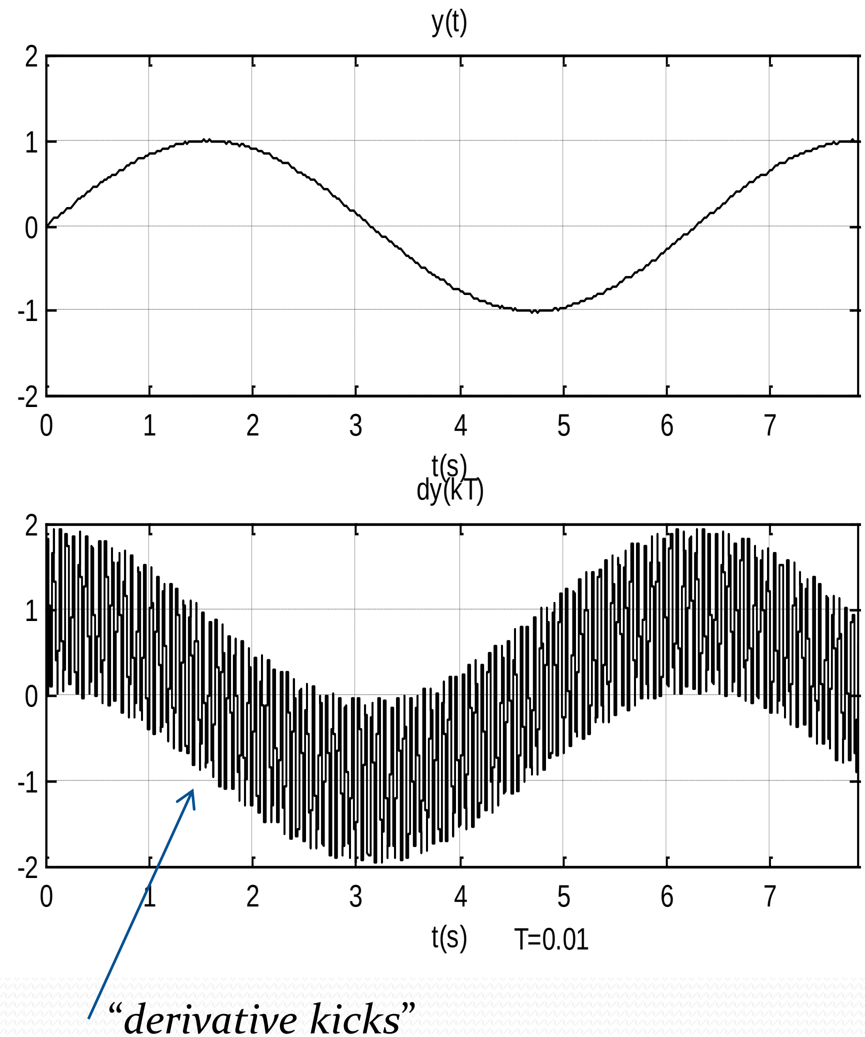
$$y(t) = \text{sen}(t) + \underbrace{a_n \cdot \text{sen}(wt)}_{ruído}$$

- donde:
  - sin(t) = sinal original de frequência = 1 rad/s y amplitude = 1;
  - a_n = amplitude do ruído, de frequência = 100 rad/s.

- Se aplicamos a derivada sobre o sinal anterior, mesmo que a amplitude do ruído corresponda a somente 1% da amplitude do sinal original (a_n = 0,01), teremos como resposta um sinal como mostrado na parte de baixo da figura ao lado.

- Perceba que a derivada (continua) deste sinal nos conduz a:

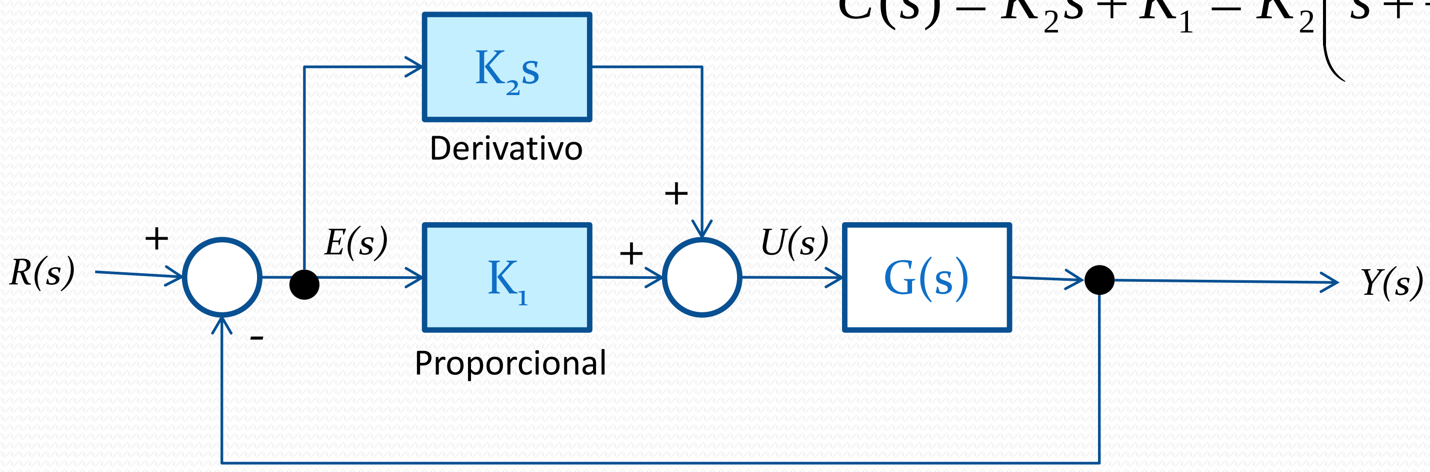
$$\frac{dy(t)}{dt} = \cos(t) + a_n \cdot w \cdot \cos(wt)$$



Compensação Derivativa Ideal (PD)

- *Ideia original:*
  - *Melhorar (acelerar) a resposta transitória*
- *Realização mediante Controlador derivativo (PD):*
  - *Desvantagens:*
    1. *Requer circuito ativo para realizar a diferenciação;*
    2. *Diferenciação pode gerar maus resultados no caso de processos ruidosos*

$$C(s) = K_2 s + K_1 = K_2\left(s + \frac{K_1}{K_2}\right)$$



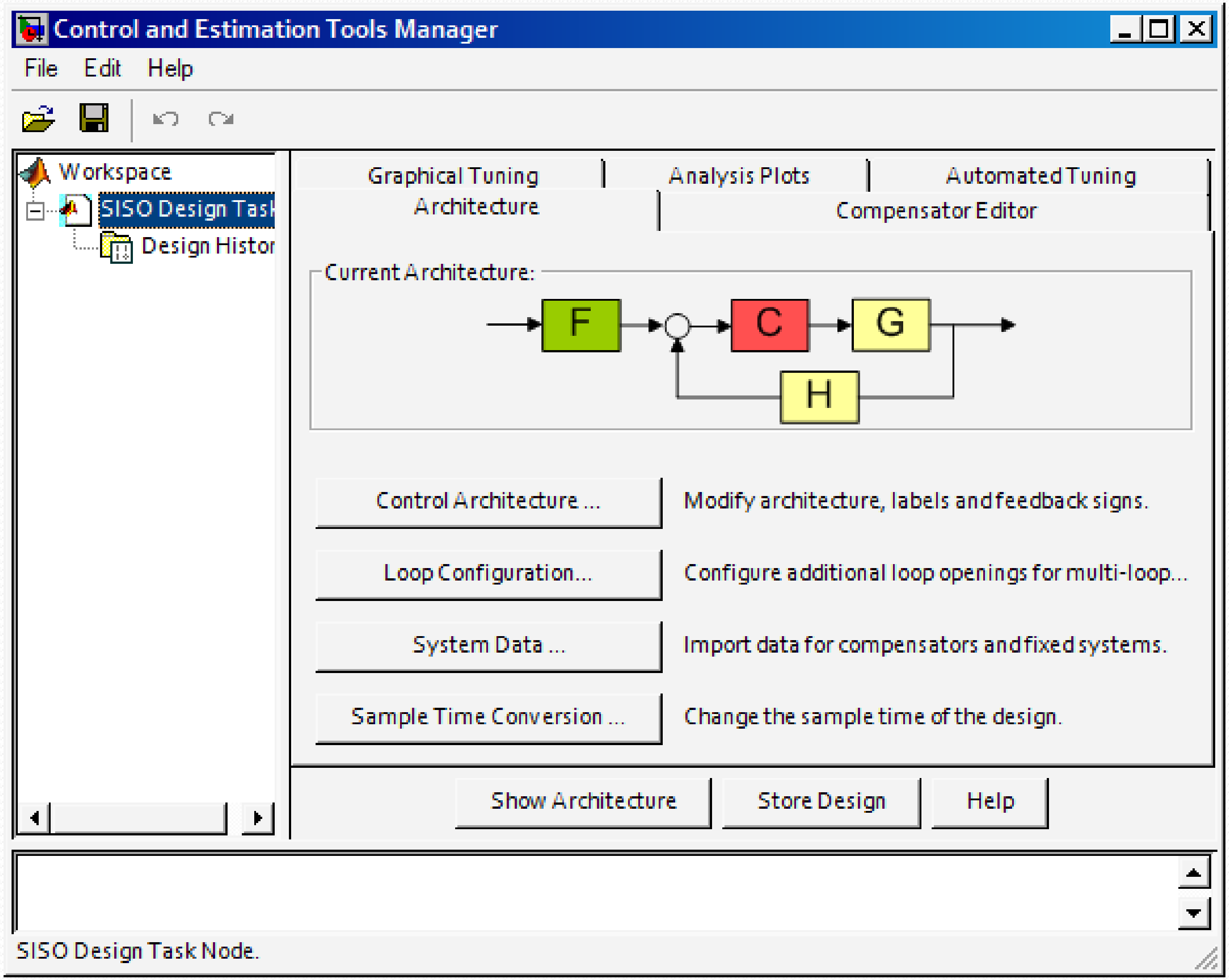
SISOTOOL SISO Design Tool.

```
SISOTOOL opens the SISO Design Tool. This Graphical User
    Interface lets you design single-input/single-output
    (SISO) compensators by graphically interacting with the
    root locus, Bode, and Nichols plots of  the open-loop
    system.  To import the plant data into the SISO Tool,
    select the Import item from the File menu. By default,
    the control system configuration is

             r -->[ F ]-->O--->[ C ]--->[ G ]----+---> y
                        - |                      |
                          +-------[ H ]----------+

    where C and F are tunable compensators.

SISOTOOL(G) specifies the plant model G to be used in the
    SISO Tool.  Here G is any linear model created with TF,
    ZPK, or SS.

SISOTOOL(G,C) and SISOTOOL(G,C,H,F) further specify values
    for the feedback compensator C, sensor H, and prefilter
    F.  By default, C, H, and F are all unit gains.

SISOTOOL(VIEWS) or SISOTOOL(VIEWS,G,...) specifies the
    initial set of views for graphically editing C and F.
    You can set VIEWS to any of the following strings or
    combination of strings:
        'rlocus'      Root locus plot
        'bode'        Bode diagram of the open-loop response
        'nichols'     Nichols plot of the open-loop response
        'filter'      Bode diagram of the prefilter F

For example
>> sisotool({'nichols','bode'})

Opens a SISO Design Tool showing the Nichols
    plot and Bode diagrams for the open loop
    CGH.

SISOTOOL(INITDATA) initializes the SISO
    Design Tool with more general control
    system configurations.  Use SISOINIT to
    build the initialization data structure
    INITDATA.

SISOTOOL(SESSIONDATA) opens the SISO Design
    Tool with a previously saved session where
    SESSIONDATA is the MAT file for the saved
    session.

See also sisoinit, ltiview, rlocus, bode,
    nichols.

Reference page in Help browser
       doc sisotool

>>
```

>> sisotool(.)

>> sisotool(g,1)

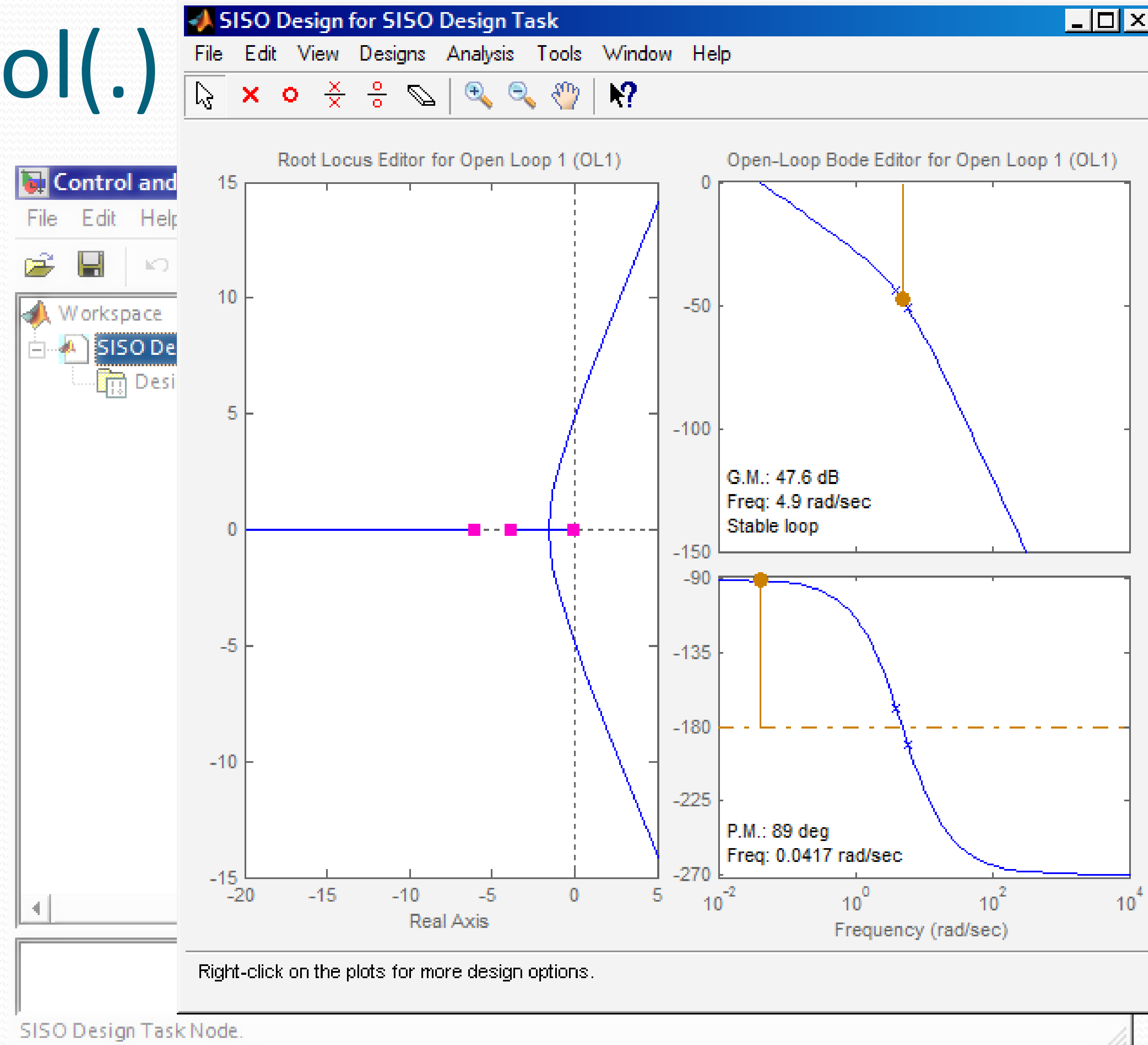
>>



>> sisotool(.)

>> sisotool(g,1)

>>

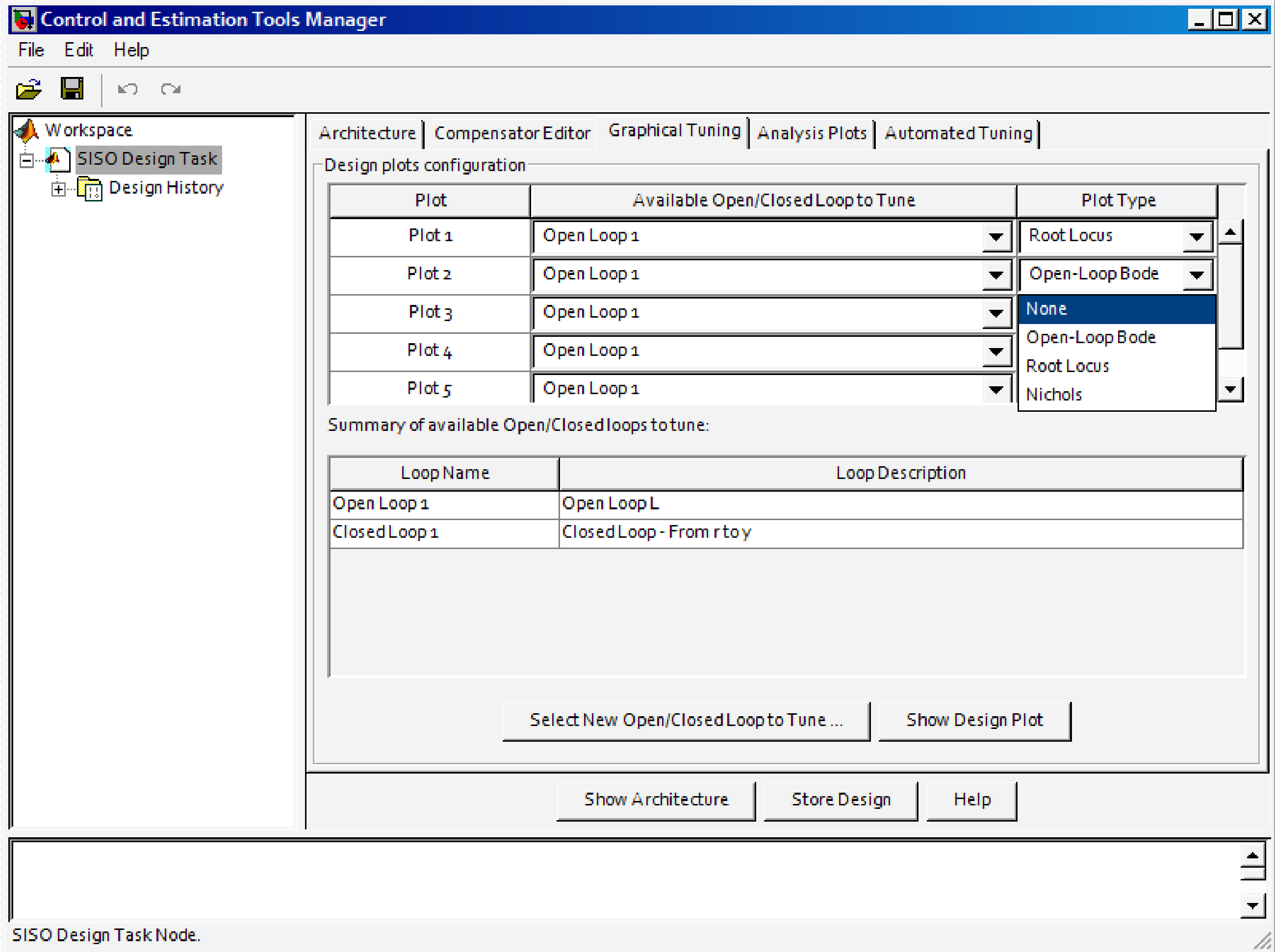


>> sisotool(.)

>> sisotool(g,1)

Editando visualização:

1) Janela “Control and Estimation Tools Manager”,
2) Aba “Graphical Tuning”,
3) Plot 2, Open Loop 1, Selecionar de “Open-Loop Bode” para “None”



>> sisotool(.)

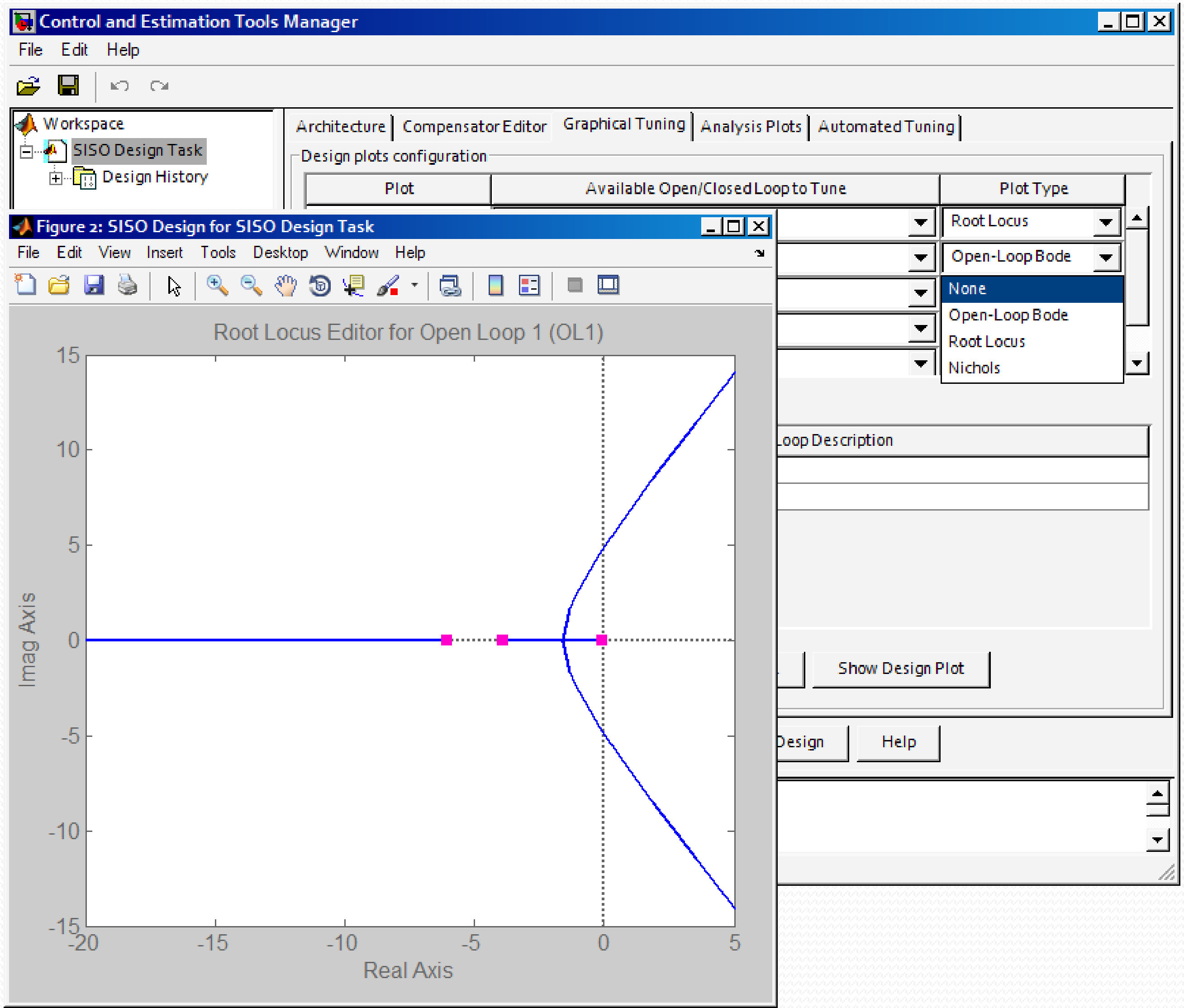
>> sisotool(g,1)

Editando visualização:

1) Janela "Control and Estimation Tools Manager",

2) Aba "Graphical Tuning",

3) Plot 2, Open Loop 1, Selecionar de "Open-Loop Bode" para "None"



>> sisotool(.)

>> sisotool(g,1)

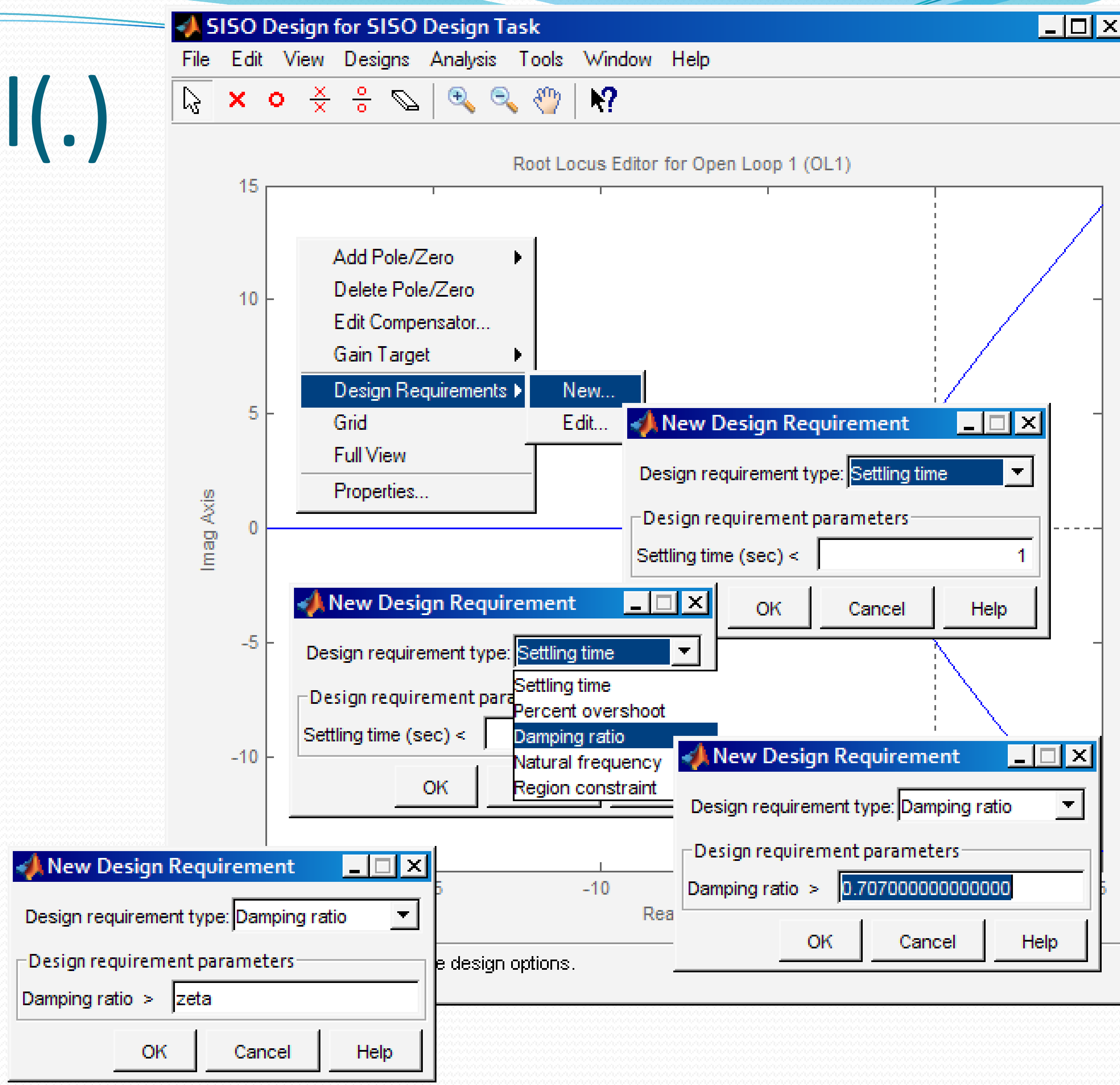
Editando visualização:

4) Ventana “Figure X: SISO…”,

5) Pressionar botão direito do mouse por sobre a janela gráfica,

6) Selecionar “Design Requirements”, New,

7) Selecionar “Damping Ratio” e alterar valor



>> sisotool(.)

>> sisotool(g,1)

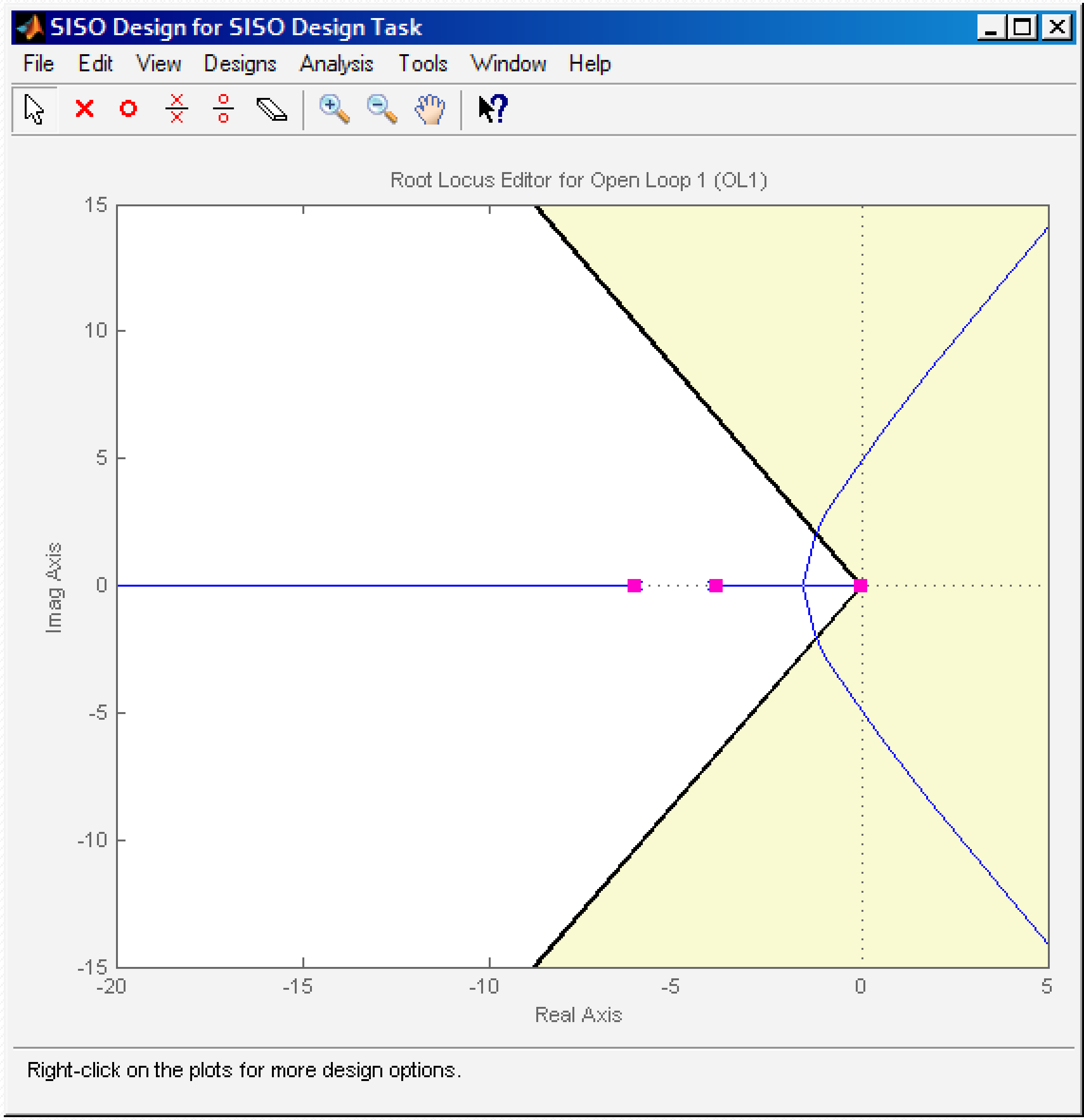
Editando visualização:

4) Ventana "Figure X: SISO...",

5) Pressionar botão direito do mouse por sobre a janela gráfica,

6) Selecionar "Design Requirements", New,

7) Selecionar "Damping Ratio" e alterar valor



>> sisotool(.)

>> sisotool(g,1)

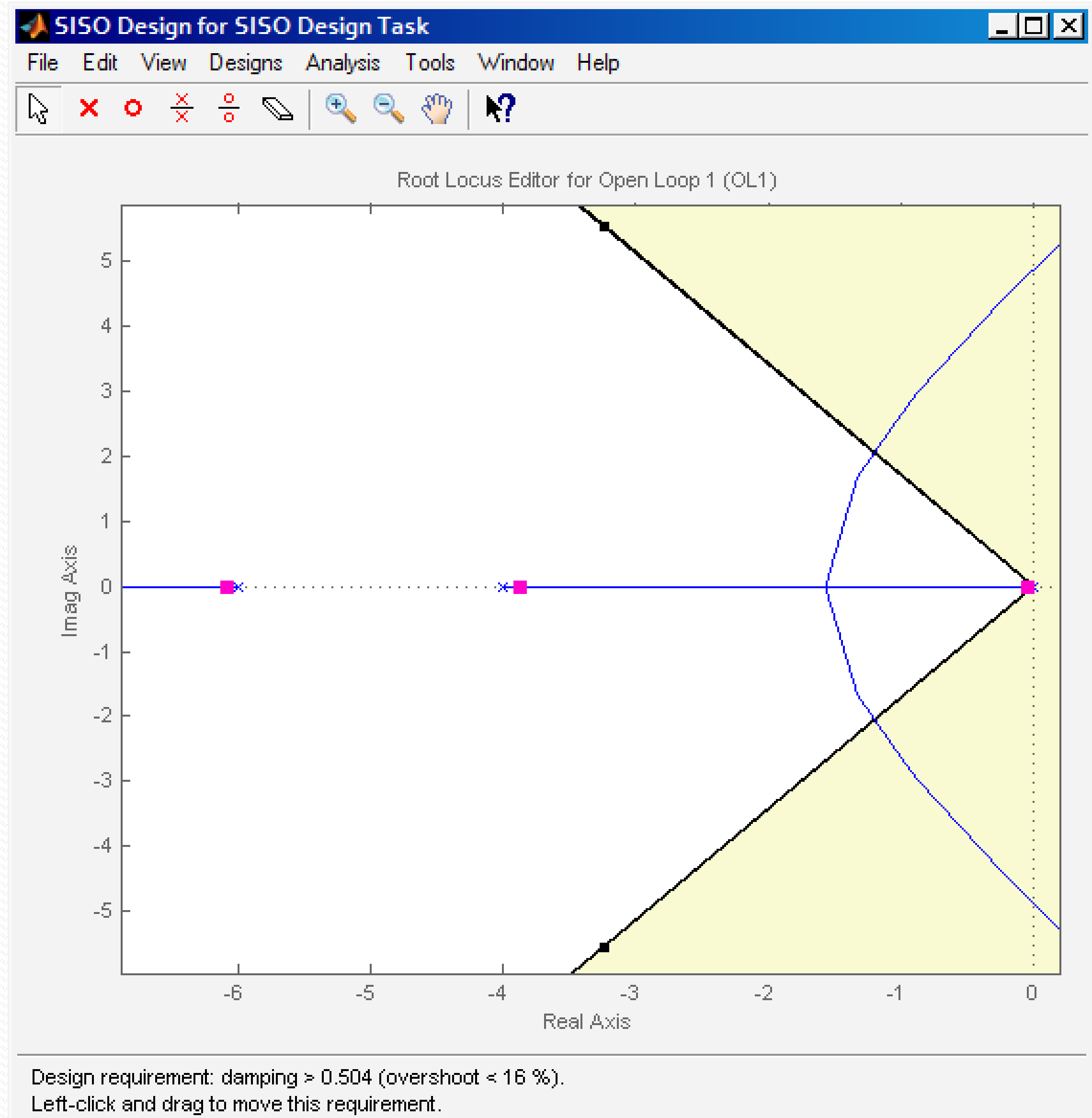
Editando visualização:

4) Ventana “Figure X: SISO...”,

5) Pressionar botão direito do mouse por sobre a janela gráfica,

6) Selecionar “Design Requirements”, New,

7) Selecionar “Damping Ratio” e alterar valor



>> sisotool(.)

>> sisotool(g,1)

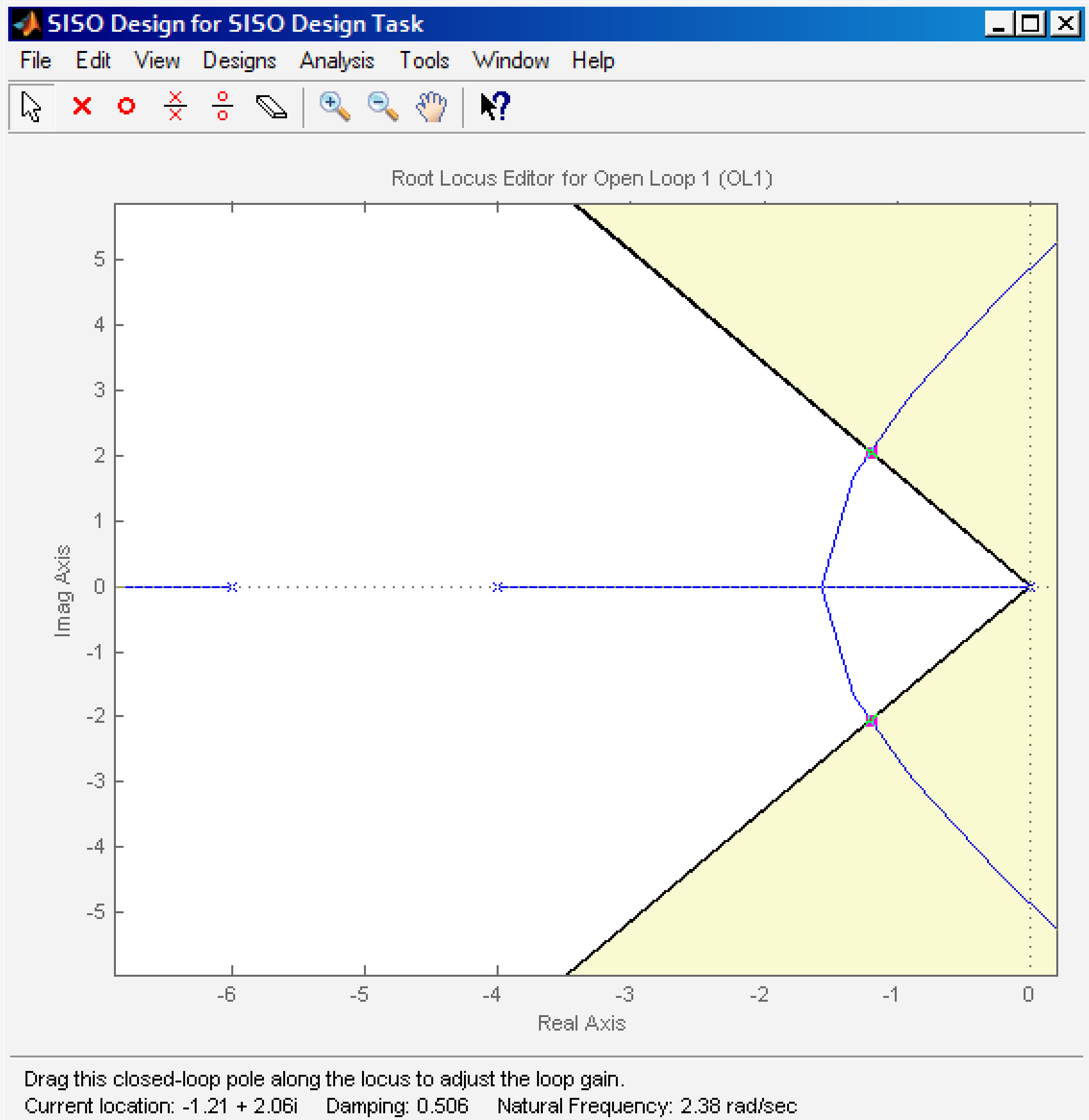
Editando visualização:

4) Ventana “Figure X: SISO…”,

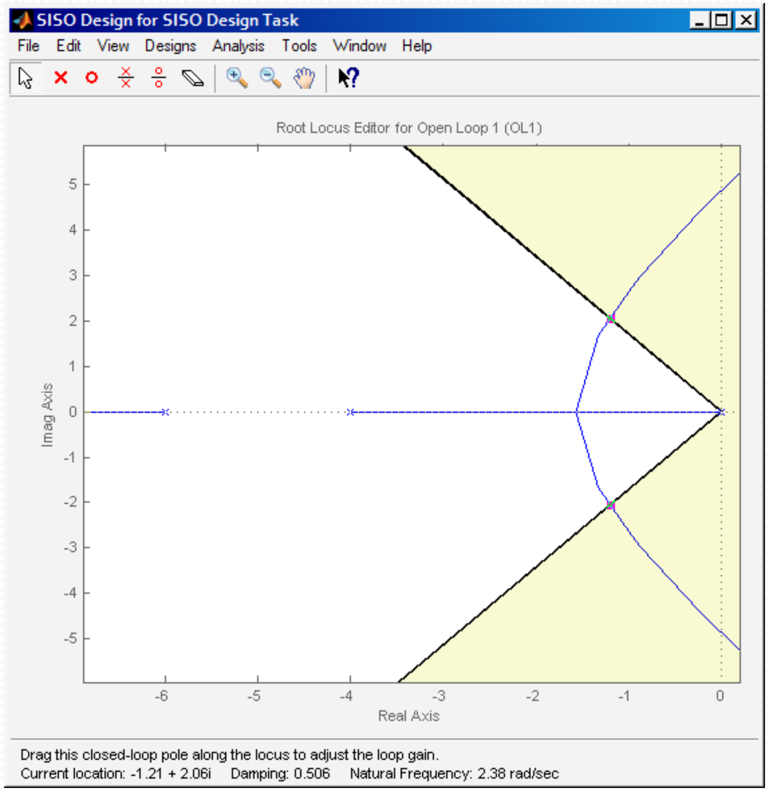
5) Pressionar botão direito do mouse por sobre a janela gráfica,

6) Selecionar “Design Requirements”, New,

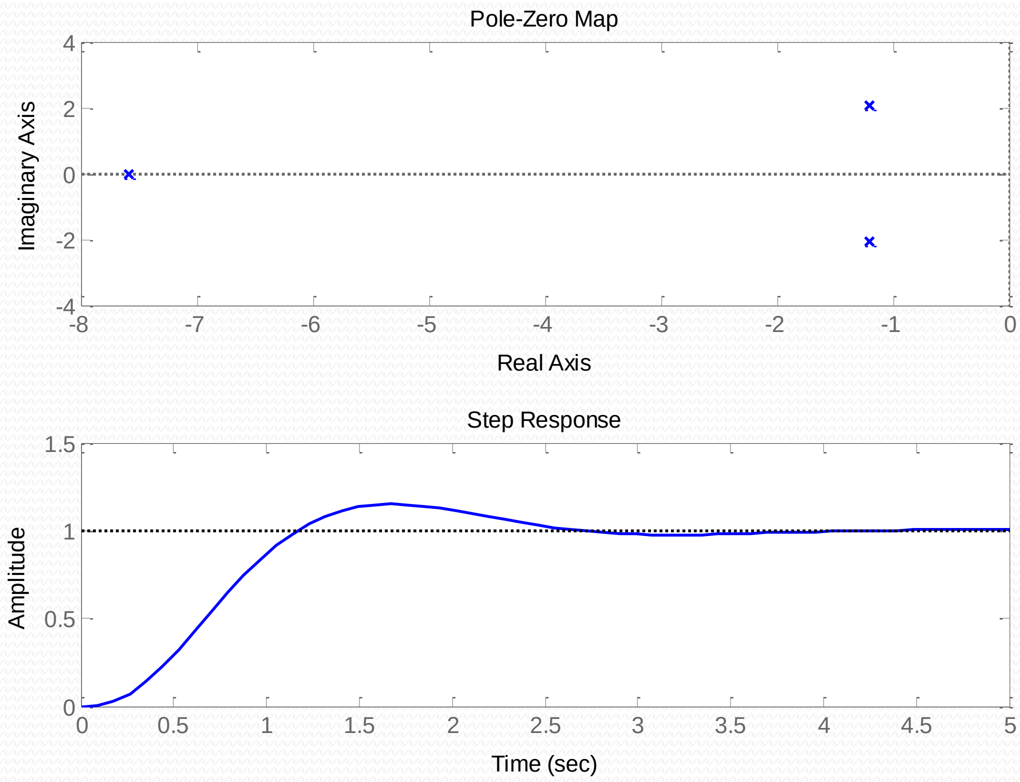
7) Selecionar “Damping Ratio” e alterar valor



>> sisotool(g)



SISO Design for SISO Design Task
File Edit View Designs Analysis Tools Window Help
Root Locus Editor for Open Loop 1 (OL1)
Imag Axis
Real Axis
Drag this closed-loop pole along the locus to adjust the loop gain.
Current location: -1.21 + 2.06i Damping: 0.506 Natural Frequency: 2.38 rad/sec



Pole-Zero Map
Imaginary Axis
Real Axis
Step Response
Amplitude
Time (sec)

>> sisotool(.)

>> sisotool(g,1)

>>