PMR3409 – CONTROLE II

EXPERIÊNCIA 5
PROJETO DE CONTROLADORES EM MATLAB

AUTOR: FELIPE LOPES DE SOUZA (ESTÁGIO PAE 2017)

1. INTRODUÇÃO

O objetivo dessa experiência é introduzir o aluno ao projeto de controladores usando a ferramenta computacional MATLAB. Diferentemente de outras experiências, em que havia uma introdução teórica seguida de uma série de experimentos, essa experiência foi concebida na forma de um tutorial passo a passo para familiarizar o aluno com as funções da toolbox de controle e da ferramenta SISOTOOL, com exercícios intermediários de fixação. Sendo assim, não se espera um relatório completo, mas sim um formulário de respostas.

ATENÇÃO: a ferramenta SISOTOOL será utilizada na próxima experiência, ou seja, espera-se que o aluno tenha alguma familiaridade com a mesma até lá. Recomenda-se que todas as instruções que sejam dadas nesse relatório de experiência sejam executadas, ao mesmo tempo, pelo aluno, para que ele ganhe experiência com a ferramenta.

2. DEFINIÇÃO DE FUNÇÕES TRANSFERÊNCIA NO MATLAB

A primeira etapa no projeto de um controlador usando MATLAB é a definição da planta, ou seja, a função transferência do sistema.

O método mais conhecido é definir a planta usando ferramentas como simulink, em que a mesma é desenhada na forma de diagramas de blocos e a simulação é feita no tempo, Figura 1

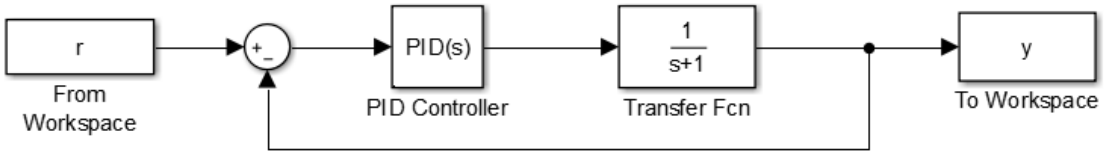


Figura 1. Exemplo de diagrama de blocos construídos usando simulink.

Essa estratégia, entretanto, só permite o projeto de controles através de simulação e tentativa e erro, desprezando todo conhecimento analítico e teórico sobre a planta. Para solucionar esse problema, existem ferramentas como a Control System Toolbox, que permitem definir as funções de transferência de forma analítica.

Supondo a função de transferência da Equação (1):

$$G = \frac{(s+1)}{s^2(s+2)(s+3)} = \frac{s+1}{s^4 + s^3 \cdot (2+3) + s^2 \cdot 2 \cdot 3} \quad (1)$$

Existem algumas formas para definir a função acima:

```
% forma 1:
% G =   numerador  =  az*S^z + ... + a1*S + a0
%       -----------   -------------------------
%      denominador    bp*S^p + ... + b1*S + b0
% numerador = [az,...,a1,a0];
% denominador = [bp,...,b1,b0];
% G = tf(numerador,denominador);
numerador = [1,1];
denominador = [1,2+3,2*3,0];
G1 = tf(numerador,denominador);

G1 =
        s + 1
  -----------------
  s^3 + 5 s^2 + 6 s
```

```
% forma 2:
% G =    zeros  =    (s-z1)*(s-z2)*...*(s-zn)
%      K*-----     K*-----------------------
%        polos       (s-p1)*(s-p2)*...*(s-pm)
% zeros = [z1,...,zn];
% polos = [p1,...,pm];
% G = zpk(zeros,polos,K);
zeros = -1;
polos = [0,0,-2,-3];
K = 1;
G2 = zpk(zeros,polos,K);

G2 =
        (s+1)
  ---------------
  s^2 (s+2) (s+3)


% forma 3
s = zpk('s');
G3 = (s + 1)/(s^2*(s+2)*(s+3));

G3 =
        (s+1)
  ---------------
  s^2 (s+2) (s+3)
```

Todas as formas resultam no mesmo sistema, e uma pode ser convertida em outra de forma simples:

```
zpk(G1) =
       (s+1)
  -------------
  s (s+3) (s+2)

tf(G2) =
         s + 1
  -------------------
  s^4 + 5 s^3 + 6 s^2
```

Funções mais complexas, como atraso de transporte, Equação (2), também podem ser escritas de forma simples:

$$e^{-Ts} \cdot G(s) \tag{2}$$

```
% atraso de 7s
s = zpk('s');
Atraso = exp(-7*s);
G4 = G3*Atraso;
G4 =
                    (s+1)
  exp(-7*s) * ---------------
              s^2 (s+2) (s+3)
```

EXERCÍCIO 1

Escreva o código que gera os sistemas abaixo em MATLAB. Cada sistema pode ser gerado de apenas uma forma (ou zpk, ou tf, ou zpk('s')), mas todas as formas devem ser usadas durante o exercício (uma em cada item):

a) $G(s) = \frac{10s+5}{s^2+2s+1}$

b) $H(s) = 7 \cdot \frac{1}{s} \frac{(s-2.5)(s+1)}{(s-3)(s-4)}$

c) $F(s) = G(s) \cdot H(s) \cdot e^{-1.3s}$

3. ANÁLISE DE SISTEMAS LINEARES

Com as funções de transferência definidas, uma série de análises podem ser feitas, como resposta ao degrau, resposta ao impulso e análises em frequência.

```
% resposta ao degrau
s = zpk('s');
G = 1/(2*s+1);
step(G);
```

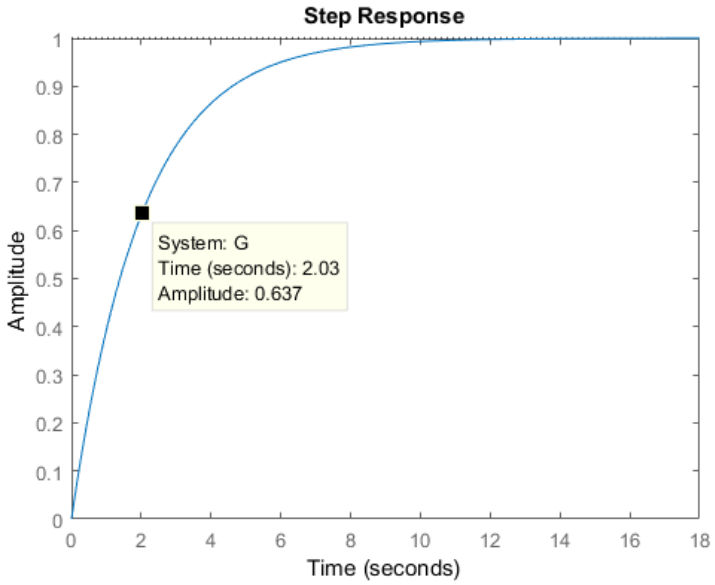


Figura 2. Resposta a um degrau unitário. Valor destacado usando a opção Data Cursor da toolbar da figura.

```
% resposta ao impulso
impulse(G);
```

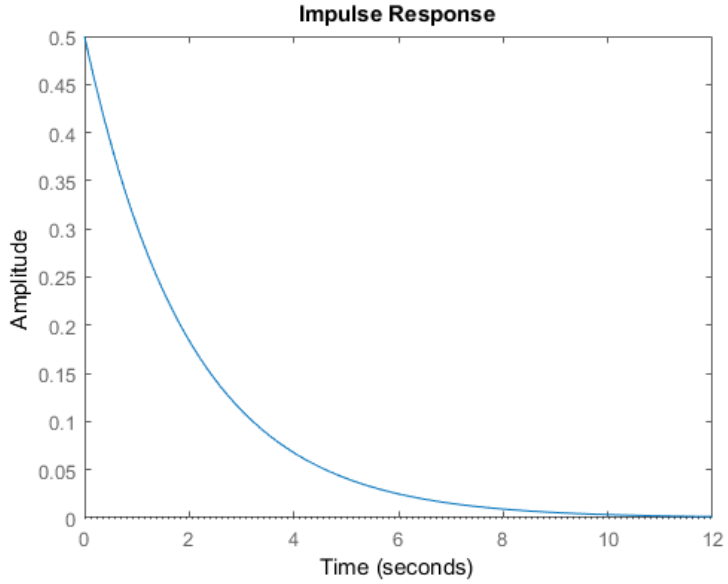


Figura 3. Resposta ao impulso.

```
% diagrama de bode
bode(G);
```

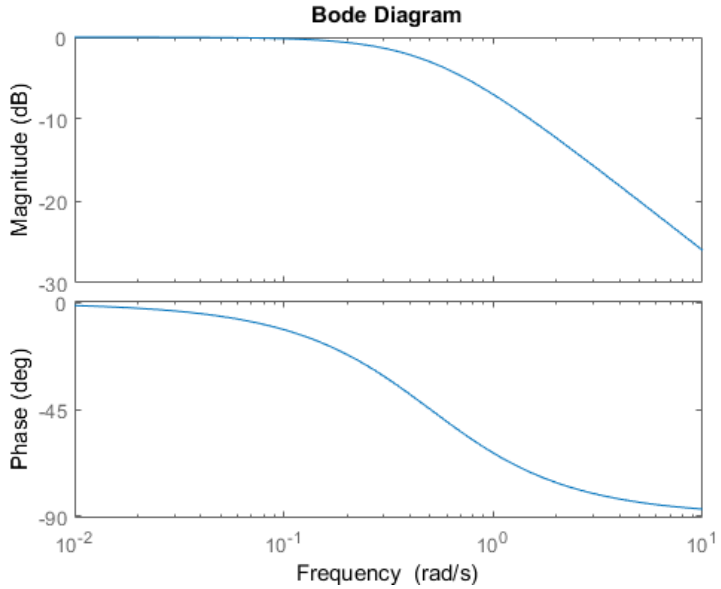


Figura 4. Diagrama de Bode.

```
% diagrama de Nyquist
nyquist(G);
```

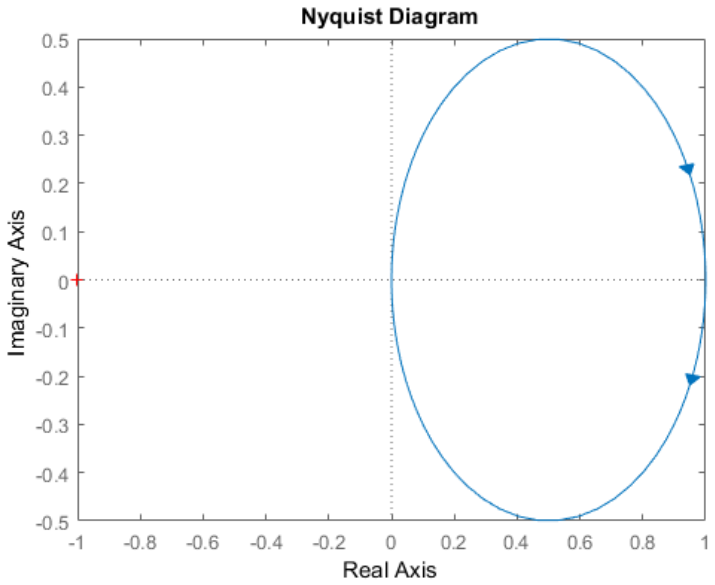


Figura 5. Diagrama de Nyquist.

As análises também podem ser feitas para o sistema com feedback:

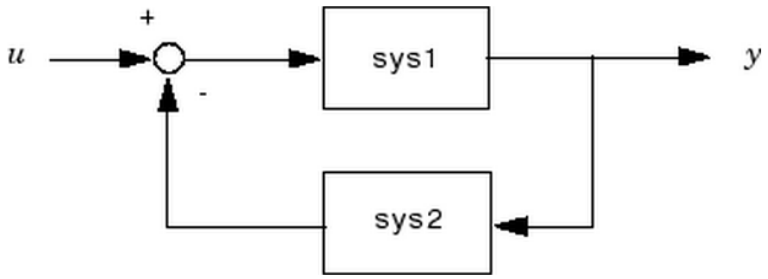


Figura 6. Sistema com feedback considerado na função feedback.

```
sys = feedback(sys1,sys2)
```

Por exemplo:

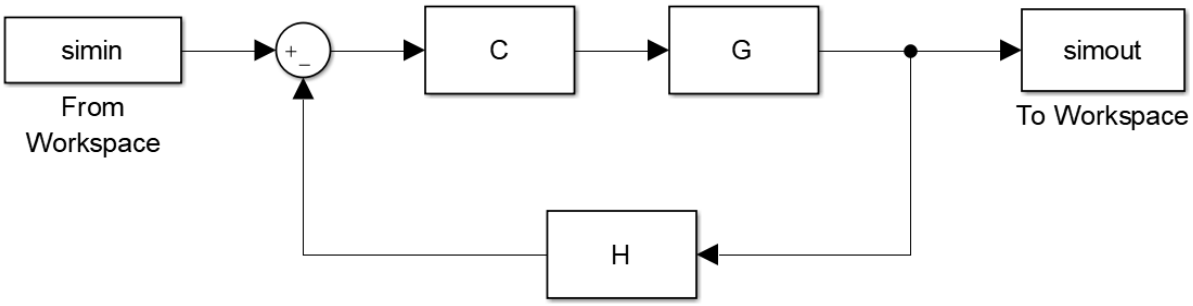


Figura 7. Sistema complexo que pode ser representado de forma simples no MATLAB.

```
% feedback
sys = feedback(C*G,H);
```

EXERCÍCIO 2

Dado o Sistema:

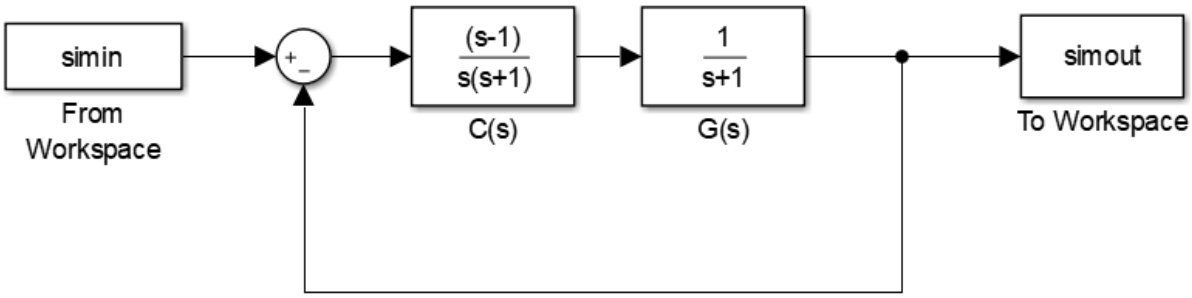


Figura 8. Sistema planta-controlador.

Descreva o que ocorre com uma entrada senoidal com frequência 0.15Hz e amplitude igual a 1 quando a mesma passa pelo sistema, ou seja, determine $y(t) = A_y \cdot \sin(\omega t + \phi)$ sendo que $r(t) = \sin(\omega t)$. Utilize, para tanto, o diagrama de bode do sistema completo (o feedback deve ser incluído, ou seja, o objetivo é verificar o diagrama de Bode da função transferência que relaciona a entrada e a saída do sistema), lembrando que:

$$f = 0.15Hz \rightarrow \omega = 2 \cdot \pi \cdot f \sim 0.94\, rad/s$$

$$A_{db}dB = 20 \cdot \log_{10} A_y$$

Ou então, usando o MATLAB:

```
Adb = mag2db(Ay);

Ay = db2mag(Adb);
```

4. PROJETO DE CONTROLADORES USANDO SISOTOOL

Com as ferramentas apresentadas até então, é possível analisar quais serão as características de uma planta $G(s)$ sob a influência de feedback e de um controle $C(s)$. O projeto do controlador $C(s)$ para se obter características desejadas, entretanto, ainda depende de técnicas ensinadas nas aulas teóricas.

A ferramenta sisotool existe justamente para auxiliar tal projeto, unindo análises teóricas, como as vistas em sala de aula, com visualizações gráficas atualizadas em tempo real e otimização, para atender requisitos complexos.

Antes de iniciar a ferramenta, defina a seguinte planta no MATLAB:

$$G_P = \frac{10}{s(0.25s + 1)} \tag{3}$$

E inicie a ferramenta digitando o comando sisotool, e apertando enter. A seguinte interface gráfica deve aparecer, Figura 9:

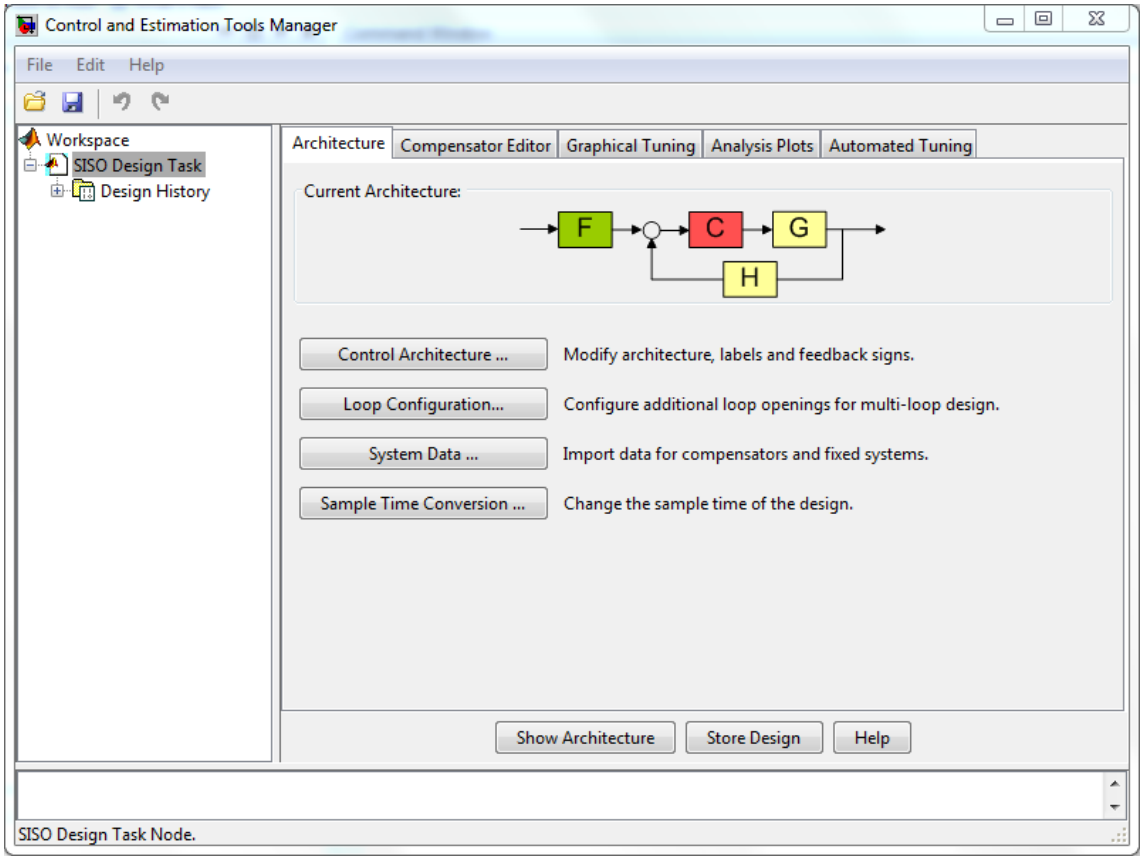


Figura 9. Tela inicial da ferramenta sisotool.

A planta do sistema pode ser carregada na interface na opção “System Data...”, Figura 10:

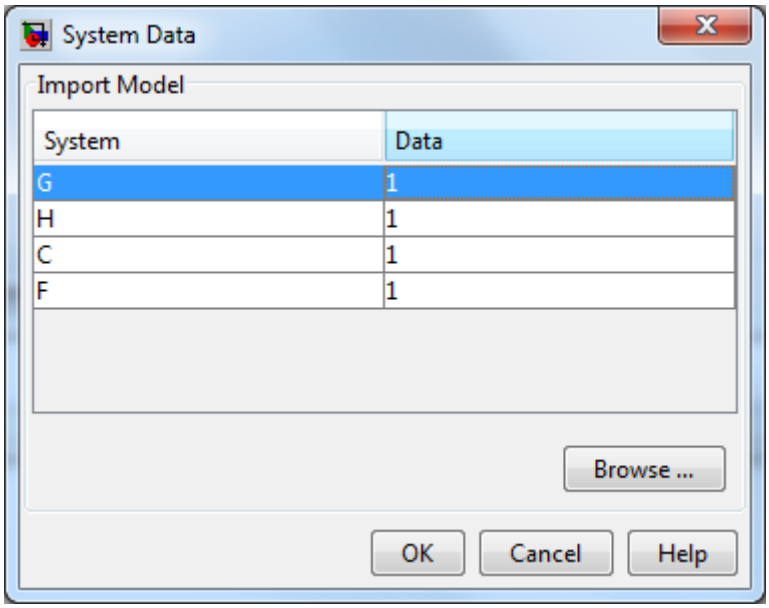


Figura 10. Interface da opção "System Data".

Em seguida, a mesma pode ser carregada no workspace abrindo-se “Browse...”, selecionando-se a variável G_P (ou qualquer outro nome que você tenha dado para a função de transferência, criada antes de abrir a interface) e clicando em “Import” e, em seguida “Close”, Figura 11.

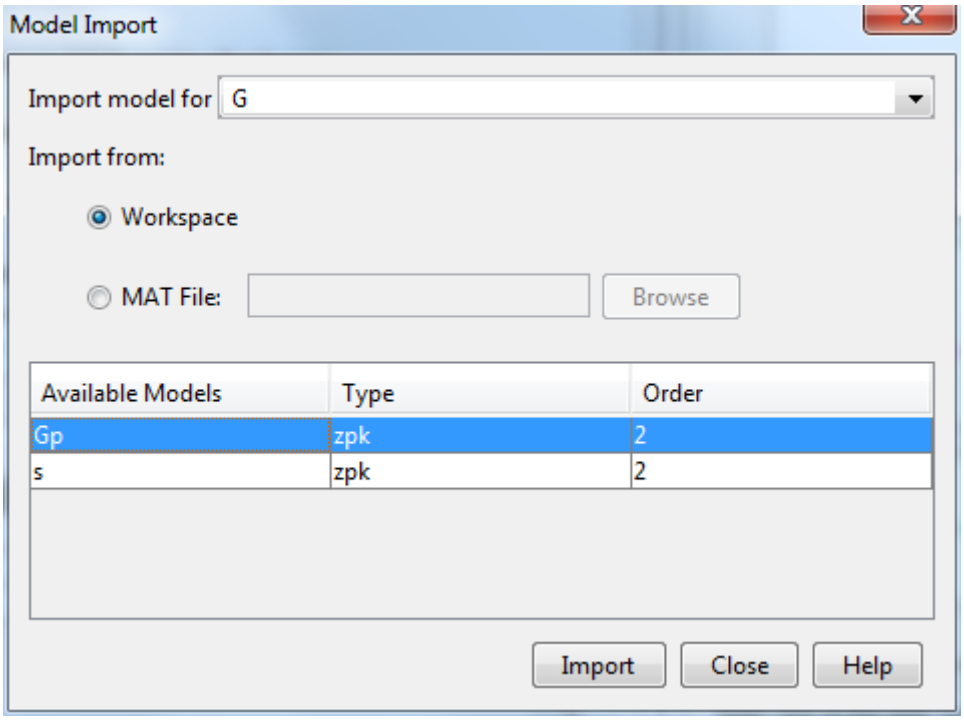


Figura 11. Interface de importação das plantas no workspace.

Assim que o OK é pressionado, os diagramas de lugar das raízes e de bode já são atualizados automaticamente, Figura 12.

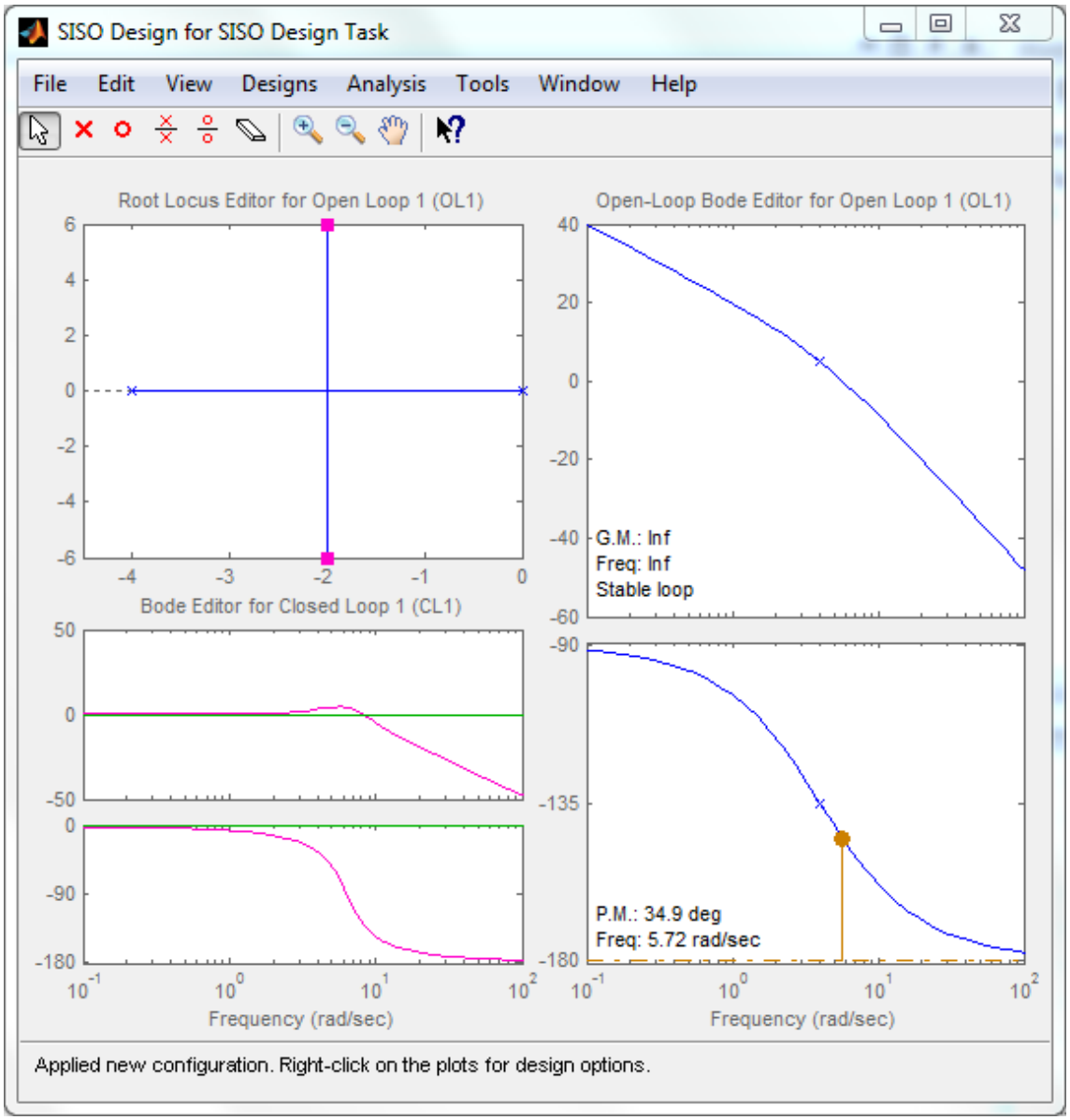


Figura 12. Lugar das raízes e diagrama de Bode atualizados automaticamente.

Os gráficos utilizados para tuning gráfico (ou seja, gráficos em que é possível modificar a função de controle $C(s)$ graficamente para atingir os requisitos, como diagrama de lugar das raízes e diagrama de Bode), podem ser abertos, fechados e modificados na aba “Graphical Tuning”.

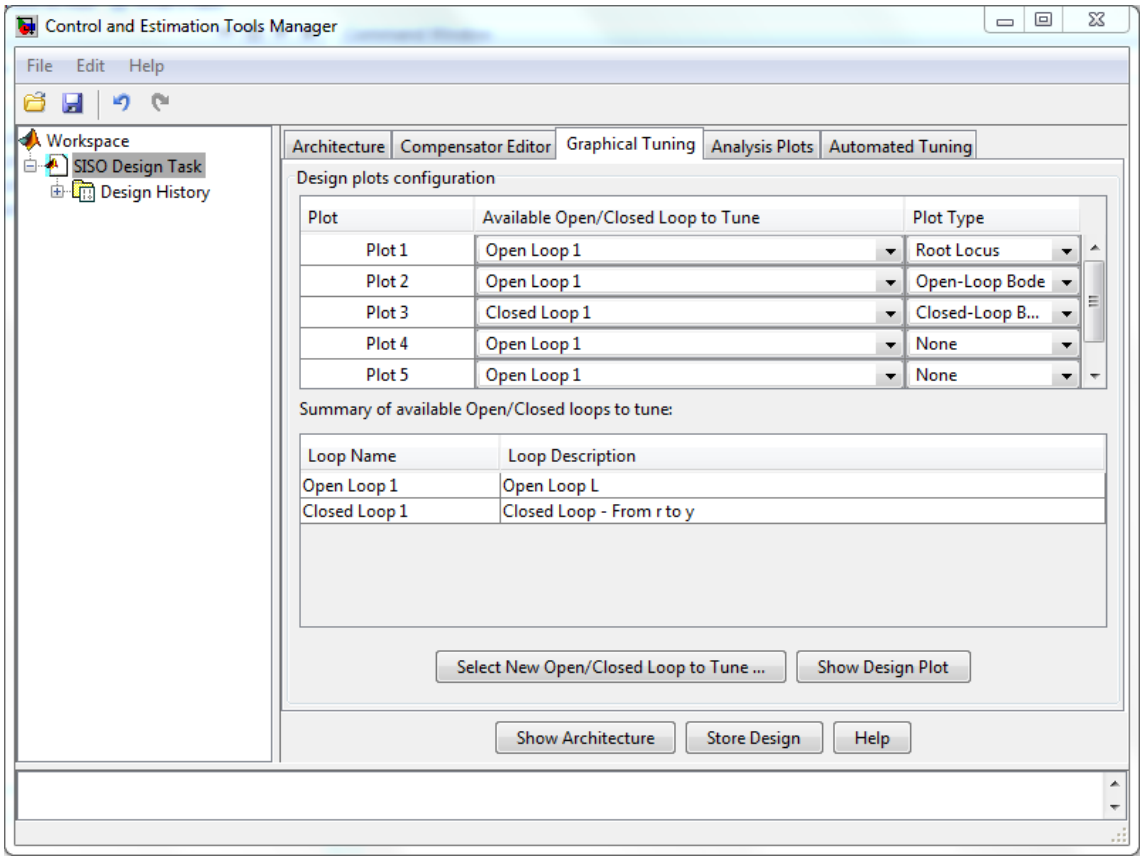


Figura 13. Aba Graphical Tuning, permite a plotagem de gráficos iterativos, como Lugar das Raízes, diagrama de Bode, etc.

4.1. Diagrama de Lugar das Raízes

O primeiro diagrama a ser estudado é o diagrama de lugar das raízes, que permite a visualização dos polos e zeros do sistema em azul, os polos e zeros do controlador em rosa, e os polos obtidos com o ajuste do ganho do controlador como quadrados rosas.

Para que o diagrama faça mais sentido, selecione a aba “Compensator Editor”, para que o ganho atual do controlador possa ser visualizado:

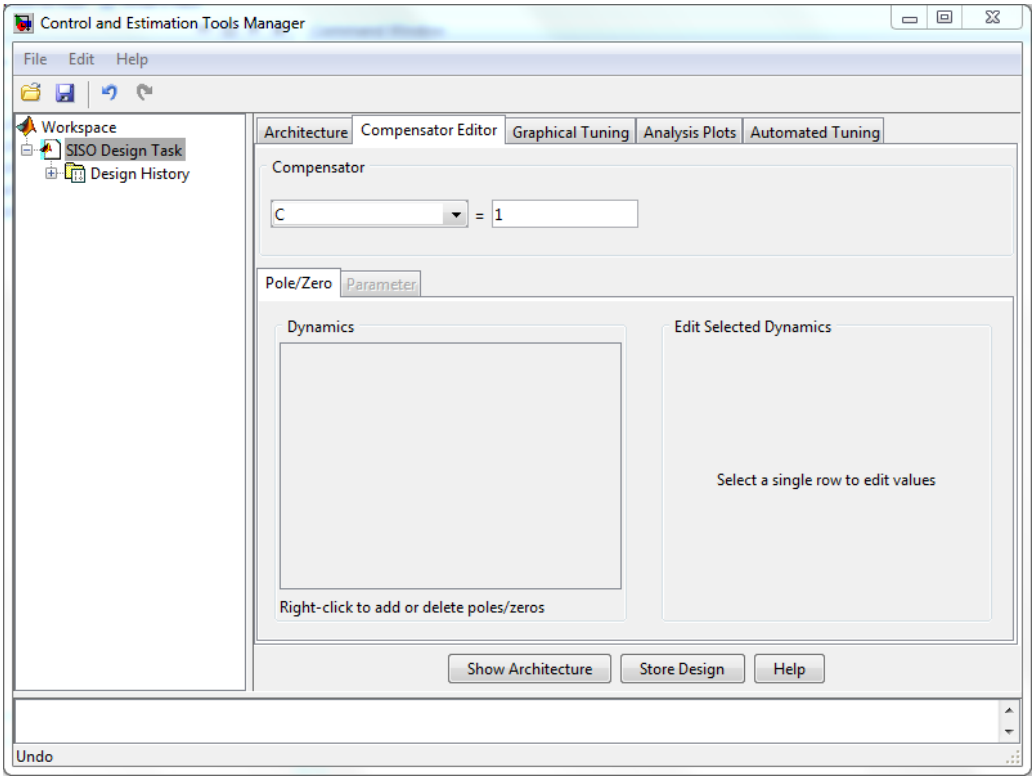


Figura 14. Aba Compensator Editor permite a visualização da dinâmica do controlador.

Em seguida, passe o mouse por cima dos quadrados rosas no gráfico de lugar das raízes, até que apareça uma mão manipuladora, e mova os quadrados pelas linhas azuis. Repare que o ganho do controlador muda automaticamente, de acordo com a posição dos quadrados rosas, ou seja, efetivamente um diagrama de lugar das raízes iterativo, em que dada a posição desejada dos polos do sistema (quadrados rosas) calcula-se o ganho correspondente.

As posições dos polos, o damping e a frequência natural podem ser visualizados no texto em baixo do gráfico quando os quadrados são movidos.

Clicando-se com o botão direito do mouse sobre o gráfico de Lugar das Raízes, abre-se o **menu de contexto** e as seguintes opções aparecem:

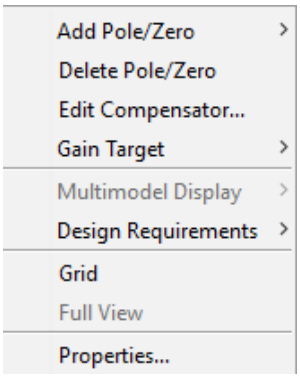


Figura 15. Menu de Contexto, utilizado para configurar os gráficos.

Inicialmente, é conveniente selecionar a opção Grid para melhorar a visualização. Na opção design requirements é possível adicionar uma série de requisitos de design, como sobressinal, tempo de assentamento, etc. Se dois requisitos forem impostos, 10% de sobressinal e tempo de assentamento de 4s, por exemplo, obtém-se as seguintes regiões proibidas (em amarelo):

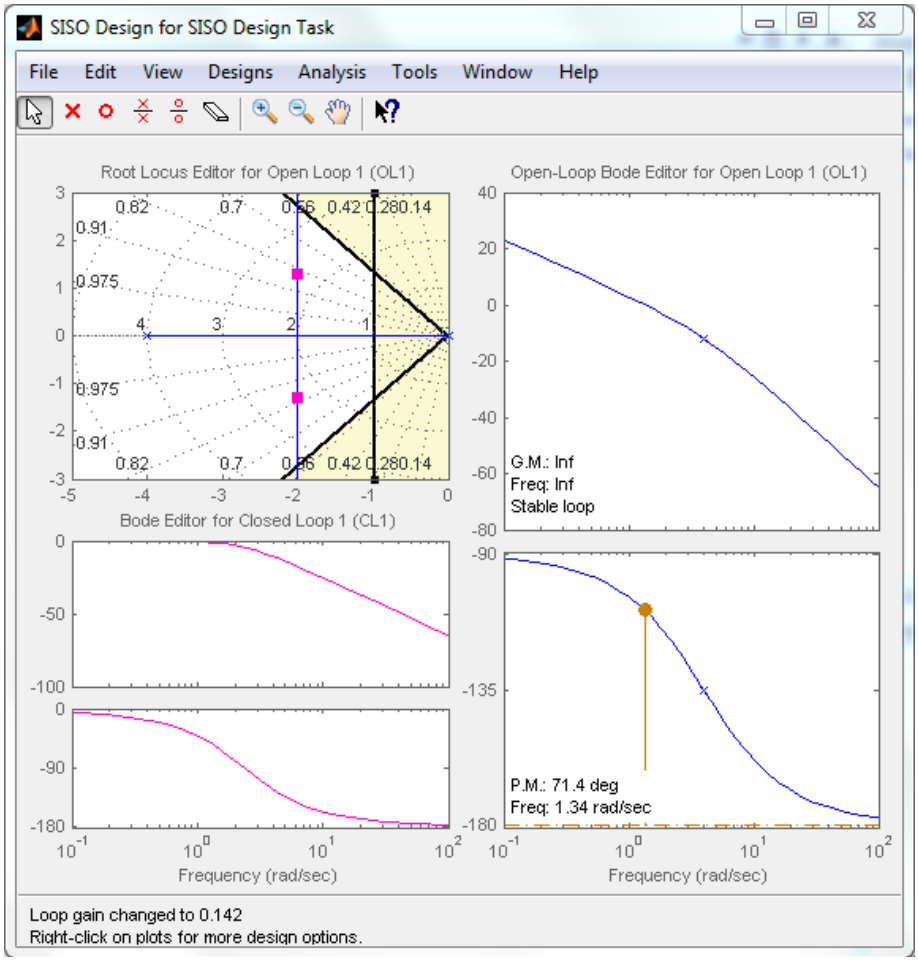


Figura 16. Requisitos de projeto adicionados no gráfico de Lugar das Raízes.

Os requisitos podem ser ajustados manipulando-se as retas pretas ou então editando os mesmos com o botão direito. Um controlador que satisfaz exatamente os requisitos pode, então, ser ajustado manualmente arrastando-se os polos representados nos quadrados rosas.

Por fim, ainda existem as opções de zoom e pan para auxiliar a visualização do gráfico, assim como atalhos para adicionar ou deletar polos e zeros do controlador (que também podem ser adicionados pelo menu de contexto).

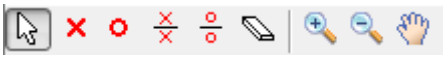


Figura 17. Toolbar de navegação.

Até então, só foi possível ajustar o ganho do controlador, de forma que apenas um controle do tipo P pôde ser ajustado. Para que seja possível ajustar um controle PD é possível adicionar um zero ao sistema de três formas:

1- Clicando no círculo vermelho na toolbar, e em seguida posicionando o mesmo no gráfico de lugar das raízes;
2- Selecionando Add Pole/Zero -> Real Zero no menu de contexto;
3- Ou então clicando com o botão direito do mouse dentro do painel "Dynamics", na aba Compensator Editor, e selecionando Add Pole/Zero -> Real Zero.

Repare que agora o valor da função controle foi atualizado, e que um círculo vermelho foi adicionado no diagrama de lugar das raízes.

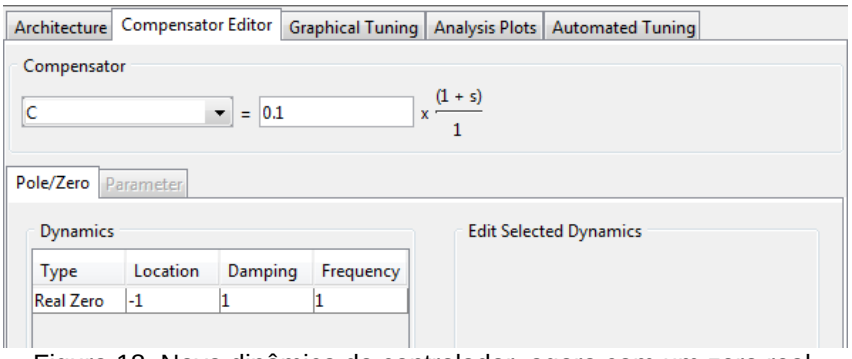


Figura 18. Nova dinâmica do controlador, agora com um zero real.

Com os ganhos proporcional e derivativo dados pela relação a seguir, Equação (4):

$$C(s) = D\left(\frac{P}{D} + s\right) \tag{4}$$

Ou então, para o exemplo:

$$D = 0.1 \tag{5}$$

$$P = D \cdot 1 = 0.1 \tag{6}$$

Todos os zeros e polos adicionados ao controle podem ser deletados pelo menu de contexto Delete Pole/Zero, ou pela toolbar clicando-se na borracha.

Retornando ao exemplo, o zero adicionado pode ser setado exatamente na aba Compensator Editor:

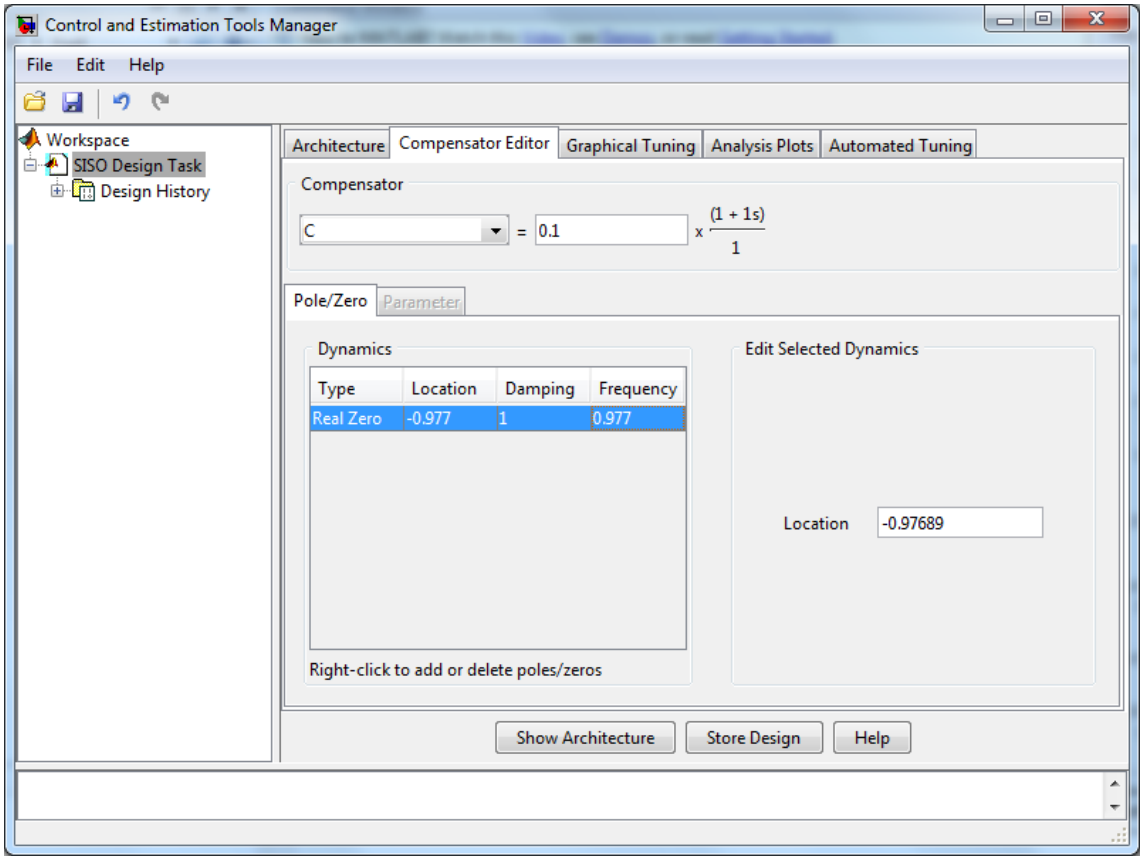


Figura 19. Clicando no zero é possível ajustar seu valor manualmente.

Além disso, ele pode ser movido livremente pelo gráfico de lugar das raízes, podendo ser posicionado antes, Figura 20, ou depois Figura 21, do polo original do sistema:

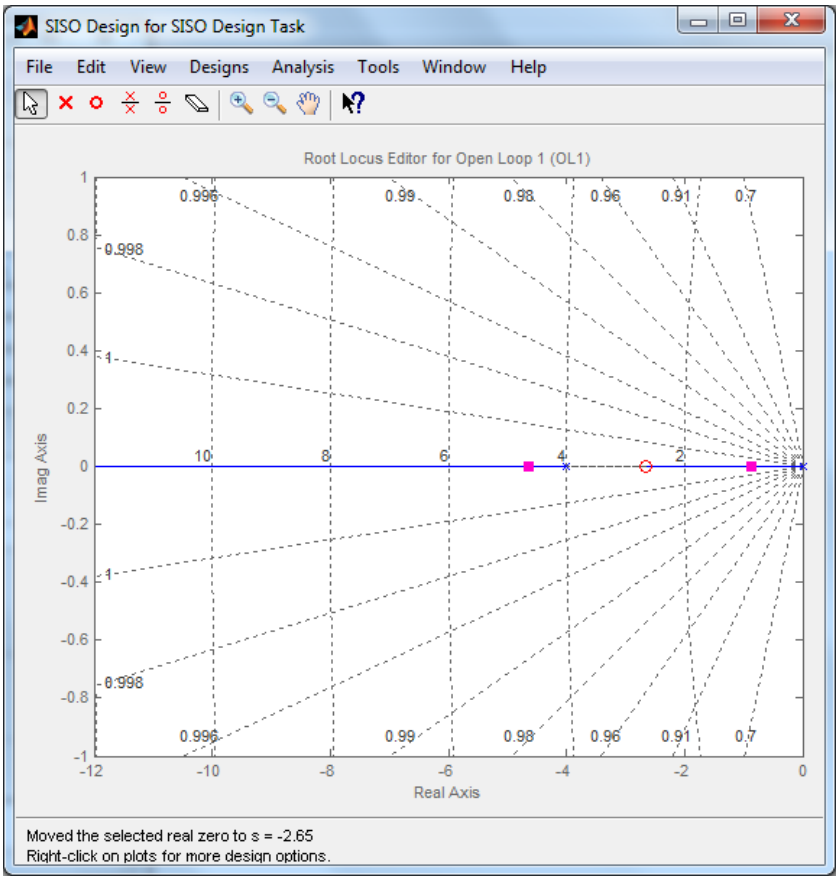


Figura 20. Diagrama quando o zero é posicionado após o polo do sistema.

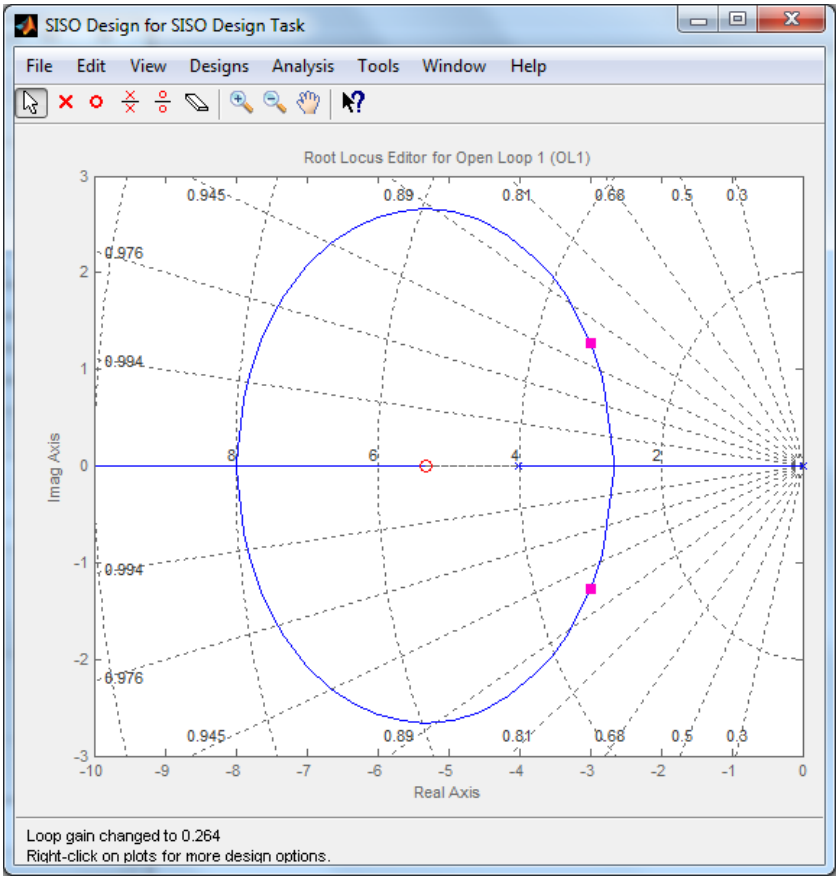


Figura 21. Diagrama quando o zero é posicionado antes do polo do sistema.

Repare que quando o zero é posicionado antes do polo, não há sobressinal, não importando o ganho (polos sempre reais negativos), porém o polo dominante (ou seja, o mais próximo de zero) sempre será mais lento que o polo do sistema (ou seja, o quadrado rosa nunca passa do x azul).

Quando o zero é posicionado após o polo, é possível que os polos estejam mais à esquerda que o x azul, ou seja, o sistema fica mais rápido, porém o sistema passa a ter sobressinal, já que tenho polos complexos conjugados).

Por fim, para que seja possível projetar um controle PID deve-se ter em mente que o controlador é na forma:

$$C(s) = P + Ds + \frac{I}{s} = \frac{Ds^2 + Ps + I}{s} \tag{7}$$

Ou seja, deve-se adicionar **dois zeros reais** e um **integrador** (é possível adicionar o integrador, ou seja, um polo exatamente em zero, pelo menu de contexto Add Pole/Zero -> Integrator).

EXERCÍCIO 3

Utilizando o diagrama de Lugar das Raízes, projete um controle do tipo PD tal que o sistema final tenha 5% de sobressinal e tempo de assentamento de 0.5s. Qual o valor das constantes P e D nesse caso? Inclua uma imagem do diagrama de lugar das raízes.

4.2. Resposta ao Degrau

Para plotar a resposta ao degrau do sistema abra a aba Analysis Plot, defina o Plot Type para step, e selecione a caixa 1, Closed Loop r to y, ou seja, defina que se deseja plotar a resposta ao degrau de uma função de transferência do sinal r para o y (desejo verificar o comportamento de y quando faço r = 1):

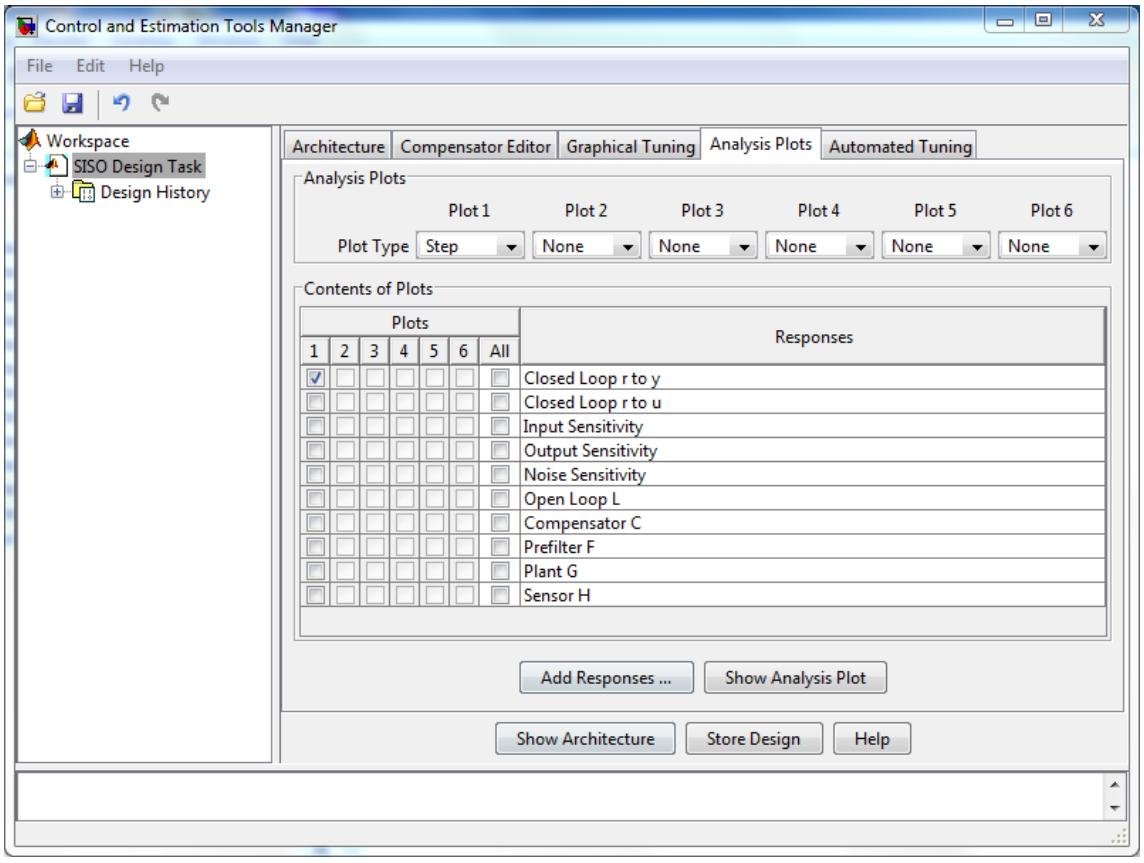


Figura 22. Aba Analysis Plots, que permite analisar, entre outras, a resposta em degrau.

Da mesma forma que o diagrama anterior, é possível ligar o grid, para facilitar a visualização. Entretanto, o menu de contexto passa a trazer algumas opções novas:

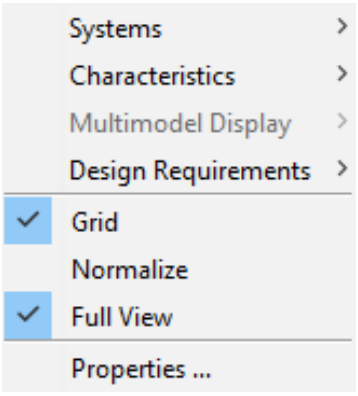


Figura 23. Menu de contexto dos Analysis Plots.

A opção Characteristics permite destacar características tradicionais da resposta ao degrau, como o pico da resposta, o tempo de subida e o sobressinal.

Selecionando, por exemplo, tempo de subida, um ponto azul aparece no gráfico destacando a característica. Clicando-se no ponto azul é possível ver o valor exato do mesmo.

A opção Design Requirements abre um diálogo mais complexo:

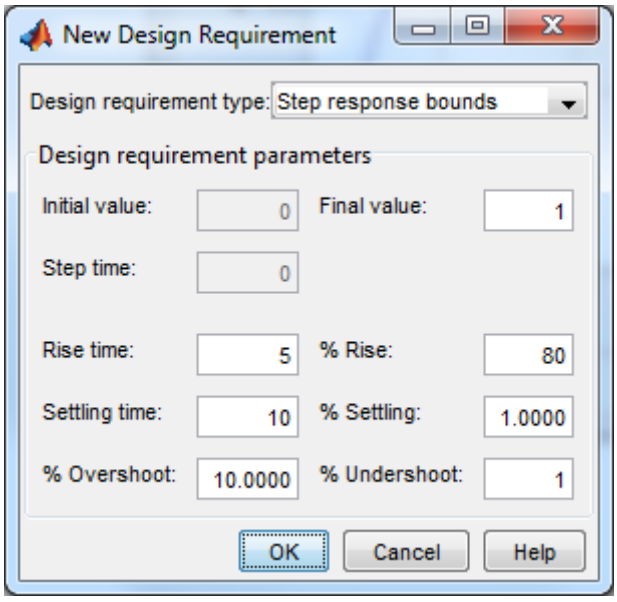


Figura 24. Design Requirements para a resposta ao degrau.

Nesse diálogo, é possível estabelecer uma série de restrições, incluindo restrições não convencionais.

Um exemplo seria: desejo que a planta atinja o valor 50% em 1s. Nesse caso, basta ajustar % rise para 50 e Rise time para 1s. **É importante observar**

que o settling time deve ser sempre menor que o rise time (é impossível setar o settling time para 0.5s, por exemplo, se o rise time for 1s).

É possível abrir ambos os gráficos, Root Locus e Step Response, e ir ajustando o ganho no primeiro, de forma que o segundo é automaticamente atualizado, o que permite designs com restrições mais complexas ainda usando tuning gráfico.

Em seguida, um exemplo de restrições:

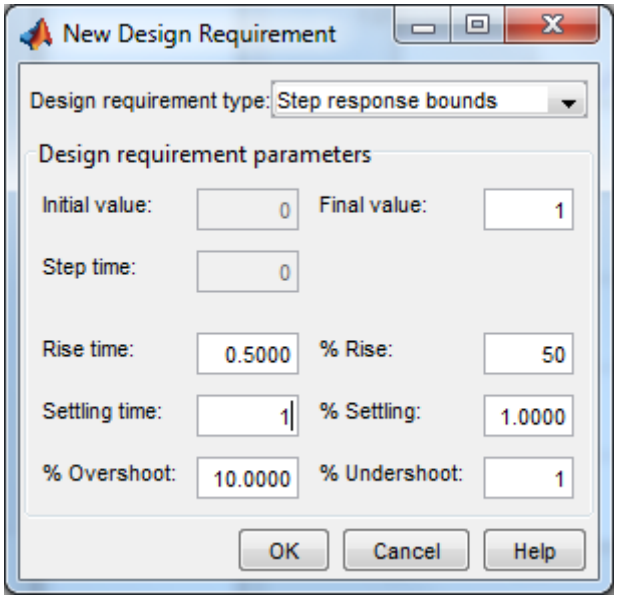


Figura 25. Exemplo de Design Requirements.

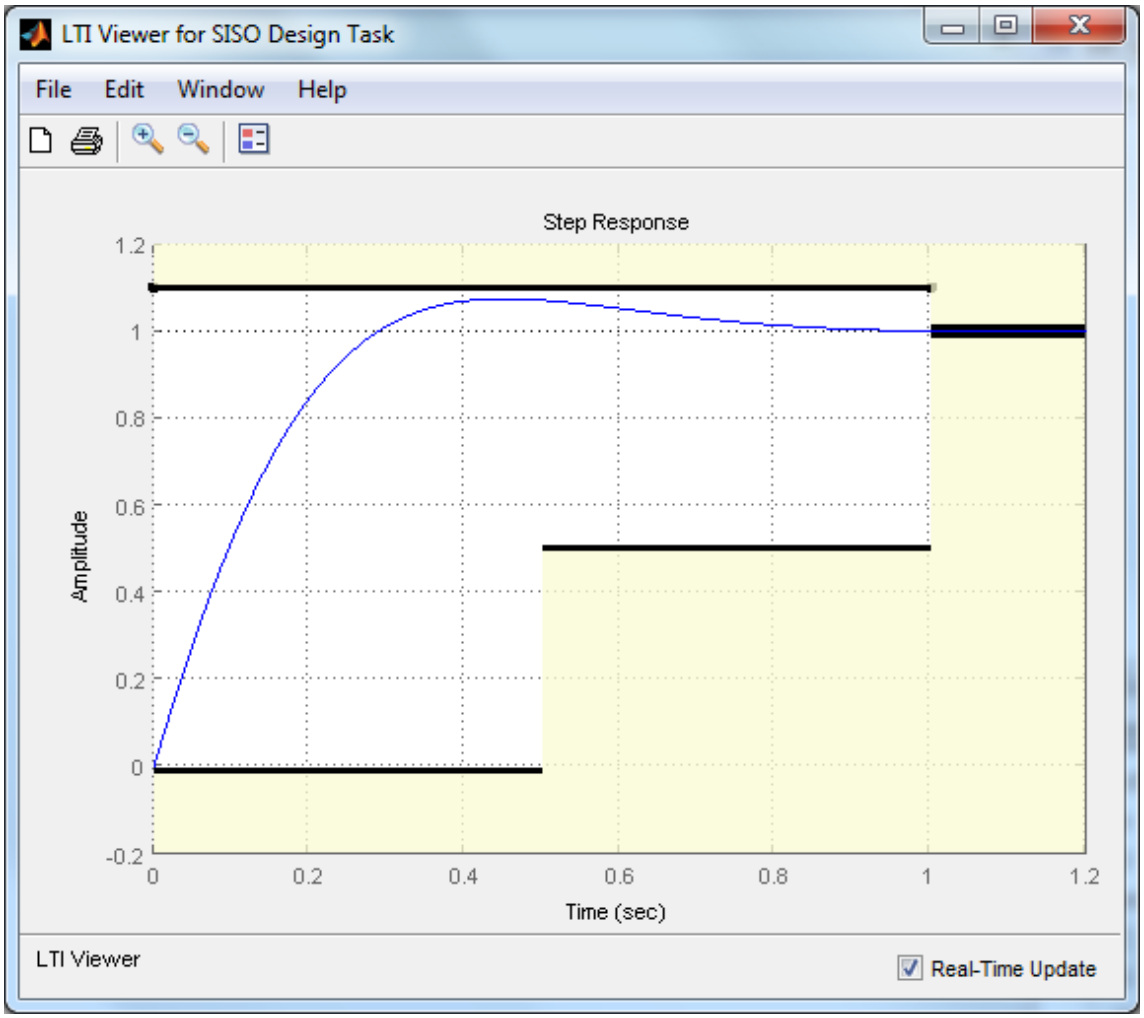


Figura 26. Restrições adicionadas ao gráfico com a resposta em degrau.

EXERCÍCIO 4

Projete um controle do tipo PD tal que o sistema final tenha 10% de sobressinal e tempo de assentamento de 1s. Inicie o projeto pelo diagrama do lugar das raízes (projeto aproximado, similar ao exercício anterior), em seguida ajuste a posição do zero e o ganho no diagrama de Lugar das Raízes para que as condições sejam satisfeitas no gráfico de resposta em degrau (crie o Design Requirement para o gráfico step, e destaque os valores obtidos de pico e tempo de assentamento usando a opção characteristics). Inclua imagens com o lugar das raízes e o gráfico de degrau.

4.3. Respostas Auxiliares

Antes de iniciar essa atividade, configure $C(s)$ para ser apenas um ganho (ou seja, delete o zero do controlador PD no diagrama de lugar das raízes).

Para facilitar a visualização dos gráficos, é possível deletar os Design Requirements clicando com botão direito -> Delete, e esconder as características da resposta em degrau deselecionando elas no menu de contexto.

Sete o valor de $C(s) = 1$ na aba Compensator Editor..

Agora adicione dois novos plotes de degrau em Analysis Plot, sendo que um é a resposta r to u, ou seja, o que acontece com o sinal de controle quando defino a referência igual a 1; e o segundo é definido em Add Responses, Input du e Output y, ou seja, o que ocorre com y quando existe um distúrbio de entrada igual a 1. Repare que para tal o Plot Type deve ser configurado para Step para o Plot1, Plot2 e Plot3, a resposta de du para y deve ser adicionada em Add Responses, e os check box correspondentes devem ser acionados.

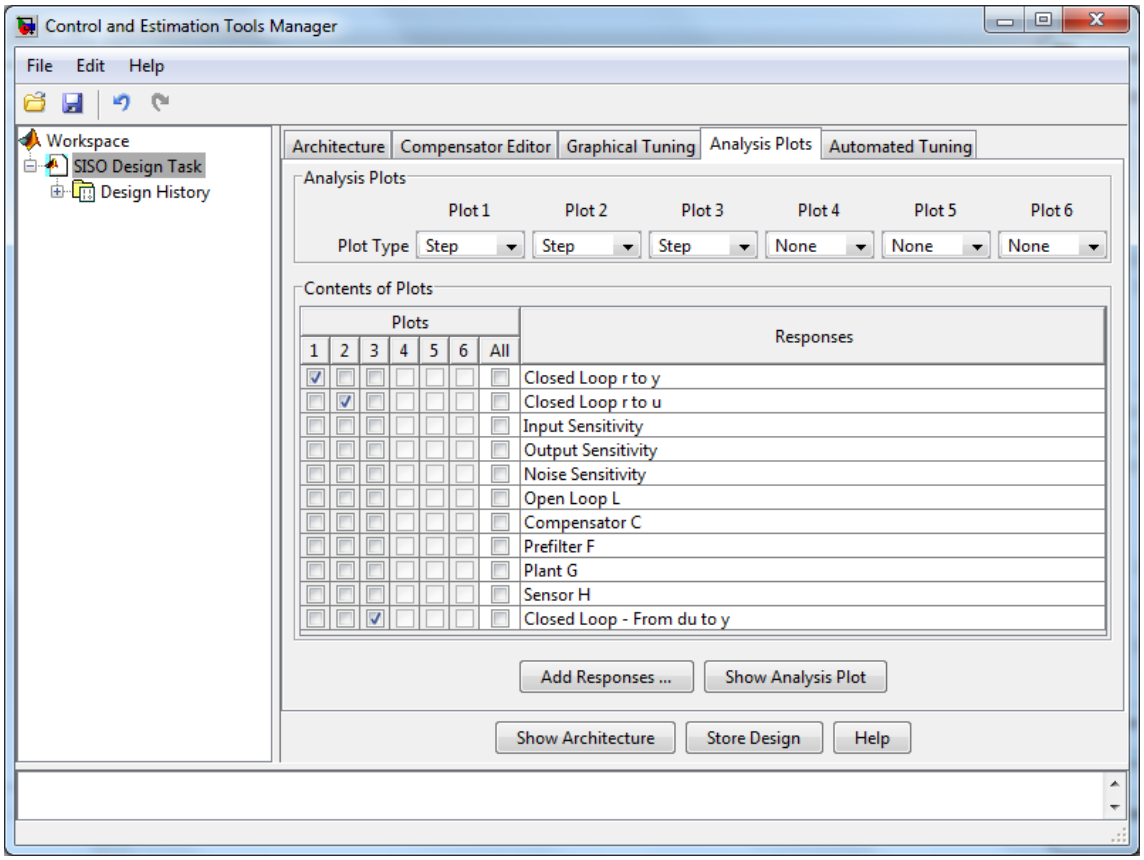


Figura 27. Adição de gráficos do tipo step adicionais.

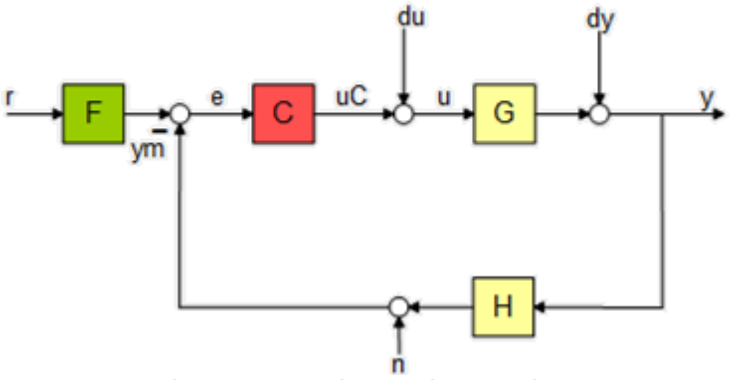


Figura 28. Arquitetura de controle.

O distúrbio de entrada du pode ser utilizado para antever aproximadamente o efeito do atrito μ na resposta final do motor, já que a entrada u efetivamente é convertida para torque (com os ganhos apropriados que relacionam a tensão de entrada e o torque do motor), e o atrito é um distúrbio de torque. A previsão será exata se não houver sobressinal, ou seja, se a velocidade do motor não mudar de sinal, já que, nesse caso, o efeito do atrito é igual a um distúrbio de torque de valor constante.

$$y(\infty) = r2y(s) \cdot r + du2y(s) \cdot \mu \tag{8}$$

O gráfico de resposta de r para u, por sua vez, pode ser usado para verificar se o controlador não satura, já que ele mostra o histórico do controle para um input de referência $r = 1$ no sistema.

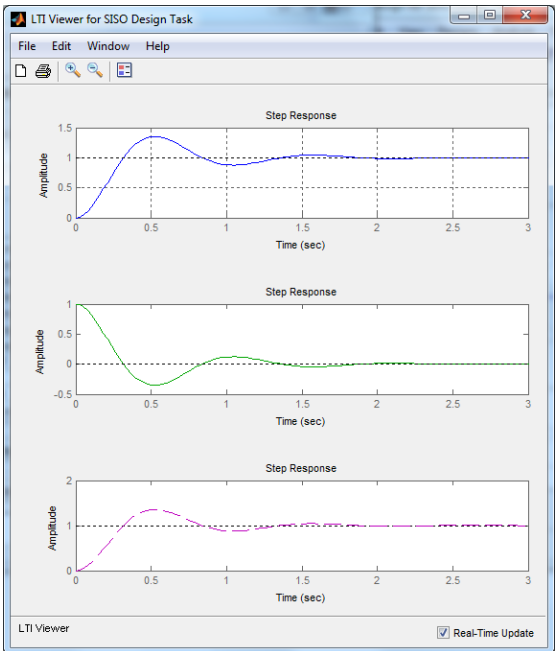


Figura 29. Respsotas ao degrau r2y, y2u e du2y.

Adicionando-se um zero em -1 e um integrador, chega-se em um controlador proporcional integral na forma:

$$C(s) = P + \frac{I}{s} = \frac{P\left(s + \frac{I}{P}\right)}{s}, P = 1, I = 1 \quad (9)$$

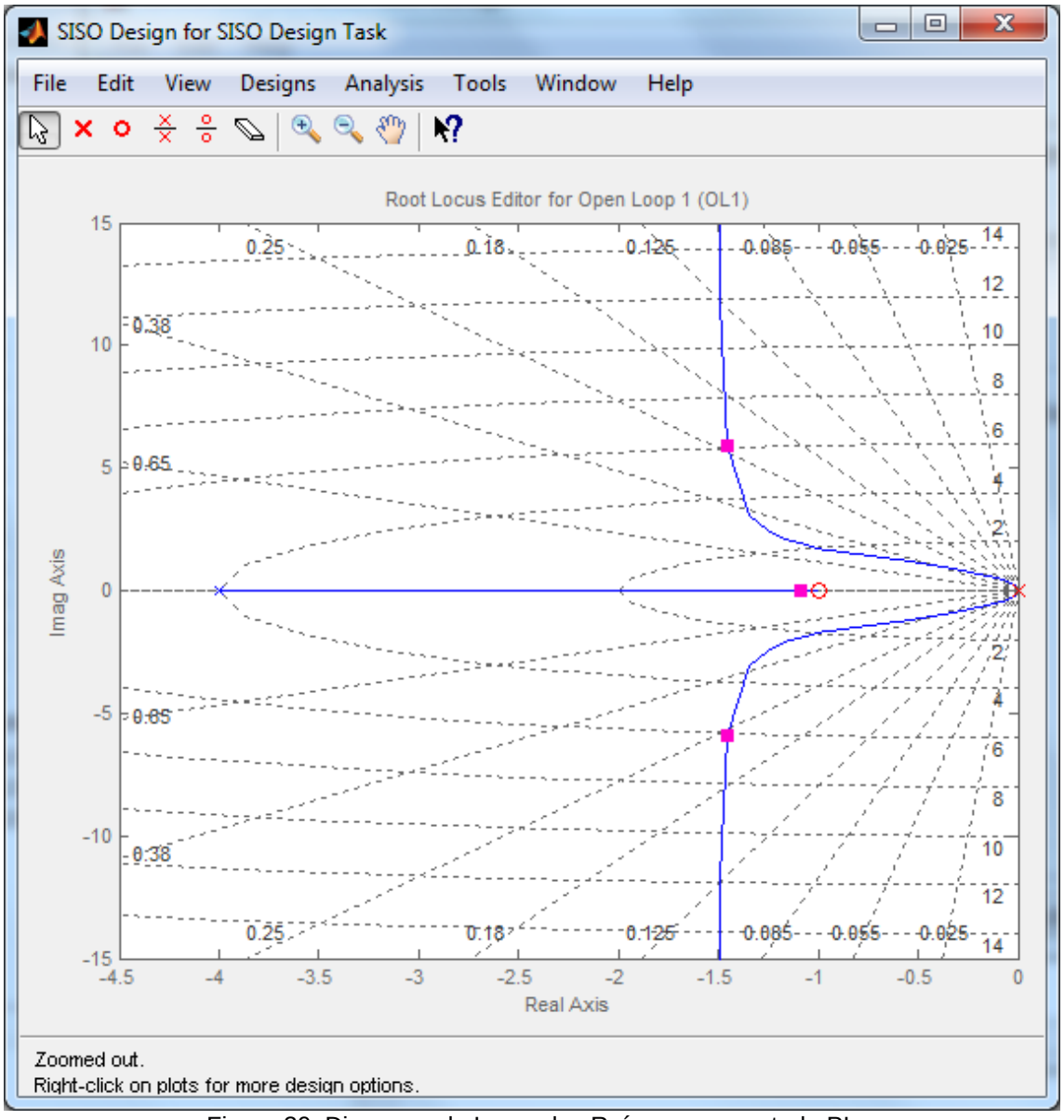


Figura 30. Diagrama de Lugar das Raízes para controle PI.

E as respostas do sistema passam a ser:

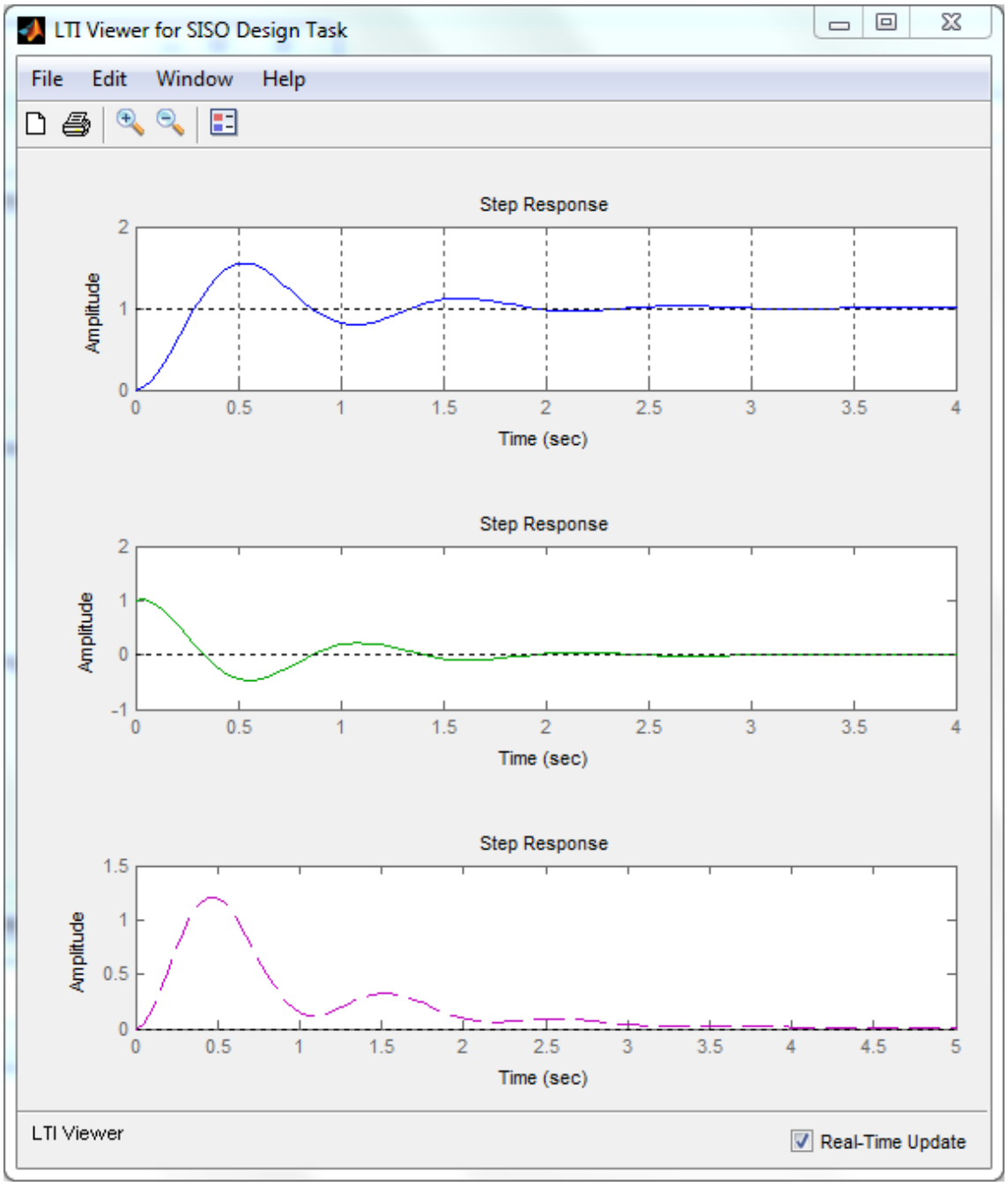


Figura 31. Respostas do sistema para controle PI.

Repare que, nesse caso, o sistema leva o distúrbio de atrito para zero, ou seja, o erro estático ao final será zero.

Ajustando-se o ganho do controle, ou seja, movendo-se os quadrados rosas no diagrama de lugar das raízes, é possível perceber que um aumento no ganho zera o erro estático mais rapidamente, mas aumenta a resposta máxima de u, ou seja, maior a chance de saturação.

Nesses casos, é interessante setar um Design Requirements diferente, clicando em Design Requirements -> New e escolhendo Design Requirement Type para ser Upper Time Response, é possível setar restrições genéricas em relação as respostas em degrau.

Por exemplo, é possível evitar saturação de 5 no sinal de controle para uma referência igual a 1 adicionando um limite superior ao gráfico de degrau r2u:

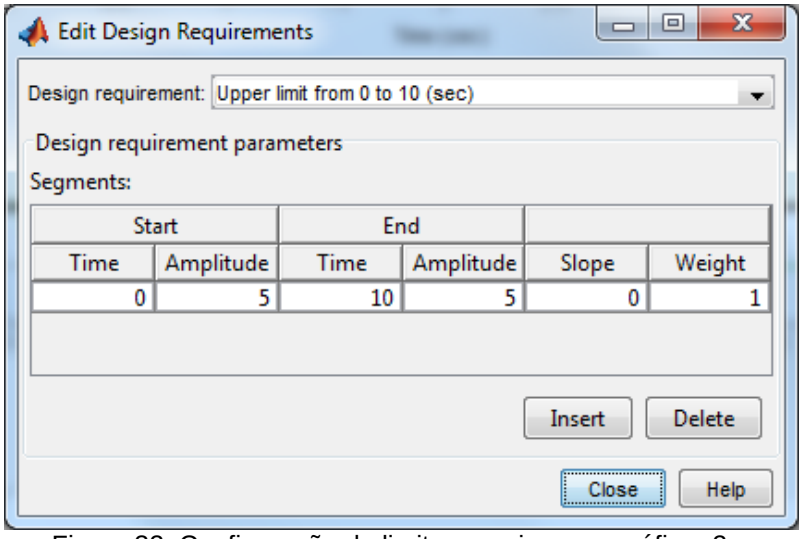


Figura 32. Configuração de limite superior para gráfico r2u.

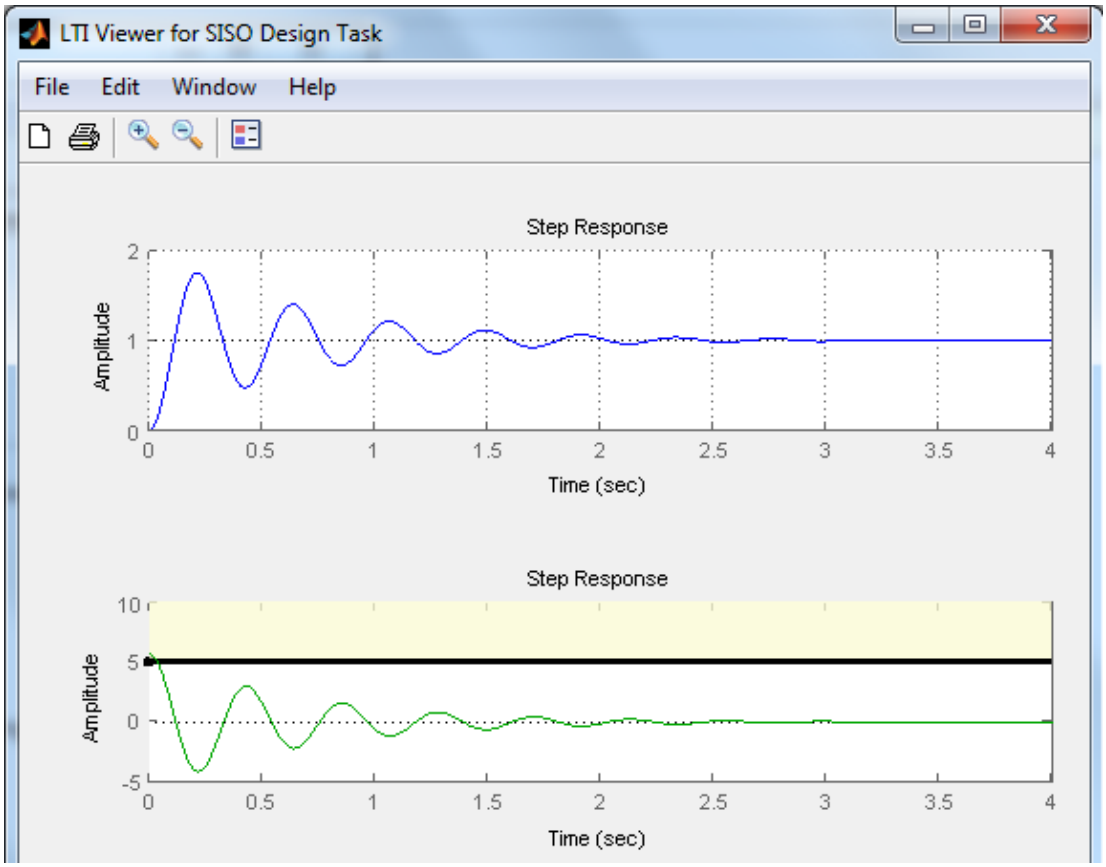


Figura 33. Com essa nova restrição, o limite de saturação fica explícito.

Ou então, é possível definir restrições para o erro estático adicionando limites superiores no gráfico de degrau du2y. Por exemplo, definindo erro estático menor que 5% após 2s:

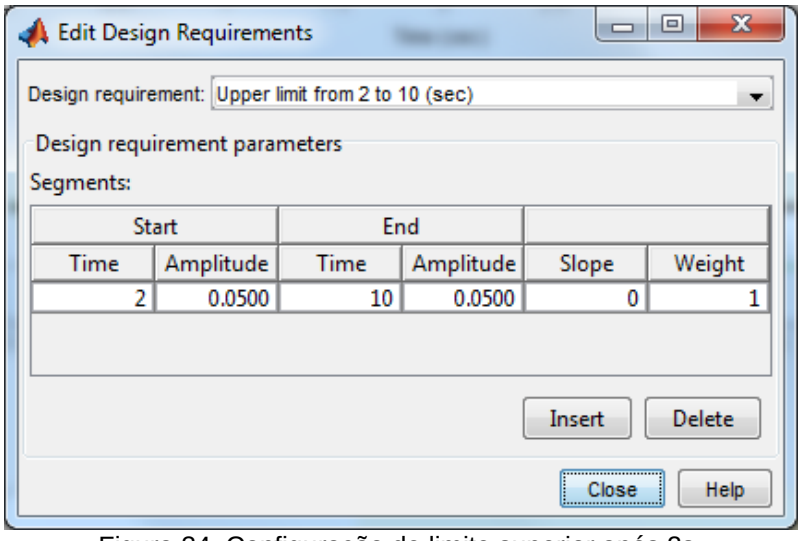


Figura 34. Configuração de limite superior após 2s.

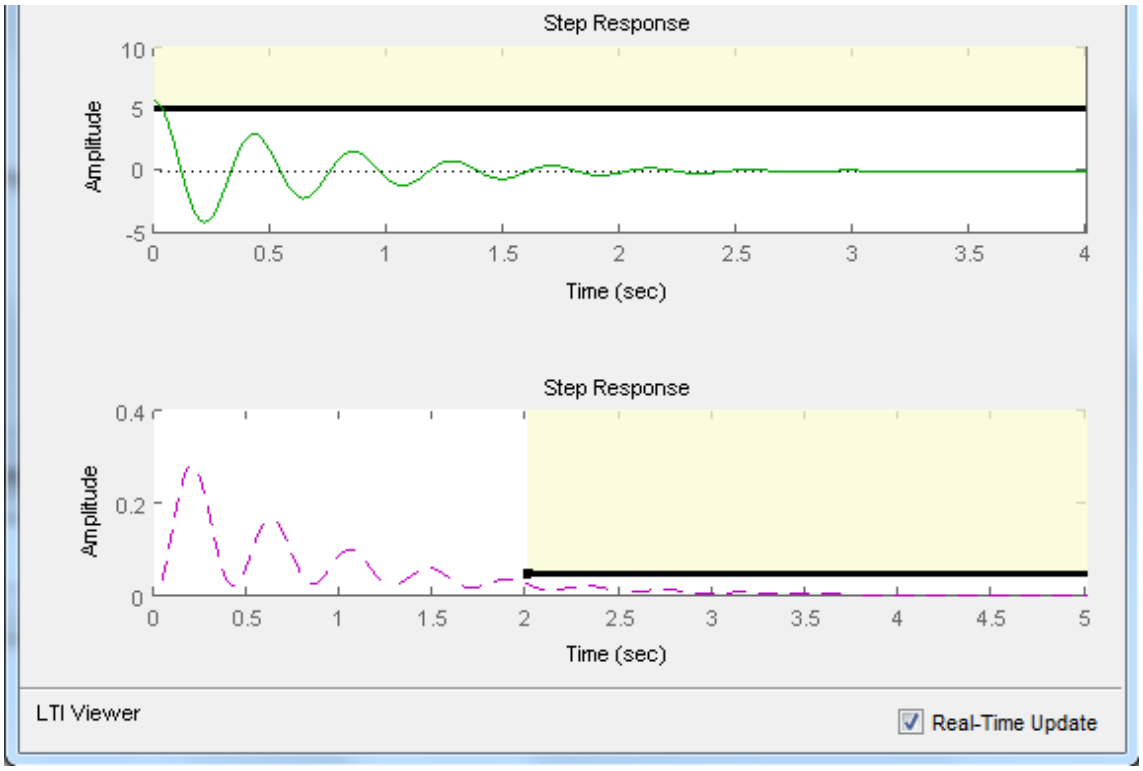


Figura 35. Com a nova configuração, o erro estático (causado pelo atrito), passa a ter uma restrição explícita.

Um problema, entretanto, ocorre quando um controlador do tipo PD é definido. Ou seja, um controlador na forma:

$$C(s) = D\left(s + \frac{P}{D}\right) \tag{10}$$

Nesse caso, a resposta em degrau de r para u, ou seja $r2u$ some. É possível entender o que ocorre quando calculamos a função de transferência:

$$y(s) = G(s) \cdot u(s)$$

$$u(s) = C(s)[y(s) - r(s)]$$

$$u = C[Gu - r]$$

$$u = CGu - Cr$$

$$(1 - CG)u = -Cr$$

$$u = -\frac{C}{(1 - CG)}r$$

$$u = -\frac{a(s+b)}{1 - a(s+b)\frac{10}{s(0.25s+1)}}r$$

$$u = -\frac{a(s+b)s(0.25s+1)}{s(0.25s+1) - a(s+b)10}r$$

Ou seja, a função de transferência é da forma:

$$u = -\frac{a_3 s^3 + a_2 s^2 + a_1 s + a_0}{b_2 s^2 + b_1 s + a_0}r \tag{11}$$

Entretanto, essa função de transferência é imprópria, ou seja, há mais zeros que polos, de forma que a mesma não pode ser representada como uma equação diferencial ordinária em função de um sinal de input, pois resultaria em algo como:

$$u(t) = \frac{dr(t)}{dt} \tag{12}$$

Enquanto sistemas próprios tem representação na forma:

$$\frac{du(t)}{dt} = r(t) \tag{13}$$

Esse problema é um problema clássico de controladores PID, que estabelece que quando o termo D é incluído, o controlador passa a não ser mais realizável, ou seja, não é possível implementá-lo como um sistema dinâmico próprio do tipo:

$$u(t) = Ax(t) \tag{14}$$

Para solucionar esse problema, a parte derivativa do controlador usualmente é alterada para incluir um novo polo, tornando a função transferência realizável:

$$Ds \sim D\frac{N_P s}{s + N_p}$$

Sendo que o valor de N_P é definido para ser grande o suficiente para que não influencie na dinâmica do sistema (quando $N_P \rightarrow \infty$, ambas as funções são iguais).

Para que seja possível determinar quão grande deve ser N_P, feche a interface gráfica sisotool e abra novamente, sem, contudo, definir uma função de transferência para G. Com isso, os valores de todas as funções transferência retornam para 1 (F, C, G e H).

Se o sistema foi reiniciado, ele passará a mostrar também o diagrama de Bode de malha aberta (se isso não ocorrer, carregue novamente o diagrama de Bode em Graphical Tuning -> Bode Open-Loop Bode).

Agora adicione um differentiator no diagrama de lugar das raízes (ou seja, um zero em zero, fazendo que $C(s) = s$:

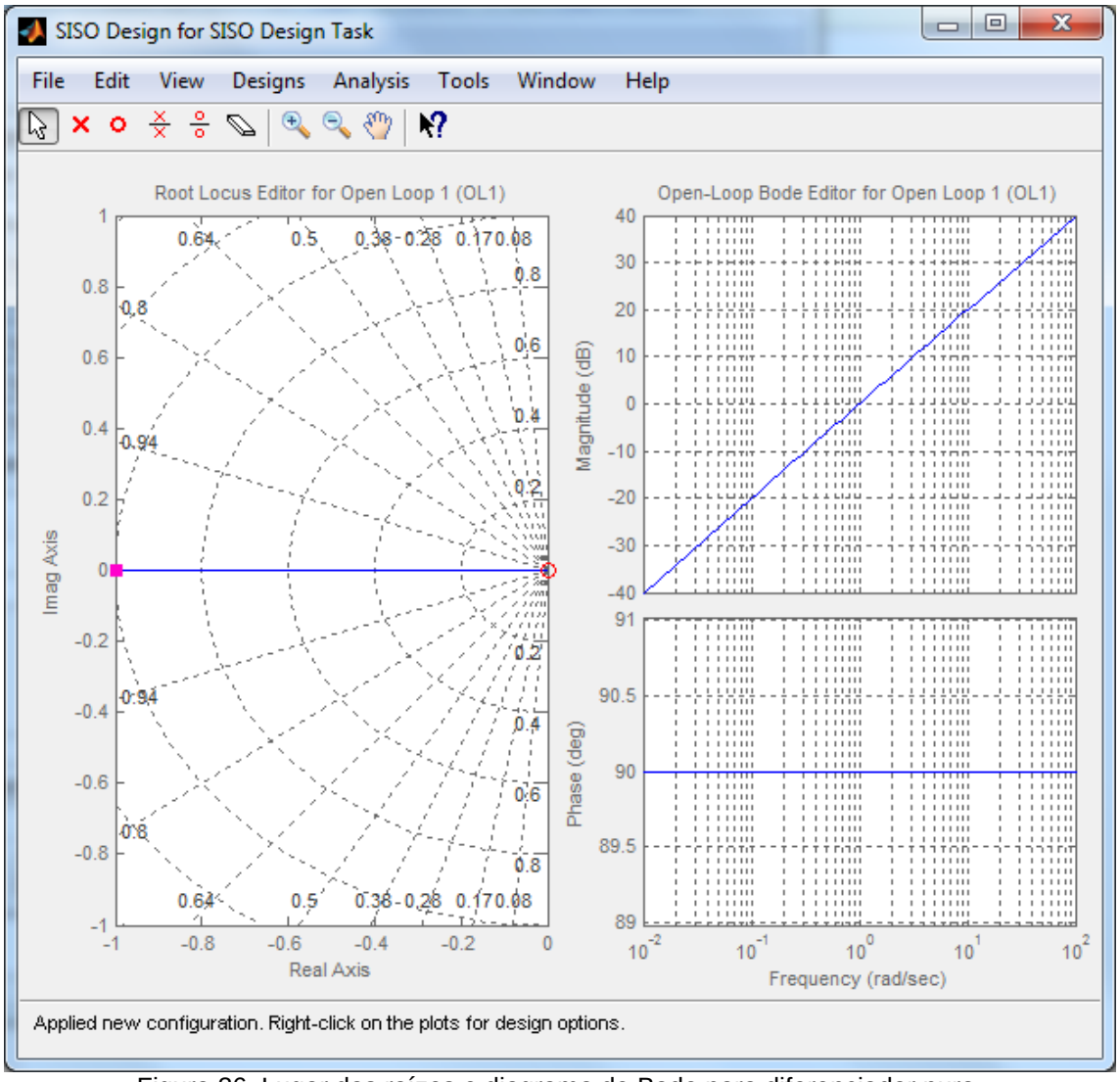


Figura 36. Lugar das raízes e diagrama de Bode para diferenciador puro.

Repare que o diagrama de Bode de malha aberta tem magnitude igual a uma reta com inclinação 20dB/década, e fase 90°, como estudado na teoria em disciplinas anteriores.

Agora adicione um polo real bem distante do zero, em -10, por exemplo:

$$C(s) = \frac{10s}{(s+10)} \tag{15}$$

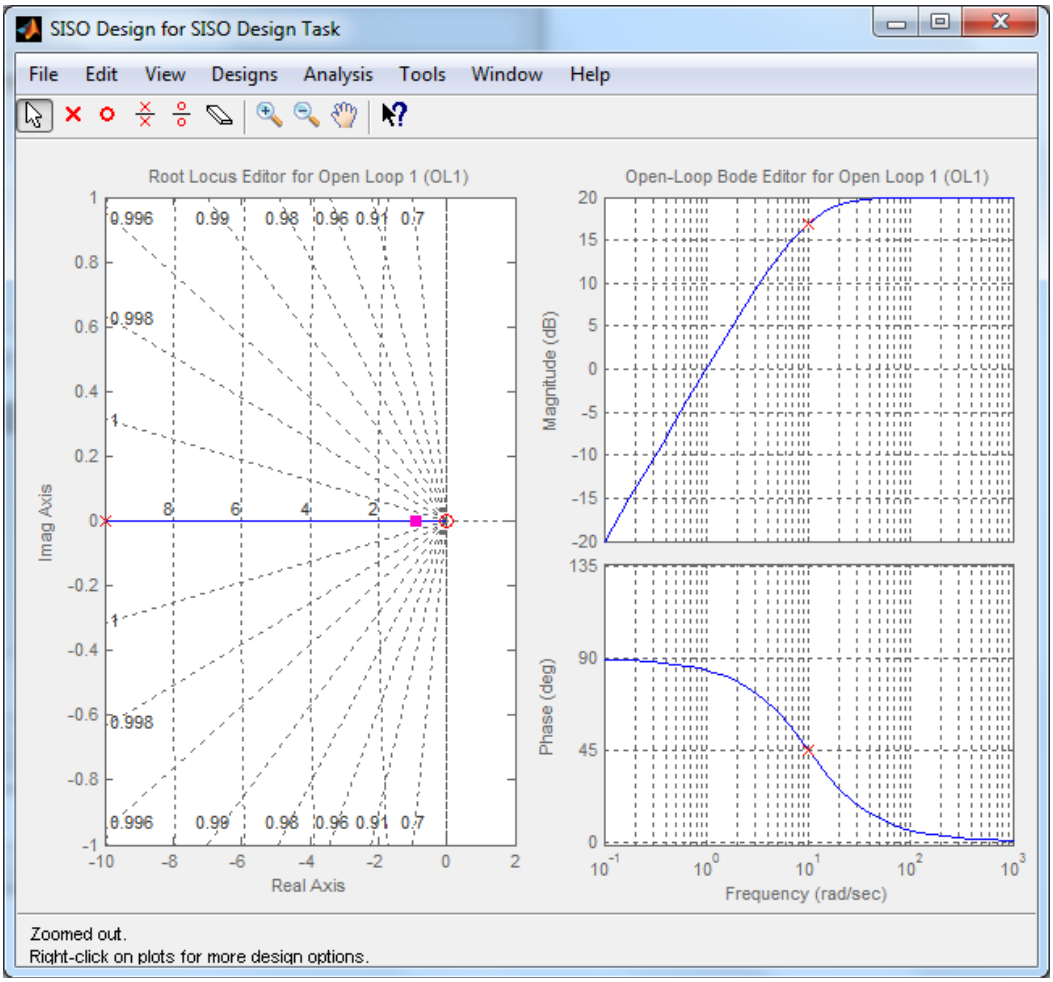


Figura 37. Lugar das raízes e diagrama de Bode de um diferenciador com filtro derivativo.

Repare que, nesse caso, até $1 rad/s$, o diagrama de Bode também tem fase aproximadamente 90° e magnitude igual a uma reta com inclinação 20dB/década. Ou seja, de fato o controlador proposto aproxima um diferenciador até uma dada frequência.

Para o caso de sistemas contínuos, normalmente define-se o diferenciador para funcionar além da frequência de corte do sistema.

Repare que é possível posicionar o polo usando o próprio diagrama de Bode, de forma que o polo representa a frequência em que a defasagem atinge 45°. Quando essa frequência é dividida por 10, a defasagem em relação ao derivativo puro é aproximadamente 6° (90°-84°), o que é considerável aceitável como aproximação.

Sendo assim, N_P deve ser cerca de 10 vezes a máxima frequência de interesse do sistema para que o polo adicional não impacte na resposta.

EXERCÍCIO 5

Verifique, utilizando o diagrama de Bode em malha aberta, qual a frequência de corte da planta $G_P(s)$ com um controlador na forma $C_1(s) = s$. Dada a frequência de interesse, calcule qual deve ser o valor de N_P adequado. Para avaliar o resultado, compare os diagramas de Bode obtidos quando:

$$C_1(s) = s \; e \; C_2(s) = \frac{N_P s}{s + N_P}$$

Repare que para o primeiro controlador deve-se adicionar um differentiator, e para o segundo basta adicionar um polo em N_P.

5. OTIMIZAÇÃO DO CONTROLADOR

Sumarizando o que foi visto até então, o controle PID pode ser definido na forma:

$$C(s) = P + D\frac{N_P s}{s + N_P} + \frac{I}{s}$$
$$C(s) = \frac{Ps^2 + PN_P s + DN_P s^2 + Is + IN_P}{s(s + N_P)} \quad (16)$$
$$C(s) = \frac{(P + DN_P)s^2 + (PN_P + I)s + IN_P}{s(s + N_P)}$$

Ou seja, para definir um controle PID devo adicionar:

- Um integrador;
- Um polo em N_P;
- Dois zeros reais;

E é possível adicionar diversas restrições, em diferentes gráficos, que permitem que eu defina características como sobressinal, tempo de assentamento, tempo de subida, evite saturação e estabeleça tempo de assentamento levando em conta atrito estático.

O projeto do controle, entretanto, ainda é feito de forma manual, ajustando-se os zeros e os ganhos até que todas as restrições sejam satisfeitas.

A última função do sisotool que será apresentada nessa apostila é justamente o tuning automático do controle para satisfazer as restrições, que será apresentado na forma de um exemplo. Nesse caso, suponha que:

1- O objetivo do controle é alternar entre dois set-points, em -3 e 3;
2- A saturação de controle ocorre quando $u = \pm 120$;
3- O motor não se mexe até que $u = 0.25$, ou seja, o atrito pode ser definido como $\mu = 0.25$;
4- O valor de N_P adotado será 100, nesse caso;
5- Desejo que meu sistema tenha sobressinal de 20%, tempo de assentamento (erro $\pm 2\%$) de 2s, e tempo de subida de 50% em 0.2s.

Sendo a planta:

$$G_P = \frac{10}{s(0.25s + 1)}$$

Inicialmente, o sisotool será aberto e a planta será carregada, e os seguintes gráficos serão colocados em destaque: Lugar das raízes, degrau de r para y, degrau de r para u e degrau de du para y.

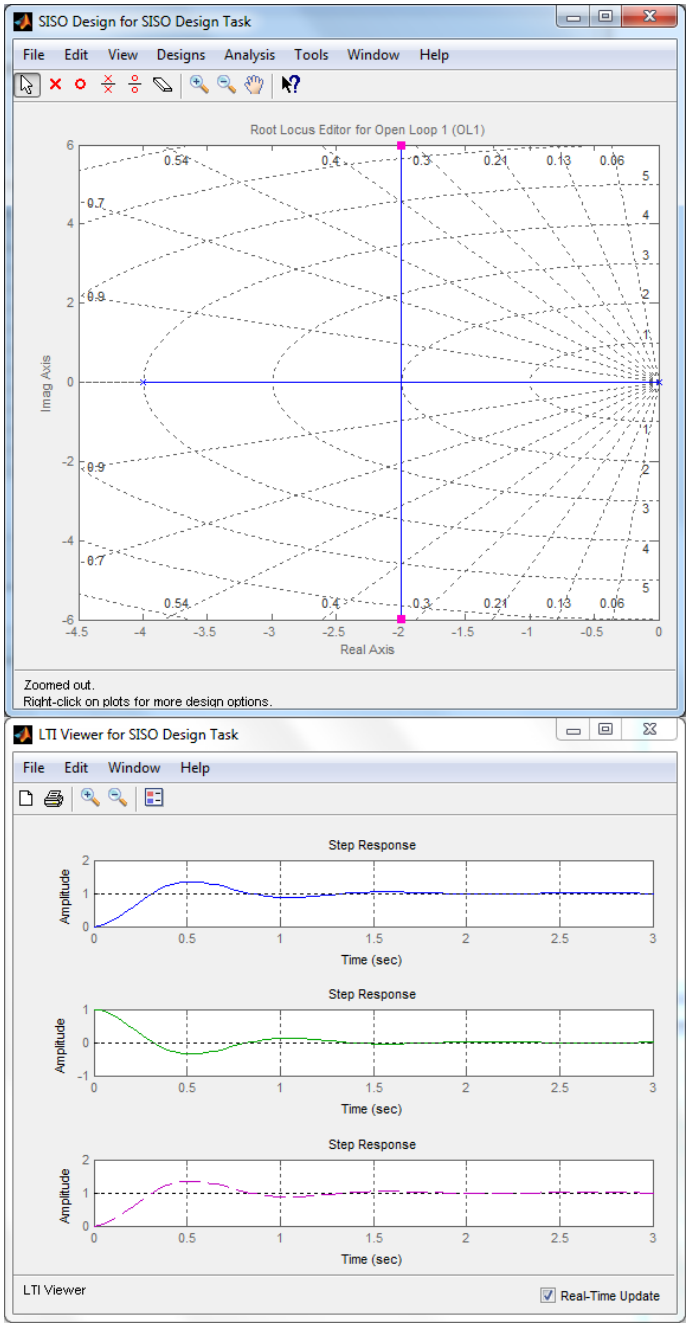


Figura 38. Diagrama de Lugar das Raízes e respostas r2y, r2u, du2y.

As respostas em degrau são sempre plotadas para input igual a 1. Quando o input alterna de -3 para 3 a amplitude do degrau passa a ser 6, sendo assim, os critérios passam a ser:

- $saturação \pm 120\ para\ 6 \rightarrow\ saturação\ de \pm \frac{120}{6} = 20\ para\ 1$
- $assentamento\ de\ 2\%\ em\ 2s \rightarrow 0.02 \cdot 6 = 0.12\ de\ erro\ estático$
- $assentamento\ 2\% \rightarrow erro < 0.12\ em\ 2s, \pm 0.12\ para\ \mu = 0.25 \rightarrow$ $\pm \frac{0.12}{0.25} = 0.5\ para\ du =\ 1$

O último critério implica que o efeito do atrito será compensado até 2s, de forma que o erro que o mesmo causa será inferior a 0.12.

Inicialmente é possível projetar de forma aproximada um controle PD usando o método do lugar das raízes. Para tanto, devo adicionar um polo real em $N_P = -100$, e um zero real, que será adicionado próximo do polo da planta em -4, já que isso simplifica a ordem do sistema completo e facilita o projeto do controlador.

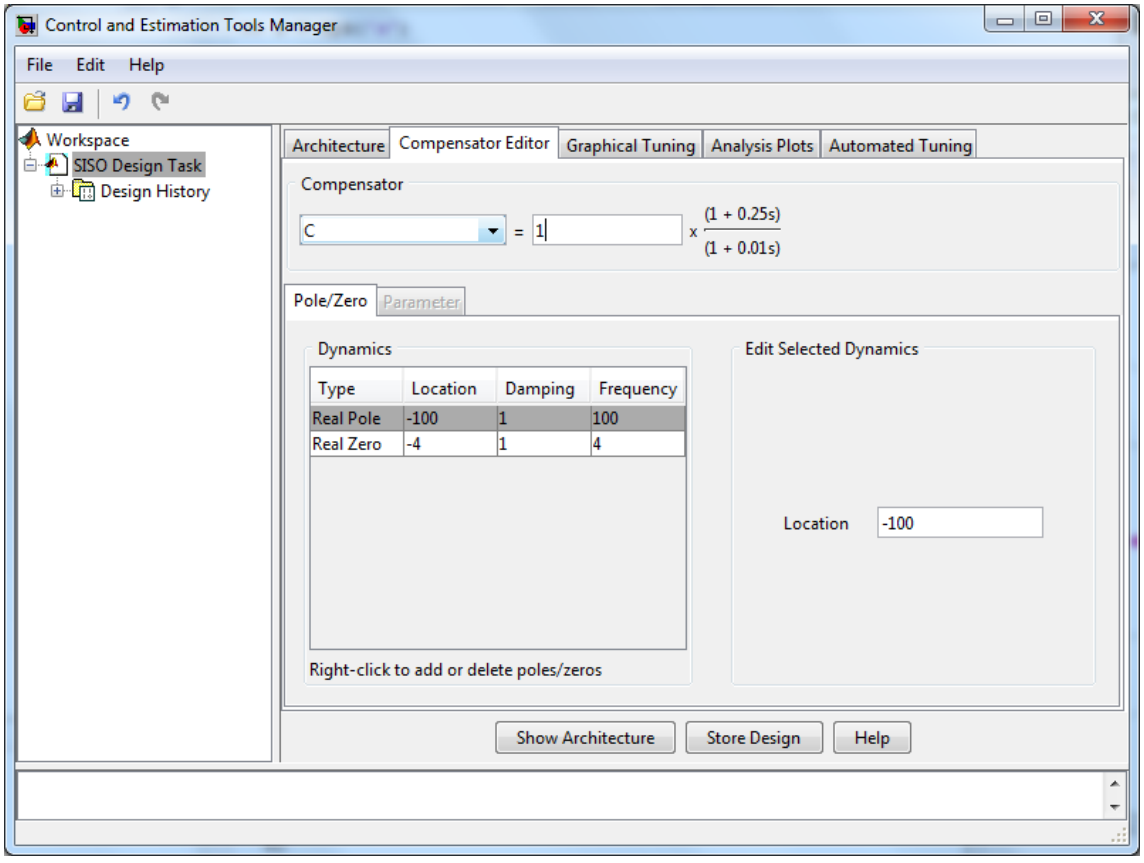
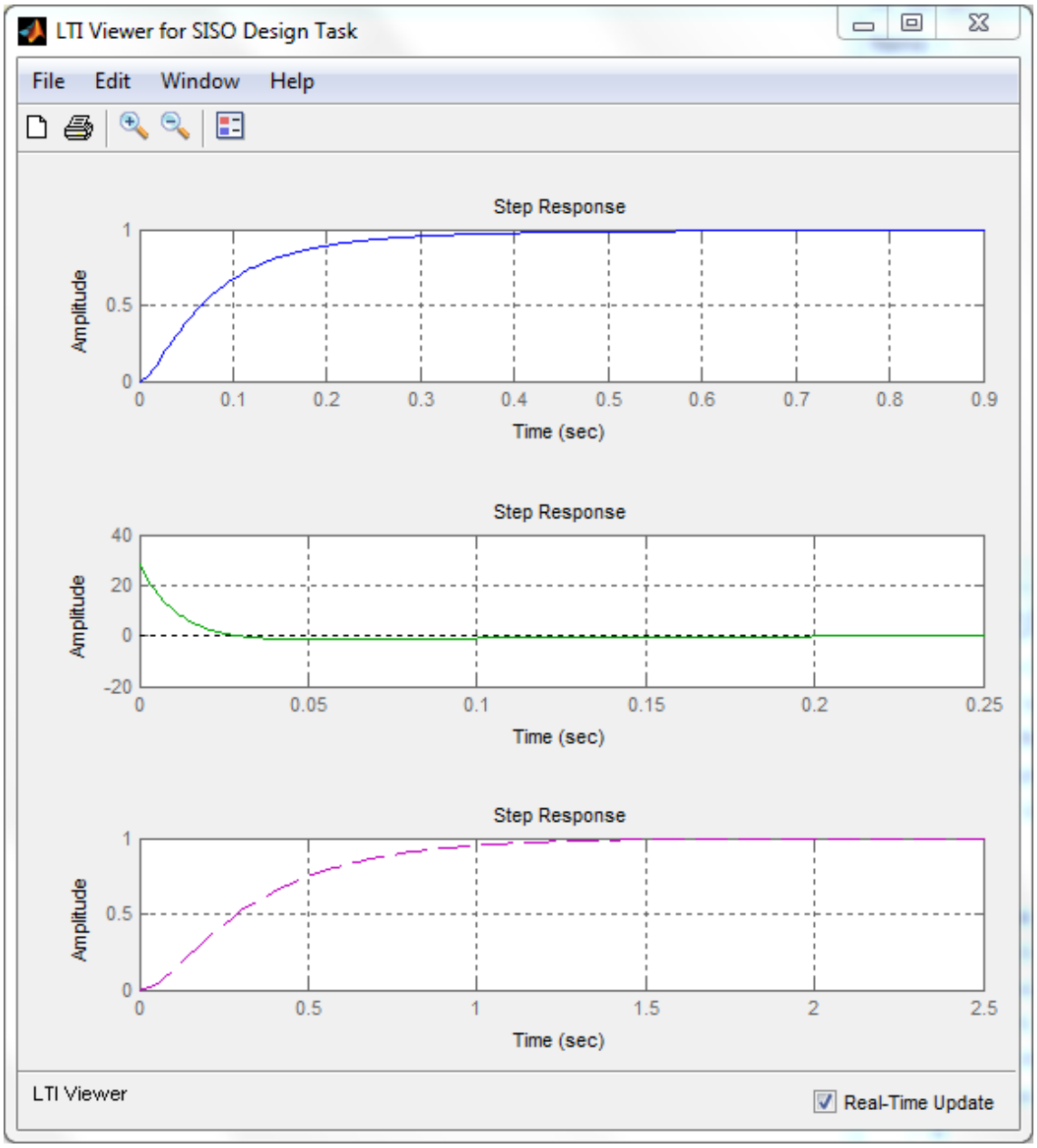


Figura 39. Configuração do controlador.



Com isso já é possível obter um controle PD com boas características de desempenho. Apesar disso, é possível observar que o erro estático devido ao atrito (gráfico du2y) é maior que 0.5, ou seja, as características de erro estático não foram satisfeitas; além disso o gráfico r2u indica um valor máximo acima de 20, ou seja, se um degrau de 6 for o input do sistema, a saturação de $6 \cdot 20 = 120$ não será respeitada.

Para diminuir o erro do atrito existem duas possibilidades:

- Aumentar o ganho do controlador (ou seja, aumentar P);
- Introduzir o termo integral para formar um controlador PID;

Ajustando-se o ganho para 2, o erro devido ao atrito cai para 0.5, como esperado, entretanto o valor de controle máximo excede muito o valor de saturação de ± 20:

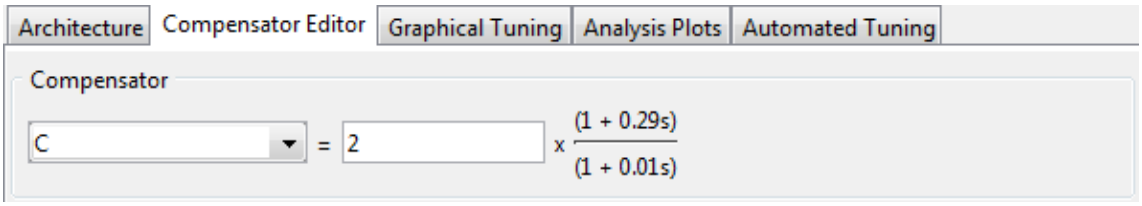


Figura 40. Ganho do controle ajustado para dois.

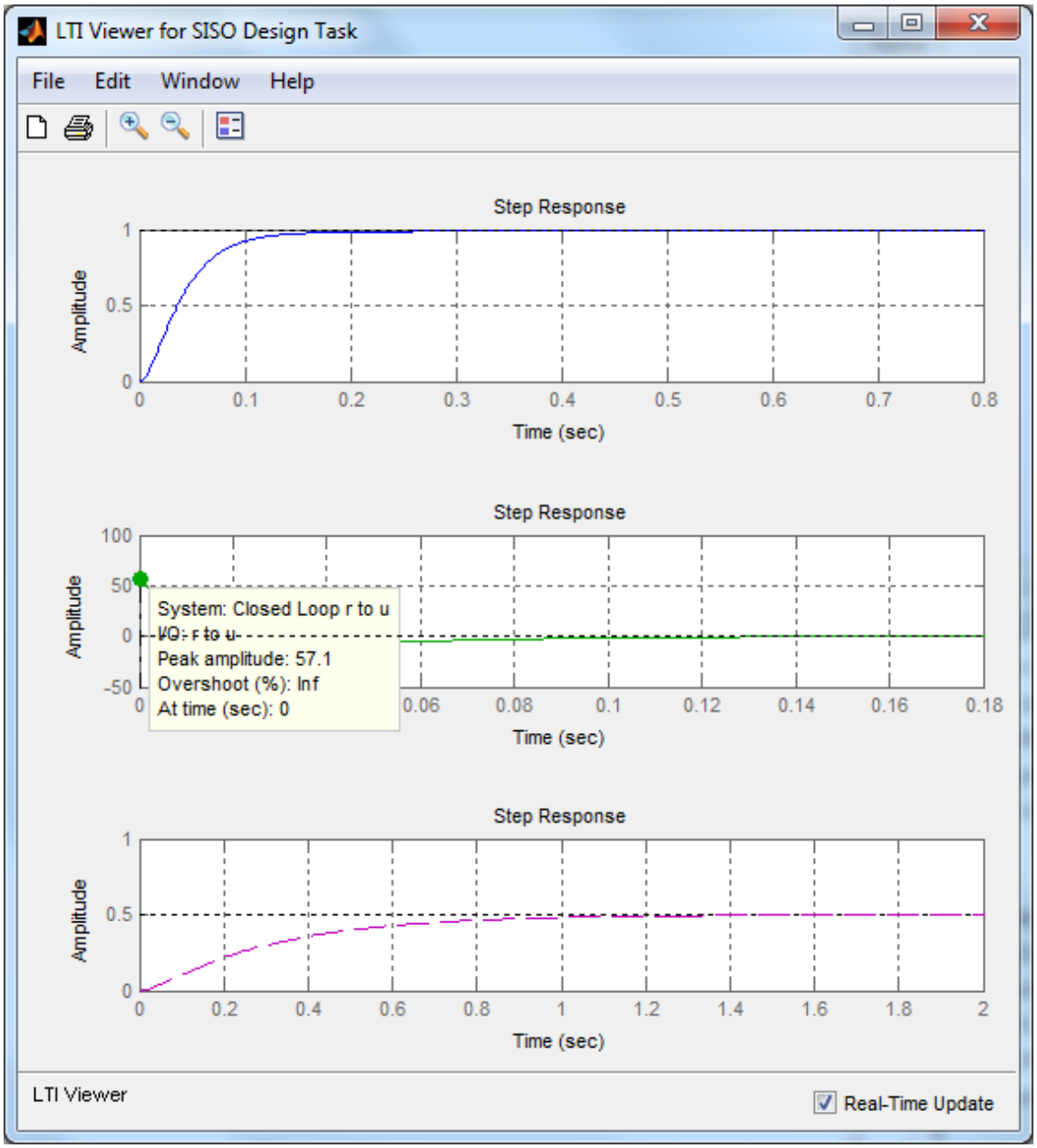


Figura 41. Respostas do sistema, mostrando que o sistema responde mais rápido (observando a resposta r2y), que o critério de saturação é violado (no gráfico verde o valor máximo deveria ser 20), e que o critério de erro estático devido ao atrito é respeitado (inferior a 0.5).

Sendo assim, a próxima opção é retornar ao ganho igual a 1 e adicionar o termo integral. Recuperando a forma completa de controle:

$$C(s) = \frac{(P + DN_P)s^2 + (PN_P + I)s + IN_P}{s(s + N_P)} \tag{17}$$

É necessário adicionar mais um integrador e um zero real negativo ao controle. Alguns valores tentativos desse novo zero foram testados (-4, -3, -2 e finalmente -1), até que o valor -1 foi selecionado como razoável, com características que atendem as especificações em quase todos os requisitos, menos no requisito saturação, como pode ser visualizado na figura a seguir:

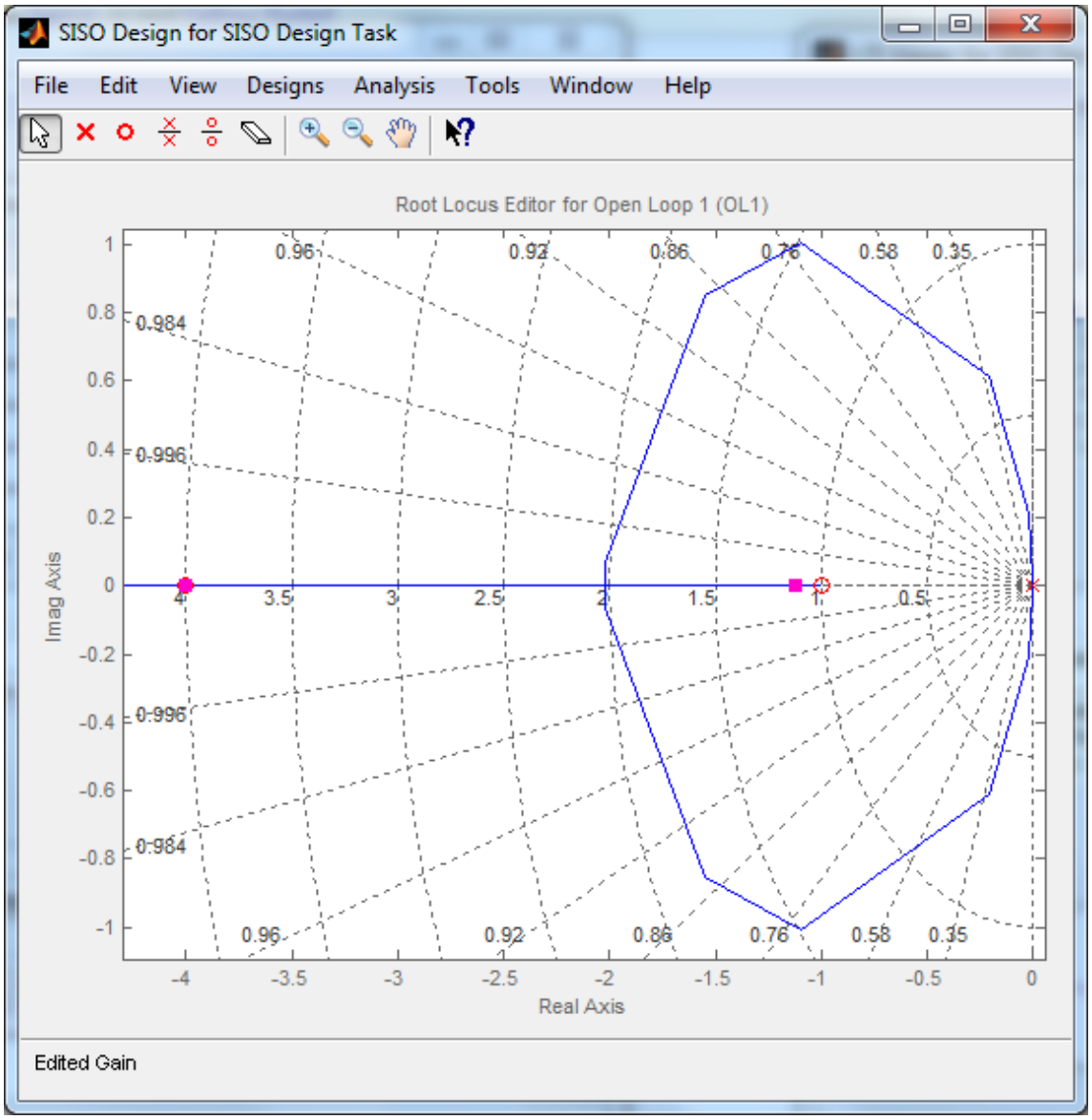


Figura 42. Lugar das raízes para o controle PID com o novo zero adicionado em -1.

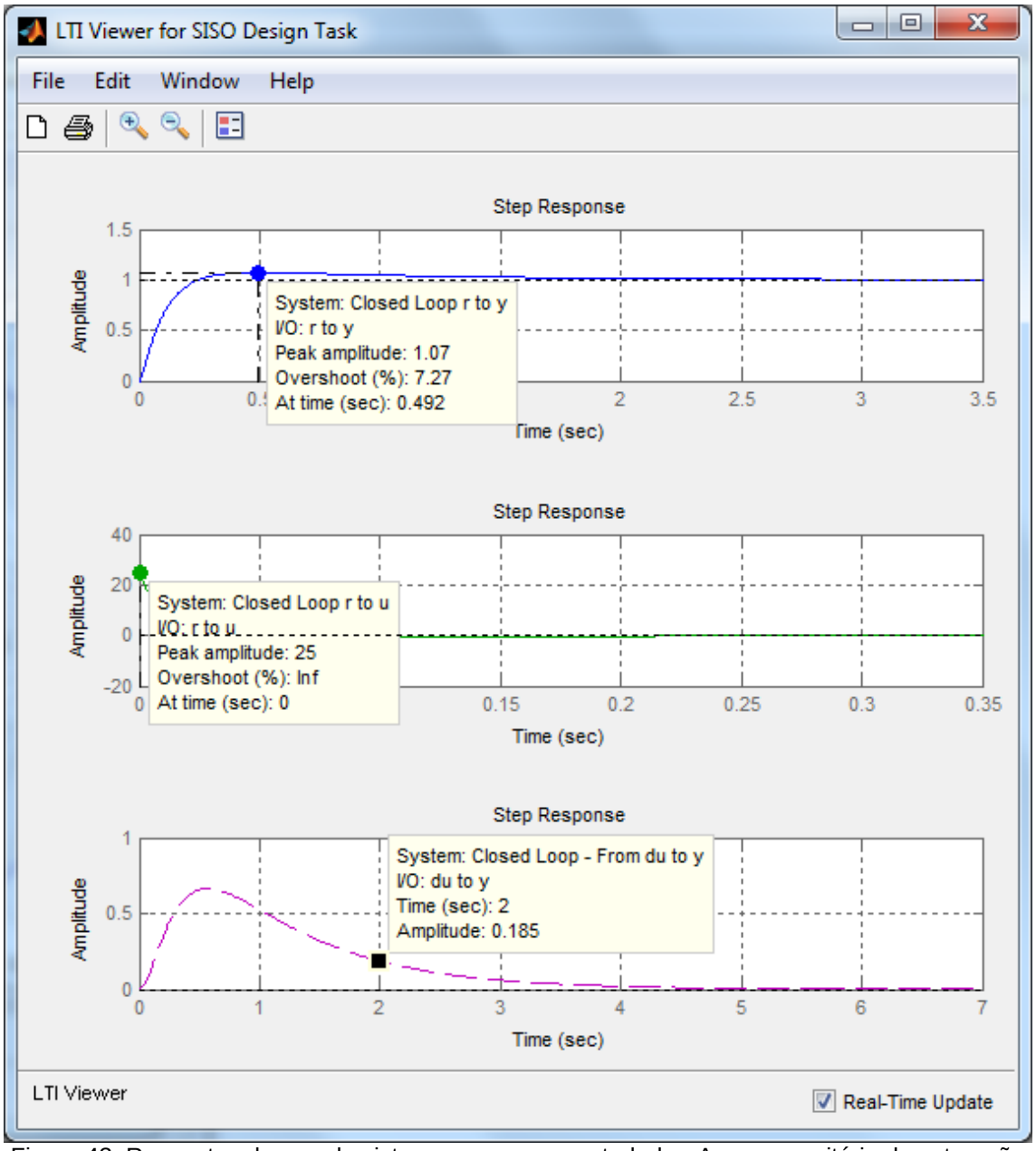


Figura 43. Respostas degrau do sistema com novo controlador. Apenas o critério de saturação não é cumprido, porém está próximo de ser cumprido.

Para se atingir os requisitos, as restrições são então adicionadas em cada um dos gráficos das respostas em degrau:

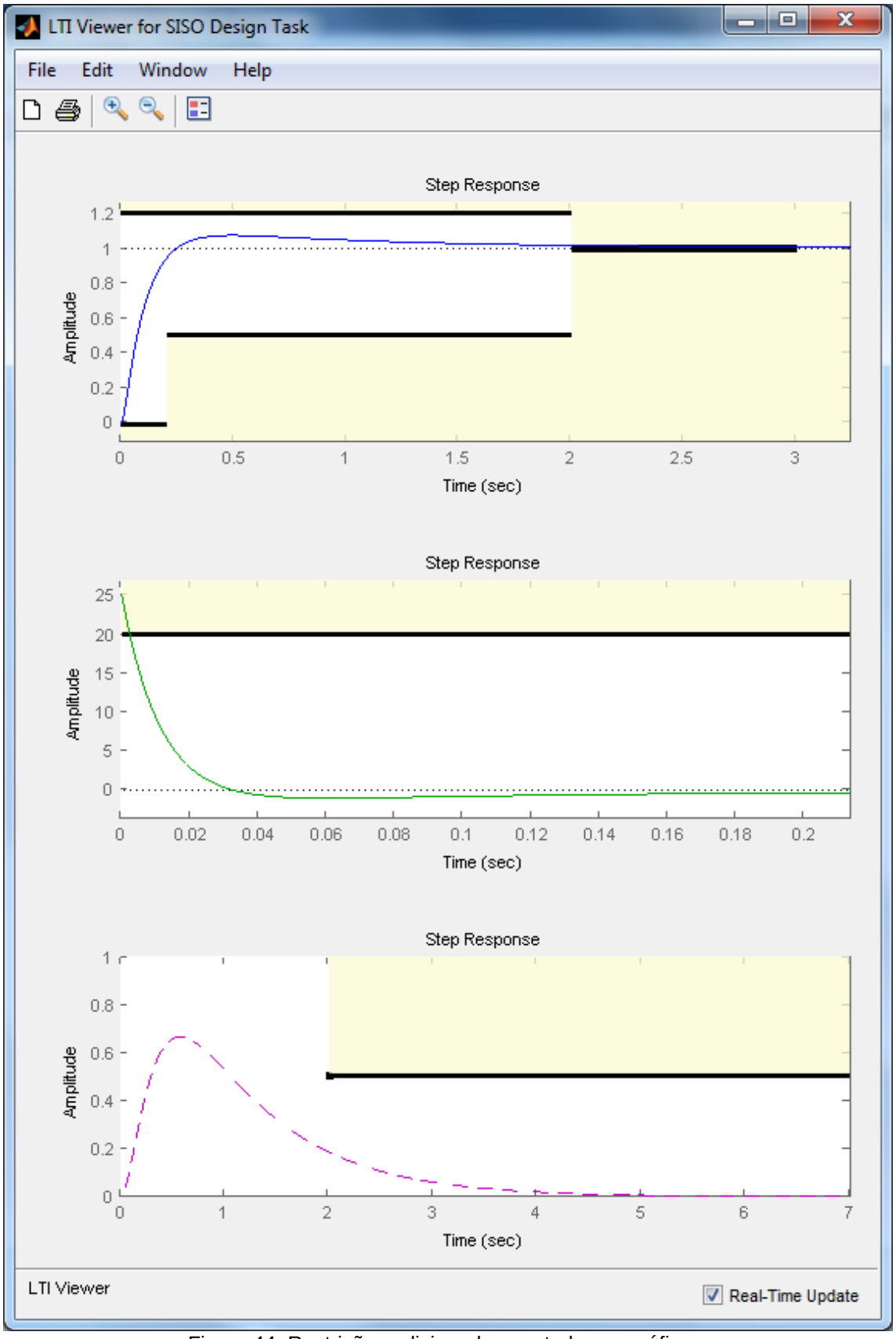


Figura 44. Restrições adicionadas em todos os gráficos.

Com essas restrições setadas, é possível selecionar, na aba Automated Tuning, Optimization Based Tuning > Optimize Compensators.

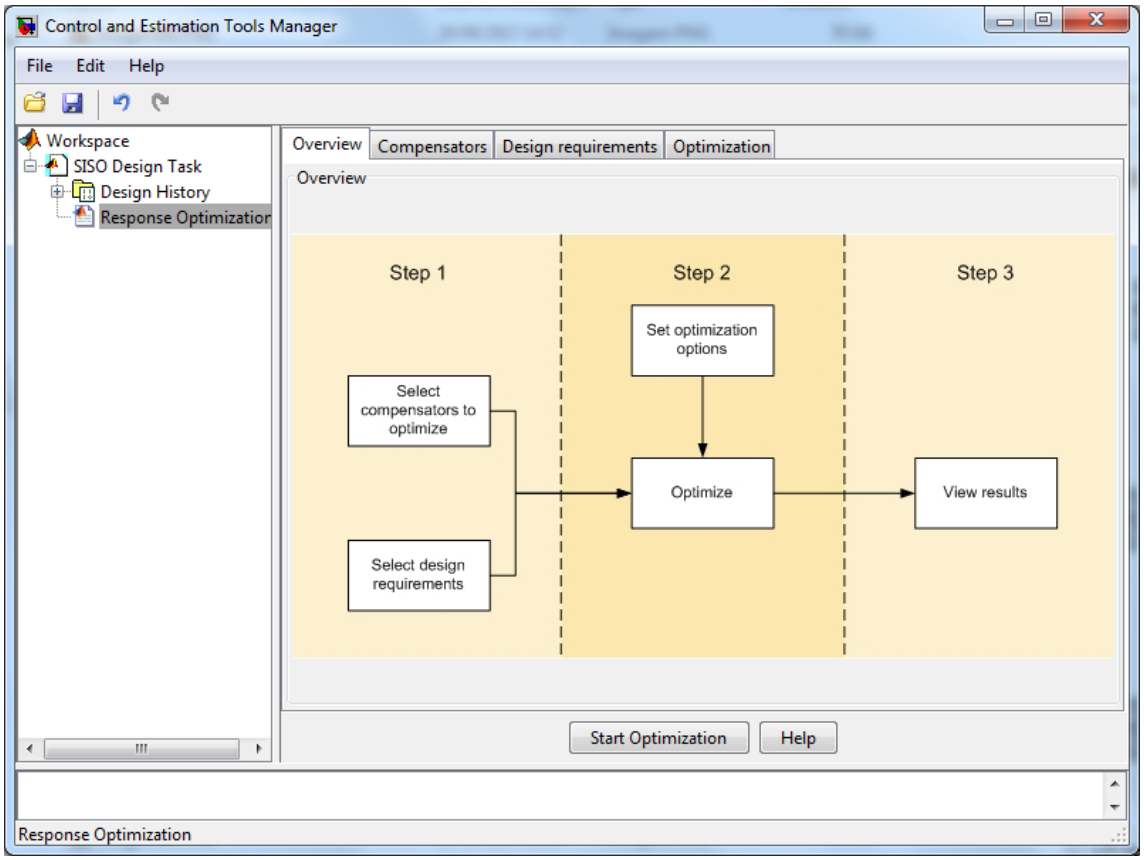


Figura 45. Janela inicial para o Response Optimization. Repare que pode-se voltar na janela do sisotool clicando-se em SISO Design Task na árvore de projeto à esquerda.

Na aba compensator, é interessante configurar que o objetivo é apenas mexer no ganho do controle e nos dois zeros reais (evitando mexer no integrador e no polo devido ao filtro derivativo):

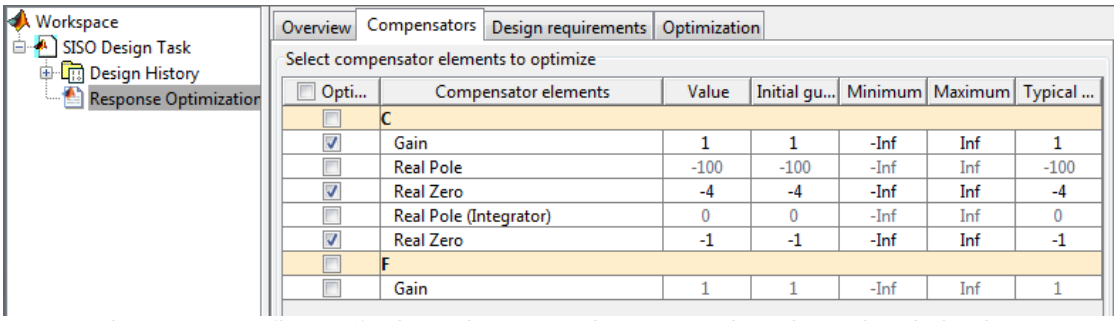


Figura 46. Configuração dos valores que devem ser alterados pelo otimizador.

Também é interessante limitar os valores para que o espaço de busca não seja muito extenso. Valores razoáveis para o problema em questão são os seguintes (lembrado de limitar os zeros para valores positivos para evitar um sistema instável):

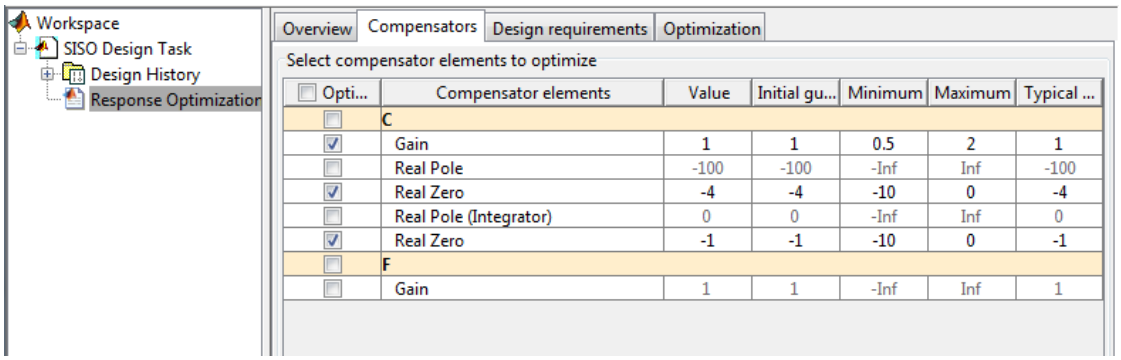


Figura 47. Restrição dos valores de busca.

Na aba Design Requirements é importante selecionar apenas os requisitos dos degraus (no caso não existem outros requisitos, mas é possível que outros requisitos tenham sido adicionados no lugar das raízes durante as fases preliminares de projeto):

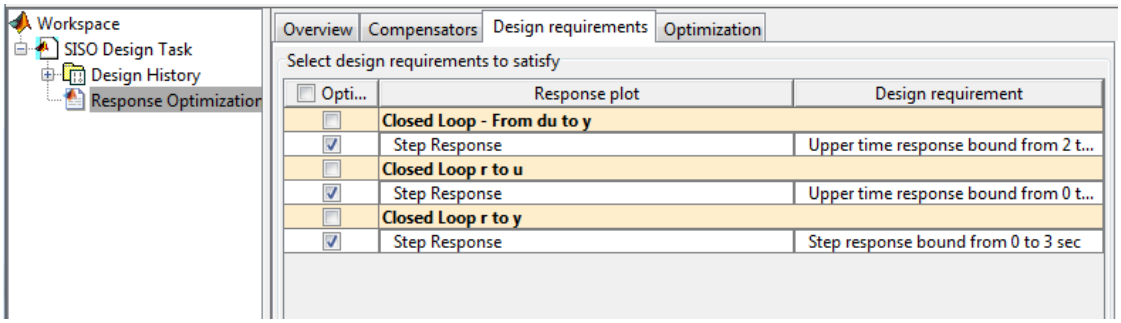


Figura 48. Requisitos de projeto selecionados, usando as resposta em degrau.

Na aba Optimization, opção Optimization options é possível observar qual é o algoritmo de otimização que será utilizado.

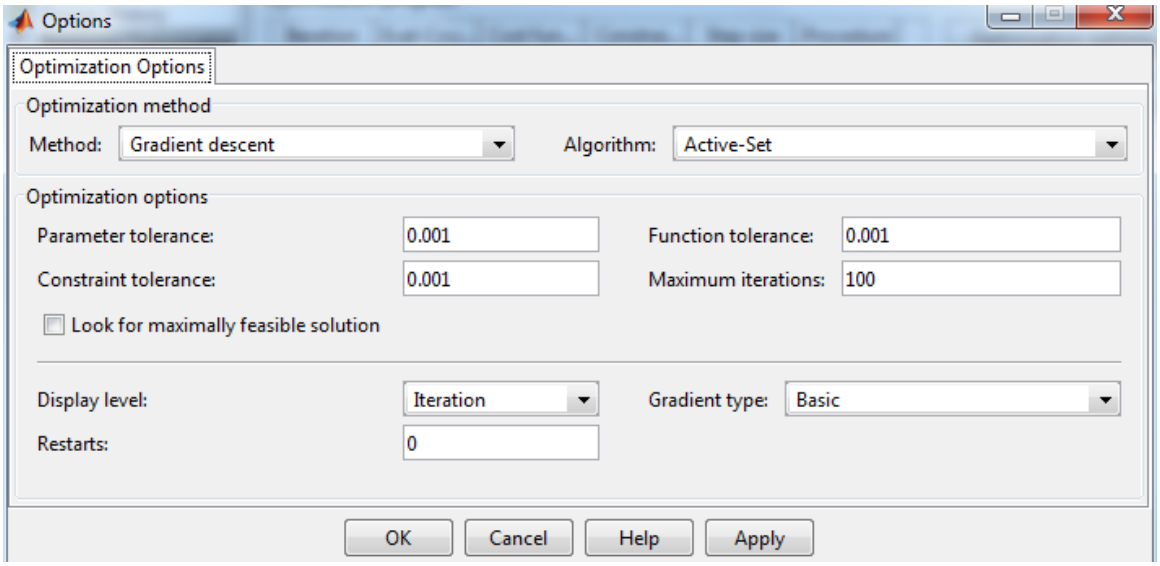


Figura 49. Algoritmo de otimização, tipo gradiente.

Repare que o algoritmo é do tipo Gradiente Descent, isso significa que é importante um chute inicial razoável para o controle, ou seja, **as técnicas clássicas que você aprendeu durante o curso ainda são necessárias**.

Com tudo concluído, basta clicar em Start Optimization. Ao longo das iterações é possível acompanhar as variações na resposta à medida que o algoritmo vai otimizando o problema.

Após 5-6 iterações o controle a seguir é encontrado satisfazendo todos os requisitos (pequenas variações numéricas podem ocorre):

$$C(s) = \frac{20(s + 8.736)(s + 0.5723)}{s(s + 100)} \tag{18}$$

Como pode ser atestado na figura a seguir, mostrando os valores de todos os parâmetros relevantes ao problema:

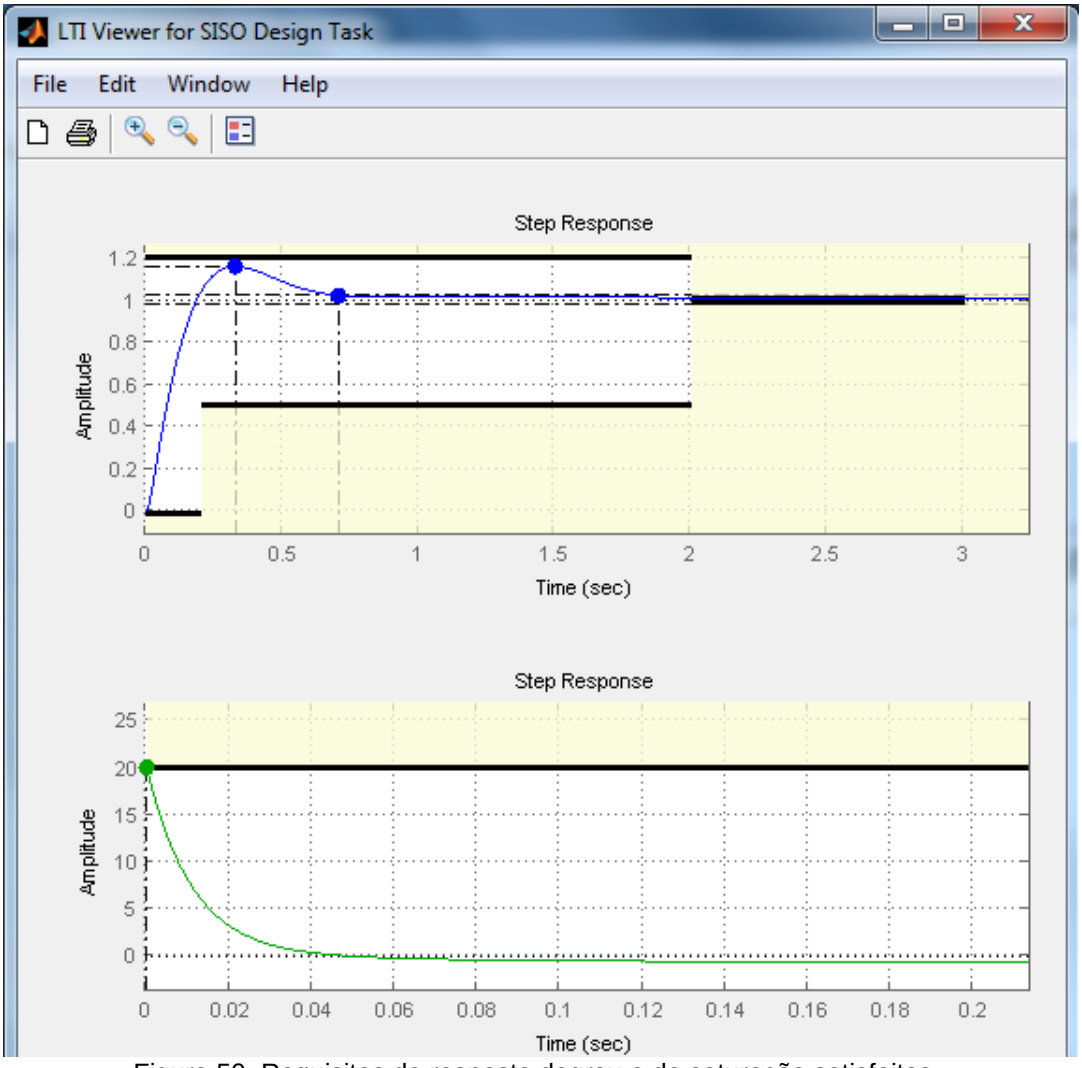


Figura 50. Requisitos da resposta degrau e da saturação satisfeitos.

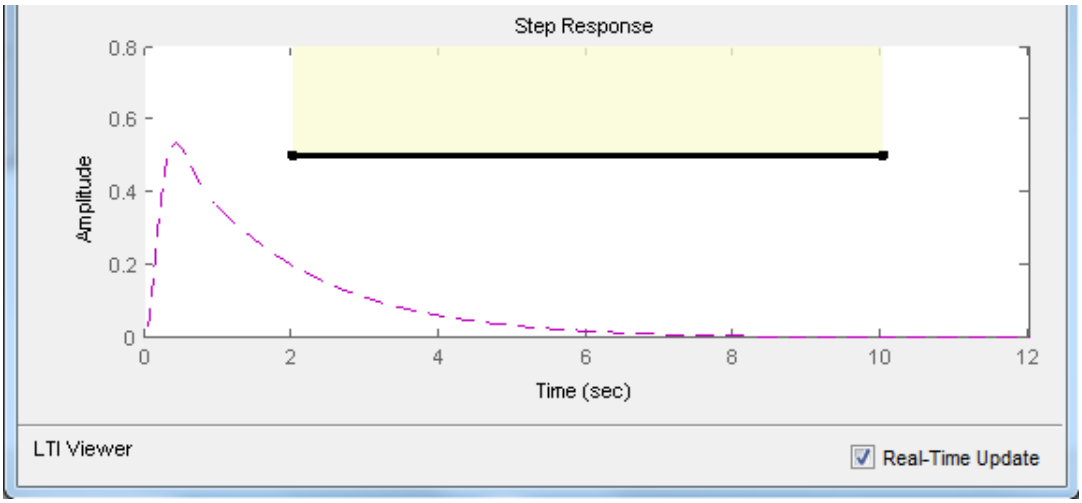


Figura 51. Requisito do erro estático devido ao atrito satisfeito.

Para calcular quais os ganhos do controlador, é possível exportar a função de transferência $C(s)$ selecionando SISO Design Task na árvore à esquerda, e usando o comando File-Export, selecionando Compensator C e, em seguida, Export to Workspace:

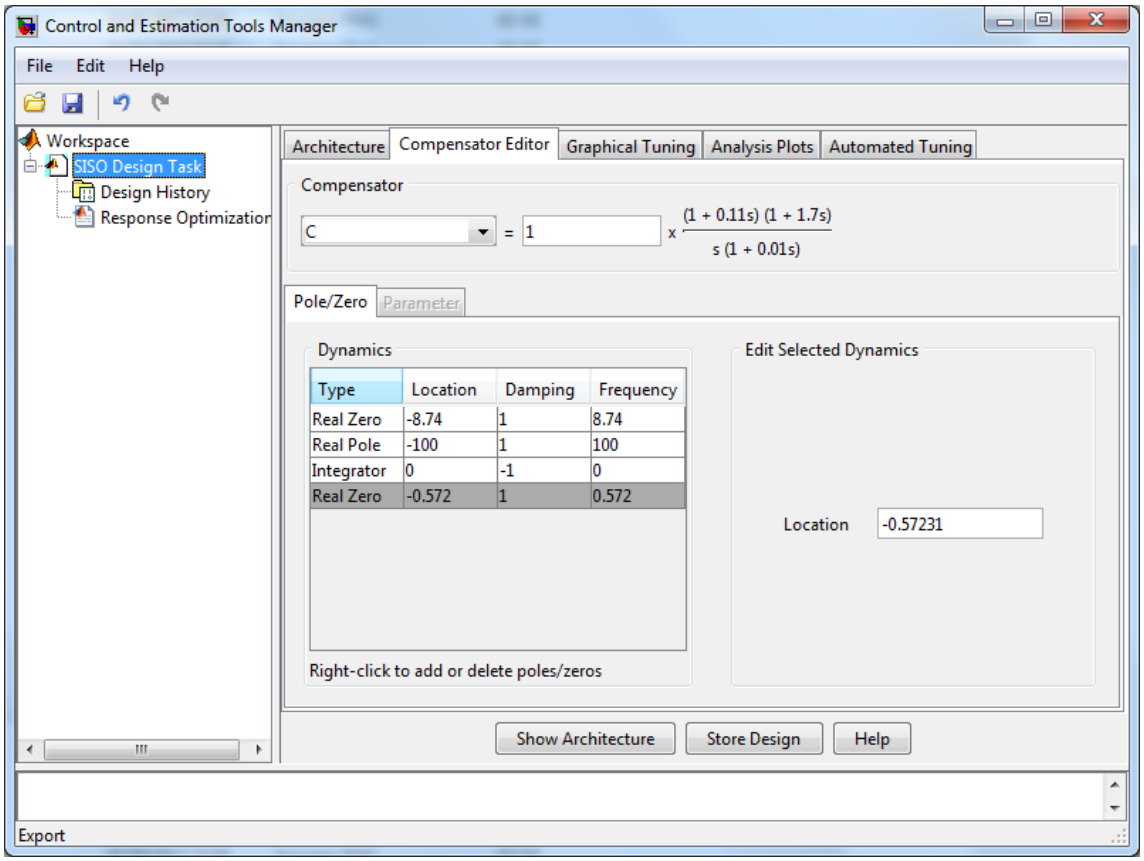


Figura 52. Retornando a janela principal do sisotool ao selecionar SISO Design Task na árvore à esquerda. É possível retornar à otimização selecionando Response Optimization.

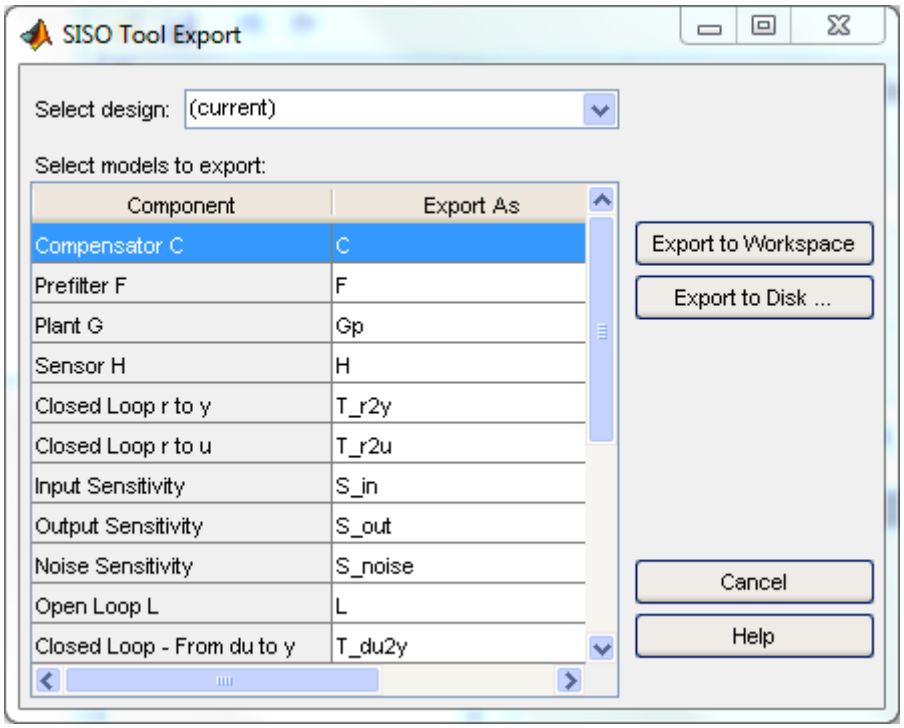


Figura 53. Exportando a função C.

Uma vez no ambiente matlab, é possível verificar:

```
tf(C) =
  20 s^2 + 186.2 s + 100
  ----------------------
       s^2 + 100 s
```

Recuperando a forma do controle, é possível calcular os ganhos:

$$C(s) = \frac{(P + DN_P)s^2 + (PN_P + I)s + IN_P}{s(s + N_P)}$$

```
I = 100/Np
P = (186.2-I)/Np
D = (20-P)/Np
```

$$I = 1, P = 1.8520 \; e \; D = 0.1815$$

É importante lembrar que o controle até então foi projetado sem levar em conta realmente o efeito do atrito, apenas considerando que o mesmo pode ser aproximado para um distúrbio. Dessa forma, é interessante testar os ganhos obtidos em um modelo simulink mais completo, “PID.mdl”, sendo que o bloco PID pode ser preenchido de forma simples usando apenas o nome das variáveis no workspace:

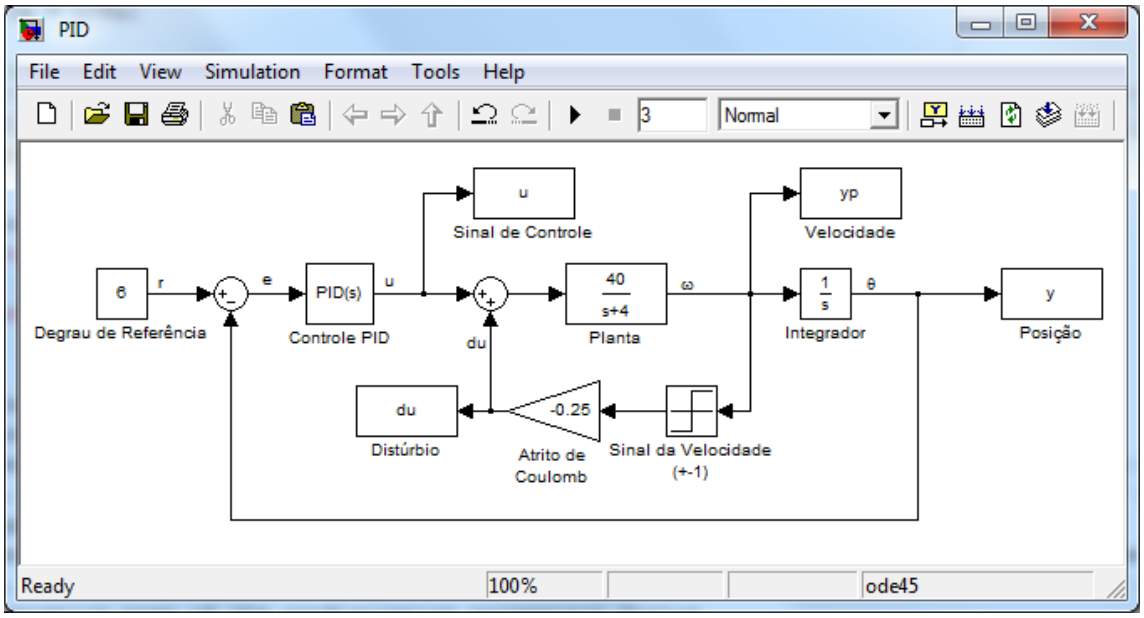


Figura 54. Modelo em simulink que deve ser utilizado para testar o controle projetado.

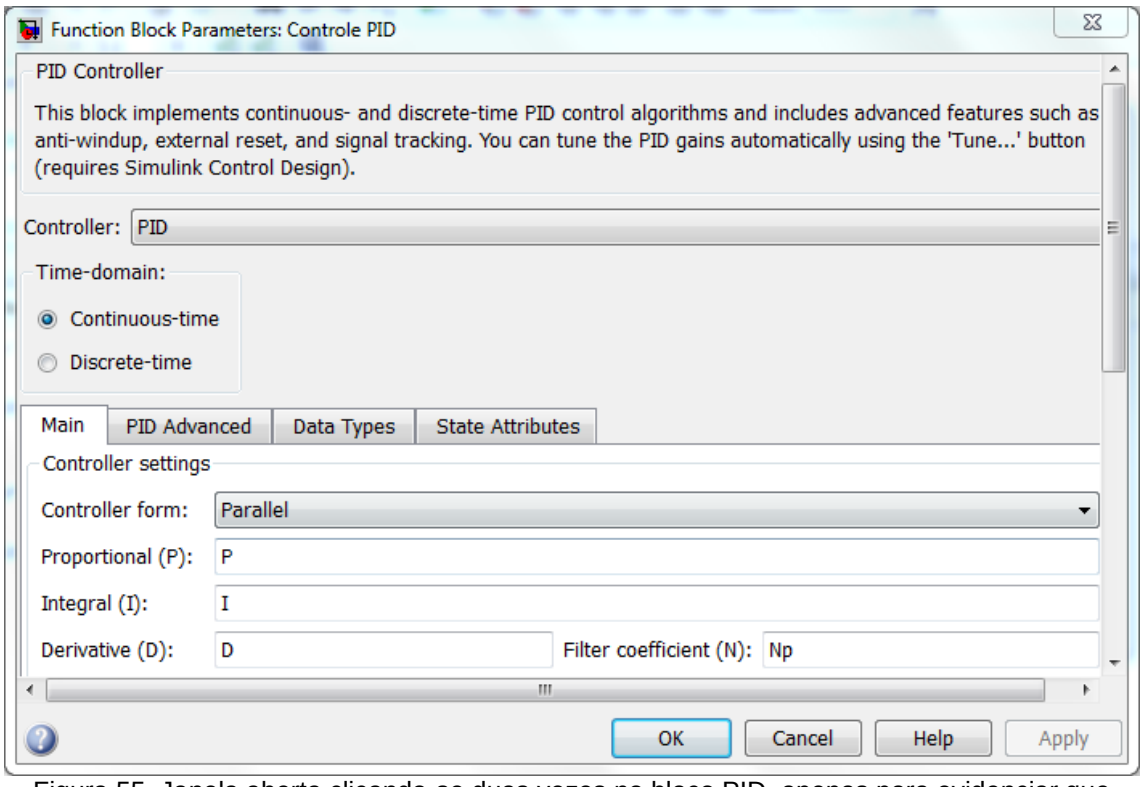


Figura 55. Janela aberta clicando-se duas vezes no bloco PID, apenas para evidenciar que basta especificar os parâmetros no workspace usando as variáveis P, I, D e Np, e os mesmos serão carregados automaticamente no simulink.

A simulação permite que os seguintes resultados sejam obtidos:

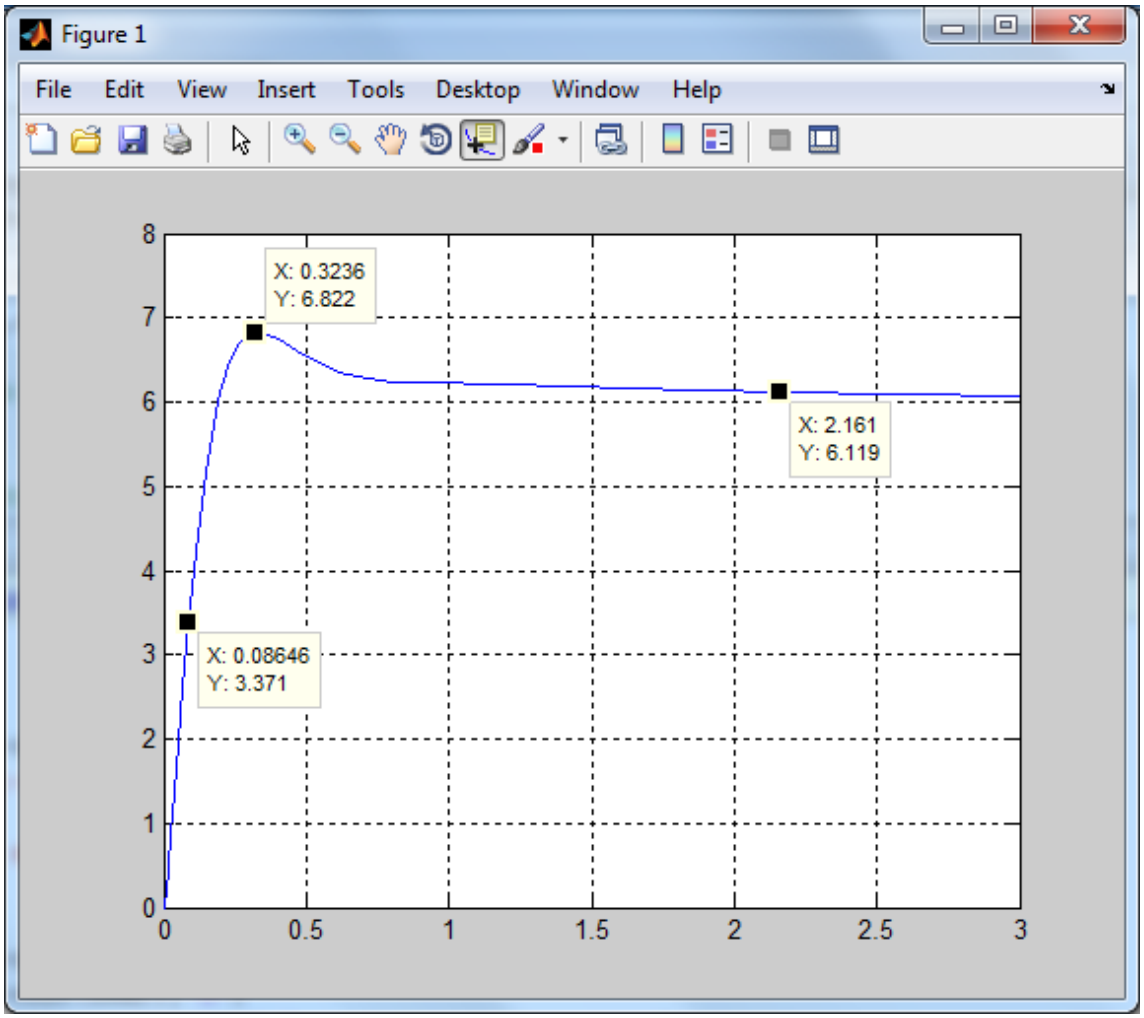


Figura 56. Resposta ao degrau, reflexo do gráfico r2y.

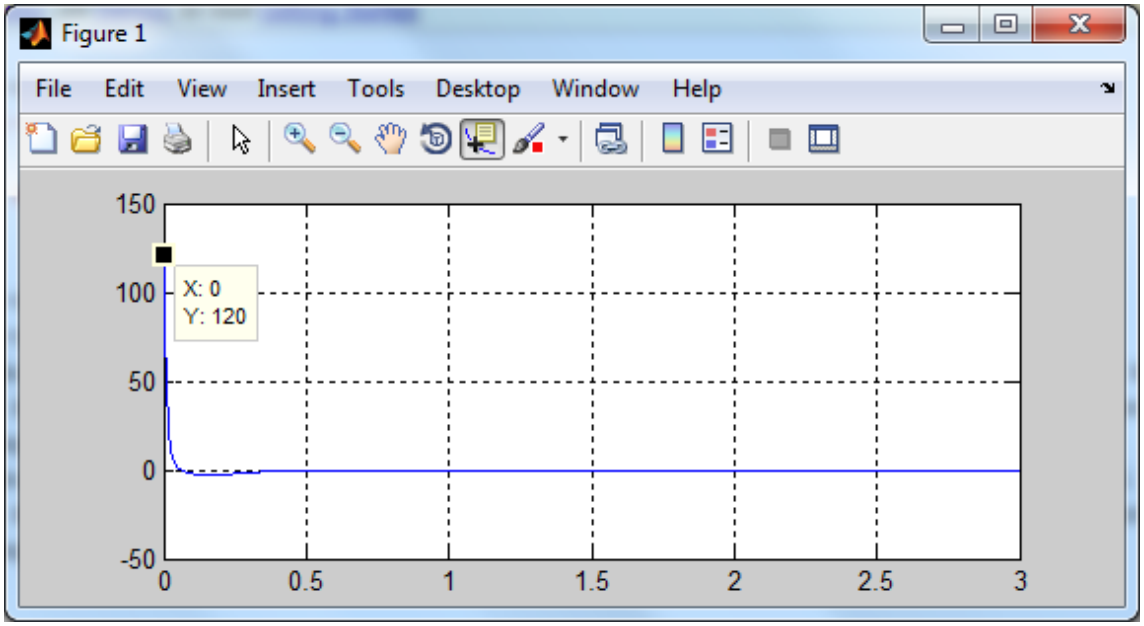


Figura 57. Valor do controle (input do sistema), reflexo do gráfico r2u.

Ou seja:

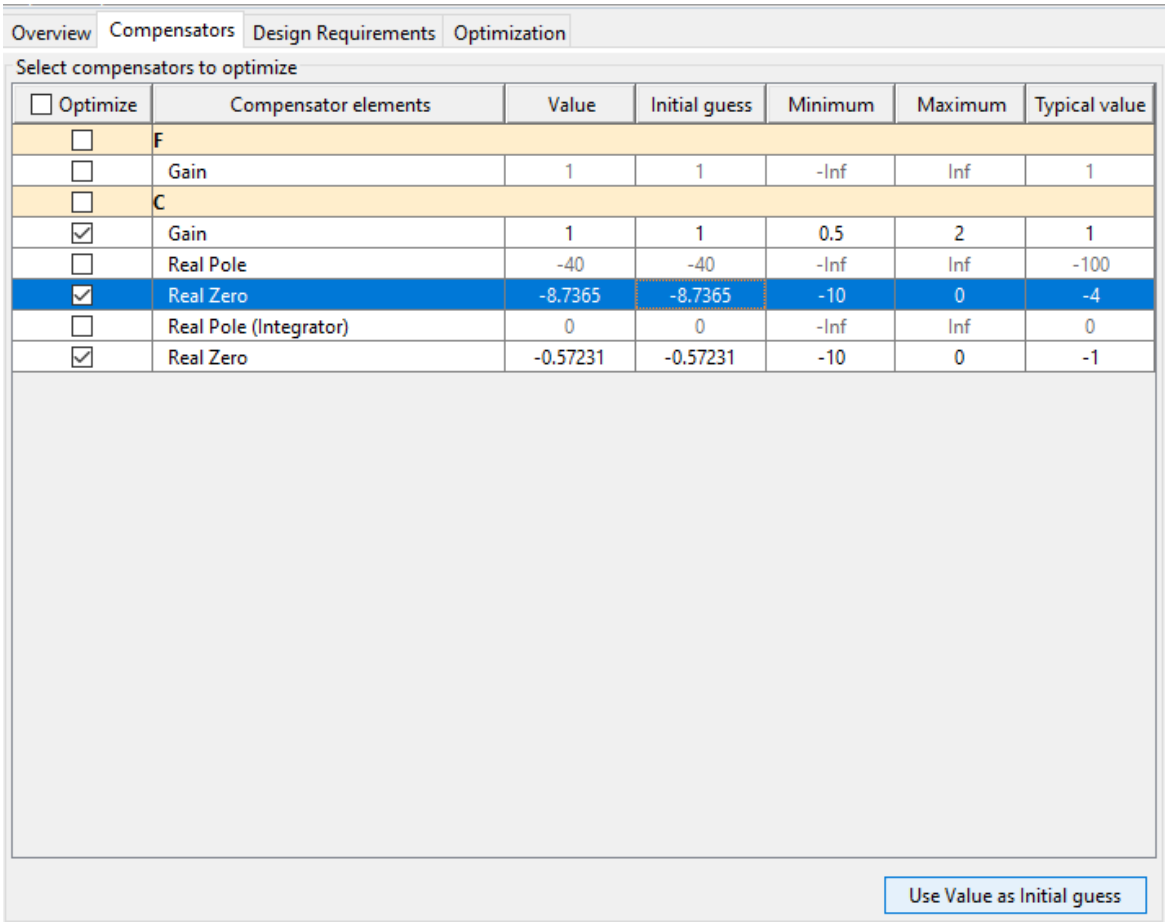
- Saturação de ± 120 respeitada;
- Sobressinal de 13.7%, ligeiramente menor que o projetado de 15.8%, e dentro do especificado de 20%;
- Tempo de subida de 50% de 0.08s, ligeiramente superior que o de 0.07 projetado e dentro do especificado de 0.2s;
- Erro menor que 2% em 2.16s, ligeiramente superior ao especificado de 2s;

A diminuição do sobressinal, e o aumento do tempo de subida e do tempo de assentamento são esperados dada a aproximação utilizada (o sinal do atrito como distúrbio deve ser sobreposto ao sinal em degrau r2y com sinal invertido, de forma que o pico diminui e o erro estático aumenta), mas é possível verificar que a aproximação permite obter um controlador que garante as especificações dentro de margens adequadas, mesmo incluindo efeitos como saturação e atrito (ou qualquer distúrbio em forma de degrau).

EXERCÍCIO 6

O projeto do controlador anterior foi feito utilizando $N_P = 100$, repita o projeto com o valor de N_P calculado no exercício 5, porém com saturação de ± 60 (lembre-se de converter a saturação pensando em um degrau de amplitude 6) e erro estático de 2% em 1s (lembrando de atualizar as restrições r2y e du2y).

Dado que o projeto do controlador prévio já está disponível, parta dos valores do mesmo para a otimização, o que pode ser feito clicando em Use Value as Initial Guess na aba Compensator do Optimization Tuning:



A opção de usar os valores anteriores pode ser interessante quando se usa uma estratégia de ir aumentando as restrições progressivamente, até que se encontre os limites possíveis do controlador.

Inclua uma imagem destacando as respostas em degrau do controle encontrado. Em seguida, calcule os ganhos do controlador PID e implemente o mesmo em Simulink, verificando a concordância com o projetado.