UNIVERSIDADE TÉCNOLÓGICA FEDERAL DO PARANÁ

DEPARTAMENTO ACADÊMICO DE ENGENHARIA ELETRÔNICA

ELEC20 – S71 - (1/2024)

FÁBIO ZHAO YUAN WUANG - 2358310

VICTOR AUGUSTO DEL MONEGO – 2378345

Este relatório tem como propósito responder as questões feitas no trabalho final da disciplina Controle 1. A dupla Fábio/Victor possui número **p** igual a 5, e os enunciados já foram ajustados para tal:

A figura 01 abaixo representa o sistema a ser analisado para todas as questões:

Uma imagem contendo Diagrama

Descrição gerada automaticamente

Figura 1: Sistema a ser tratado

1. **Faça uma análise em malha aberta para este sistema, indicando:**
2. **Polos e zeros. A partir do diagrama de lugar das raízes, comente sobre a resposta temporal esperada em malha fechada dados os polos dominantes.**
3. **Utilizando um controlador proporcional determine utilizando o critério de Routh-Hurwitz, o ganho K do controlador, a partir do qual o sistema se torna instável. Discuta a estabilidade do sistema.**
4. **Qual o tipo de sistema e sua constante de erro associada. É possível reduzir o erro associado á metade, utilizando o controle proporcional sem que o sistema perca a estabilidade?**

O código abaixo foi utilizado para responder à questão 1:

|  |
| --- |
| % comandos iniciais  clear all  close all  clc  % declarando o sistema  s = tf('s');  sys = (**10**\***5**)/(s\*(s+**5**)\*(**0.1**\*s+**5**))  zpk(sys)  % polos e zeros  pole(sys)  %lugar das raízes  figure(**1**);  rlocus(sys)  %Critério de estabilidade de routh Hurwitz  K = **1**;  K\_crit = **27.49**;  sys\_fb = feedback((K\_crit)\*sys,**1**)  figure(**2**)  step(sys\_fb)  %Constante de erro  %sistema é do tipo 1 (possui polo na origem)  Ktst = K\_crit/**2**  Kv = **10**\*Ktst\***5**/(**5**\***5**)  errv = **1**/Kv |

Abaixo estão anexos os polos, e o diagrama de lugar das raízes.

Gráfico, Gráfico de dispersão

Descrição gerada automaticamente

Analisando o diagrama, e considerando o polo em zero e em -50, é de se esperar que o polo em -5 defina o ganho crítico do sistema, bem como as suas demais propriedades e, portanto, sua estabilidade, por estar mais próximo do eixo imaginário.

A obtenção do K crítico (K de Routh), foi feita pela tabela de routh, ilustrada no PDF de apoio “Trabalho Final – Obtenção do K de Routh”. Realizando os cálculos, o sistema se torna instável a partir de K = 27,5. Partindo deste ponto, a conclusão que podemos tirar é que o sistema possui uma boa margem de escalabilidade quando se trata de estabilidade, visto que existe uma boa janela de ganho que o sistema pode receber antes de se desestabilizar.

Os cálculos para este sistema deram um erro aproximado de 0,037, e considerando a redução deste erro pela metade utilizando o cálculo proporcional, o trecho de código abaixo ilustra esta possibilidade:

|  |
| --- |
| %Constante de erro  %sistema é do tipo 1 (possui polo na origem)  Ktst = K\_crit/**2**  Kv = **10**\*Ktst\***5**/(**5**\***5**)  errv = **1**/Kv  % Visto que Kv é proporcional ao Ktst, e  % Kv é inversamente proporcional ao errv  % então, o único jeito de reduzir o erro associado à  % metade é dobrando o Kv, isto é, temos de  % dobrar o Ktst. Portanto, se reduzimos o erro à metade  % temos de dobrar o Ktst (I). Tendo (I) em mente:  % Para controladores com Ktst<K\_routh/2,  % ao dobrar o valor de Ktst, garantimos a estabilidade  % já que 2\*Ktst<K\_routh, e, de (I)  % conseguimos reduzir o erro à metade.  % Para controladores com K\_routh/2 < Ktst < K\_routh,  % ao dobrarmos Ktst, afim de reduzir a metade  % o erro associado, não garantimos a  % estabilidade, já que K\_routh < 2\*Ktst. |

1. **Projetar um controlador utilizando resposta em frequência tal que a margem de fase seja de (50±1) °. Este controlador é de avanço ou atraso de fase. Justifique**

O código abaixo foi utilizado para responder à questão 02:

|  |
| --- |
| clear all  close all  clc  %declarando o sistema  s = tf('s');  sys = (**10**\***5**)/(s\*(s+**5**)\*(**0.1**\*s+**5**));  %ganho arbitrário  K = **11**;  %margem máxima usando a margem desejada  phi = **50** + **7**;  zpk(sys)  %diagrama de Bode do sistema descompensado  figure(**1**)  margin(K\*sys)  %determinação dos parâmetros de fase  alph = **10**^(**15.7**/**20**);  tau = **1**/(**0.1**\***2.85**);  %Sistema compensado com controlador de atraso de fase  D = K \* (tau\*s+**1**)/(alph\*tau\*s+**1**);  margin(D\*sys)  figure(**2**)  %teste de sobressalto  figure(**3**)  step(feedback(D\*sys,**1**))  stepinfo(feedback(D\*sys,**1**)) |

Abaixo, os gráficos de bode do sistema descompensado, bem como o sistema após a compensação de fase:

Gráfico, Gráfico de linhas

Descrição gerada automaticamente

O approach utilizado foi um controlador por atraso de fase, pois era cômodo que a porcentagem de overshoot não fosse tanto afetada. Tendo isso em mente, definindo um overhead de 7 graus para a margem desejada, e ajustando o ganho a fim de tornar a margem precisa chegamos em um valor de K = 11. A justificativa para utilizar um controlador de atraso de fase é ilustrada abaixo:

Gráfico, Gráfico de linhas

Descrição gerada automaticamente

A resposta ao degrau do sistema manteve um overshoot relativamente baixo de 20%, o que é um efeito característico de um controlador de atraso de fase

1. **Projete um controlador por alocação de polos utilizando o lugar das raízes tal que o tempo de assentamento seja menor que 0,2 segundos e a taxa de amortecimento seja de 0,5. Comente o projeto e justifique os resultados.**

O código abaixo foi utilizado para responder à questão 3:

|  |
| --- |
| clear all  close all  clc  %declaração do sistema  s = tf('s');  sys = (**10**\***5**)/(s\*(s+**5**)\*(**0.1**\*s+**5**));  %lugar das raízes  rlocus(sys)  grid on;  hold on;  % Polos desejados  polos\_desejados = [-**20** + **34.64**i, -**20** - **34.64**i];  plot(real(polos\_desejados), imag(polos\_desejados), 'rx', 'MarkerSize', **10**, 'LineWidth', **2**)  %matrizes na forma canônica de controlabilidade  A = [**0** **1** **0** ; **0** **0** **1** ; **0** -**25** -**5.5**];  B = [**0** ; **0** ; **1**];  C = [**50** **0** **0**];  %Operações matriciais para construção do sistema (slides aula 16)  P = [-**20**+**1**i\***34.64** -**20**-**1**i\***34.64** -**50.5**];  K= place(A,B,P);  I=eye(size(A));  MJ=C\*inv(-A+B\*K)\*B;  J=**1**/MJ;  %controlador feito  controller=ss(A-B\*K, B\*J,C,[]);  figure(**2**)  step(controller)  hold on;  grid on;  stepinfo(controller)  damp(controller) |

O PDF de apoio “Trabalho Final – Primeira Tentativa de Projeto de Alocação de Polos” ilustra o raciocínio (apresentado na aula 16) de projeto de controlador utilizando realimentação de estado. O código do Matlab apresenta um polo real mais próximo do polo mais distante do sistema base, para um melhor overshoot. Tendo estas coisas em mente, as imagens abaixo ilustram a resposta ao degrau, os dados de respostas, e o fator de amortecimento dos polos deste controlador.

Gráfico, Gráfico de linhas

Descrição gerada automaticamente

Tabela

Descrição gerada automaticamente

Analisando as informações apresentadas, vemos que o tempo de assentamento permanece menor que 0,2 segundos, e a taxa de amortecimento dos polos complexos conjugados se situa em 0,5, de acordo com o requisitado.

O projeto por Alocação de Polos requer um trabalho braçal maior do que os demais, porém calcular os polos necessários para se manter a característica necessitada possui uma chance de acerto maior quando se trata de tempo de assentamento e amortecimento, o que pode ser uma vantagem para o projetista.

1. **Projete um controlador tipo PID que tenha resposta temporal em malha fechada semelhante ao controlador da questão 3. Comente o projeto e justifique os resultados.**

O código a seguir foi utilizado para responder à questão 4:

|  |
| --- |
| clear all  close all  clc  %objetivo:  %atingir as especificações do exercicio anterior  %overshooting = 10.6%  %settling time = 0.161s  %Rise time = 0.052  %declarando o sistema  s=tf('s');  sys = (10\*5)/(s\*(s+5)\*(0.1\*s+5));  %Tentativa 1:  %ver a freq de cruzamento  margin(sys)  figure(1)  %frequência de cruzamento de fase obtida observando o gráfico  wcf = 15.8;  %achar o K utilizando G  G = -28.8;  K = 10^(-G/20);  %comparando com o K\_routh, foi decidido utilizar  %o próprio K\_routh  Kr = 27.49;  Tu = 2\*pi/wcf;  %Parâmetros tabelados da tabela de Ziegler-Nichols  Kp = 0.6\*Kr;  Ki = 2\*Kp/Tu;  Kd = Kp\*Tu/8;  C = Kp + Ki/s + Kd\*s;  %fechamento da malha  sys\_fb\_c = feedback(C\*sys, 1);  sys\_fb = feedback(sys, 1);  sys\_crit\_fb = feedback(sys\*(Kr), 1);  %comparação entre a malha fechada do  %sistema sem o controlador (vermelho)  %e com o controlador  step(sys\_fb, "red")  hold on  step(sys\_fb\_c)  %por este método, podemos ver que o  %overshooting e o tempo de assentamento  %não foram cumpridas.  %Por isso, decidimos criar um pseudo-PID  %e, das características do pseudo-PID  %achar uma expressão para um PID que  %tenha um comportamento prox do pseudo  %tentativa 2:  %utilizando controlSystemDesigner() e na  %tentativa e erro  %foi possível chegar na seguinte expressão  %para o pseudo-PID com as especificações desejadas:  %Cm = 0.0075205\*(0.2\*s+1)\*(1+2\*10^3\*s)/(s\*(1+0.001\*s))  %utilizando Cm como base, foi possível projetar  % o seguinte PID:  a1 = 3.008\*1000;  a2 = 15.04\*1000;  a3 = 0.007521\*1000;  b1 = 1000;  %a1, a2, a3, b1 são coeficientes dos polinomios  %do numerador e do denominador de Cm  %que quando escritos em termos de Kp, Ki, Kd:  Kpm = (-a3+a2\*b1)/(b1^2);  Kim = a3/b1;  Kdm = (a3-a2\*b1+a1\*(b1^2))/(b1^3);  %daqui, equacionamos o PID:  Cm1 = Kpm + Kim/s + Kdm\*s;  sys\_fb\_cm1 = feedback(Cm1\*sys, 1);  %comparação gráfica das duas malhas fechadas  %PID <- azul  %controlador ex3 <- vermelho  figure()  step(sys\_fb\_cm1)  hold on  A = [0 1 0 ; 0 0 1 ; 0 -25 -5.5];  B = [0 ; 0 ; 1];  C = [50 0 0];  P = [-20+1i\*34.64 -20-1i\*34.64 -50.5 ];  K= place(A,B,P);  MJ=C\*inv(-A+B\*K)\*B;  J=1/MJ;  controller=ss(A-B\*K, B\*J,C,[]);  step(controller, "red") |

Abaixo, temos a resposta ao degrau do sistema PID comparada com o sistema de alocação de polos, que está em vermelho, e suas informações de degrau:

Gráfico, Gráfico de linhas

Descrição gerada automaticamente

Texto

Descrição gerada automaticamenteTexto

Descrição gerada automaticamente

Baseando-se no gráfico e dados como fontes de análise, percebemos que o controlador PID consegue estabelecer uma performance ainda melhor que o controlador por alocação de polos, visto que ele apresenta tanto overshoot quanto tempo de assentamento menor. O ponto negativo é que como o próprio código mostra, o controlador PID é extremamente sensível a alterações, e a manipulação de cada parte do controlador acaba sendo um desafio de projeto, e sendo um custo alto pela performance apresentada.

1. **Compare o desempenho dos 3 controladores acima e indique as dificuldades/facilidades dos projetos.**

Analisando os 3 controladores, podemos tomar algumas observações:

* O controlador por compensação de fase, possui um overshoot relativamente baixo, mesmo que não o mais baixo dos 3, e um longo tempo de assentamento. Para aplicações em que o sistema precisaria atingir o regime permanente com agilidade, não seria interessante utilizar este modelo. Porém quando o objetivo é manter a margem de fase, é uma boa alternativa pois o controlador consegue compensar a fase do sistema sem causar alterações drásticas no sobrepasso da resposta do sistema.
* O controlador por PID é o controlador mais flexível de todos os controladores, visto que cada “parte” do controlador pode ser ajustada para as devidas características de projeto necessárias. A desvantagem deste controlador se apresenta na sua estabilidade, ou ao menos a sensibilidade de sua estabilidade. Modificar as partes do controlador para atingir certos patamares torna-se um desafio para o projetista. Mesmo atingindo uma precisão maior que o controlador por alocação de polos, o PID possui um grau de liberdade tão alto, com tantas variáveis, que sua projeção torna-se em certo ponto instável, viabilizando o uso de algo como um controlador de alocação de polos.
* O controlador por alocação de polos possui a vantagem de ter um overshoot e tempo de assentamento baixos em uma boa estabilidade, mesmo que não tão baixos quanto os parâmetros obtidos pelo PID. De todos, é o controlador mais preciso nestes dois quesitos, o que o torna eficiente para aplicações precisas por alcançar patamares de precisão altos, e ser estável. O custo desta eficiência é que o projeto do controlador exige uma quantidade considerável de trabalho técnico, em comparação com os outros.

Em conclusão, a aplicação do controlador evidentemente impacta no nível de precisão e construção que deve ser investido.

**OBS:** Os PDFs de apoio mencionados no relatório acima serão enviados em anexo com os códigos do Matlab, e o Relatório em si.