UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
CENTRO TECNOLÓGICO
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
SISTEMAS REALIMENTADOS

ALAF DO NASCIMENTO SANTOS
FELIPE ANTONIO MOREIRA SILVA

SEGUNDO TRABALHO COMPUTACIONAL DE SISTEMAS REALIMENTADOS

VITÓRIA - ES

MARÇO
2022

Alaf do Nascimento Santos
Felipe Antonio Moreira Silva

SEGUNDO TRABALHO COMPUTACIONAL DE SISTEMAS REALIMENTADOS

Parte manuscrita do Segundo Trabalho Computacional de Sistemas Realimentados dos alunos Alaf do Nascimento Santos e Felipe Antonio Moreira Silva, apresentado ao Prof. Dr. Jose Leandro Felix Salles, como requisito parcial para aprovação na disciplina.

Vitória - ES

Março
2022

LISTA DE FIGURAS

SUMÁRIO

1 ESPECIFICAÇÃO

1.1 Especificação do Grupo

- Nome: Alaf do Nascimento Santos. Matrícula: 2017100781
- Nome: Felipe Antonio Moreira Silva. Matrícula: 2018205316
- Número do Grupo: 9

Segundo Trabalho Computacional de Sistemas Realimentados

Componentes do Grupo (no máximo 2 alunos):
Número do Grupo (N):
Data da entrega do trabalho no Google Classroom: até 25/03

1 - A dinâmica de um veículo lunar (ver Fig. P12.10 pag 571 do livro do Dorf e Bishop) é dada por:

$$G = \frac{100N}{(s+(25-N))(s+\frac{25}{N})} e^{-(T)s}$$

1.1 Projetar um controlador PID, usando resposta em frequência, para que o sistema tenha as seguintes especificações:

Erro à entrada degrau e ao distúrbio de degrau menores ou iguais a 0,1, largura de banda da função de transferência em malha aberta maior possível e margem de fase maior ou igual a 60 graus .

Faça o projeto para os seguintes casos :

i) T=0 seg. (sem atraso de transporte), e
ii) T=0.1/N seg.

Analise os dois casos mencionados acima e discuta as diferenças.

OBS: faça o projeto para o caso (ii) sem usar a aproximação de padé, considerando o comando do matlab g=tf(num,den,'InputDelay', T); onde num e dem são o numerador e o denominador da FT sem atraso de transporte.

1.2 Para o caso em que T=0 seg, multiplique o controlador projetado pela FT ***G*** do veículo lunar e determine a equação de estados deste sistema em malha aberta. Em seguida, desenvolva um código no matlab, semelhante ao fornecido para simular o avião (programa "simulaviao.m" fornecido em anexo), para simular o sistema de controle de direção do veículo lunar em malha fechada no espaço de estados. Considere nas simulações os seguintes casos:

i) Existe na saída do sistema de controle um ruído graussiano (comando "rand") no sensor do veículo lunar.
ii) Não existe na saída do sistema de controle o ruído de medição.

Analise os dois casos mencionados acima e discuta as diferenças.

1.3 Compare a resposta ao degrau obtida usando o simulador desenvolvido no item 1.2 (sem o ruído de medição), com a resposta ao degrau (sobressinal, tempo de subida) obtida no item 1.1 para T=0 s.

1.4 – Repita os exercícios 1.1, 1.2, e 1.3 anteriores para o controlador Atraso-Avanço de fase.

1.5 - Compare as respostas à entrada degrau (sobressinal, tempo de subida) e o erro em regime às entradas degrau para os dois sistemas com os controladores obtidos nos itens 1.1 e 1.4 para T=0.0 s.

1.6 – Considerando o item 1.5, escolha o melhor controlador obtido e determine os polos da FT em malha fechada. A partir destes polos, projete um controlador usando a realimentação de estados para controlar a dinâmica deste sistema.
.
1.7 - Compare as respostas ao degrau e o respectivo erro à entrada degrau dos sistemas realimentados projetados no 1.1 e 1.4 para T=0.0 s com o sistema realimentado projetado no item 1.6. A pontuação até o item 1.7 vale 10.

1.8 – Desenvolver um projeto do observador de estados para o sistema realimentado projetado no item 1.6. Simule este sistema em malha fechada considerando realimentação de estados, e compare o seu desempenho com as respostas à entrada degrau obtidas no item 1.7. (ponto adicional)

2 Considerações sobre os Projetos

A execução destes projetos deve seguir os seguintes passos:

1. Desenvolver o controlador usando a resposta em frequência e realimentação de estados, através do Matlab. Aplicar o controlador desenvolvido ao modelo da planta e simular a resposta em malha fechada, verificando se as especificações são atendidas;
2. Caso não satisfaça, refazer os passos 1 e 2.

3 O Relatório final escrito, feito em grupo de até dois alunos, deverá ser entregue até a data especificada e deverá conter

Projeto do Controlador:

- Descrever o passo a passo dos projetos baseados em respostas em frequências e espaço de estados, mostrando as equações utilizadas para a escolha dos parâmetros dos controladores;
- FT do controlador e FT de malha fechada (planta+controlador) utilizadas para a simulação;
- Comandos do Matlab usados para simulação;
- Resultados da simulação no Matlab do controlador+planta, apresentando os gráficos dos sinais de saída no domínio do tempo e da frequência, especificando o sobressinal, o tempo de subida, o erro em regime as margens de ganho e de fase, a largura de banda obtidas com o controlador projetado.

2 ITEM 1.1

Para o primeiro item do trabalho computacional, foi solicitado a realização do projeto de um controlador PID, usando resposta em frequência, para que o sistema tenha as seguintes especificações:

- Erro à entrada degrau e ao distúrbio de degrau menores ou iguais a 0,1;
- Largura de banda da função de transferência em malha aberta maior possível;
- Margem de fase maior ou igual a 60 graus.

Além disso, foi fornecido que a função de transferência que representa a dinâmica de um veículo lunar em malha aberta é:

$$G = \frac{100N}{(s + (25 - N))(s + \frac{25}{N})} e^{-(T)s}$$

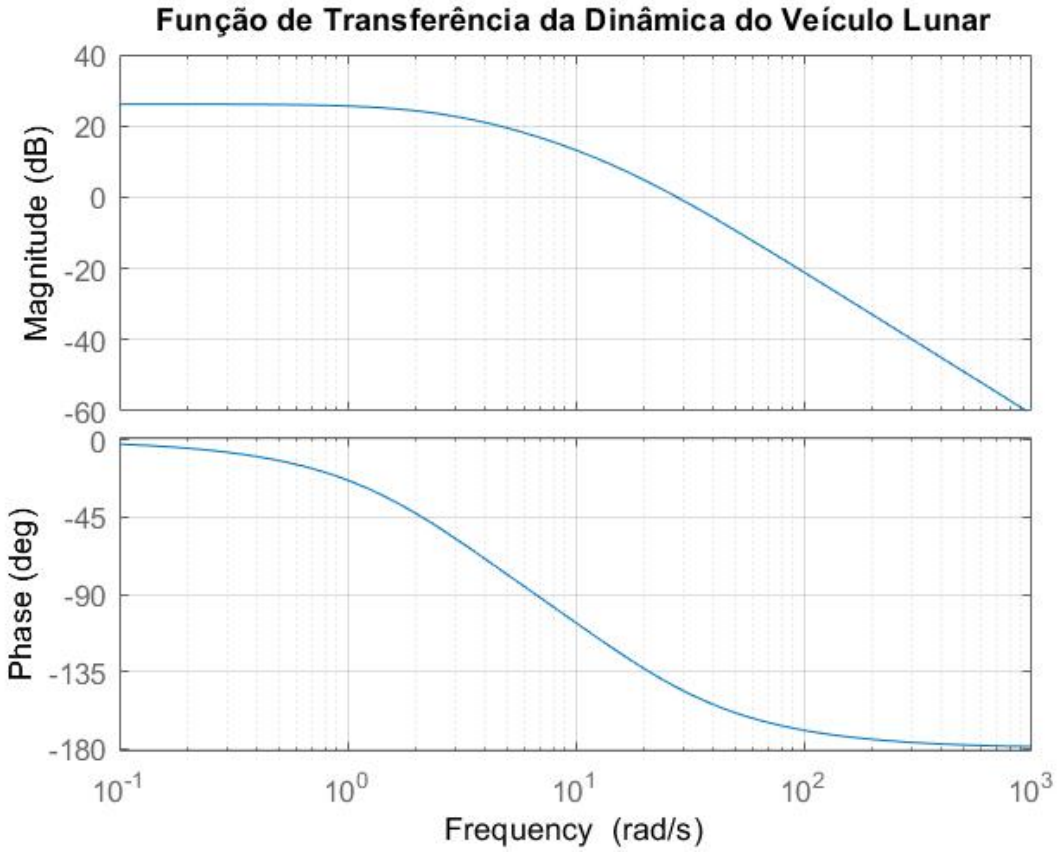
Deste modo, foi realizado o projeto para dois casos distintos da função de transferência:

- Sem atraso de transporte, $T = 0seg$;
- Com atraso de transporte, $T = \frac{0,1}{N} seg$;

2.1 Primeiro caso (sem atraso de transporte)

Para o primeiro caso (sem o atraso), iniciamos o desenvolvimento implementando os dados básicos do sistema, como a função de transferência em malha aberta, e definindo o valor de $N = 9$. Após esta primera implementação, realizamos o diagrama de Bode da FTMA para análises posteriores.

Figura 1 – Diagrama de Bode da FTMA



Fonte: Produção do próprio autor.

Sendo assim, para o projeto do controlador PID, iniciamos com a etapa de determinação da GM, PM, frequências de cruzamento de ganho e de fase e a largura de banda da planta dada pela função de transferência Gma, nos quais obtemos:

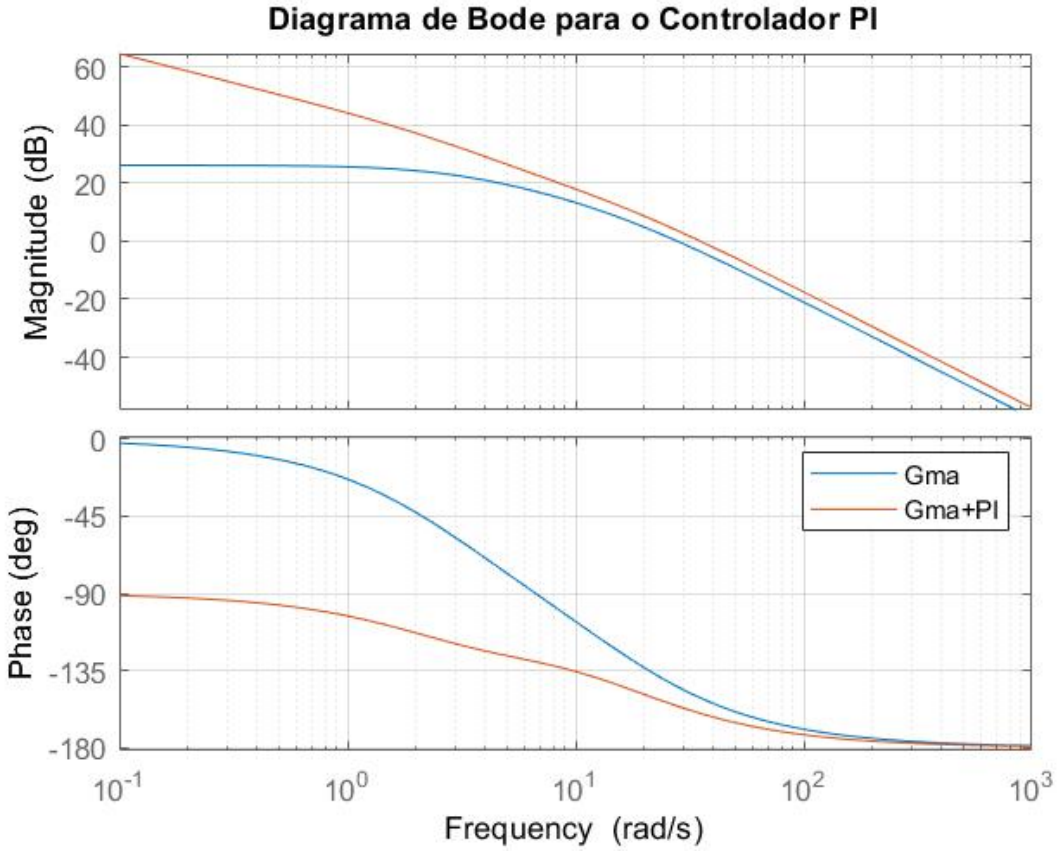
- GM = Inf db;
- PM = 35.5531 graus;
- Frequência de cruzamento de ganho = Inf rad/s ;
- Frequência de cruzamento de fase = 27.8687 rad/s ;

Após essa etapa, obtivermos que a nova frequência de cruzamento de ganho, dada por ω_{0dB} através da relação 25° = 180° + Fase de GP ω_{0dB}, será de 0.6694 rad/s. Deste modo, calculamos o ganho proporcional do PI (KPI) tal que que GMA = GP × KPI tenha a margem de fase aproximadamente igual a 25° na nova frequência de cruzamento de ganho

(ω_{0dB}), ou seja, escolher KPI tal que $20log(K_p) + 20log(Gp_{\omega_{0dB}})$. Portanto, temos que o ganho Kpi será de 1.4938.

Para proxima etapa do projeto, escolhemos a frequência de corte do PI tal que o atraso de fase do PI ocorra um pouco abaixo da nova frequência de cruzamento de ganho, sendo $K_{ii} = 8.3260$. Deste modo, temos o nosso controlador PI e podemos realizar sua simuação com a planta a fim de verificar se o projeto é satisfeito.

Figura 2 – Diagrama de Bode da Gma com Controlador PI



Fonte: Produção do próprio autor.

Após obitido o controlador integrativo, podemos projetar o PD dado por 1 + sKdd considerando a FTMA de GMA = GP × [Kpi × (Kiis)] e que a MF especificada seja maior ou igual a 60° na nova frequência de cruzamento de ganho obtida anteriormente. Isto nos dá uma frequência de corte do PD de $K_{dd} = 0.0285$. Sendo assim, temos ao final, um controlador PID resultante:

$$PID = \frac{0.04251s^2 + 1.731s + 8.326}{s}$$

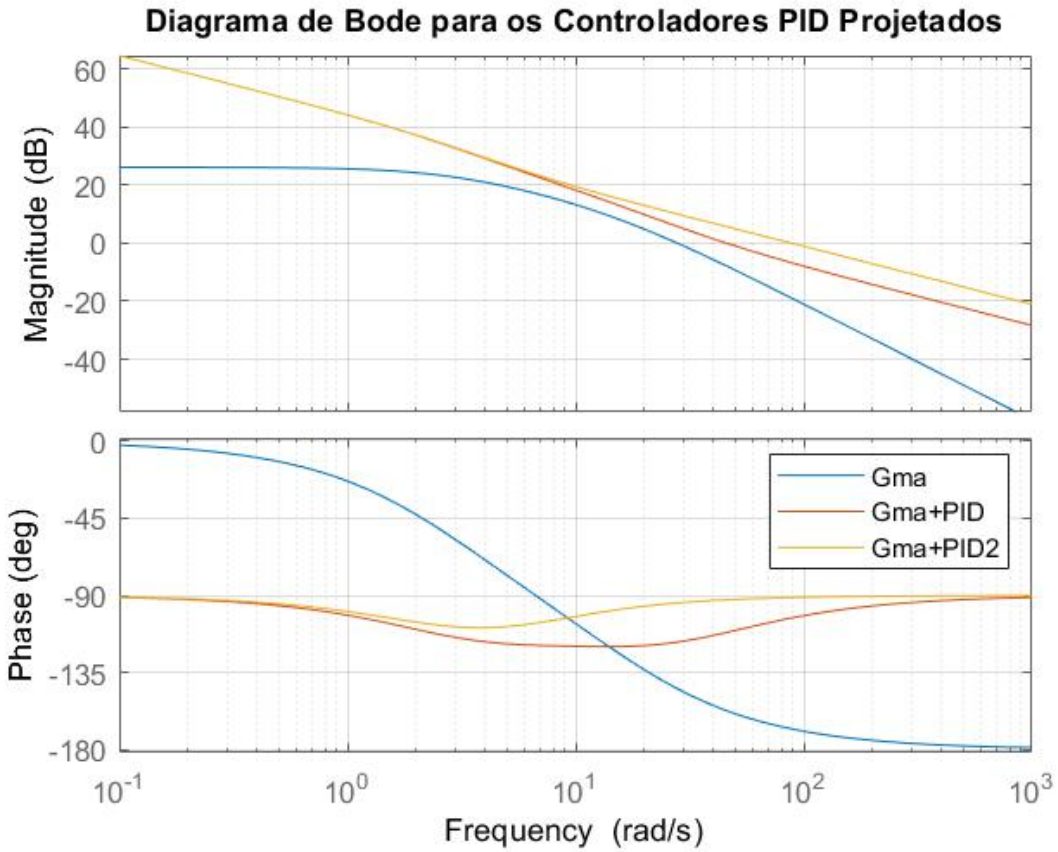
Antes de realizar as simulações e análises da resposta ao degrau em malha fechada, realizamos a repetição do procedimento anterior utilizando como parâmetro no cálculo

de Kdd, o valor da frequência de corte do PID anterior. Deste modo temos um novo controlador PID:

$$PID_2 = \frac{0.09789s^2 + 2.039s + 8.326}{s}$$

Sendo assim, temos as simulações de malha aberta e fechada para realizar as análises necessárias para escolher o melhor controlador ou realizar um novo projeto.

Figura 3 – Diagrama de Bode da FTMA com Controladores PID projetados



Fonte: Produção do próprio autor.

Figura 4 – Tabela realizada no Matlab para comparar os diferentes projetos de controladores

T =

4×10 table

FTMAs	Controladores	Overshoot	SettlingTime	RiseTime	Gms	BWs	MFs	Ess_Degrau	Ess_Disturbio
'Gma'	'0'	36.377	37.496	4.3955	Inf	27.869	35.553	0.047059	0.95294
'GGma'	'PI'	58.825	56.334	3.2733	Inf	35.143	19.986	0	0
'GGGma'	'PID'	13.217	16.384	3.193	Inf	45.78	68.284	0	0
'GGGma2'	'PID2'	1.9639	4.7675	2.3946	Inf	88.106	88.653	0	0

Fonte: Produção do próprio autor.

Figura 5 – Resposta ao Degrau de Malha Fechada com PI gerada pelo comando stepinfo

```
struct with fields:

     RiseTime: 3.2733
  SettlingTime: 56.3338
   SettlingMin: 0.6882
   SettlingMax: 1.5881
     Overshoot: 58.8246
    Undershoot: 0
          Peak: 1.5881
      PeakTime: 10
```

Fonte: Produção do próprio autor.

Figura 6 – Resposta ao Degrau de Malha Fechada com PID gerada pelo comando stepinfo

```
ans =

  struct with fields:

       RiseTime: 3.1930
    SettlingTime: 16.3843
     SettlingMin: 0.9740
     SettlingMax: 1.1322
       Overshoot: 13.2170
      Undershoot: 0
            Peak: 1.1322
        PeakTime: 8
```

Fonte: Produção do próprio autor.

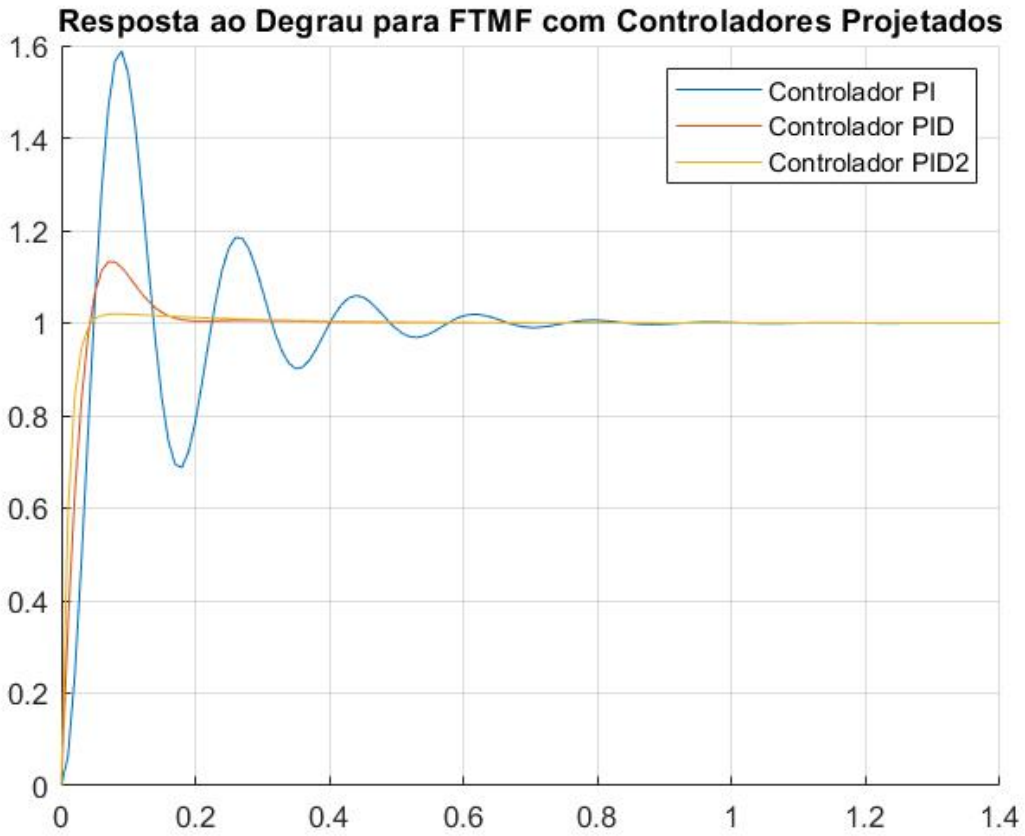
Figura 7 – Resposta ao Degrau de Malha Fechada com PID2 gerada pelo comando stepinfo

```
ans =

  struct with fields:

       RiseTime: 2.3946
    SettlingTime: 4.7675
     SettlingMin: 0.9460
     SettlingMax: 1.0197
       Overshoot: 1.9639
      Undershoot: 0
            Peak: 1.0197
        PeakTime: 9
```

Fonte: Produção do próprio autor.

Portanto, ao analisar as Figuras 4, 5, 6, 7 e 8, podemos determinar que o melhor controlador PID foi o segundo (PID_2), pois possui uma resposta rápida, um sobressinal baixo, erros ao degrau e ao distúrbio nulo e atender às especificações iniciais do projeto.

Figura 8 – Resposta ao Degrau de Malha Fechada com os Controladores projetados

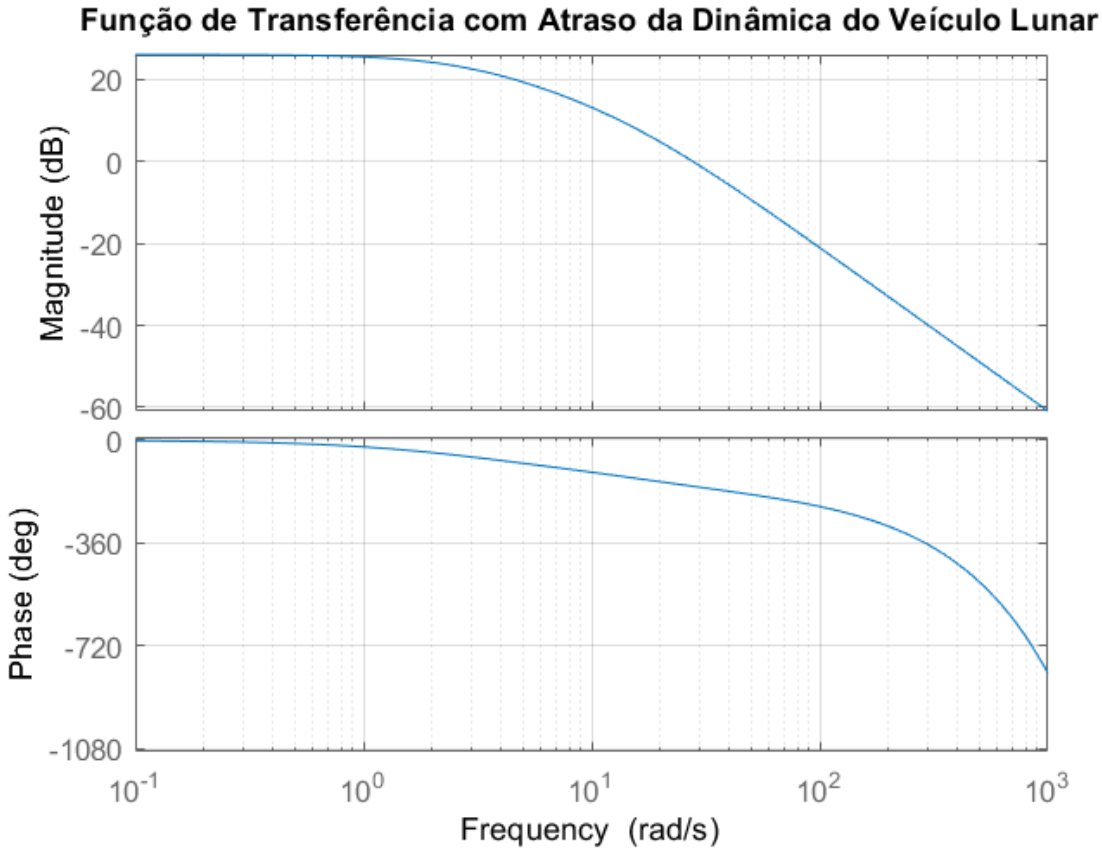


Fonte: Produção do próprio autor.

2.2 Segundo caso (com atraso de transporte)

Já para o segundo caso (com o atraso), iniciamos o desenvolvimento do segundo projeto implementando novamente os dados básicos do sistema, como a nova função de transferência em malha aberta, e definindo o valor de $N = 9$. Após essa primera implementação, realizamos o diagrama de Bode da FTMA com atraso para análises posteriores.

Figura 9 – Diagrama de Bode da FTMA com atraso de transporte



Fonte: Produção do próprio autor.

Sendo assim, para o projeto do controlador PID deste segundo caso, iniciamos com a etapa de determinação da GM, PM, frequências de cruzamento de ganho e de fase e a largura de banda da planta dada pela função de transferência Gma com atraso de transporte, nos quais obtivemos:

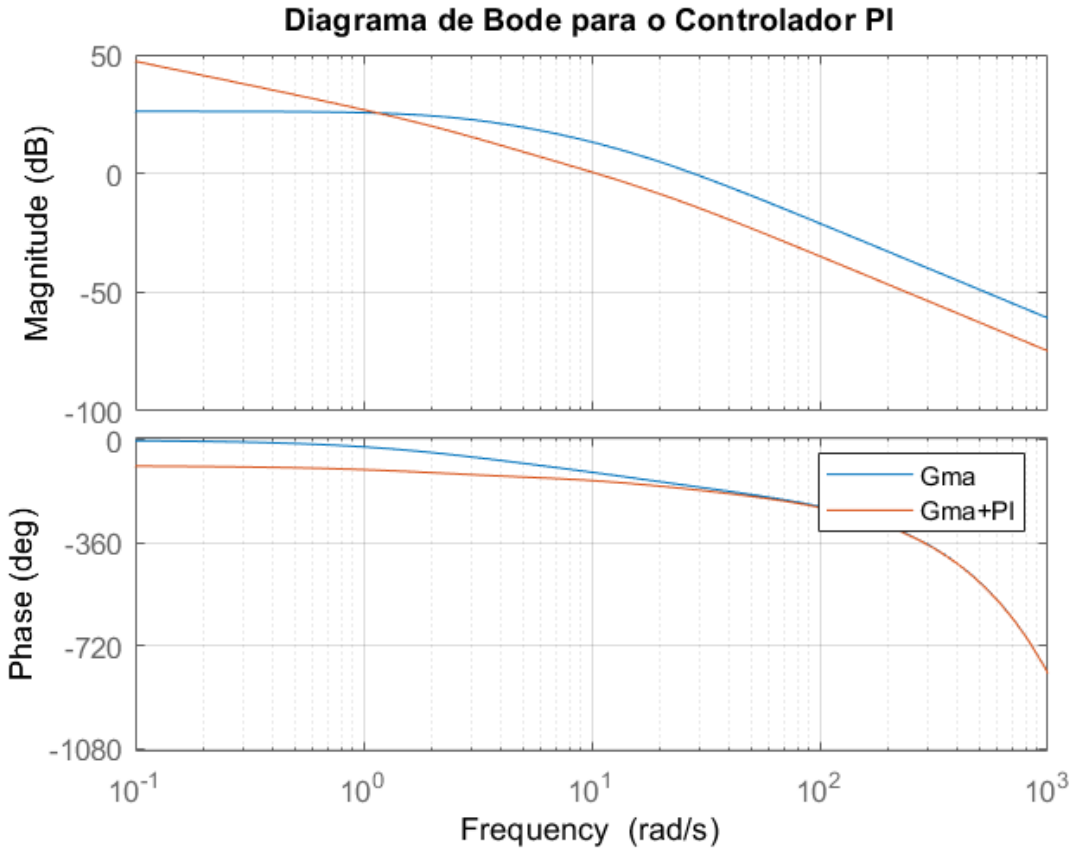
- GM = 1.9419 db;
- PM = 17.8114 graus;
- Frequência de cruzamento de ganho = 40.2519 rad/s ;
- Frequência de cruzamento de fase = 27.8687 rad/s ;

Após essa etapa, temos que a nova frequência de cruzamento de ganho ω_{0dB} será de 0.4912 rad/s. Deste modo, calculando o ganho proporcional do PI (KPI) assim como para o Gma sem atraso, obtivemos um ganho Kpi de 2.0360, porém, quando se utiliza tal valor de ganho, a função de transferência em malha fechada apresenta instabilidade e impossibilita

a continuação das análises, portanto, determinamos que o valor de Kpi fosse menor que o calculado, ou seja, Kpi = 0.2036 (10% do Kpi teórico).

Dando continuidade ao projeto, escolhemos a frequência de corte do PI tal que o atraso de fase do PI ocorra um pouco abaixo da nova frequência de cruzamento de ganho, sendo $K_{ii} = 1.1348$. Deste modo, obtivemos o nosso controlador PI e realizamos sua simuação com a planta a fim de verificar se o projeto é satisfeito.

Figura 10 – Diagrama de Bode da Gma com atraso de transporte com Controlador PI



Fonte: Produção do próprio autor.

Após obitido o controlador integrativo anteriormente, podemos realizar o projeto do PD de mesmo modo que o caso anterior, o que nos dá uma frequência de corte do PD de $K_{dd} = 0.0954$. Sendo assim, temos ao final, um controlador PID resultante:

$$PID = \frac{0.01942s^2 + 0.3118s + 1.135}{s}$$

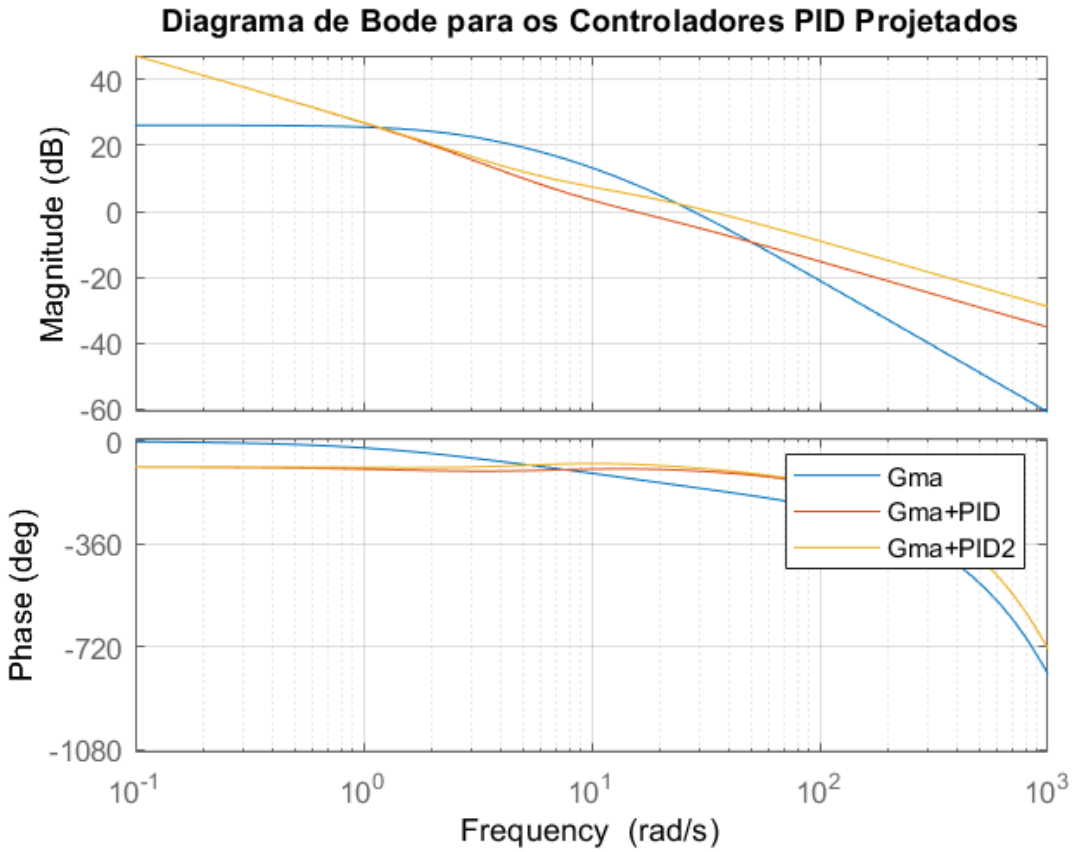
Antes de realizar as simulações e análises da resposta ao degrau em malha fechada, realizamos a repetição do procedimento anterior utilizando como parametro do cálculo de Kdd, o valor da frequência de corte do PID anterior. Deste modo temos um novo

controlador PID:

$$PID_2 = \frac{0.03983s^2 + 0.4256s + 1.135}{s}$$

Sendo assim, temos as simulações de Malha Aberta e Fechada de Gma com atraso de transporte para realizar as análises necessárias para escolher o melhor controlador ou realizar um novo projeto.

Figura 11 – Diagrama de Bode da FTMA com atraso de transporte com Controladores PID projetados



Fonte: Produção do próprio autor.

Figura 12 – Tabela realizada no Matlab para comparar os diferentes projetos de controladores

FTMAs_atraso	Controladores_atraso	Overshoot_atraso	SettlingTime_atraso	RiseTime_atraso	Gms_atraso	BWs_atraso	MFs_atraso	Ess_Degrau_atraso	Ess_Disturbio_atraso
'Gma_atraso'	'0'	63.51	89.641	3.9017	1.9419	27.869	17.811	0.047059	0.95294
'GGma_atraso'	'PI'	36.392	88.732	10.538	6.5556	10.486	36.929	0	0
'GGGma_atraso'	'PID'	4.2607	62.528	11.206	8.2107	15.337	82.382	0	0
'GGGma2_atraso'	'PID2'	0.88603	29.535	10.85	4.1015	32.978	81.297	0	0

Fonte: Produção do próprio autor.

Figura 13 – Resposta ao Degrau de Malha Fechada com atraso de transporte e com PI gerada pelo comando stepinfo

```
ans =

  struct with fields:

      RiseTime: 10.5379
  SettlingTime: 88.7323
   SettlingMin: 0.9247
   SettlingMax: 1.3640
     Overshoot: 36.3920
    Undershoot: 0
          Peak: 1.3640
      PeakTime: 29
```

Fonte: Produção do próprio autor.

Figura 14 – Resposta ao Degrau de Malha Fechada com atraso de transporte e com PID gerada pelo comando stepinfo

```
ans =

  struct with fields:

      RiseTime: 11.2060
  SettlingTime: 62.5275
   SettlingMin: 0.9032
   SettlingMax: 1.0426
     Overshoot: 4.2607
    Undershoot: 0
          Peak: 1.0426
      PeakTime: 37
```

Fonte: Produção do próprio autor.

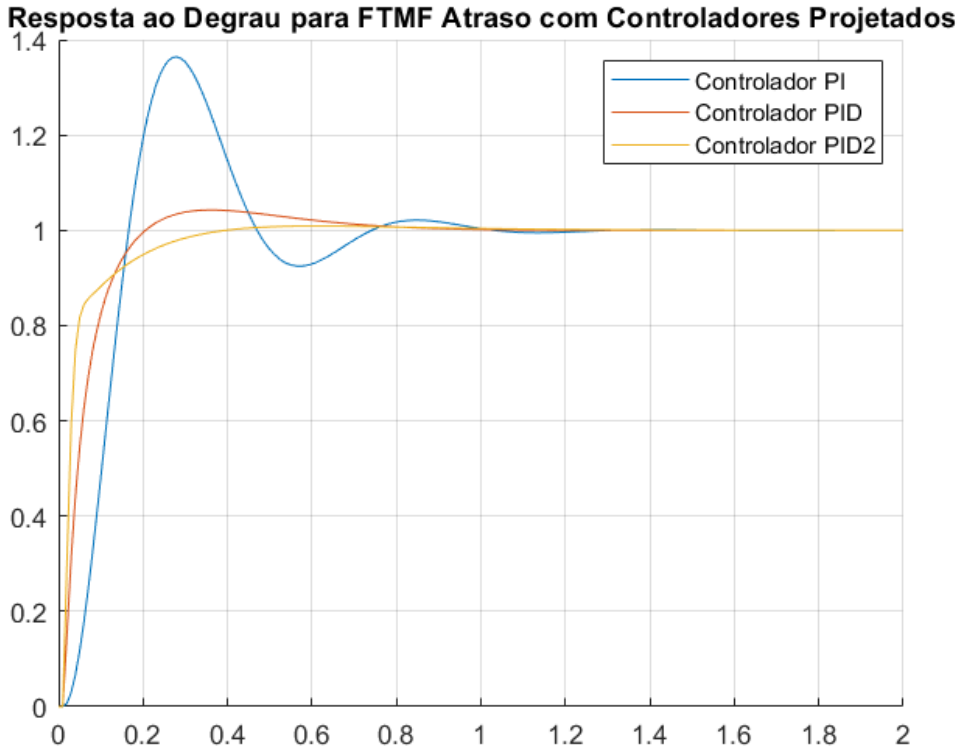
Figura 15 – Resposta ao Degrau de Malha Fechada com atraso de transporte e com PID2 gerada pelo comando stepinfo

```
ans =

  struct with fields:

      RiseTime: 10.8496
  SettlingTime: 29.5346
   SettlingMin: 0.9064
   SettlingMax: 1.0089
     Overshoot: 0.8860
    Undershoot: 0
          Peak: 1.0089
      PeakTime: 62
```

Fonte: Produção do próprio autor.

Portanto, ao analisar as Figuras 12, 13, 14, 15 e 16, podemos determinar que o melhor controlador PID para Gma com atraso de transporte foi o segundo novamente (PID_2), pois tem uma resposta rápida, um sobressinal baixo, erros ao degrau e ao distúrbio nulo e atender as especificações iniciais do projeto.

Figura 16 – Resposta ao Degrau de Malha Fechada com atraso de transporte e com os Controladores projetados



Fonte: Produção do próprio autor.

Por fim, observando as Figuras 4 e 12. podemos concluir que o controlador projetado sem atraso de transporte apresentou uma oscilação maior do que o controlador com atraso, pois obteve uma sobreelevação de quase 2% enquanto o projetado com atraso não atingiu 1%. Porém, quando analisamos o tempo de estabelecimento, a resposta do controlador sem atraso foi muito mais rápida, o que torna o sobressinal justificável. Deste modo, fazendo um balanceamento dos dados coletatos, temos que o controlador sem o atraso de transporte é superior.

3 ITEM 1.2

Inicialmente, para o caso em que T = 0 segundos do item anterior, multiplicou-se o controlador projetado pela função de transferência do veículo lunar. Tal implementação é descrita no trecho de código a seguir:

```
clear all, close all, clc;
op = 1; % 1 se item (i), 2 se item (ii)
global F A B;
load('PID');
load('Gma');
Gpid = Gma*PID;
```

Na segunda linha de código, a escolha da variável **op** deve ser feita de acordo com a característica em relação ao ruído de medição que se deseja. Se ***op = 1*** existe na saída do sistema de controle um ruído graussiano no sensor do veículo lunar. Se ***op = 2*** não há ruído de medição na saída do sistema. A função de malha aberta gerada pela multiplicação do controlador pela Gma é dada por 3.1.

$$Gpid = \frac{88.1s^2 + 1835s + 7493}{s^3 + 18.78s^2 + 44.44s} \tag{3.1}$$

Para obtenção da equação de estados do sistema em malha aberta apresentado, fez-se necessário o uso da teoria de representação em variáveis de estados, sendo possível descrever um sistema LIT da seguinte forma:

$$\dot{X}(t) = A \cdot X(t) + B \cdot U(t)$$
$$Y(t) = C \cdot X(t) + D \cdot U(t)$$

Onde X é o vetor de estados do sistema, Y é a saída e U a entrada. As matrizes A, B, C e D são definidas como:

- A: matriz do sistema

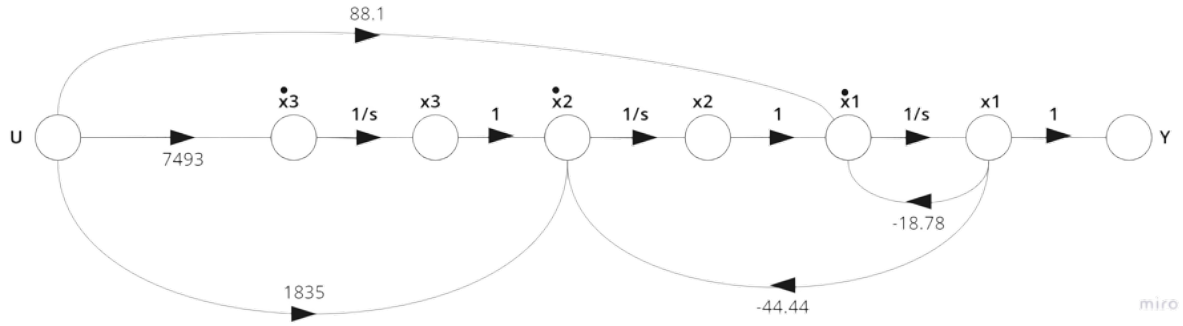
- B: matriz de entrada
- C: matriz de saída
- D: matriz de transmissão direta

Para se obter tais matrizes, faz-se necessário analisar o diagrama de fluxo de sinal a partir da função *Gpid* multiplicada por s^{-3}/s^{-3}, ou seja:

$$Gpid = \frac{88.1s^2 + 1835s + 7493}{s^3 + 18.78s^2 + 44.44s} \cdot \frac{s^{-3}}{s^{-3}} = \frac{88.1s^{-1} + 1835s^{-2} + 7493s^{-3}}{1 + 18.78s^{-1} + 44.44s^{-2}}$$

Sendo possível esboçar o seguinte diagrama de fluxo:

Figura 17 – Diagrama de Fluxo de Sinal



Fonte: Produção do próprio autor.

Do diagrama, pode-se obter

$$\dot{x}_1 = -18.78 \cdot x_1 + x_2 + 88.1 \cdot u$$
$$\dot{x}_2 = -44.44 \cdot x_1 + x_3 + 1835 \cdot u$$
$$\dot{x}_3 = 7493 \cdot u$$

e , portanto, para a *Gpid* dada, as matrizes do espaço de estados são definidas como:

$$A = \begin{pmatrix} -18.78 & 1.00 & 0.00 \\ -44.44 & 0.00 & 1.00 \\ 0.00 & 0.00 & 0.00 \end{pmatrix},\ B = \begin{pmatrix} 88.1 \\ 1835 \\ 74930 \end{pmatrix},\ C = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix},\ D = 0$$

Em código,

```
% Definição das Matrizes A, B, C e D
A = [-18.78 1 0;
     -44.44 0 1;
       0 0 0];
B = [88.1 1835 7493]';
C = [1 0 0];
D = 0;
```

Com a definição das matrizes do espaço de estados dada em código, foi necessário fazer a inicialização das variáveis de simulação e estado inicial, tal como definidas no script *simulaviao.m* (caso exemplo dado pelo professor).

```
% vetor estados inicial
Xs = [0 0 0];
F = 0;

% Inicialização de variáveis
tfinal = 5; % Tempo total de simulação
ref = 1; % Referência
passo = 0;
tempo = 0;
dT = 0.01; % tempo de amostragem
```

A simulação do sistema de controle de direção em malha fechada é feita por meio de um loop do tipo while com critério de parada dado pelo tempo de simulação, o qual foi de 0 à 5 segundos, com passo de 0.1 segundos dado por cada iteração do laço.

```
while (tempo < tfinal)
     passo = passo + 1;
     ts = passo*dT;

     % simulação da equação diferencial
     [T,X] = ode45('modelo_linearVeiculo',[tempo ts], Xs);

     % Armazenamento de variaveis
     n = length(T);
     tempo = ts;
```

```
        Xs = X(n,:);
        if(op == 2);
            F = ref - Xs(1);
        end
        if(op == 1);
            F = ref-(Xs(1) + 0.05*randn(1));
        end
        vetout(passo,:) = X(n,1);
        vettime(passo) = T(n,:);
        vetvref(passo,:) = ref;
    end
```

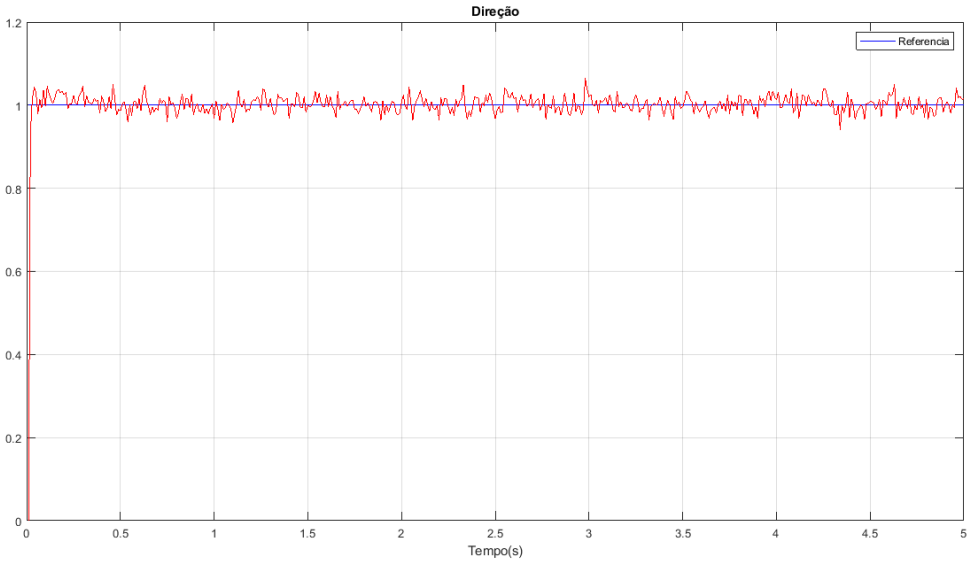
Ao fim dos 5 segundos de simulação, tem-se a saída da resposta do controle de direção dada pela plotagem do vetor vetout em relação ao tempo (dado pelo vettime), para melhor visualização, areferência também é exibida. Segue o trecho de código para plotagem:

```
figure
plot(vettime, vetvref,'b');
hold on;
plot(vettime,vetout,'r'); %multiplica por p1 para reduzir o erro
xlabel('Tempo(s)');title('Direção');
legend('Referencia');
grid;
```

A Figura 18 é resultado da simulação do sistema de controle de direção em malha fechada no espaço de estados, com ruído gaussiano no sensor do veículo lunar na saída do sistema de controle.

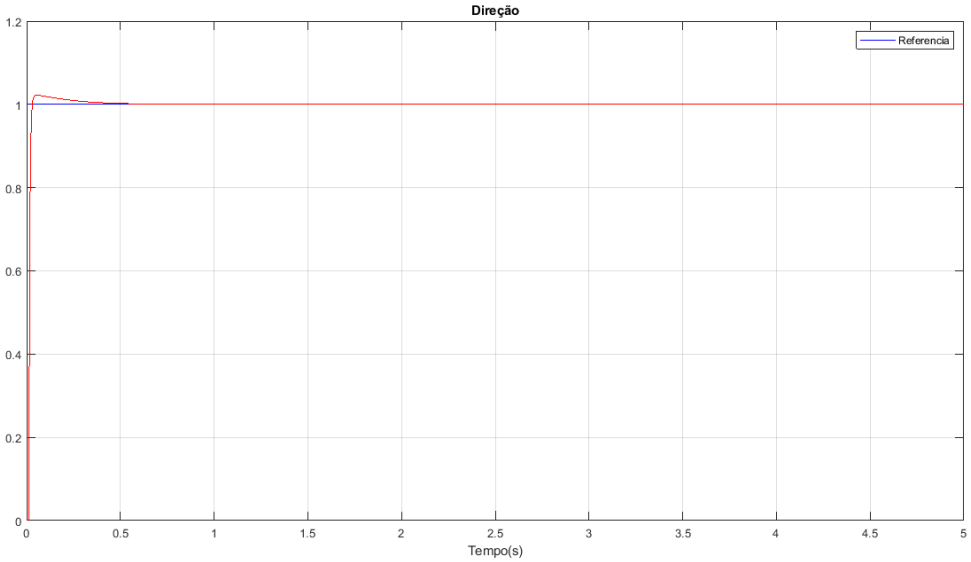
A Figura 19 é resultado da simulação do sistema de controle de direção em malha fechada no espaço de estados, sem ruído de medição na saída do sistema de controle.

Figura 18 – Resposta com ruído de medição



Fonte: Produção do próprio autor.

Figura 19 – Resposta sem ruído de medição



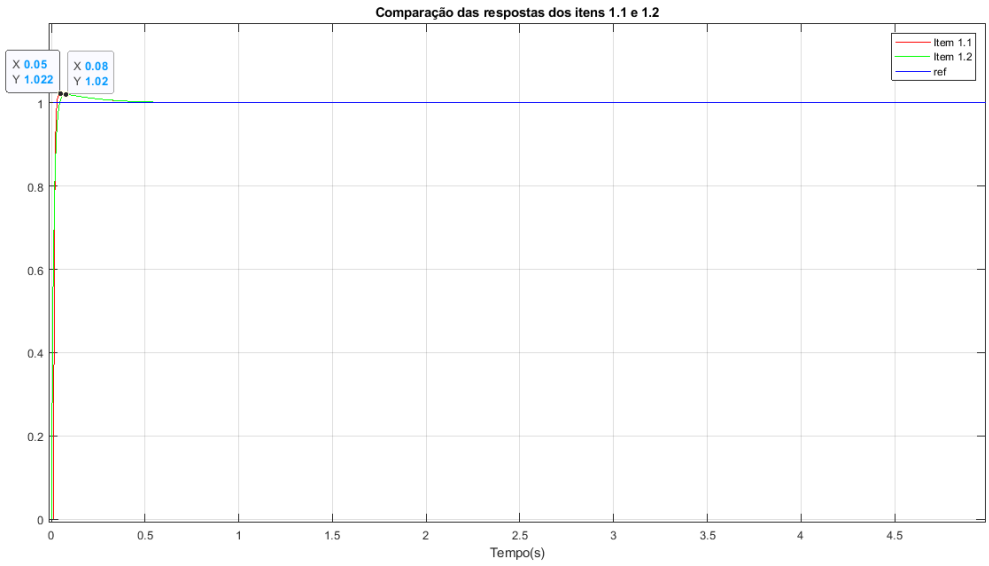
Fonte: Produção do próprio autor.

4 ITEM 1.3

Neste item foi solicitado realizar comparação das respostas ao degrau obtidas a partir do controlador PID projetado no item 1.1 para $T = 0s$ e a simulação desenvolvida no item 1.2 sem o ruído de medição na saída.

A Figura 20 mostra a comparação entre as curvas obtidas nos itens anteriores como solicitado. Nota-se que a simulação do sistema de controle de direção do veículo lunar obtida se assemelha a resposta ao degrau do item 1.1 em todo o tempo de simulação. A tabela 1 demonstra um comparativo quantitativo dessa semelhança e reforça que a simulação desenvolvida se mostrou satisfatória para a dada planta. O tempo de subida, o sobressinal e o erro foram encontrados através das curvas apresentadas na Figura 20.

Figura 20 – Comparação das respostas ao degrau



Fonte: Produção do próprio autor.

Tabela 1 – Comparação entre as curvas dos itens 1.1 e 1.2

	Item 1.1	Item 1.2
Sobressinal	2.2 %	2.0 %
Tempo de subida	0.03 s	0.05 s
Erro	0	0

Fonte: Produção do próprio autor.

5 ITEM 1.4

5.1 Item 1.1 (Atraso-Avanço)

Para a primeira parte do item 1.4, foi solicitado a realização do projeto para os dois casos do item 1.1, porém com um controlador Atraso-Avanço tendo novamente como requisitos o erro à entrada degrau e ao distúrbio menores ou iguais a 0.1, largura de banda da função de transferência em malha aberta maior possível e margem de fase maior ou igual a 60 graus.

Para o primeiro caso (sem o atraso), iniciamos o desenvolvimento do projeto do compensador Atraso. Deste modo, determinamos o ganho $K1$ que atenda ao erro menor que 0.1. Sendo assim, obtivemos o valor do ganho estático proporcional através do diagrama de bode da Figura 1 de valor de $Kp = 26.1db$ ou, tirando da escala db, $Kp = 20.1837$.

Através do erro estático, pode-se determinar o valor de $K1$ realizando a limitação de 0.1 para a entrada degrau e distúrbio, obtendo assim:

$$essDegrau = \lim_{s \to 0} \frac{1}{1 + K1 \cdot G_{ma}} = 0.1$$

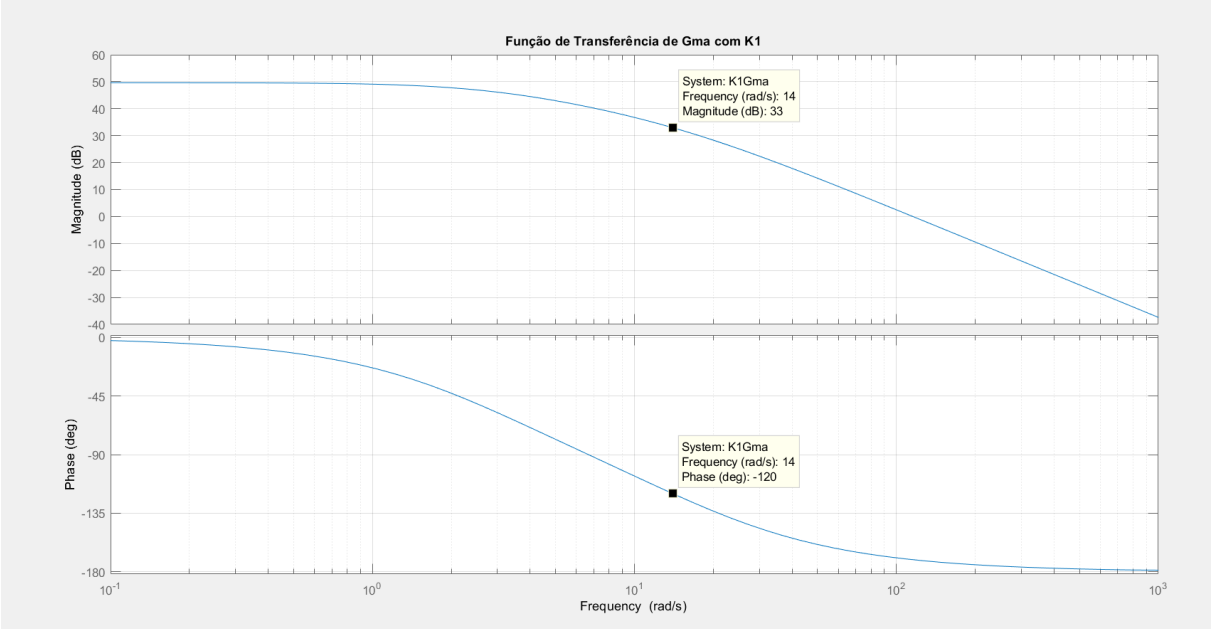
$$essDistrubio = \lim_{s \to 0} \frac{G_{ma}}{1 + K1 \cdot G_{ma}} = 0.1$$

$$K1_{degrau} = 0.4444$$
$$K1_{disturbio} = 9.9506$$

Analisando ambos valores de encontrados, definimos um gannho $K1 = 15$ pois consegue atender ambas especificações de erro.

Uma vez determinado o ganho, podemos iniciar o projeto do controlador de atraso de fase escolhendo a nova frequência de cruzamento de ganho, sendo determinada pela relação $fase_{cruzGanho} = MFdesejada - 180$, portanto temos que a nova fase será de -120 graus, no qual terá uma frequência de cruzamento de ganho de 14 rad/s e um módulo de 33 db utilizando o diagrama de bode de $K1G_{ma}$.

Figura 21 – Diagrama de Bode de K1Gma



Fonte: Produção do próprio autor.

Sabendo que a relação de alfa para o controlador atraso é $20log_{10}(\alpha) = 20log_{10}(modulo_w 0db)$, temos que $\alpha = 0.0303$.

Deste modo, após α determinado, calculamos o τ através das frequências de corte do zero do compensador, utilizando a nova frequência de cruzamento de ganho uma década abaixo, e a frequências de corte do polo do compensador, ou seja:

$$f_c zero = \frac{1}{\alpha\tau} - w0db_{(-1dec/abaixo)} \rightarrow \tau = 8.2500$$

$$f_c polo = \frac{1}{\tau} = 0.1212 rad/s$$

Portanto, uma vez determinado os valores de $K1$, α e τ, podemos definir o controlador de atraso de fase:

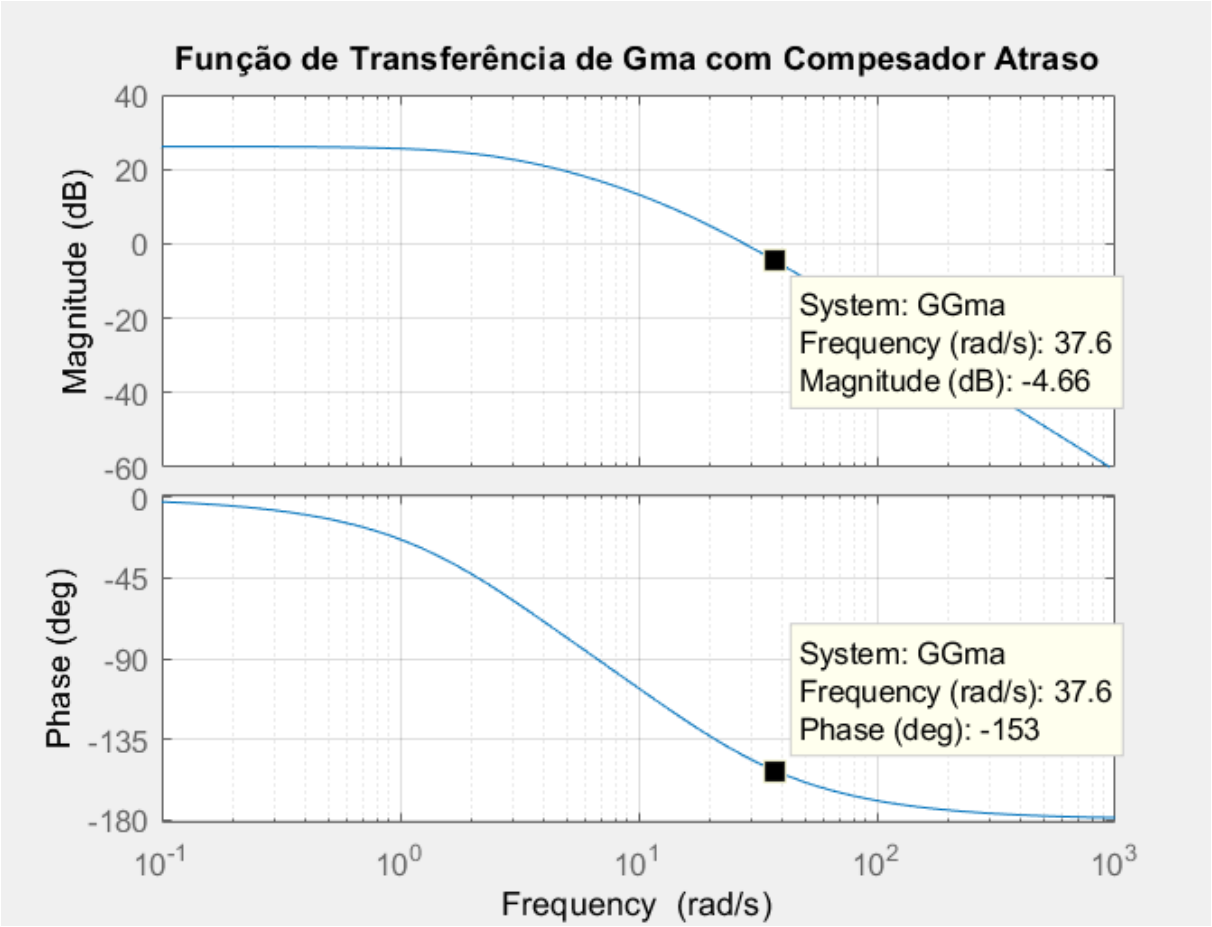
$$C_{atraso} = K1\frac{1+\alpha\tau j\omega}{1+\tau j\omega} = 15\frac{1+0.0303\cdot 8.2500 j\omega}{1+8.2500 j\omega}$$

Continuando o projeto, iniciamos o desenvolvimento da parte do controlador em avanço de fase. Deste modo, escolhemos o valor do ganho de avanço como $K2 = 1$ e determinamos o valor da nova fase, dada por $\phi_m = 75 - MF_{C_{atraso}Gma}$ que em nosso caso, obtivemos uma $MF_{C_{atraso}Gma} = 35.5531 graus$ e escolhemos uma nova margem de fase de 60 graus com

um adicional de 15 graus (75 graus) para ter uma boa margem de sobra. Após definido a nova fase, utilizamos este valor para determinar a constante α para o controlador avanço, sendo dado por $tg(\phi_m) = \frac{\alpha-1}{2\sqrt{\alpha}}$, o que nos deu $\alpha = 4.4849$

Deste modo, após α calculado, determinamos a frequência onde ocorre o máximo avanço de fase através de seu módulo calculado por $Modulo_{dB} = -10log_{10}(\alpha)$, que neste nosso caso nos deu $Modulo_{dB} = -4.6747$ aproximadamente, e avaliamos este valor no diagrama de bode de $C_{atraso}Gma$.

Figura 22 – Diagrama de Bode de C_{atraso}Gma



Fonte: Produção do próprio autor.

Através de bode, temos que a frequência onde irá ocorrer o máximo avanço de fase será em torno de $Wm = 37.6rad/s$. Sendo assim, podemos determinar a constante τ para o avanço pela relação $\tau = \frac{1}{\omega_m\sqrt{\alpha}}$, o que nos dá $\tau = 0.0126$

Portanto, uma vez determinado os valores de $K2$, α e τ, podemos definir o controlador de avanço de fase:

$$C_{avanco} = K2\frac{1+\alpha\tau j\omega}{1+\tau j\omega} = 1\frac{1+4.4849\cdot 0.0126j\omega}{1+0.0126j\omega}$$

E,deste modo, determinamos o controlador atraso-avanço desejado.

$$C_{AtrasoAvanco} = \frac{2.742\cdot 10^{32}s^2 + 2.964\cdot 10^{33}s + 1.947\cdot 10^{34}}{6.113\cdot 10^{31}s^2 + 5.112\cdot 10^{33}s + 1.947\cdot 10^{34}}$$

Por fim, em termos de análises e especificações, adicionamos o controlador atraso e atraso-avaço na planta e realizamos simulações de em malha aberta e fecha a fim de analisar suas respostas, gerando a tabela com os dados e a resposta ao degrau.

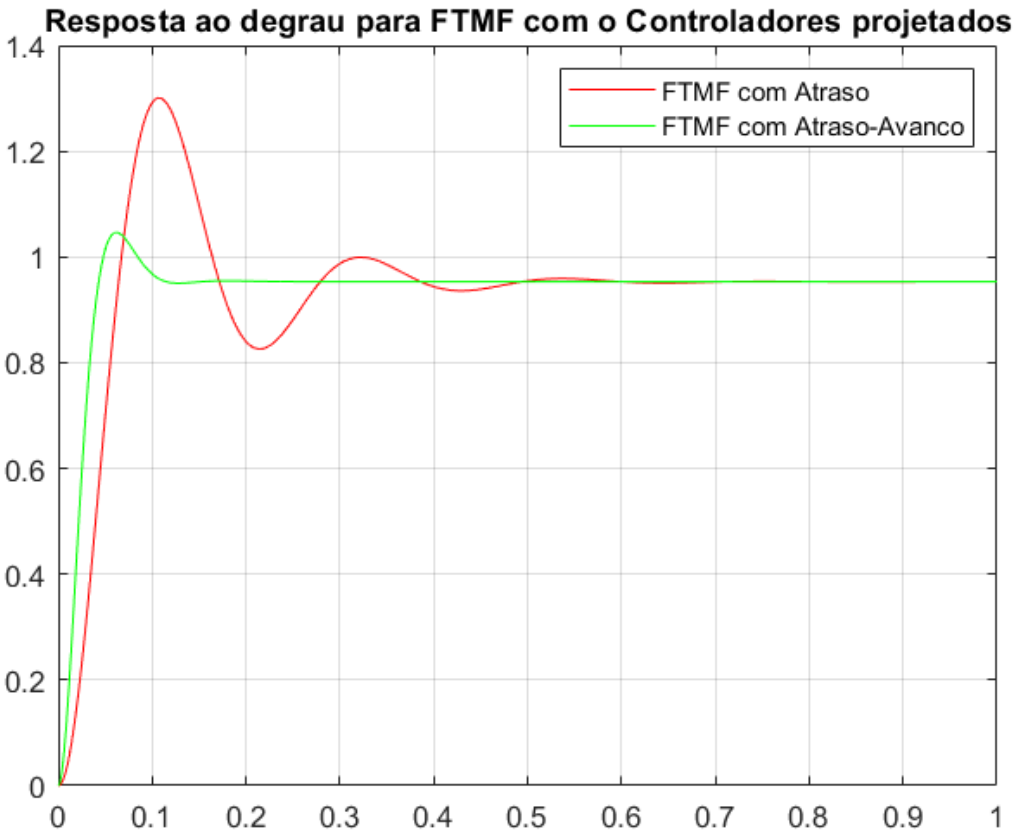
Figura 23 – Tabela realizada no Matlab para comparar os projetos dos controladores na planta de atraso e atraso-avanço

```
T =

  2×10 table
```

FTMAs	Controladores	Overshoot	SettlingTime	RiseTime	Gms	BWs	MFs	Ess_Degrau	Ess_Disturbio
'GGma'	'Atraso'	36.484	365.9	43.239	Inf	27.869	35.553	0.0032814	0.066448
'GGGGma'	'Avanco-Atraso'	9.7853	98.392	28.902	Inf	44.694	62.283	0.0032814	0.066448

Fonte: Produção do próprio autor.

Figura 24 – Resposta ao Degrau de Malha Fechada com os Controladores projetados



Fonte: Produção do próprio autor.

Para o segundo e último caso (com o atraso de transporte), realizamos os mesmo processos de desenvolvimento do projeto do compensador anterior. Deste modo, determinamos o ganho $K1$ que atenda ao erro menor que 0.1. Sendo assim, determinarmos o valor do ganho estático proporcional através do diagrama de bode da Figura 9, obtendo um valor de $Kp = 26.1db$ ou, tirando da escala db, $Kp = 20.1837$.

Através da erro estático, pode ser determinar o valor de $K1$ realizando a limitação de 0.1 para a entrada degrau e distúrbio, obtendo:

$$essDegrau = \lim_{s \to 0} \frac{1}{1 + K1 \cdot G_{ma}} = 0.1$$

$$essDistrubio = \lim_{s \to 0} \frac{G_{ma}}{1 + K1 \cdot G_{ma}} = 0.1$$

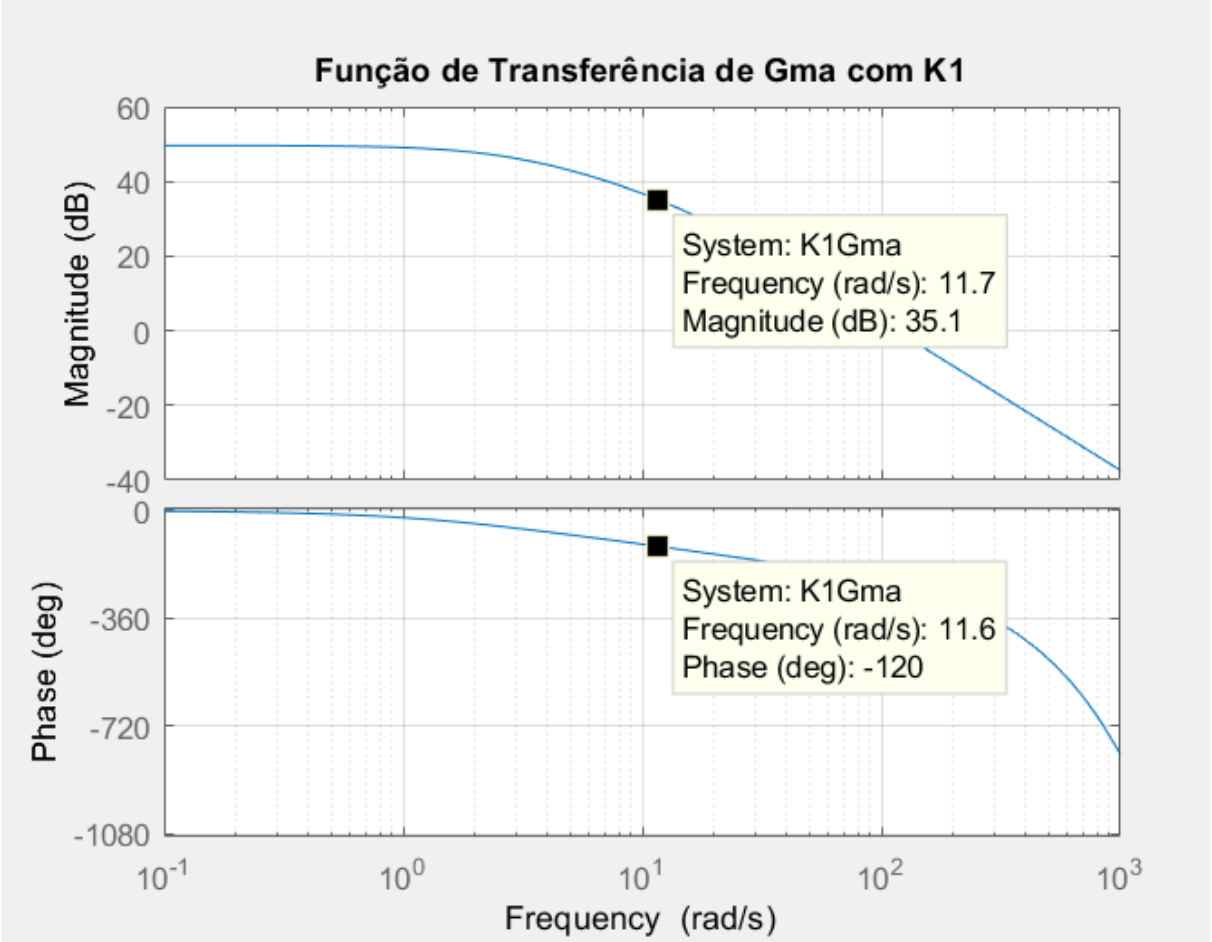
$$K1_{degrau} = 0.4444$$

$$K1_{disturbio} = 9.9506$$

Assim, analisando ambos valores de encontrados, definimos um gannho $K1 = 15$ pois atende ambas especificações.

Uma vez determinado o ganho, podemos iniciar o projeto do controlador de atraso de fase escolhendo a nova frequência de cruzamento de ganho, sendo determinada pela relação $fase_{cruzGanho} = MFdesejada - 180$, portanto temos que a nova fase será de -120 graus, no qual dará uma frequência de cruzamento de ganho de 11.6 rad/s e um módulo de 25.1 db utilizando o diagrama de bode de $K1G_{ma}$.

Figura 25 – Diagrama de Bode de K1Gma com atraso de transporte



Fonte: Produção do próprio autor.

Sabendo que a relação de alfa para o controlador atraso é $20log_{10}(\alpha) = 20log_{10}(modulo_w 0db)$, temos que $\alpha = 0.0398$.

Deste modo, após α determinado, calculamos o τ através da frequências de corte do zero do compensador, com nova frequência de cruzamento de ganho uma década abaixo, e a frequências de corte do polo do compensador, ou seja:

$$f_c zero = \frac{1}{\alpha\tau} - w0db_{(-1dec/abaixo)} \rightarrow \tau = 15.6875$$

$$f_c polo = \frac{1}{\tau} = 0.0637 rad/s$$

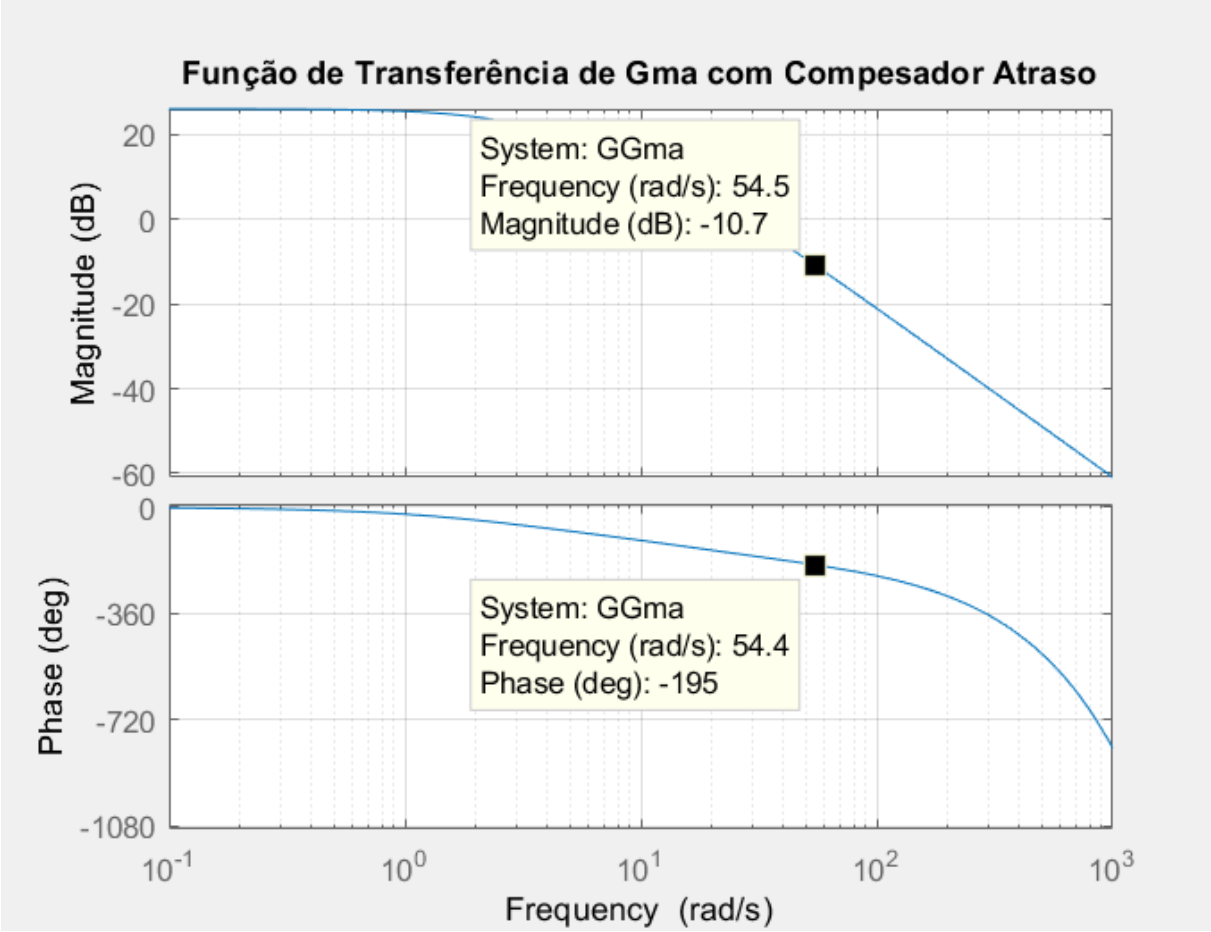
Portanto, uma vez determinado os valores de $K1$, α e τ, podemos definir o controlador de atraso de fase:

$$C_{atraso} = K1\frac{1 + \alpha\tau j\omega}{1 + \tau j\omega} = 15\frac{1 + 0.0398 \cdot 15.6875 j\omega}{1 + 15.6875 j\omega}$$

Continuando o projeto, iniciamos o desenvolvimento da parte do controlador em avanço de fase. Deste modo, escolhemos o valor do ganho de avanço como $K2 = 1$ e determinamos o valor da nova fase, dada por $\phi_m = 75 - MF_{C_{atraso}Gma}$ que em nosso caso, obtivemos uma $MF_{C_{atraso}Gma} = 17.8114 graus$ e escolhemos uma nova MF de 60 graus com um adicional de 15 graus (75 graus) para ter uma boa margem de sobra. Após definido a nova fase, utilizamos este valor para determinar a constante α para o controlador avanço, sendo dado por $tg(\phi_m) = \frac{\alpha-1}{2\sqrt{\alpha}}$, o que nos deu$\alpha = 11.5360$

Deste modo, após α calculado, determinamos a frequência onde ocorre o máximo avanço de fase através de seu módulo calculado por $Modulo_{dB} = -10log_{10}(\alpha)$, que neste nosso caso nos deu $Modulo_{dB} = -10.6205$ aproximadamente, e avaliamos este valor no diagrama de bode de $C_{atraso}Gma$.

Figura 26 – Diagrama de Bode de C_{atraso}Gma com atraso de transporte



Fonte: Produção do próprio autor.

Através de bode, temos que a frequência onde irá ocorrer o máximo avanço de fase serpa em torno $Wm = 54.5rad/s$. Sendo assim, podemos determinar a constante τ para Avanço pela relação $\tau = \frac{1}{\omega_m\sqrt{\alpha}}$, o que nos dá $\tau = 0.0054$

Portanto, uma vez determinado os valores de $K2$, α e τ, podemos definir o controlador de avanço de fase:

$$C_{avanco} = K2\frac{1+\alpha\tau j\omega}{1+\tau j\omega} = 1\frac{1+11.5360\cdot 0.0054j\omega}{1+0.0054j\omega}$$

E,deste modo, determinamos o controlador atraso-avanço desejado.

$$C_{AtrasoAvanco} = \frac{9.48\cdot 10^{31}s^2 + 1.673\cdot 10^{33}s + 2.434\cdot 10^{33}}{8.218\cdot 10^{30}s^2 + 1.534\cdot 10^{33}s + 2.434\cdot 10^{33}}$$

Por fim, em termos de análises e especificações, adicionamos o controlador atraso e atraso-avaço na planta e realizamos simulações de em malha aberta e fecha a fim de analisar suas respostas, gerando a tabela com os dados e a resposta ao degrau.

Figura 27 – Tabela realizada no Matlab para comparar os projetos dos controladores na planta de atraso e atraso-avanço com atraso de transporte

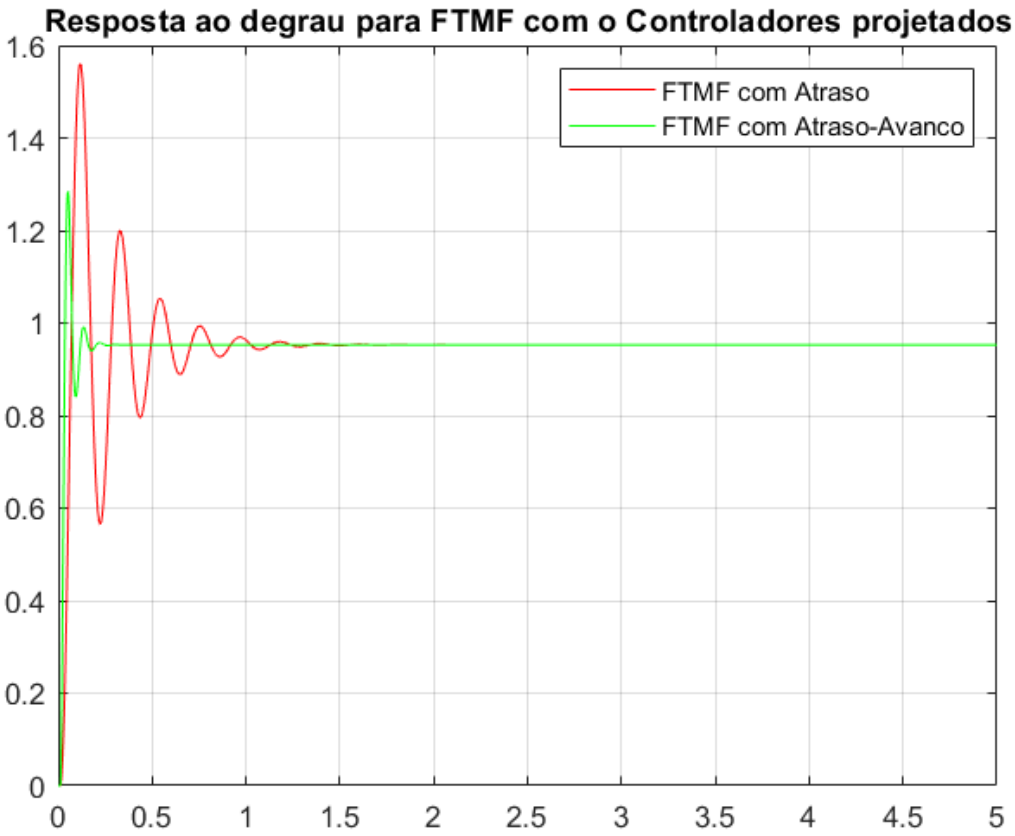
T =

2×10 table

FTMAs	Controladores	Overshoot	SettlingTime	RiseTime	Gms	BWs	MFs	Ess_Degrau	Ess_Disturbio
'GGma'	'Atraso'	63.758	887.47	38.236	1.9419	27.869	17.811	0.0032814	0.066448
'GGGGma'	'Avanco-Atraso'	34.793	151.29	16.48	2.0138	53.796	42.458	0.0032814	0.066448

Fonte: Produção do próprio autor.

Figura 28 – Resposta ao Degrau de Malha Fechada com atraso de transporte com os Controladores projetados



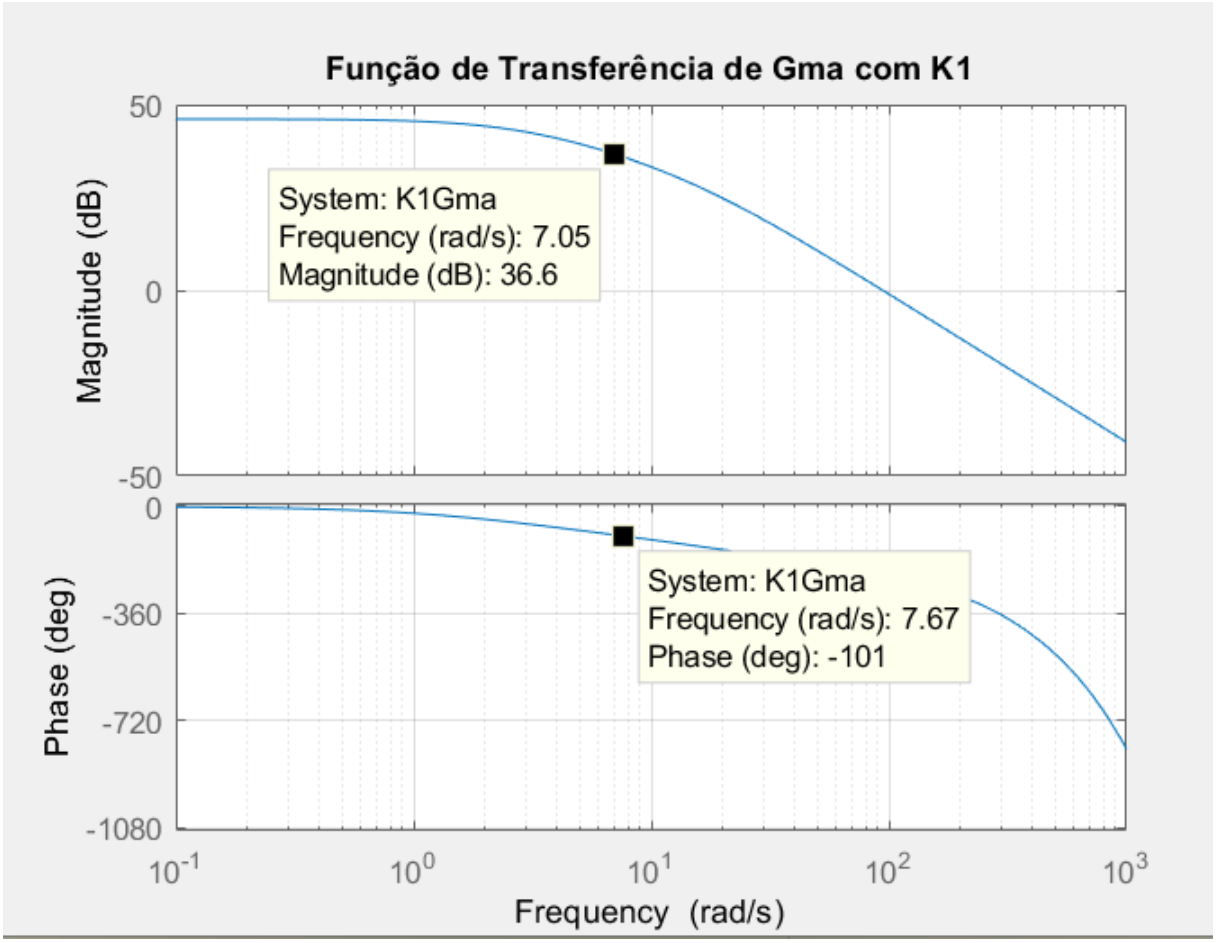
Fonte: Produção do próprio autor.

Como observado pela Tabela 27, não foi atendido ao requisito de margem de fase superior à 60 graus, deste modo, como podemos abaixar ainda mais o ganho K1 para tenha um aumento de margem fase e continue atendendo ao requisito de erro, realizamos outra vez o projeto avanço-atraso definimos um gannho $K1 = 10$.

Uma vez determinado o ganho, podemos iniciar o projeto do controlador de atraso de fase escolhendo a nova frequência de cruzamento de ganho, sendo determinada pela relação

$fase_{cruzGanho} = MFdesejada - 180$, portanto temos que a nova fase será de -100 graus, no qual dará uma frequência de cruzamento de ganho de 7.65 rad/s e um módulo de 37 db utilizando o diagrama de bode de $K1G_{ma}$.

Figura 29 – Diagrama de Bode de K1Gma com atraso de transporte



Fonte: Produção do próprio autor.

Sabendo que a relação de alfa para o controlador atraso é $20log_{10}(\alpha) = 20log_{10}(modulo_w 0db)$, temos que $\alpha = 0.0270$.

Deste modo, após α determinado, calculamos o τ através da frequências de corte do zero do compensador, com nova frequência de cruzamento de ganho uma década abaixo, e a frequências de corte do polo do compensador, ou seja:

$$f_c zero = \frac{1}{\alpha\tau} - w0db_{(-1dec/abaixo)} \rightarrow \tau = -15.7447$$

$$f_c polo = \frac{1}{\tau} = -0.0635 rad/s$$

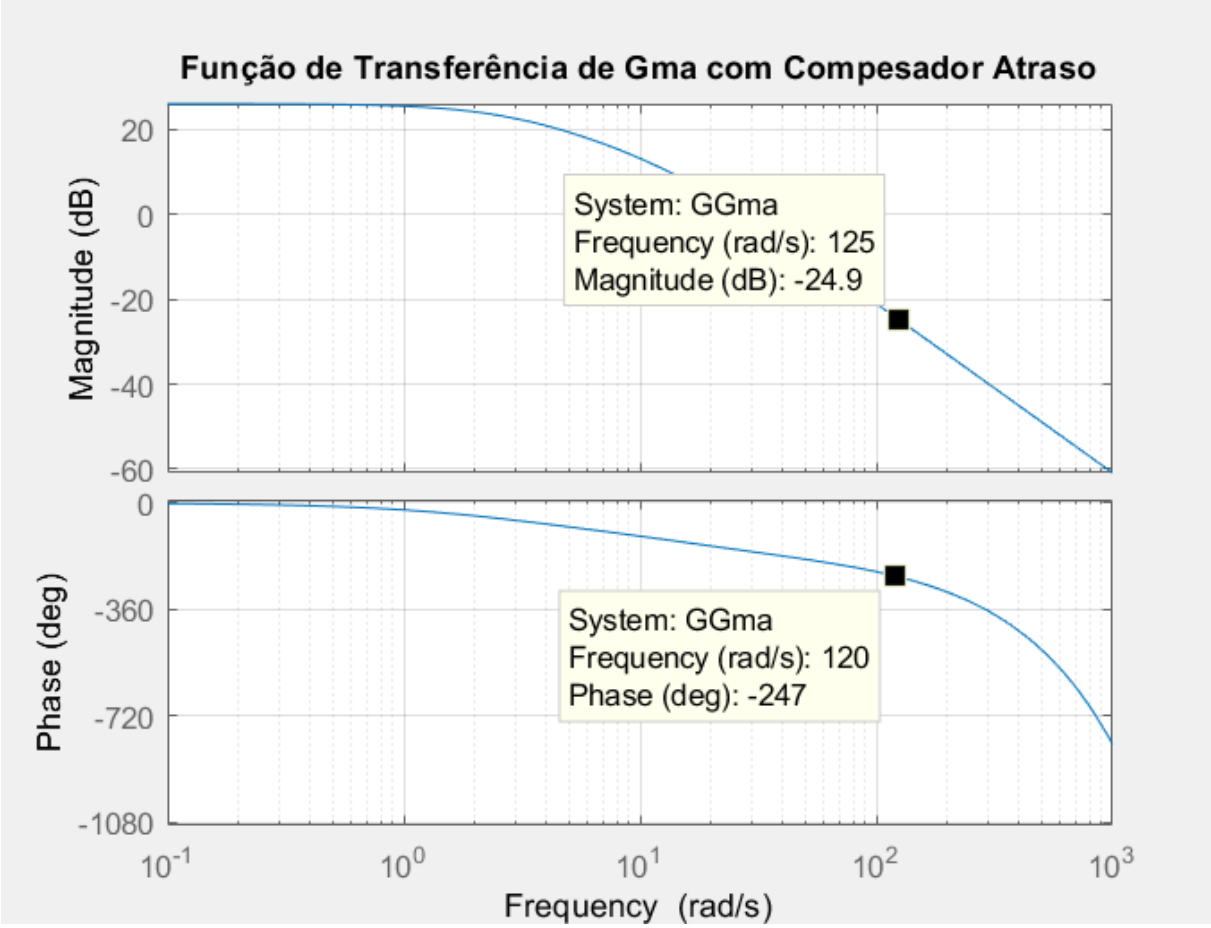
Portanto, uma vez determinado os valores de $K1$, α e τ, podemos definir o controlador de atraso de fase:

$$C_{atraso} = K1\frac{1 + \alpha\tau j\omega}{1 + \tau j\omega} = 10\frac{1 + 0.0270 \cdot -15.7447 j\omega}{1 - 15.7447 j\omega}$$

Continuando o projeto, iniciamos o desenvolvimento da parte do controlador em avanço de fase. Deste modo, escolhemos o valor do ganho de avanço como $K2 = 1$ e determinamos o valor da nova fase, dada por $\phi_m = 100 - MF_{C_{atraso}Gma}$ que em nosso caso, obtivemos uma $MF_{C_{atraso}Gma} = 17.8114 graus$ e escolhemos uma nova MF de 60 graus com um adicional de 40 graus (100 graus) para ter uma boa margem de sobra. Após definido a nova fase, utilizamos este valor para determinar a constante α para o controlador avanço, sendo dado por $tg(\phi_m) = \frac{\alpha-1}{2\sqrt{\alpha}}$, o que nos deu$\alpha = 214.5370$

Deste modo, após α calculado, determinamos a frequência onde ocorre o máximo avanço de fase através de seu módulo calculado por $Modulo_{dB} = -10log_{10}(\alpha)$, que neste nosso caso nos deu $Modulo_{dB} = -23.3150$ aproximadamente, e avaliamos este valor no diagrama de bode de $C_{atraso}Gma$.

Figura 30 – Diagrama de Bode de C_{atraso}Gma com atraso de transporte



Fonte: Produção do próprio autor.

Através de bode, temos que a frequência onde irá ocorrer o máximo avanço de fase serpa em torno $Wm = 250rad/s$. Sendo assim, podemos determinar a constante τ para Avanço pela relação $\tau = \frac{1}{\omega_m\sqrt{\alpha}}$, o que nos dá $\tau = 2.7309 \cdot 10^{-04}$

Portanto, uma vez determinado os valores de $K2$, α e τ, podemos definir o controlador de avanço de fase:

$$C_{avanco} = K2\frac{1+\alpha\tau j\omega}{1+\tau j\omega} = 1\frac{1+214.5370 \cdot 2.7309 \cdot 10^{-04} j\omega}{1+2.7309 \cdot 10^{-04} j\omega}$$

E,deste modo, determinamos o controlador atraso-avanço desejado.

$$C_{AtrasoAvanco} = \frac{3.803 \cdot 10^{33}s^2 + 5.597 \cdot 10^{34}s - 1.525 \cdot 10^{35}}{1.772 \cdot 10^{30}s^2 + 6.486 \cdot 10^{34}s - 1.525 \cdot 10^{33}}$$

Por fim, em termos de análises e especificações, adicionamos o controlador atraso e atraso-avaço na planta e realizamos simulações de em malha aberta e fecha a fim de analisar suas respostas, gerando a tabela com os dados e a resposta ao degrau.

Figura 31 – Tabela realizada no Matlab para comparar os projetos dos controladores na planta de atraso e atraso-avanço com atraso de transporte

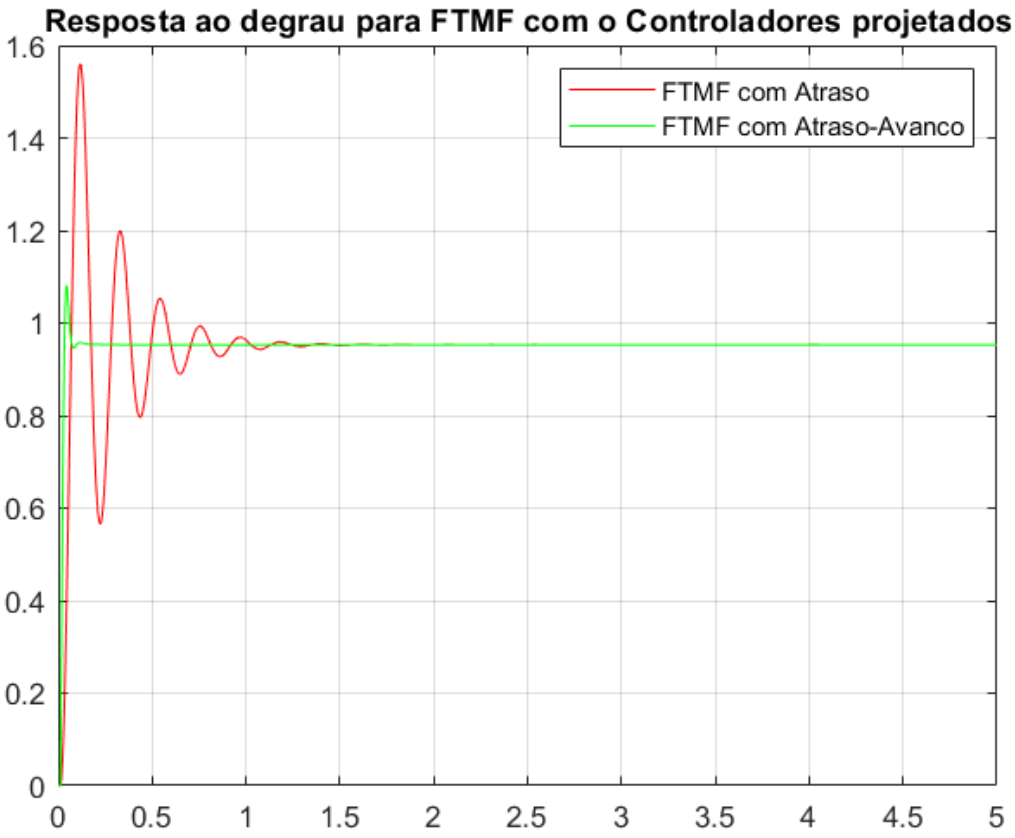
T =

2×10 table

FTMAs	Controladores	Overshoot	SettlingTime	RiseTime	Gms	BWs	MFs	Ess_Degrau	Ess_Disturbio
'GGma'	'Atraso'	63.753	887.35	38.233	1.9419	27.869	17.811	0.004914	0.099509
'GGGGma'	'Avanco-Atraso'	13.411	66.724	15.723	2.6369	52.956	57.408	0.004914	0.099509

Fonte: Produção do próprio autor.

Figura 32 – Resposta ao Degrau de Malha Fechada com atraso de transporte com os Controladores projetados



Fonte: Produção do próprio autor.

Como observado pela Tabela 31, mesmo assim não foi atendido ao requisito de margem de fase superior à 60 graus, porém conseguimos atigir um valor bem próximo fazendo a repetição do projeto mantendo os outros requisitos aceitáveis.

5.2 Item 1.2 (Atraso-Avanço)

Considerando a função de transferência de quarta ordem resultante da multiplicação entre a planta e o controlador atraso-avanço projetado no item anterior, tem-se a função de malha aberta dada por 5.1

$$Gma_c = \frac{2.468 \cdot 10^{35}s^2 + 5.368 \cdot 10^{36}s + 1.752 \cdot 10^{37}}{6.113 \cdot 10^{31}s^4 + 6.26 \cdot 10^{33}s^3 + 1.182 \cdot 10^{35}s^2 + 5.928 \cdot 10^{35}s + 8.654 \cdot 10^{35}} \quad (5.1)$$

Dividindo o numerador e o denominador por $6.113 \cdot 10^{31}$:

$$Gma_c = \frac{2.468 \cdot 10^{35}s^2 + 5.368 \cdot 10^{36}s + 1.752 \cdot 10^{37}}{6.113 \cdot 10^{31}s^4 + 6.26 \cdot 10^{33}s^3 + 1.182 \cdot 10^{35}s^2 + 5.928 \cdot 10^{35}s + 8.654 \cdot 10^{35}} \cdot \frac{6.113 \cdot 10^{31}}{6.113 \cdot 10^{31}}$$

$$\implies Gma_c = \frac{4037.30s^2 + 87812.86s + 286602.32}{s^4 + 102.40s^3 + 1933.58s^2 + 9697.37s + 14156.72}$$

Para obter-se as matrizes do espaço de estados, deve-se seguir um procedimento semelhante ao visto anteriormente e portanto, deve-se multiplicar Gma_c por s^{-4}/s^{-4}, como a seguir:

$$Gma_c = \implies Gma_c = \frac{4037.30s^2 + 87812.86s + 286602.32}{s^4 + 102.40s^3 + 1933.58s^2 + 9697.37s + 14156.72} \cdot \frac{s^{-4}}{s^{-4}}$$

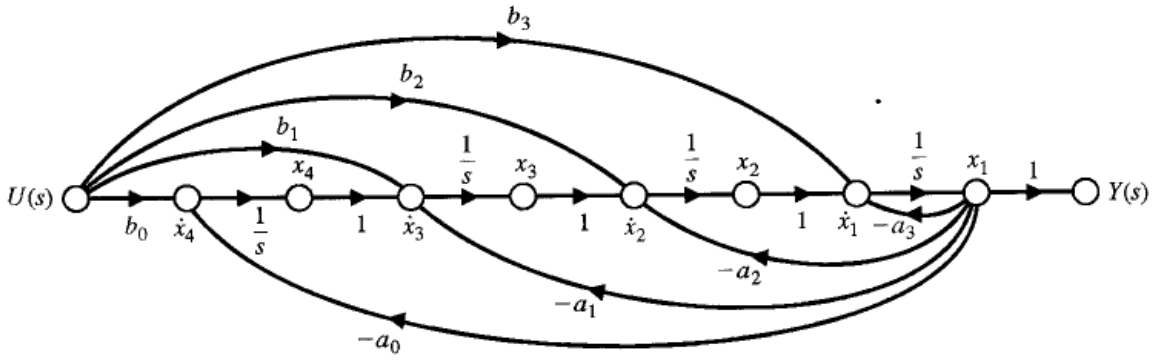
$$\implies Gma_c = \frac{4037.30s^{-2} + 87812.86s^{-3} + 286602.32s^{-4}}{1 + 102.40s^{-1} + 1933.58s^{-2} + 9697.37s^{-3} + 14156.72s^{-4}} \quad (5.2)$$

Da função 5.2 e utilizando-se o modelo *feedfoward* para o diagrama de fluxo de final, é possível se obter a Figura 33, considerando-se a forma 5.3.

$$Gma_c = \frac{b3s^3 + b2s^2 + b1s + b0}{s^4 + a3s^3 + a2s^2 + a1s + a0} \quad (5.3)$$

Do diagrama de fluxo temos as seguintes equações diferenciais:

Figura 33 – Diagrama de Fluxo de Sinal



Fonte: (DORF; BISHOP, 1998)

$$\dot{x}_1 = -a_3 x_1 + x_2 + b_3 u,$$
$$\dot{x}_2 = -a_2 x_1 + x_3 + b_2 u,$$
$$\dot{x}_3 = -a_1 x_1 + x_4 + b_1 u,$$
$$\dot{x}_4 = -a_0 x_1 + b_0 u$$

Onde,

- $a0 = 14156.72$;
- $a1 = 9697.37$;
- $a2 = 1933.58$;
- $a3 = 102.40$
- $b0 = 286602.32$;
- $b1 = 87812.86$;
- $b2 = 4037.30$;
- $b3 = 0$;

Chegando-se a definição das matrizes em código:

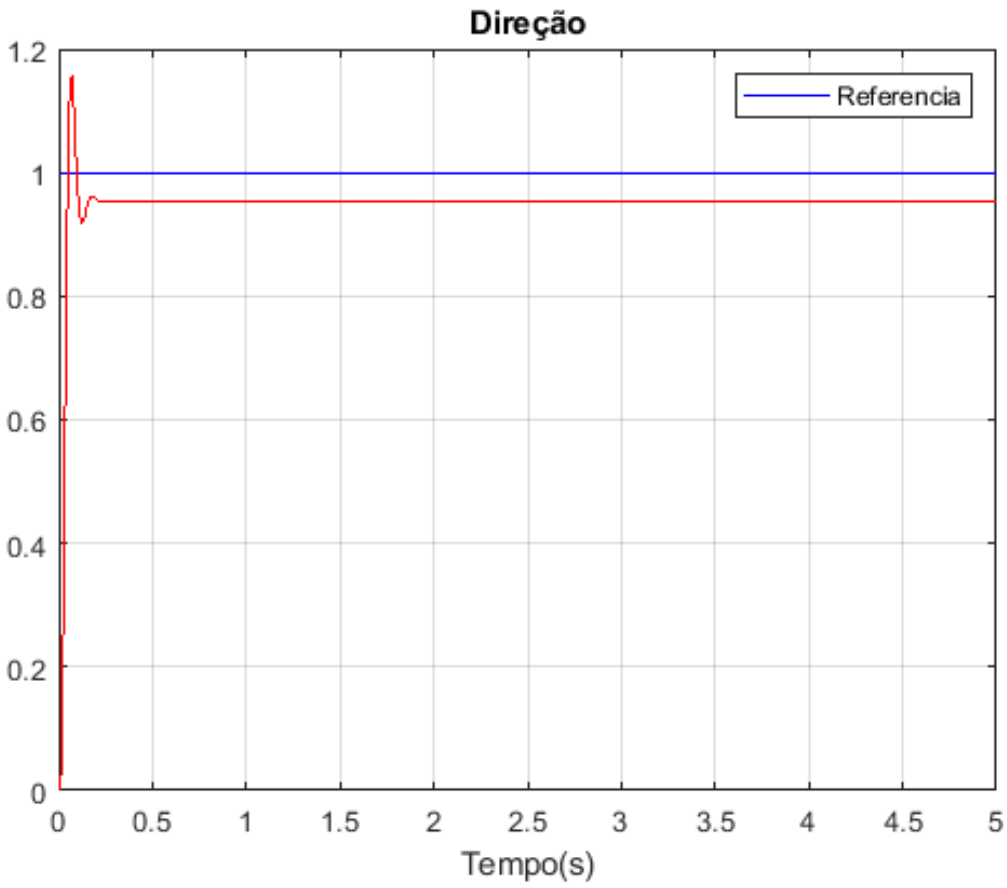
```
A = [-a3 1 0 0;
 -a2 0 1 0;
 -a1 0 0 1;
 -a0 0 0 0];

B = [b3 b2 b1 b0]';
C = [1 0 0 0];
D = 0;
```

Por fim, a simulação como no item 1.2:

```
while (tempo < tfinal)
     passo = passo + 1;
     ts = passo*dT;

     % simulação da equação diferencial
     [T,X] = ode45('modelo_linear',[tempo ts], Xs);

     % Armazenamento de variaveis
     n = length(T);
     tempo = ts;
     Xs = X(n,:);
     if(op == 2);
         F = ref - Xs(1);
     end
     if(op == 1);
         F = ref-(Xs(1) + 0.02*randn(1));
     end
     vetout(passo,:) = X(n,1);
     vettime(passo) = T(n,:);
     vetvref(passo,:) = ref;

end
```

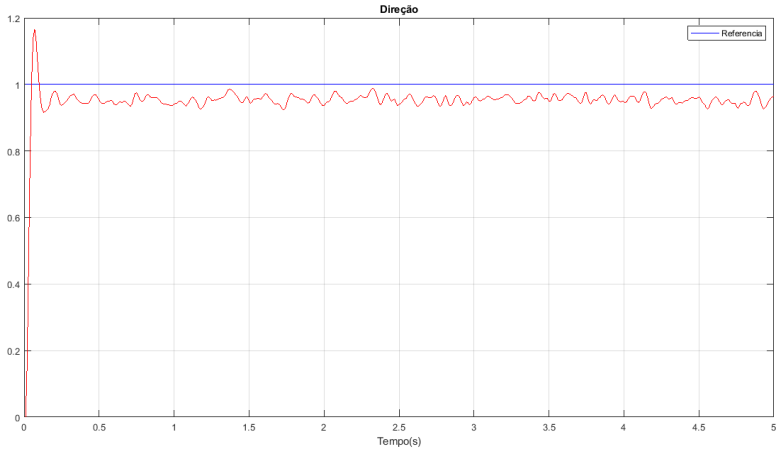
Vale lembrar que para simulação com ruído na medição basta alterar-se a variável op de 2 para 1. As saídas da simulação do controle de direção do veículo lunar para os casos com e sem ruído de medição são dadas, respectivamente, pelas Figuras 34, 35.

Figura 34 – Resposta com ruído de medição



Fonte: Produção do próprio autor.

Figura 35 – Resposta sem ruído de medição



Fonte: Produção do próprio autor.

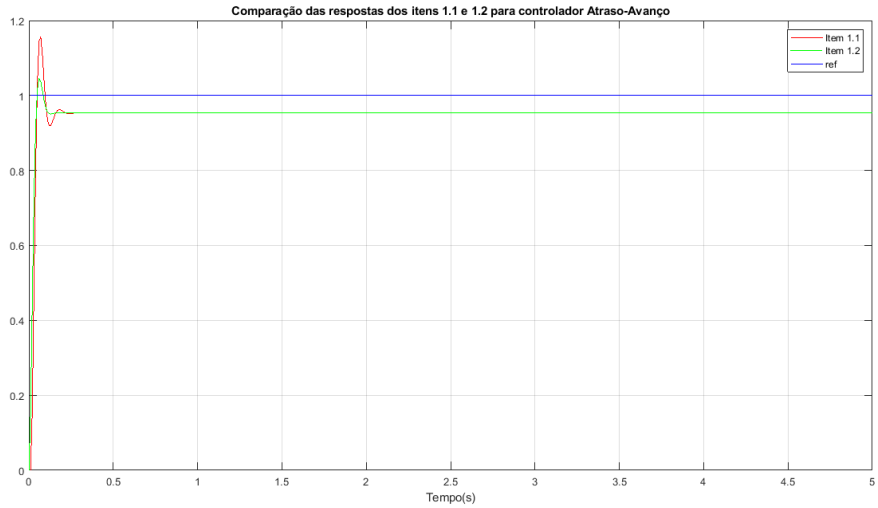
5.3 Item 1.3 (Atraso-Avanço)

Neste item foi solicitado realizar comparação das respostas ao degrau obtidas a partir do controlador Atraso-Avanço de fase no item 1.1 para $T = 0s$ e a simulação desenvolvida no

item 1.2 sem o ruído de medição na saída também para o controlador Atraso-Avanço de fase.

A Figura 36 mostra a comparação entre as curvas obtidas nos itens anteriores como solicitado. Nota-se que a simulação do sistema de controle de direção do veículo lunar obtida se assemelha a resposta ao degrau do item 1.1 em todo o tempo de simulação. A tabela 2 demonstra um comparativo quantitativo dessa semelhança e reforça que a simulação desenvolvida se mostrou satisfatória para a dada planta. O tempo de subida, o sobressinal e o erro foram encontrados através das curvas apresentadas na Figura 36.

Figura 36 – Comparação das respostas (Atraso-Avanço)



Fonte: Produção do próprio autor.

Tabela 2 – Comparação entre as curvas dos itens 1.1 e 1.2 (Atraso-Avanço)

	Item 1.1	Item 1.2
Sobressinal	16 %	5 %
Tempo de subida	0.04 s	0.04 s
Erro	0.05	0.05

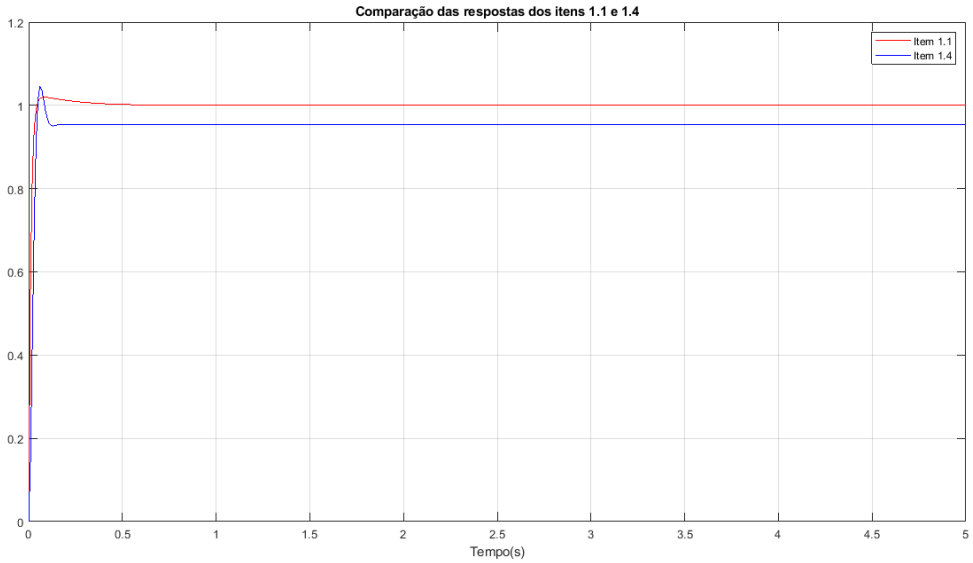
Fonte: Produção do próprio autor.

A tabela 2 foi construída a partir da análise da Figura 36 e, a partir dela, pode-se dizer verificar que na simulação do item 1.2 houve uma vantagem no sobressinal em relação a resposta ao degrau do item 1.1, cuja diferença é de mais de 10%. Os outros parâmetros se mostraram iguais, sendo importante destacar a característica mais oscilatória observada na resposta do item 1.1.

6 ITEM 1.5

A Figura 37 mostra a comparação entre as curvas obtidas nos itens anteriores como solicitado. Nota-se que a resposta do sistema com controlador PID escolhido no item 1.1 difere da resposta com controlador Atraso-Avanço do item 1.4, sendo o erro em regime a maior diferença entre esses sinais. Ambos atendem as especificações de projeto, com erros inferiores a 0.1, sendo no caso do PID um erro nulo e no atraso-avanço 0.05 de erro em regime, com tempo de subida de 0.04 segundos em ambos casos e sobressinal de 2% no primeiro caso e 8.8% no segundo. Estas informações são resumidas na Tabela 3.

Figura 37 – Comparação das respostas ao degrau controladores item 1.1 e 1.4



Fonte: Produção do próprio autor.

Tabela 3 – Comparação entre as curvas dos itens 1.1 e 1.4

	Item 1.1	Item 1.4
Sobressinal	2.0 %	8.8 %
Tempo de subida	0.04 s	0.04 s
Erro	0	0.05

Fonte: Produção do próprio autor.

7 ITEM 1.6

7.1 Análise do item 1.5

Do item 1.5, fez-se a escolha do controlador PID que foi projetado no item 1.1 para T = 0.0 s. Tal escolha se justifica pelo fato deste controlador ter gerado um menor sobressinal e erro em regime no sistema, quando comparado ao controlador Atraso-Avanço do item 1.4. A tabela 3 exemplifica tais observações e torna evidente que, no caso do tempo de subida, os controladores se mostraram equivalentes e nos outros parâmetros observados o PID se sobressaiu.

Em malha fechada, para o controlador escolhido, tem-se a função de transferência 7.1. Esta função e seus polos foram gerados pelo trecho de código abaixo:

```
load('PID');
load('Gma');
Gpid = Gma*PID2;

GMF=feedback(Gpid,1);
polos_mf = pole(GMF);
```

$$Gmf = \frac{88.1s^2 + 1835s + 7493}{s^3 + 106.9s^2 + 1880s + 7493} \tag{7.1}$$

Para a Gmf encontrada, obteve-se 3 polos, sendo eles: -86.0424, -15.0497 e -5.7868. A partir destes polos, na seção 7.2 fez-se o projeto do controlador usando a realimentação de estados para controlar a dinâmica deste sistema dado.

7.2 Projeto de controlador via realimentação de estados

Como em itens anteriores, definiu-se as matrizes do espaço de estados desejado. Neste caso tem-se a planta

$$G = \frac{900}{s^2 + 18.78s + 44.44}$$

A qual, pela forma canônica feedfoward apresentada no item 1.2, obteve-se as matrizes A, B, C e D, como no trecho de código abaixo:

```
clear all, close all, clc;
global F A B;
load('PID');
load('Gma');
Gpid = Gma*PID2;

GMF=feedback(Gpid,1);
polos_mf = pole(GMF);

A = [18.78 1;
    44.44 0];
B = [0 900]';
C = [1 0];
D = 0;
```

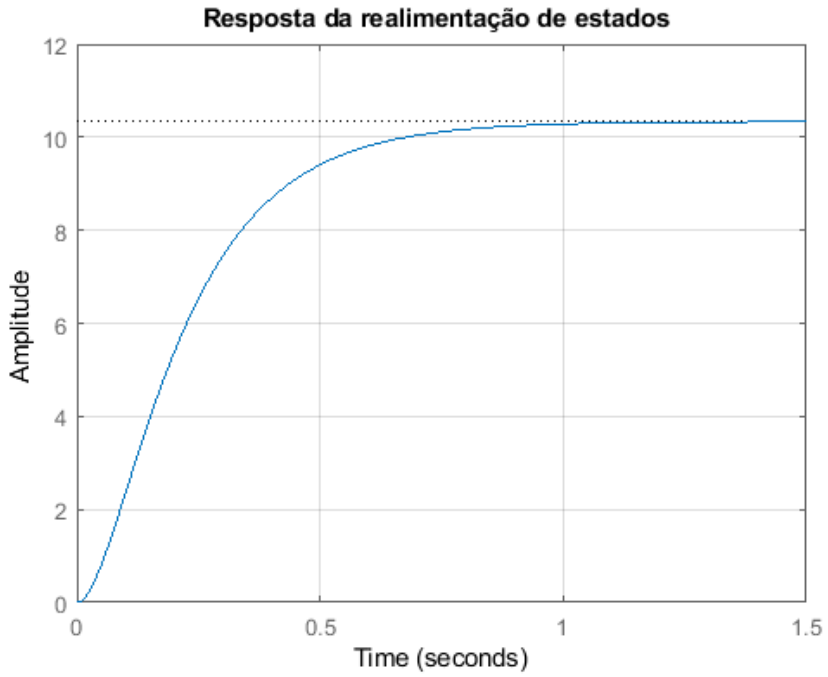
Em seguida, dos polos de malha fechada encontrados a partir da escolha do melhor controlador do item 1.5 e eliminando-se o polo menos dominante, pode-se encontrar a função de transferência de malha fechada da realimentação de estados, como segue:

```
%eliminando o polo menos dominante
polos_mf_ordem2 = [polos_mf(2), polos_mf(3)];

K = place(A, B, polos_mf_ordem2);
Ak = A-B*K;
m = ss(Ak,B,C,D); % FT de MF
```

O polo em -86.0424 foi rejeitado pois era o de menor influência na resposta do sistema. A matriz K posiciona os polos desejados a partir do parâmetro Ak (matriz A da realimentação de estados) do espaço de estados e assim pode-se gerar a função de malha fechada **m** via realimentação de estados. A resposta de **m** é dada pela figura 38 e exibida pelo seguinte trecho de código:

```
figure;
step(m);title('Resposta da realimentação de estados ');
grid
```

Figura 38 – Resposta da realimentação de estados

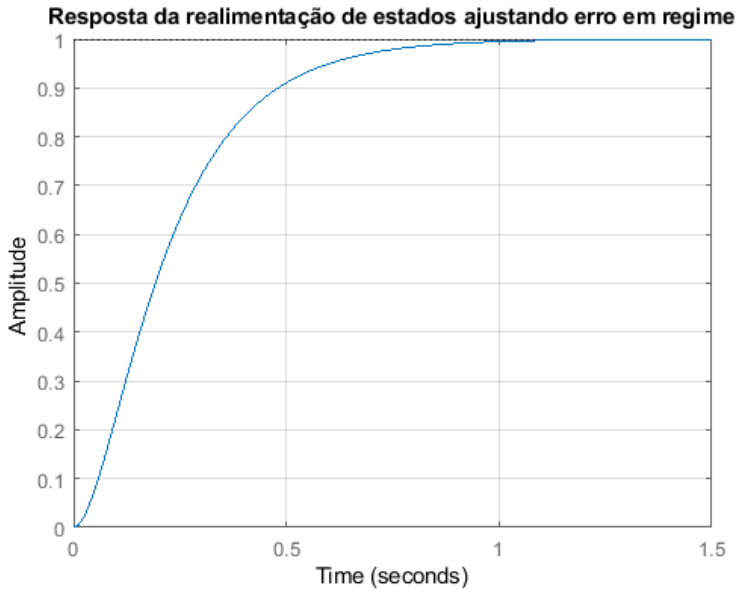


Fonte: Produção do próprio autor.

Tem-se que a realimentação de estados garante a estabilidade, mas não garante um bom erro em regime, como visto na Figura 38. Contudo, dada a função de transferência de malha fechada do sistema, o **m** calculado anteriormente, pode-se obter a inversa do ganho de regime da mesma e assim ajustar a saída para ser igual à referência na Figura 39, através da multiplicação da FTMF por p1 (inverso do ganho de regime). O p1 calculado foi de 0.0968. A resposta corrigida e o cálculo de p1 são dados pelo seguinte código:

```
k0 = freqresp(m,0);
p1 = 1/k0;

figure
step(p1*m);
title('Resposta da realimentação de estados ajustando erro em regime');
grid
```

Figura 39 – Resposta da realimentação de estados com correção de erro

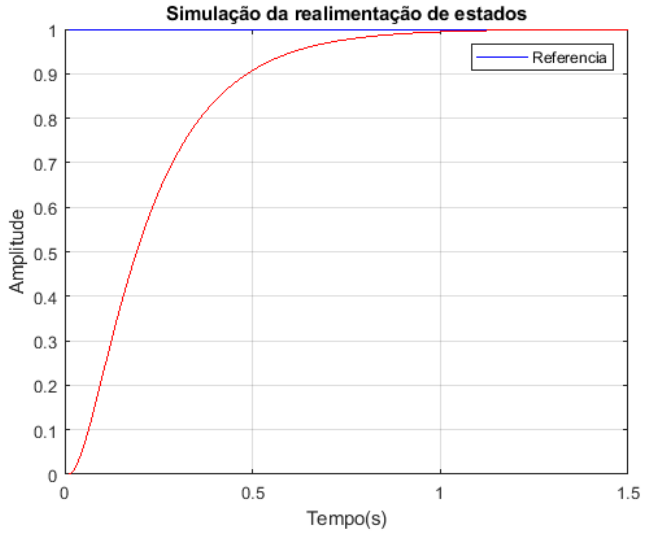


Fonte: Produção do próprio autor.

7.3 Simulação do controle via realimentação de estados

Por fim, como feito para o caso da simulação do veículo lunar para os controladores PID e Atraso-Avanço, realizou-se a simulação do sistema em malha fechada com realimentação de estados, obtendo-se uma resposta satisfatória se compararmos as curvas da Figura 40 e 39. O código da simulação está no anexo .6.

Figura 40 – Simulação da resposta em malha fechada com realimentação de estados

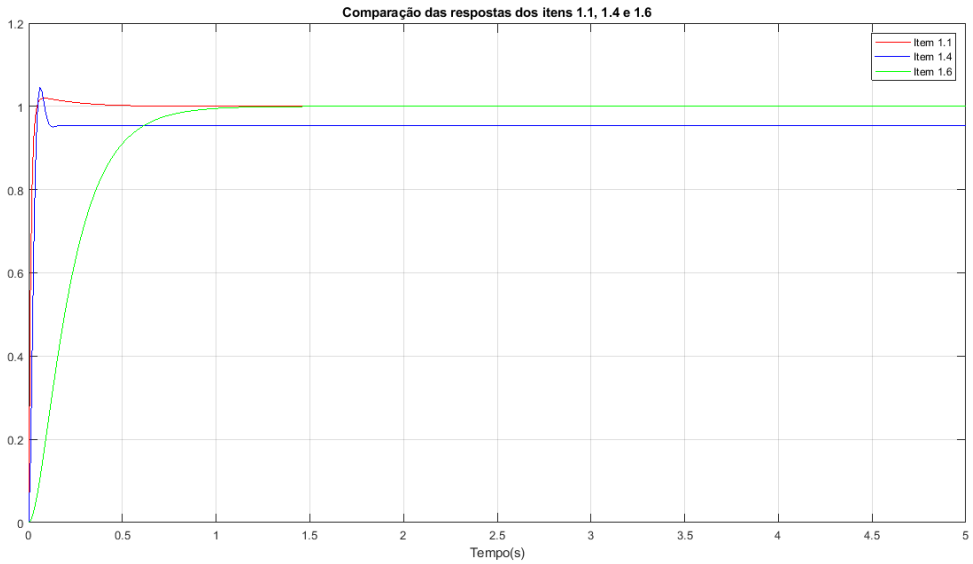


Fonte: Produção do próprio autor.

8 ITEM 1.7

A Figura 41 mostra a comparação entre as respostas dos sistemas para cada controlador projetado obtidos nos itens 1.1, 1.4 e 1.6. Nota-se que a resposta do sistema com controlador PID escolhido no item 1.1 difere da resposta com controlador Atraso-Avanço do item 1.4, sendo o erro em regime a maior diferença entre esses sinais. Ambos atendem as especificações de projeto, com erros inferiores a 0.1, sendo no caso do PID um erro nulo e no atraso-avanço 0.05 de erro em regime, com tempo de subida de 0.04 segundos em ambos casos e sobressinal de 2% no primeiro caso e 8.8% no segundo. Estas informações são resumidas na Tabela 4. Quanto ao controle por realimentação de estados, pode-se notar que o erro em regime e o sobressinal são nulos, porém há um aumento significativo no tempo de subida da curva, sendo 10 vezes maior que para os outros controladores.

Figura 41 – Comparação das respostas ao degrau controladores item 1.1, 1.4 e 1.6



Fonte: Produção do próprio autor.

Tabela 4 – Comparação entre as curvas dos itens 1.1, 1.4 e 1.6

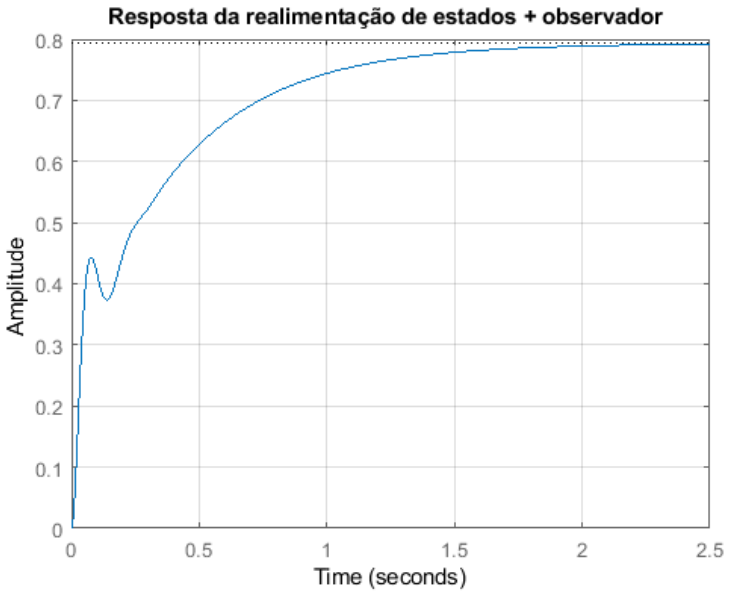
	Item 1.1	Item 1.4	Item 1.6
Sobressinal	2.0 %	8.8 %	0
Tempo de subida	0.04 s	0.04 s	0.42
Erro	0	0.05	0

Fonte: Produção do próprio autor.

9 ITEM 1.8

Multiplicou-se os polos de malha fechada do nosso sistema por um fator no intuito de obter-se os polos do observador. Para um fator de 1.5 obteve-se as curvas 42 e 43, onde a primeira representa a resposta da realimentação de estados sem correção de erro em regime e a segunda já com a correção.

Figura 42 – Resposta da realimentação de estados + Observador (fator 1.5)



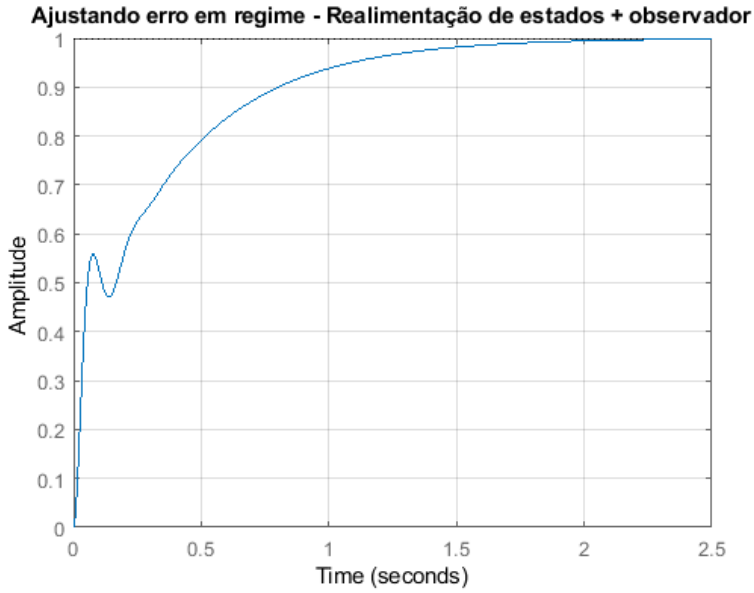
Fonte: Produção do próprio autor.

Para um fator de 5 obteve-se as curvas 44 e 45, onde a primeira representa a resposta da realimentação de estados sem correção de erro em regime e a segunda já com a correção.

A figura 46 é o comparativo entre as respostas solicitadas para cada método de controle solicitado. Pode-se verificar que o controle com observador não se mostrou satisfatório, sendo necessário revisar o projeto, pois este não deveria afetar a resposta ao degrau do sistema.

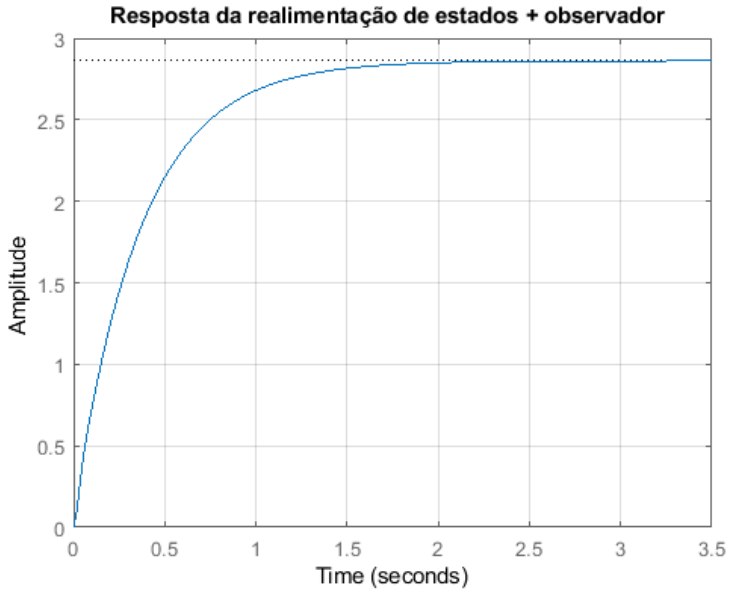
Os códigos deste item estão no anexo .8 e foram baseados no laboratório 12 da disciplina de Laboratório de Controle do semestre 2021/2 EARTE.

Figura 43 – Ajuste de erro - Resposta da realimentação de estados + Observador (fator 1.5)



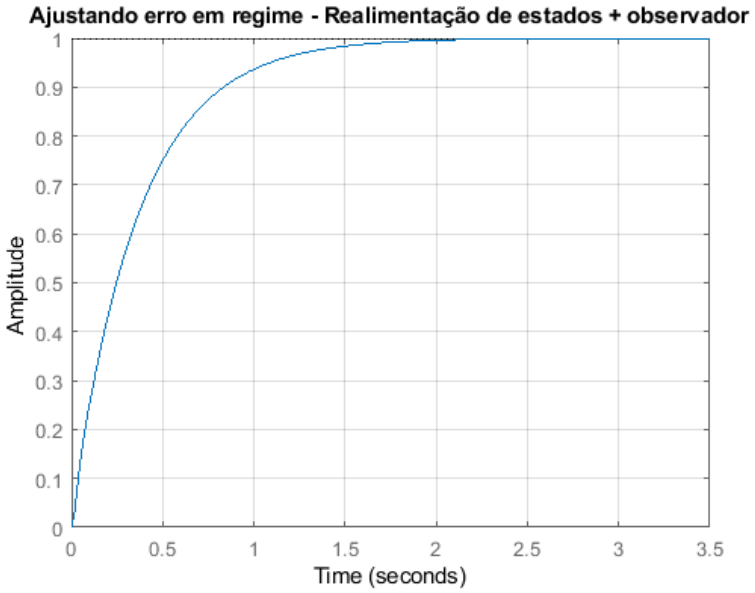
Fonte: Produção do próprio autor.

Figura 44 – Resposta da realimentação de estados + Observador (fator 5)



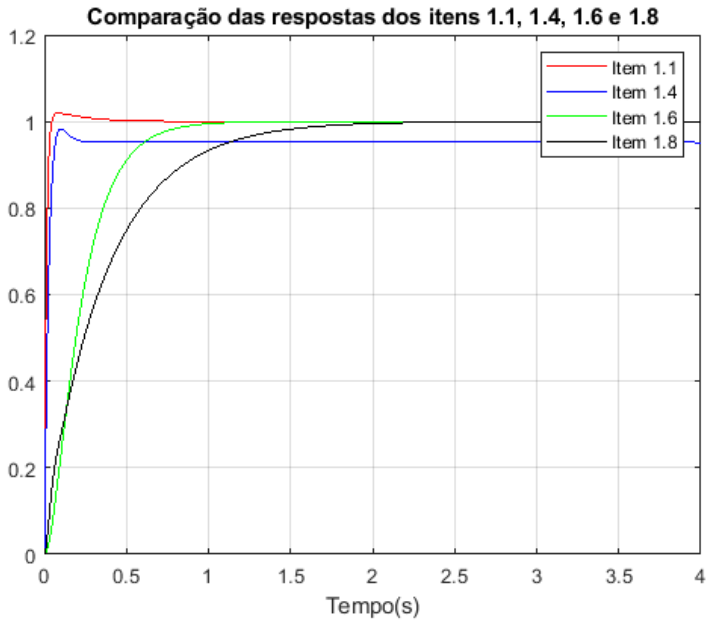
Fonte: Produção do próprio autor.

Figura 45 – Ajuste de erro - Resposta da realimentação de estados + Observador (fator 5)



Fonte: Produção do próprio autor.

Figura 46 – Resposta da realimentação de estados + Observador



Fonte: Produção do próprio autor.

REFERÊNCIAS BIBLIOGRÁFICAS

DORF, R. C.; BISHOP, R. H. Sistemas de controle modernos. 8ª edição. Tradução: Bernardo Severo da Silva Filho, LTC–Livros Técnicos e Científicos Editora SA, Rio de Janeiro, RJ, 1998. Citado na página 40.

Anexos

.1 Item 1.1

```
% ========================================================================
%
%         SEGUNDO TRABALHO COMPUTACIONAL DE SISTEMAS REALIMENTADOS
%
% Alaf do Nascimento Santos - 20171007181
% Felipe Antonio Moreira Silva - 2018205316
% ========================================================================

% Definição da FTMA
clear all, close all, clc;
syms s x;
N = 9; % Numero da Dupla
num = 100*N;   % Implemetação do numerador FTMA
d = (s+(25-N))*(s+25/N);
denom = sym2poly(d);   % Implemetação do denominador FTMA
Gma = tf(num, denom); % Definição da FTMA
save('Gma','Gma');

%% Questão 1.1 - (i) T=0 seg. (sem atraso de transporte)

Gma_poly = poly2sym(num,s)/poly2sym(d,s);

% Definição da FTMA
Gma = tf(num, denom); % Definição da FTMA


% ETAPA 1 ================================================================

[Gm,Pm,Wgm,Wpm] = margin(Gma);

BW_Gma = Wpm; % Largura de Banda de Gma
MF_Gma = Pm; % Margem de Fase de Gma

GMF = feedback(Gma,1); % Funcao com Malha Fechada

% Calculo de Erros
```

```
Lim_ErrDegrau_y = 1/(1+Gma_poly);
Lim_ErrDist_y = Gma_poly/(1+Gma_poly);

% Erro a entrada a Degrau
ErroDegrau_y = double(limit(Lim_ErrDegrau_y,s,0,'right'));

% Erro a entrada a Disturbio
ErroDist_y = double(limit(Lim_ErrDist_y,s,0,'right'));

% ETAPA 2 e 3 ============================================================

% Nova frequencia de cruzamento de ganho
G_w0 = evalfr(Gma, BW_Gma);

% Calculo do Ganho proporcinal
relacao = 20*log10(x)+ 20*log10(G_w0)==0;
Kpi_sym = solve(relacao,x);
Kpi = sym2poly(Kpi_sym);

Gma2 = Gma*Kpi; % Adicao do controlador P


% ETAPA 4 ================================================================

% Frequencia de corte do PI
Kii=Kpi*BW_Gma/5;
PI=tf([Kpi Kii],[1 0]);  % Implementecao do Controlador PI
PI_poly = (Kpi*s + Kii)/s;

GGma = Gma*PI; % Adicao do controlador PI

[Gm_PI,Pm_PI,Wgm_PI,Wpm_PI]=margin(GGma);
BW_GGma = Wpm_PI; % Largura de Banda de GGma
MF_GGma = Pm_PI; % Margem de Fase de GGma

GGMF=feedback(GGma,1); % Funcao com Malha Fechada

% Calculo de Erros
Lim_ErrDegrau_y1 = 1/(1+PI_poly*Gma_poly);
```

```
Lim_ErrDist_y1 = Gma_poly/(1+PI_poly*Gma_poly);

% Erro a entrada a Degrau
ErroDegrau_y1 = double(limit(Lim_ErrDegrau_y1,s,0,'right'));

% Erro a entrada a Disturbio
ErroDist_y1 = double(limit(Lim_ErrDist_y1,s,0,'right'));

% ETAPA 5 ============================================================

Kdd=1/BW_GGma; % Frequencia de corte do PD
npid=conv([Kpi Kii],[Kdd 1]);
dpid=[1 0];
PID=tf(npid,dpid); % Implementecao do Controlador PID

GGGma=Gma*PID; % Adicao do controlador PID

[Gm_PID,Pm_PID,Wgm_PID,Wpm_PID]=margin(GGGma);
BW_GGGma = Wpm_PID; % Largura de Banda de GGGma
MF_GGGma = Pm_PID; % Margem de Fase de GGGma

GGGMF=feedback(GGGma,1); % Funcao com Malha Fechada

% Calculo de Erros
PID_poly = poly2sym(npid,s)/poly2sym(dpid,s);
Lim_ErrDegrau_y2 = 1/(1+PID_poly*Gma_poly);
Lim_ErrDist_y2 = Gma_poly/(1+PID_poly*Gma_poly);

% Erro a entrada a Degrau
ErroDegrau_y2 = double(limit(Lim_ErrDegrau_y2,s,0,'right'));

% Erro a entrada a Disturbio
ErroDist_y2 = double(limit(Lim_ErrDist_y2,s,0,'right'));

% REPETIÇÃO 1 ETAPA 5 ================================================

Kdd2 = 3/Wpm_PID; % Frequencia de corte do PD
npid2 = conv([Kpi Kii],[Kdd2 1]);
dpid2 = [1 0];
```

```
PID2 = tf(npid2,dpid2);  % Implementecao do Controlador PID2

GGGma2=Gma*PID2; % Adicao do controlador PID2

[Gm_PID2, Pm_PID2, Wgm_PID2, Wpm_PID2]=margin(GGGma2);
BW_GGGma2 = Wpm_PID2; % Largura de Banda de GGGma2
MF_GGGma2 = Pm_PID2; % Margem de Fase de GGGma2

GGGMF2=feedback(GGGma2,1); % Funcao com Malha Fechada

% Calculo de Erros
PID2_poly = poly2sym(npid2,s)/poly2sym(dpid2,s);
Lim_ErrDegrau_y3 = 1/(1+PID2_poly*Gma_poly);
Lim_ErrDist_y3 = Gma_poly/(1+PID2_poly*Gma_poly);

% Erro a entrada a Degrau
ErroDegrau_y3 = double(limit(Lim_ErrDegrau_y3,s,0,'right'));

% Erro a entrada a Disturbio
ErroDist_y3 = double(limit(Lim_ErrDist_y3,s,0,'right'));



t=0.0:0.01:1.4;
y0 = step(GMF,t);
q0 = stepinfo(y0)

y1 = step(GGMF,t);
q1 = stepinfo(y1)

y2=step(GGGMF,t);
q2 = stepinfo(y2)

y3 = step(GGGMF2,t);
q3 = stepinfo(y3)

% TABELA ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Overshoot = [q0.Overshoot; q1.Overshoot; q2.Overshoot; q3.Overshoot];
SettlingTime = [q0.SettlingTime;
```

```
                q1.SettlingTime;
                q2.SettlingTime;
                q3.SettlingTime];
RiseTime = [q0.RiseTime; q1.RiseTime; q2.RiseTime; q3.RiseTime];

FTMAs = {'Gma';'GGma';'GGGma';'GGGma2'};
Controladores = {'0';'PI';'PID';'PID2'};
Gms = [Gm; Gm_PI; Gm_PID; Gm_PID2];
BWs = [BW_Gma;BW_GGma;BW_GGGma;BW_GGGma2];
MFs = [MF_Gma;MF_GGma;MF_GGGma;MF_GGGma2];
Ess_Degrau= [ErroDegrau_y;ErroDegrau_y1;ErroDegrau_y2;ErroDegrau_y3];
Ess_Disturbio= [ErroDist_y;ErroDist_y1;ErroDist_y2;ErroDist_y3];

T = table(FTMAs,Controladores,Overshoot,
          SettlingTime,RiseTime,Gms,BWs,
          MFs,Ess_Degrau,Ess_Disturbio)

% PLOTAGENS ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

% ************************ DIAGRAMA DE BODE ******************************

% Diagrama de Bode da FTMA
figure;
bode(Gma);
title('Função de Transferência da Dinâmica do Veículo Lunar');
grid on;


% Diagrama de Bode da Gma com PI
figure;
hold on;
bode(Gma);
bode(GGma);
title('Diagrama de Bode para o Controlador PI');
grid on;
legend('Gma','Gma+PI');
hold off;
```

```
% Diagrama de Bode da Gma com PID, PID2 e PID3
figure;
hold on;
bode(Gma);
bode(GGGma);
bode(GGGma2);
title('Diagrama de Bode para os Controladores PID Projetados');
grid on;
legend('Gma','Gma+PID','Gma+PID2');
hold off;


% Diagrama de Bode da Gma com PI, PID, PID2 e PID3
figure;
hold on;
bode(Gma);
bode(GGma);
bode(GGGma);
bode(GGGma2);
title('Diagrama de Bode para os Controladores Projetados');
grid on;
legend('Gma','Gma+PI','Gma+PID','Gma+PID2');
hold off;


% ********************** RESPOSTA AO DEGRAU ****************************
figure;
hold on;

plot(t,y1);
plot(t,y2);
plot(t,y3);

title('Resposta ao Degrau para FTMF com Controladores Projetados');
grid on;
legend('Controlador PI','Controlador PID','Controlador PID2');
hold off;

save('PID','PID2'); %para chamar no item 1.2
```

```
%% Questão 1.1 - (ii) T = 0.1/N seg.

% PROJETO PID COM ATRASO DE TRANSPORTE

T = 0.1/N; % Atraso de transporte [seg]
Gma_atraso_poly = poly2sym(num*exp(-T*s),s)/poly2sym(d,s)

% Definicao da Gma com atraso
Gma_atraso = tf(num, denom, 'InputDelay',T);


% ETAPA 1 ================================================================

[Gm_atraso, Pm_atraso, Wgm_atraso, Wpm_atraso] = margin(Gma_atraso);

BW_Gma_atraso = Wpm_atraso; % Largura de Banda de Gma
MF_Gma_atraso = Pm_atraso; % Margem de Fase de Gma

GMF_atraso = feedback(Gma_atraso,1); % Funcao com Malha Fechada

% Calculo de Erros
Lim_ErrDegrau_y_atraso = 1/(1+Gma_atraso_poly);
Lim_ErrDist_y_atraso = Gma_atraso_poly/(1+Gma_atraso_poly);

% Erro a entrada a Degrau
ErroDegrau_y_atraso = double(limit(Lim_ErrDegrau_y_atraso, s, 0, 'right'));

% Erro a entrada a Disturbio
ErroDist_y_atraso = double(limit(Lim_ErrDist_y_atraso, s, 0, 'right'));


% ETAPA 2 e 3 ============================================================

% Nova frequencia de cruzamento de ganho
G_w0_atraso = evalfr(Gma_atraso, BW_Gma_atraso);

% Calculo do Ganho proporcinal
relacao_atraso = 20*log10(x)+ 20*log10(G_w0_atraso)==0;
```

```
Kpi_sym_atraso = solve(relacao_atraso,x);
Kpi_atraso = sym2poly(Kpi_sym_atraso);

% Diminuição do Kp para o sistema ser estável em MF
Kpi_atraso = 0.1 * Kpi_atraso;

Gma2_atraso = Gma_atraso*Kpi_atraso; % Adicao do controlador P


% ETAPA 4 ================================================================

% Frequencia de corte do PI
Kii_atraso=Kpi_atraso * BW_Gma_atraso/5;

% Implementecao do Controlador PI
PI_atraso=tf([Kpi_atraso Kii_atraso],[1 0]);
PI_poly_atraso = (Kpi_atraso*s + Kii_atraso)/s;

GGma_atraso = Gma_atraso*PI_atraso;  % Adicao do controlador PI

[Gm_PI_atraso,Pm_PI_atraso,Wgm_PI_atraso,Wpm_PI_atraso]=margin(GGma_atraso);
BW_GGma_atraso = Wpm_PI_atraso; % Largura de Banda de GGma
MF_GGma_atraso = Pm_PI_atraso; % Margem de Fase de GGma

GGMF_atraso=feedback(GGma_atraso,1);  % Funcao com Malha Fechada

% Calculo de Erros
Lim_ErrDegrau_y1_atraso = 1/(1+PI_poly_atraso*Gma_atraso_poly);
Lim_ErrDist_y1_atraso = Gma_atraso_poly/(1+PI_poly_atraso*Gma_atraso_poly);

% Erro a entrada a Degrau
ErroDegrau_y1_atraso = double(limit(Lim_ErrDegrau_y1_atraso,s,0,'right'));

% Erro a entrada a Disturbio
ErroDist_y1_atraso = double(limit(Lim_ErrDist_y1_atraso,s,0,'right'));


% ETAPA 5 ================================================================
```

```
Kdd_atraso=1/BW_GGma_atraso; % Frequencia de corte do PD
npid_atraso=conv([Kpi_atraso Kii_atraso],[Kdd_atraso 1]);
dpid_atraso=[1 0];

PID_atraso = tf(npid_atraso,dpid_atraso);  % Implementecao do Controlador PID
GGGma_atraso = Gma_atraso*PID_atraso; % Adicao do controlador PID

[Gm_PID_atraso,Pm_PID_atraso,Wgm_PID_atraso,Wpm_PID_atraso]=margin(GGGma_atraso);
BW_GGGma_atraso = Wpm_PID_atraso; % Largura de Banda de GGGma
MF_GGGma_atraso = Pm_PID_atraso; % Margem de Fase de GGGma

GGGMF_atraso=feedback(GGGma_atraso,1); % Funcao com Malha Fechada

% Calculo de Erros
PID_poly_atraso = poly2sym(npid_atraso,s)/poly2sym(dpid_atraso,s);
Lim_ErrDegrau_y2_atraso = 1/(1+PID_poly_atraso*Gma_atraso_poly);
Lim_ErrDist_y2_atraso = Gma_atraso_poly/(1+PID_poly_atraso*Gma_atraso_poly);

% Erro a entrada a Degrau
ErroDegrau_y2_atraso = double(limit(Lim_ErrDegrau_y2_atraso,s,0,'right'));

% Erro a entrada a Disturbio
ErroDist_y2_atraso = double(limit(Lim_ErrDist_y2_atraso,s,0,'right'));


% REPETIÇÃO 1 ETAPA 5 ================================================

Kdd2_atraso = 3/Wpm_PID_atraso; % Frequencia de corte do PD
npid2_atraso = conv([Kpi_atraso Kii_atraso],[Kdd2_atraso 1]);
dpid2_atraso = [1 0];
PID2_atraso = tf(npid2_atraso,dpid2_atraso); % Implementecao do Controlador PID2

GGGma2_atraso=Gma_atraso*PID2_atraso; % Adicao do controlador PID2

[Gm_PID2_atraso, Pm_PID2_atraso,
 Wgm_PID2_atraso, Wpm_PID2_atraso]=margin(GGGma2_atraso);
BW_GGGma2_atraso = Wpm_PID2_atraso; % Largura de Banda de GGGma2
MF_GGGma2_atraso = Pm_PID2_atraso; % Margem de Fase de GGGma2
```

```
GGGMF2_atraso=feedback(GGGma2_atraso,1); % Funcao com Malha Fechada

% Calculo de Erros
PID2_poly_atraso = poly2sym(npid2_atraso,s)/poly2sym(dpid2_atraso,s);
Lim_ErrDegrau_y3_atraso = 1/(1+PID2_poly_atraso*Gma_atraso_poly);
Lim_ErrDist_y3_atraso = Gma_atraso_poly/(1+PID2_poly_atraso*Gma_atraso_poly);

% Erro a entrada a Degrau
ErroDegrau_y3_atraso = double(limit(Lim_ErrDegrau_y3_atraso,s,0,'right'));

% Erro a entrada a Disturbio
ErroDist_y3_atraso = double(limit(Lim_ErrDist_y3_atraso,s,0,'right'));

t=0.0:0.01:2;
y0_atraso = step(GMF_atraso,t);
q0_atraso = stepinfo(y0_atraso)

y1_atraso=step(GGMF_atraso,t);
q1_atraso = stepinfo(y1_atraso)

y2_atraso=step(GGGMF_atraso,t);
q2_atraso = stepinfo(y2_atraso)

y3_atraso=step(GGGMF2_atraso,t);
q3_atraso = stepinfo(y3_atraso)

% TABELA ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Overshoot_atraso = [q0_atraso.Overshoot; q1_atraso.Overshoot; q2_atraso.Overshoot;

SettlingTime_atraso = [q0_atraso.SettlingTime;
                       q1_atraso.SettlingTime;
                       q2_atraso.SettlingTime;
                       q3_atraso.SettlingTime];

RiseTime_atraso = [q0_atraso.RiseTime;
                   q1_atraso.RiseTime;
                   q2_atraso.RiseTime;
                   q3_atraso.RiseTime];
```

```
FTMAs_atraso = {'Gma_atraso';'GGma_atraso';'GGGma_atraso';'GGGma2_atraso'};
Controladores_atraso = {'0';'PI';'PID';'PID2'};
Gms_atraso = [Gm_atraso; Gm_PI_atraso; Gm_PID_atraso; Gm_PID2_atraso];
BWs_atraso = [BW_Gma_atraso; BW_GGma_atraso; BW_GGGma_atraso; BW_GGGma2_atraso];
MFs_atraso = [MF_Gma_atraso; MF_GGma_atraso; MF_GGGma_atraso; MF_GGGma2_atraso];

Ess_Degrau_atraso = [ErroDegrau_y_atraso;
                     ErroDegrau_y1_atraso;
                     ErroDegrau_y2_atraso;
                     ErroDegrau_y3_atraso];

Ess_Disturbio_atraso = [ErroDist_y_atraso;
                        ErroDist_y1_atraso;
                        ErroDist_y2_atraso;
                        ErroDist_y3_atraso];

T_atraso = table(FTMAs_atraso,
                 Controladores_atraso,
                 Overshoot_atraso,
                 SettlingTime_atraso,
                 RiseTime_atraso,
                 Gms_atraso,
                 BWs_atraso,
                 MFs_atraso,
                 Ess_Degrau_atraso,
                 Ess_Disturbio_atraso)

% PLOTAGENS ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

% ************************ DIAGRAMA DE BODE *******************************

% Diagrama de Bode da FTMA
figure;
bode(Gma_atraso);
title('Função de Transferência com Atraso da Dinâmica do Veículo Lunar');
grid on;


% Diagrama de Bode da Gma com PI
```

```
figure;
hold on;
bode(Gma_atraso);
bode(GGma_atraso);
title('Diagrama de Bode para o Controlador PI');
grid on;
legend('Gma','Gma+PI');
hold off;


% Diagrama de Bode da Gma com PID, PID2 e PID3
figure;
hold on;
bode(Gma_atraso);
bode(GGGma_atraso);
bode(GGGma2_atraso);
title('Diagrama de Bode para os Controladores PID Projetados');
grid on;
legend('Gma','Gma+PID','Gma+PID2');
hold off;


% Diagrama de Bode da Gma com PI, PID, PID2 e PID3
figure;
hold on;
bode(Gma_atraso);
bode(GGma_atraso);
bode(GGGma_atraso);
bode(GGGma2_atraso);
title('Diagrama de Bode para os Controladores Projetados');
grid on;
legend('Gma','Gma+PI','Gma+PID','Gma+PID2');
hold off;


% ********************** RESPOSTA AO DEGRAU ****************************
figure;
hold on;
```

```
plot(t,y1_atraso);
plot(t,y2_atraso);
plot(t,y3_atraso);

title('Resposta ao Degrau para FTMF Atraso com Controladores Projetados');
grid on;
legend('Controlador PI','Controlador PID','Controlador PID2');
hold off;
```

.2 Item 1.2

```
% ========================================================================
%
%         SEGUNDO TRABALHO COMPUTACIONAL DE SISTEMAS REALIMENTADOS
%
% Alaf do Nascimento Santos - 20171007181
% Felipe Antonio Moreira Silva - 2018205316
% ========================================================================

clear all, close all, clc;
op = 2; % 1 se item (i), 2 se item (ii)

global F A B;
load('PID');
load('Gma');
Gpid = Gma*PID2;

GGGMF2=feedback(Gpid,1);

% Definição das Matrizes A, B, C e D
A = [-18.78 1 0;
     -44.44 0 1;
       0 0 0];
B = [88.1 1835 7493]';
C = [1 0 0];
D = 0;
```

```
%%%%%% Agora simular o veículo lunar no espaço de estados %%%%%%
% vetor estados inicial
Xs = [0 0 0];
F = 0;

% Inicialização de variáveis
tfinal = 5; % Tempo total de simulação
ref = 1; % Referência
passo = 0;
tempo = 0;
dT = 0.01; % tempo de amostragem

while (tempo < tfinal)
     passo = passo + 1;
     ts = passo*dT;

     % simulação da equação diferencial
     [T,X] = ode45('modelo_linearVeiculo',[tempo ts], Xs);

     % Armazenamento de variaveis
     n = length(T);
     tempo = ts;
     Xs = X(n,:);
     if(op == 2);
         F = ref - Xs(1);
     end
     if(op == 1);
         F = ref-(Xs(1) + 0.02*randn(1));
     end
     vetout(passo,:) = X(n,1);
     vettime(passo) = T(n,:);
     vetvref(passo,:) = ref;
end

% Plotagem dos gráficos
figure
plot(vettime, vetvref,'b');
hold on;
plot(vettime,vetout,'r'); %multiplica por p1 para reduzir o erro
```

```
xlabel('Tempo(s)');title('Direção');
legend('Referencia');
grid;
```

.3 Item 1.3

```
% ========================================================================
%
%         SEGUNDO TRABALHO COMPUTACIONAL DE SISTEMAS REALIMENTADOS
%
% Alaf do Nascimento Santos - 20171007181
% Felipe Antonio Moreira Silva - 2018205316
% ========================================================================

clear all, close all, clc;
global F A B;
load('PID');
load('Gma');
Gpid = Gma*PID2;

GGGMF2=feedback(Gpid,1);

% Definição das Matrizes A, B, C e D
A = [-18.78 1 0;
     -44.44 0 1;
       0 0 0];
B = [88.1 1835 7493]';
C = [1 0 0];
D = 0;


%%%%%% Agora simular o veículo lunar no espaço de estados %%%%%%
% vetor estados inicial
Xs = [0 0 0];
F = 0;

% Inicialização de variáveis
tfinal = 5; % Tempo total de simulação
```

```
ref = 1; % Referência
passo = 0;
tempo = 0;
dT = 0.01; % tempo de amostragem

while (tempo < tfinal)
     passo = passo + 1;
     ts = passo*dT;

     % simulação da equação diferencial
     [T,X] = ode45('modelo_linearVeiculo',[tempo ts], Xs);

     % Armazenamento de variaveis
     n = length(T);
     tempo = ts;
     Xs = X(n,:);
     F = ref - Xs(1);
     vetout(passo,:) = X(n,1);
     vettime(passo) = T(n,:);
     vetvref(passo,:) = ref;
end

% Plotagem dos gráficos

t=0.0:0.01:5;
y3=step(GGGMF2,t);

figure;
plot(vettime, vetout,'r');
hold on;
plot(t,y3,'g');
plot(vettime,vetvref,'b');
xlabel('Tempo(s)');title('Direção');
legend('Referencia');
grid;

title('Comparação das respostas dos itens 1.1 e 1.2');
grid on;
legend('Item 1.1','Item 1.2', 'ref');
```

```
hold off;
```

.4 Item 1.4

.4.1 1.1 (Atraso-Avanço)

.4.1.1 (i) T = 0s

```
% ========================================================================
%
%         SEGUNDO TRABALHO COMPUTACIONAL DE SISTEMAS REALIMENTADOS
%
% Alaf do Nascimento Santos - 20171007181
% Felipe Antonio Moreira Silva - 2018205316
% ========================================================================

% Definição da FTMA
clear all, close all, clc;
syms s x;
N = 9; % Numero da Dupla
num = 100*N;   % Implemetação do numerador FTMA
d = (s+(25-N))*(s+25/N);
denom = sym2poly(d);   % Implemetação do denominador FTMA
Gma = tf(num, denom); % Definição da FTMA
GMF = feedback(Gma,1); % Definição da FTMF
Gma_poly = poly2sym(num,s)/poly2sym(d,s); % FTMA Simbolica
save('Gma','Gma');
t=0:.001:1;

%% Questão 1.4 (1.1) - (i) T=0 seg. (sem atraso de transporte)

% Especificações:
% * Erro a entrada degrau e ao disturbio menores ou iguais a 0.1
% * Largura de Banda da FTMA maior possivel
% * Margem de Fase maior ou igual a 60 graus

[Gm1,Pm1,Wgm1,Wpm1] = margin(Gma);
```

```
% 1 ETAPA ========================================================================
% Determinação do ganho K1 que atenda ao erro menor que 0.1


% Desempenho em estado estacionário
Eq_Ess_Degrau = 1/(1 + Gma_poly); % Erro a entrada degrau
Eq_Ess_Disturbio = Gma_poly/(1 + Gma_poly); % Erro a entrada disturbio

Ess_Deg = double(limit(Eq_Ess_Degrau,s,0,'right'));
Ess_Dist = double(limit(Eq_Ess_Disturbio,s,0,'right'));

% Determinação do Kp
figure;
bode(Gma);
title('Função de Transferência da Dinâmica do Veículo Lunar');
grid on;

 % Atraves do diagrama de bode, temos que Kp = 26.1 db
Kp_db = 26.1; % Ganho proporcional estático em db
Kp = 10^(Kp_db/20); % Ganho proporcional estático

erro_est = 1/1+Kp; % Definição do erro estático utilizando Kp

% Especificaçãode K1 para erro ao degrau e disturbio requisitado

Erro_deg_k1 = 1/(1 + x*Gma_poly);
Erro_dist_k1 = Gma_poly/(1 + x*Gma_poly);

Eq_deg_K1 = limit(Erro_deg_k1,s,0,'right')== 0.1;
Eq_dist_K1 = limit(Erro_dist_k1,s,0,'right')== 0.1;

K1_Degrau_sym = solve(Eq_deg_K1,x); % K1 para atender erro ao Degrau
K1_Degrau = sym2poly(K1_Degrau_sym); % K1 = 0.4444

K1_Dist_sym = solve(Eq_dist_K1,x); % K1 para atender erro ao Disturbio
K1_Dist = sym2poly(K1_Dist_sym); % K1 = 9.9506

% Analisando ambos valores de K1, definimos K1 para atender ambos
```

```
K1 = 15; % Ganho K1 para Atraso

Eq_Erro_deg = 1/(1 +K1*Gma_poly);
Eq_Erro_dist = Gma_poly/(1 + K1*Gma_poly);

Erro_deg = double(limit(Eq_Erro_deg,s,0,'right')); % Erro a entrada a Degrau
Erro_dist = double(limit(Eq_Erro_dist,s,0,'right'));% Erro a entrada a Disturbio

% 2 ETAPA  ================================================================

% Projeto do Controlador de Atraso de Fase

K1Gma = K1*Gma;

[Gm,Pm,Wgm,Wpm] = margin(K1Gma);

BW_Gma = Wpm; % Largura de Banda de K1Gma
MF_Gma = Pm; % Margem de Fase de K1Gma

% 3 ETAPA  ================================================================

% Determinação da frequência de cruzamento de ganho nova
fase_K1Gma = 60 - 180;
% Fase K1Gma = -120

figure;
bode(K1Gma);
title('Função de Transferência de Gma com K1');
grid on;

% Através do Bode de K1Gma
fase_w0db = 14; % rad/s
md_w0db = 33;

% 4 ETAPA  ================================================================

% Determinação da constante alpha

eq_alfa = - 20*log10(x) == 20*log10(md_w0db); % Relação de Alfa
```

```
alfa_sym = solve(eq_alfa,x);
alfa = sym2poly(alfa_sym);

% 5 ETAPA  ================================================================

% Determinação das a frequência de corte

% Frequencia de corte Zero do Compensador - 1/alfa*tau (-1 dec de md_w0db)
syms tau_eq
w0db_decAbaixo = fase_w0db - 10;
eq_fcorte_zero = 1/(alfa*tau_eq) - w0db_decAbaixo;
tau_sym = solve(eq_fcorte_zero,tau_eq);
tau = sym2poly(tau_sym);
fc_zero = 1/alfa*tau;

% Frequencia de corte Polo do Compensador - 1/tau
fc_polo = 1/tau;

% 6 ETAPA  ================================================================


% Definição do Controlador Atraso
syms jw
C_atraso_sym = K1*(1+jw*alfa*tau)/(1+jw*tau); % Equação do Controlador Atraso

[num_Catraso, deno_Catraso] = numden(C_atraso_sym);

C_atraso = tf(sym2poly(num_Catraso),sym2poly(num_Catraso));

GGma = C_atraso*Gma; % Controlador Atraso + Gma

[Gm_GGma,Pm_GGma,Wgm_GGma,Wpm_GGma]=margin(GGma);
MG_GGma = Gm_GGma; % Margem de Ganho de GGma
BW_GGma = Wpm_GGma; % Largura de Banda de GGma
MF_GGma = Pm_GGma; % Margem de Fase de GGma


GGMF = feedback(GGma,1);
```

```
C_atraso_poly = subs(C_atraso_sym,jw,s);
 % Erro a entrada degrau
Eq_Ess_Degrau_GGma = 1/(1 + C_atraso_poly*Gma_poly);

 % Erro a entrada disturbio
Eq_Ess_Disturbio_GGma = Gma_poly/(1 + C_atraso_poly*Gma_poly);

Ess_Deg_GGma = double(limit(Eq_Ess_Degrau_GGma,s,0,'right'));
Ess_Dist_GGma = double(limit(Eq_Ess_Disturbio_GGma,s,0,'right'));

% 7 ETAPA  ================================================================

% Determinação do ganho K2 para o Avanço
K2 = 1;

% Determinação da nova fase
phi_m = 75 - Pm_GGma; % Margem de fase especificada 60 com adicional


% 8 ETAPA  ================================================================

% Determinação da constante alfa para o controlador Avanço

syms alfa_Avanco_x
Eq_Alfa_Avanco =  tand(phi_m) == (alfa_Avanco_x-1)/(2*sqrt(alfa_Avanco_x));
alfa_avanco_sym = solve(Eq_Alfa_Avanco,alfa_Avanco_x);
alfa2 = sym2poly(alfa_avanco_sym); % Alfa do Controlador Avanço


% 9 ETAPA  ================================================================

% Determinação da frequência onde ocorre o máximo avanço de fase

relacao_alfa = -10*log10(alfa2); % Parte do alfa da relacao
% relacao alfa =  -6.5175 db

% Analisando o diagrama de bode para GGma, temos que Wm = 37.6 rad/s
figure;
bode(GGma);
```

```
title('Função de Transferência de Gma com Compesador Atraso');
grid on;

wm = 37.6; % Frequência onde ocorre o máximo avanço de fase

% 10 ETAPA  ============================================================

% Definição da Constante Tau para Avanço pela relação

tau2 = 1/(wm*sqrt(alfa2));

% Frequencia de corte Zero do Compensador
fc_zero2 = 1/(alfa2*tau2);

% Frequencia de corte Polo do Compensador
fc_polo2 = 1/(tau2);

% 11 ETAPA  ============================================================

% Definição do Controlador Avanco

C_avanco_sym = K2*(1+jw*alfa2*tau2)/(1+jw*tau2);
[num_Cavanco, deno_Cavanco] = numden(C_avanco_sym);

C_avanco = tf(sym2poly(num_Cavanco),sym2poly(deno_Cavanco));

GGGma = C_avanco*Gma; % Controlador Avanco + Gma

GGGMF = feedback(GGGma,1);

GGGGma = C_avanco*GGma; % Controlador Avanco + Controlador Atraso + Gma

C_avanco_atraso = C_avanco*C_atraso;


[Gm_GGGma,Pm_GGGma,Wgm_GGGma,Wpm_GGGma]=margin(GGGma);
MG_GGGma = Gm_GGGma; % Margem de Ganho de GGGma
BW_GGGma = Wpm_GGGma; % Largura de Banda de GGGma
MF_GGGma = Pm_GGGma; % Margem de Fase de GGGma
```

```
[Gm_GGGGma,Pm_GGGGma,Wgm_GGGGma,Wpm_GGGGma]=margin(GGGGma);
MG_GGGGma = Gm_GGGGma; % Margem de Ganho de GGGGma
BW_GGGGma = Wpm_GGGGma; % Largura de Banda de GGGGma
MF_GGGGma = Pm_GGGGma; % Margem de Fase de GGGGma

GGGGMF = feedback(GGGGma,1);
C_avanco_atraso_sym = C_atraso_sym*C_avanco_sym;
C_avanco_atraso_poly = subs(C_avanco_atraso_sym,jw,s);

% Erro a entrada degrau
Eq_Ess_Degrau_GGGGma = 1/(1 + C_avanco_atraso_poly*Gma_poly);

% Erro a entrada disturbio
Eq_Ess_Disturbio_GGGGma = Gma_poly/(1 + C_avanco_atraso_poly*Gma_poly);

Ess_Deg_GGGGma = double(limit(Eq_Ess_Degrau_GGGGma,s,0,'right'));
Ess_Dist_GGGGma = double(limit(Eq_Ess_Disturbio_GGGGma,s,0,'right'));


% Malha Fechada ------------------------------------------------------

y0 = step(GMF,t);
q0 = stepinfo(y0);

y1 = step(GGMF,t);
q1 = stepinfo(y1);

y2 = step(GGGMF,t);
q2 = stepinfo(y2);

y3 = step(GGGGMF,t);
q3 = stepinfo(y3);

% TABELA ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

Overshoot = [q1.Overshoot; q3.Overshoot];
SettlingTime = [q1.SettlingTime;q3.SettlingTime];
```

```
RiseTime = [q1.RiseTime; q3.RiseTime];

FTMAs = {'GGma';'GGGGma'};
Controladores = {'Atraso';'Avanco-Atraso'};
Gms = [MG_GGma; MG_GGGGma];
BWs = [BW_GGma; BW_GGGGma];
MFs = [MF_GGma; MF_GGGGma];
Ess_Degrau= [Ess_Deg_GGma;Ess_Deg_GGGGma];
Ess_Disturbio= [Ess_Dist_GGma;Ess_Dist_GGGGma];

T = table(FTMAs,Controladores,Overshoot,SettlingTime,
          RiseTime,Gms,BWs,MFs,Ess_Degrau,Ess_Disturbio)


% PLOTAGENS ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

% ************************ DIAGRAMA DE BODE ******************************

% Diagrama de Bode da Gma, com C_Atraso, com C_Avanco, C_Avanco_C_Atraso
figure;
hold on;
bode(Gma);
bode(GGma);
bode(GGGma);
bode(GGGGma);
title('Diagrama de Bode para o Controladores projetados');
grid on;
legend('Gma','Gma + Controlador Atraso', 'Gma + Controlador Avanco',
        'Gma + Controlador Avanco-Atraso');
hold off;

% ************************ RESPOSTA AO DEGRAU ****************************

figure;
plot(t,y1,'r',t,y2,'g');
title('Resposta ao degrau para FTMF com o Controladores projetados');
legend('FTMF com Atraso','FTMF com Atraso-Avanco');
grid on;
save('C_avanco_atraso','C_avanco_atraso');
```

.4.1.2 (ii) T = 0.1/N s

```
% ========================================================================
%
%          SEGUNDO TRABALHO COMPUTACIONAL DE SISTEMAS REALIMENTADOS
%
% Alaf do Nascimento Santos - 20171007181
% Felipe Antonio Moreira Silva - 2018205316
% ========================================================================

% Definição da FTMA
clear all, close all, clc;
syms s x;
N = 9; % Numero da Dupla
num = 100*N;   % Implemetação do numerador FTMA
d = (s+(25-N))*(s+25/N);
denom = sym2poly(d);   % Implemetação do denominador FTMA
Gma = tf(num, denom); % Definição da FTMA
GMF = feedback(Gma,1); % Definição da FTMF
Gma_poly = poly2sym(num,s)/poly2sym(d,s); % FTMA Simbolica
save('Gma','Gma');
t=0:.001:5;

%% Questão 1.4 (1.1) - (ii) T = 0.1/N seg.

T = 0.1/N; % Atraso de transporte [seg]
Gma_poly = poly2sym(num*exp(-T*s),s)/poly2sym(d,s);

% Definicao da Gma com atraso
Gma = tf(num, denom, 'InputDelay',T);

[Gm1,Pm1,Wgm1,Wpm1] = margin(Gma);

% 1 ETAPA ================================================================
% Determinação do ganho K1 que atenda ao erro menor que 0.1


% Desempenho em estado estacionário
Eq_Ess_Degrau = 1/(1 + Gma_poly); % Erro a entrada degrau
```

```
Eq_Ess_Disturbio = Gma_poly/(1 + Gma_poly); % Erro a entrada disturbio

Ess_Deg = double(limit(Eq_Ess_Degrau,s,0,'right'));
Ess_Dist = double(limit(Eq_Ess_Disturbio,s,0,'right'));

% Determinação do Kp
figure;
bode(Gma);
title('FT da Dinâmica do Veículo Lunar com Atraso de Transporte');
grid on;

 % Atraves do diagrama de bode, temos que Kp = 26.1 db
Kp_db = 26.1; % Ganho proporcional estático em db
Kp = 10^(Kp_db/20); % Ganho proporcional estático

erro_est = 1/1+Kp; % Definição do erro estático utilizando Kp

% Especificaçãode K1 para erro ao degrau e disturbio requisitado

Erro_deg_k1 = 1/(1 + x*Gma_poly);
Erro_dist_k1 = Gma_poly/(1 + x*Gma_poly);

Eq_deg_K1 = limit(Erro_deg_k1,s,0,'right')== 0.1;
Eq_dist_K1 = limit(Erro_dist_k1,s,0,'right')== 0.1;

K1_Degrau_sym = solve(Eq_deg_K1,x); % K1 para atender erro ao Degrau
K1_Degrau = sym2poly(K1_Degrau_sym); % K1 = 0.4444

K1_Dist_sym = solve(Eq_dist_K1,x); % K1 para atender erro ao Disturbio
K1_Dist = sym2poly(K1_Dist_sym); % K1 = 9.9506

% Analisando ambos valores de K1, definimos K1 para atender ambos
K1 = 10; % Ganho K1 para Atraso

Eq_Erro_deg = 1/(1 +K1*Gma_poly);
Eq_Erro_dist = Gma_poly/(1 + K1*Gma_poly);

Erro_deg = double(limit(Eq_Erro_deg,s,0,'right')); % Erro a entrada a Degrau
Erro_dist = double(limit(Eq_Erro_dist,s,0,'right'));% Erro a entrada a Disturbio
```

```
% 2 ETAPA  ================================================================

% Projeto do Controlador de Atraso de Fase

K1Gma = K1*Gma;

[Gm,Pm,Wgm,Wpm] = margin(K1Gma);

BW_Gma = Wpm; % Largura de Banda de K1Gma
MF_Gma = Pm; % Margem de Fase de K1Gma

% 3 ETAPA  ================================================================

% Determinação da frequência de cruzamento de ganho nova
fase_K1Gma = 20 - 180;
% Fase K1Gma = -160

figure;
bode(K1Gma);
title('Função de Transferência de Gma com K1');
grid on;

% Através do Bode de K1Gma
fase_w0db = 26.4; % rad/s
md_w0db = 20.8;

% 4 ETAPA  ================================================================

% Determinação da constante alpha

eq_alfa = - 20*log10(x) == 20*log10(md_w0db); % Relação de Alfa
alfa_sym = solve(eq_alfa,x);
alfa = sym2poly(alfa_sym);

% 5 ETAPA  ================================================================

% Determinação das a frequência de corte
```

```
% Frequencia de corte Zero do Compensador - 1/alfa*tau (-1 dec de md_w0db)
syms tau_eq
w0db_decAbaixo = fase_w0db - 10;
eq_fcorte_zero = 1/(alfa*tau_eq) - w0db_decAbaixo;
tau_sym = solve(eq_fcorte_zero,tau_eq);
tau = sym2poly(tau_sym);
fc_zero = 1/alfa*tau;

% Frequencia de corte Polo do Compensador - 1/tau
fc_polo = 1/tau;

% 6 ETAPA  ================================================================


% Definição do Controlador Atraso
syms jw
C_atraso_sym = K1*(1+jw*alfa*tau)/(1+jw*tau); % Equação do Controlador Atraso

[num_Catraso, deno_Catraso] = numden(C_atraso_sym);

C_atraso = tf(sym2poly(num_Catraso),sym2poly(num_Catraso));

GGma = C_atraso*Gma; % Controlador Atraso + Gma

[Gm_GGma,Pm_GGma,Wgm_GGma,Wpm_GGma]=margin(GGma);
MG_GGma = Gm_GGma; % Margem de Ganho de GGma
BW_GGma = Wpm_GGma; % Largura de Banda de GGma
MF_GGma = Pm_GGma; % Margem de Fase de GGma


GGMF = feedback(GGma,1);

C_atraso_poly = subs(C_atraso_sym,jw,s);
Eq_Ess_Degrau_GGma = 1/(1 + C_atraso_poly*Gma_poly); % Erro a entrada degrau
% Erro a entrada disturbio
Eq_Ess_Disturbio_GGma = Gma_poly/(1 + C_atraso_poly*Gma_poly);

Ess_Deg_GGma = double(limit(Eq_Ess_Degrau_GGma,s,0,'right'));
Ess_Dist_GGma = double(limit(Eq_Ess_Disturbio_GGma,s,0,'right'));
```

```
% 7 ETAPA  ================================================================

% Determinação do ganho K2 para o Avanço
K2 = 1;

% Determinação da nova fase
phi_m = 90 - Pm_GGma; % Margem de fase especificada 60 com adicional


% 8 ETAPA  ================================================================

% Determinação da constante alfa para o controlador Avanço

syms alfa_Avanco_x
Eq_Alfa_Avanco =  tand(phi_m) == (alfa_Avanco_x-1)/(2*sqrt(alfa_Avanco_x));
alfa_avanco_sym = solve(Eq_Alfa_Avanco,alfa_Avanco_x);
alfa2 = sym2poly(alfa_avanco_sym); % Alfa do Controlador Avanço


% 9 ETAPA  ================================================================

% Determinação da frequência onde ocorre o máximo avanço de fase

relacao_alfa = -10*log10(alfa2); % Parte do alfa da relacao
% relacao alfa =  -10.62 db

% Analisando o diagrama de bode para GGma, temos que Wm = 60.6 rad/s
figure;
bode(GGma);
title('Função de Transferência de Gma com Compesador Atraso');
grid on;

wm = 75.3; % Frequência onde ocorre o máximo avanço de fase

% 10 ETAPA  ===============================================================

% Definição da Constante Tau para Avanço pela relação
```

```
tau2 = 1/(wm*sqrt(alfa2));

% Frequencia de corte Zero do Compensador
fc_zero2 = 1/(alfa2*tau2);

% Frequencia de corte Polo do Compensador
fc_polo2 = 1/(tau2);

% 11 ETAPA  ============================================================

% Definição do Controlador Avanco

C_avanco_sym = K2*(1+jw*alfa2*tau2)/(1+jw*tau2);
[num_Cavanco, deno_Cavanco] = numden(C_avanco_sym);

C_avanco = tf(sym2poly(num_Cavanco),sym2poly(deno_Cavanco));

GGGma = C_avanco*Gma; % Controlador Avanco + Gma

GGGMF = feedback(GGGma,1);

GGGGma = C_avanco*GGma; % Controlador Avanco + Controlador Atraso + Gma

C_avanco_atraso = C_avanco*C_atraso;


[Gm_GGGma,Pm_GGGma,Wgm_GGGma,Wpm_GGGma]=margin(GGGma);
MG_GGGma = Gm_GGGma; % Margem de Ganho de GGGma
BW_GGGma = Wpm_GGGma; % Largura de Banda de GGGma
MF_GGGma = Pm_GGGma; % Margem de Fase de GGGma


[Gm_GGGGma,Pm_GGGGma,Wgm_GGGGma,Wpm_GGGGma]=margin(GGGGma);
MG_GGGGma = Gm_GGGGma; % Margem de Ganho de GGGGma
BW_GGGGma = Wpm_GGGGma; % Largura de Banda de GGGGma
MF_GGGGma = Pm_GGGGma; % Margem de Fase de GGGGma

GGGGMF = feedback(GGGGma,1);
C_avanco_atraso_sym = C_atraso_sym*C_avanco_sym;
```

```
C_avanco_atraso_poly = subs(C_avanco_atraso_sym,jw,s);

 % Erro a entrada degrau
Eq_Ess_Degrau_GGGGma = 1/(1 + C_avanco_atraso_poly*Gma_poly);

% Erro a entrada disturbio
Eq_Ess_Disturbio_GGGGma = Gma_poly/(1 + C_avanco_atraso_poly*Gma_poly);

Ess_Deg_GGGGma = double(limit(Eq_Ess_Degrau_GGGGma,s,0,'right'));
Ess_Dist_GGGGma = double(limit(Eq_Ess_Disturbio_GGGGma,s,0,'right'));


% Malha Fechada ------------------------------------------------------------

y0 = step(GMF,t);
q0 = stepinfo(y0);

y1 = step(GGMF,t);
q1 = stepinfo(y1);

y2 = step(GGGMF,t);
q2 = stepinfo(y2);

y3 = step(GGGGMF,t);
q3 = stepinfo(y3);

% TABELA ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

Overshoot = [q1.Overshoot; q3.Overshoot];
SettlingTime = [q1.SettlingTime;q3.SettlingTime];
RiseTime = [q1.RiseTime; q3.RiseTime];

FTMAs = {'GGma';'GGGGma'};
Controladores = {'Atraso';'Avanco-Atraso'};
Gms = [MG_GGma; MG_GGGGma];
BWs = [BW_GGma; BW_GGGGma];
MFs = [MF_GGma; MF_GGGGma];
Ess_Degrau= [Ess_Deg_GGma;Ess_Deg_GGGGma];
Ess_Disturbio= [Ess_Dist_GGma;Ess_Dist_GGGGma];
```

```
T = table(FTMAs,Controladores,Overshoot,
SettlingTime,RiseTime,Gms,BWs,MFs,Ess_Degrau,Ess_Disturbio)


% PLOTAGENS ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

% ************************ DIAGRAMA DE BODE ******************************

% Diagrama de Bode da Gma, com C_Atraso, com C_Avanco, C_Avanco_C_Atraso
figure;
hold on;
bode(Gma);
bode(GGma);
bode(GGGma);
bode(GGGGma);
title('Diagrama de Bode para o Controladores projetados');
grid on;
legend('Gma','Gma + C Atraso', 'Gma + C Avanco', 'Gma + C Avanco-Atraso');
hold off;

% ************************ RESPOSTA AO DEGRAU ****************************

figure;
plot(t,y1,'r',t,y2,'g');
title('Resposta ao degrau para FTMF com o Controladores projetados');
legend('FTMF com Atraso','FTMF com Atraso-Avanco');
grid on;
save('C_avanco_atraso','C_avanco_atraso');

%%
%% Questão 1.4 (1.1) - (ii) T = 0.1/N seg.

T = 0.1/N; % Atraso de transporte [seg]
Gma_poly = poly2sym(num*exp(-T*s),s)/poly2sym(d,s);

% Definicao da Gma com atraso
Gma = tf(num, denom, 'InputDelay',T);
```

```
[Gm1,Pm1,Wgm1,Wpm1] = margin(Gma);

% 1 ETAPA ========================================================================
% Determinação do ganho K1 que atenda ao erro menor que 0.1


% Desempenho em estado estacionário
Eq_Ess_Degrau = 1/(1 + Gma_poly); % Erro a entrada degrau
Eq_Ess_Disturbio = Gma_poly/(1 + Gma_poly); % Erro a entrada disturbio

Ess_Deg = double(limit(Eq_Ess_Degrau,s,0,'right'));
Ess_Dist = double(limit(Eq_Ess_Disturbio,s,0,'right'));

% Determinação do Kp
figure;
bode(Gma);
title('Função de Transferência da Dinâmica do Veículo Lunar com
Atraso de Transporte');
grid on;

 % Atraves do diagrama de bode, temos que Kp = 26.1 db
Kp_db = 26.1; % Ganho proporcional estático em db
Kp = 10^(Kp_db/20); % Ganho proporcional estático

erro_est = 1/1+Kp; % Definição do erro estático utilizando Kp

% Especificaçãode K1 para erro ao degrau e disturbio requisitado

Erro_deg_k1 = 1/(1 + x*Gma_poly);
Erro_dist_k1 = Gma_poly/(1 + x*Gma_poly);

Eq_deg_K1 = limit(Erro_deg_k1,s,0,'right')== 0.1;
Eq_dist_K1 = limit(Erro_dist_k1,s,0,'right')== 0.1;

K1_Degrau_sym = solve(Eq_deg_K1,x); % K1 para atender erro ao Degrau
K1_Degrau = sym2poly(K1_Degrau_sym); % K1 = 0.4444

K1_Dist_sym = solve(Eq_dist_K1,x); % K1 para atender erro ao Disturbio
K1_Dist = sym2poly(K1_Dist_sym); % K1 = 9.9506
```

```
% Analisando ambos valores de K1, definimos K1 para atender ambos
K1 = 10; % Ganho K1 para Atraso

Eq_Erro_deg = 1/(1 +K1*Gma_poly);
Eq_Erro_dist = Gma_poly/(1 + K1*Gma_poly);

Erro_deg = double(limit(Eq_Erro_deg,s,0,'right')); % Erro a entrada a Degrau
Erro_dist = double(limit(Eq_Erro_dist,s,0,'right'));% Erro a entrada a Disturbio

% 2 ETAPA  ================================================================

% Projeto do Controlador de Atraso de Fase

K1Gma = K1*Gma;

[Gm,Pm,Wgm,Wpm] = margin(K1Gma);

BW_Gma = Wpm; % Largura de Banda de K1Gma
MF_Gma = Pm; % Margem de Fase de K1Gma

% 3 ETAPA  ================================================================

% Determinação da frequência de cruzamento de ganho nova
fase_K1Gma = 80 - 180;
% Fase K1Gma = -100

figure;
bode(K1Gma);
title('Função de Transferência de Gma com K1');
grid on;

% Através do Bode de K1Gma
fase_w0db = 7.65; % rad/s
md_w0db = 37;

% 4 ETAPA  ================================================================

% Determinação da constante alpha
```

```
eq_alfa = - 20*log10(x) == 20*log10(md_w0db); % Relação de Alfa
alfa_sym = solve(eq_alfa,x);
alfa = sym2poly(alfa_sym);

% 5 ETAPA  ================================================================

% Determinação das a frequência de corte

% Frequencia de corte Zero do Compensador - 1/alfa*tau (-1 dec de md_w0db)
syms tau_eq
w0db_decAbaixo = fase_w0db - 10;
eq_fcorte_zero = 1/(alfa*tau_eq) - w0db_decAbaixo;
tau_sym = solve(eq_fcorte_zero,tau_eq);
tau = sym2poly(tau_sym);
fc_zero = 1/alfa*tau;

% Frequencia de corte Polo do Compensador - 1/tau
fc_polo = 1/tau;

% 6 ETAPA  ================================================================


% Definição do Controlador Atraso
syms jw
C_atraso_sym = K1*(1+jw*alfa*tau)/(1+jw*tau); % Equação do Controlador Atraso

[num_Catraso, deno_Catraso] = numden(C_atraso_sym);

C_atraso = tf(sym2poly(num_Catraso),sym2poly(num_Catraso));

GGma = C_atraso*Gma; % Controlador Atraso + Gma

[Gm_GGma,Pm_GGma,Wgm_GGma,Wpm_GGma]=margin(GGma);
MG_GGma = Gm_GGma; % Margem de Ganho de GGma
BW_GGma = Wpm_GGma; % Largura de Banda de GGma
MF_GGma = Pm_GGma; % Margem de Fase de GGma
```

```
GGMF = feedback(GGma,1);

C_atraso_poly = subs(C_atraso_sym,jw,s);

% Erro a entrada degrau
Eq_Ess_Degrau_GGma = 1/(1 + C_atraso_poly*Gma_poly);
 % Erro a entrada disturbio
Eq_Ess_Disturbio_GGma = Gma_poly/(1 + C_atraso_poly*Gma_poly);

Ess_Deg_GGma = double(limit(Eq_Ess_Degrau_GGma,s,0,'right'));
Ess_Dist_GGma = double(limit(Eq_Ess_Disturbio_GGma,s,0,'right'));

% 7 ETAPA  ================================================================

% Determinação do ganho K2 para o Avanço
K2 = 1;

% Determinação da nova fase
phi_m = 100 - Pm_GGma; % Margem de fase especificada 60 com adicional


% 8 ETAPA  ================================================================

% Determinação da constante alfa para o controlador Avanço

syms alfa_Avanco_x
Eq_Alfa_Avanco =  tand(phi_m) == (alfa_Avanco_x-1)/(2*sqrt(alfa_Avanco_x));
alfa_avanco_sym = solve(Eq_Alfa_Avanco,alfa_Avanco_x);
alfa2 = sym2poly(alfa_avanco_sym); % Alfa do Controlador Avanço


% 9 ETAPA  ================================================================

% Determinação da frequência onde ocorre o máximo avanço de fase

relacao_alfa = -10*log10(alfa2); % Parte do alfa da relacao
% relacao alfa =  -23.3150 db

% Analisando o diagrama de bode para GGma, temos que Wm = 59 rad/s
```

```
figure;
bode(GGma);
title('Função de Transferência de Gma com Compesador Atraso');
grid on;

wm = 250; % Frequência onde ocorre o máximo avanço de fase

% 10 ETAPA  ================================================================

% Definição da Constante Tau para Avanço pela relação

tau2 = 1/(wm*sqrt(alfa2));

% Frequencia de corte Zero do Compensador
fc_zero2 = 1/(alfa2*tau2);

% Frequencia de corte Polo do Compensador
fc_polo2 = 1/(tau2);

% 11 ETAPA  ================================================================

% Definição do Controlador Avanco

C_avanco_sym = K2*(1+jw*alfa2*tau2)/(1+jw*tau2);
[num_Cavanco, deno_Cavanco] = numden(C_avanco_sym);

C_avanco = tf(sym2poly(num_Cavanco),sym2poly(deno_Cavanco));

GGGma = C_avanco*Gma; % Controlador Avanco + Gma

GGGMF = feedback(GGGma,1);

GGGGma = C_avanco*GGma; % Controlador Avanco + Controlador Atraso + Gma

C_avanco_atraso = C_avanco*C_atraso;


[Gm_GGGma,Pm_GGGma,Wgm_GGGma,Wpm_GGGma]=margin(GGGma);
MG_GGGma = Gm_GGGma; % Margem de Ganho de GGGma
```

```
BW_GGGma = Wpm_GGGma; % Largura de Banda de GGGma
MF_GGGma = Pm_GGGma; % Margem de Fase de GGGma


[Gm_GGGGma,Pm_GGGGma,Wgm_GGGGma,Wpm_GGGGma]=margin(GGGGma);
MG_GGGGma = Gm_GGGGma; % Margem de Ganho de GGGGma
BW_GGGGma = Wpm_GGGGma; % Largura de Banda de GGGGma
MF_GGGGma = Pm_GGGGma; % Margem de Fase de GGGGma

GGGGMF = feedback(GGGGma,1);
C_avanco_atraso_sym = C_atraso_sym*C_avanco_sym;
C_avanco_atraso_poly = subs(C_avanco_atraso_sym,jw,s);

 % Erro a entrada degrau
Eq_Ess_Degrau_GGGGma = 1/(1 + C_avanco_atraso_poly*Gma_poly);

% Erro a entrada disturbio
Eq_Ess_Disturbio_GGGGma = Gma_poly/(1 + C_avanco_atraso_poly*Gma_poly);

Ess_Deg_GGGGma = double(limit(Eq_Ess_Degrau_GGGGma,s,0,'right'));
Ess_Dist_GGGGma = double(limit(Eq_Ess_Disturbio_GGGGma,s,0,'right'));


% Malha Fechada ------------------------------------------------------------

y0 = step(GMF,t);
q0 = stepinfo(y0);

y1 = step(GGMF,t);
q1 = stepinfo(y1);

y2 = step(GGGMF,t);
q2 = stepinfo(y2);

y3 = step(GGGGMF,t);
q3 = stepinfo(y3);

% TABELA ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

```
Overshoot = [q1.Overshoot; q3.Overshoot];
SettlingTime = [q1.SettlingTime;q3.SettlingTime];
RiseTime = [q1.RiseTime; q3.RiseTime];

FTMAs = {'GGma';'GGGGma'};
Controladores = {'Atraso';'Avanco-Atraso'};
Gms = [MG_GGma; MG_GGGGma];
BWs = [BW_GGma; BW_GGGGma];
MFs = [MF_GGma; MF_GGGGma];
Ess_Degrau= [Ess_Deg_GGma;Ess_Deg_GGGGma];
Ess_Disturbio= [Ess_Dist_GGma;Ess_Dist_GGGGma];

T = table(FTMAs,Controladores,Overshoot,
        SettlingTime,RiseTime,Gms,BWs,MFs,Ess_Degrau,Ess_Disturbio)


% PLOTAGENS ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

% ************************ DIAGRAMA DE BODE ******************************

% Diagrama de Bode da Gma, com C_Atraso, com C_Avanco, C_Avanco_C_Atraso
figure;
hold on;
bode(Gma);
bode(GGma);
bode(GGGma);
bode(GGGGma);
title('Diagrama de Bode para o Controladores projetados');
grid on;
legend('Gma','Gma + Controlador Atraso',
       'Gma + Controlador Avanco', 'Gma + Controlador Avanco-Atraso');
hold off;

% ************************ RESPOSTA AO DEGRAU ****************************

figure;
plot(t,y1,'r',t,y2,'g');
title('Resposta ao degrau para FTMF com o Controladores projetados');
legend('FTMF com Atraso','FTMF com Atraso-Avanco');
```

```
grid on;
save('C_avanco_atraso','C_avanco_atraso');
```

.4.2 1.2 (Atraso-Avanço)

```
% ========================================================================
%
%         SEGUNDO TRABALHO COMPUTACIONAL DE SISTEMAS REALIMENTADOS
%
% Alaf do Nascimento Santos - 20171007181
% Felipe Antonio Moreira Silva - 2018205316
% ========================================================================

clear all, close all, clc;
op = 1; % 1 se item (i), 2 se item (ii)

global F A B;
load('C_avanco_atraso');
load('Gma');
Gma_c = Gma*C_avanco_atraso;

Gmf=feedback(Gma_c,1);

% Definição das Matrizes A, B, C e D
a0 = 14156.72;
a1 = 9697.37;
a2 = 1933.58;
a3 = 102.40;
b0 = 286602.32;
b1 = 87812.86;
b2 = 4037.30;
b3 = 0;

A = [-a3 1 0 0;
     -a2 0 1 0;
     -a1 0 0 1;
     -a0 0 0 0];
```

```
B = [b3 b2 b1 b0]';
C = [1 0 0 0];
D = 0;

%%%%%% Agora simular o veículo lunar no espaço de estados %%%%%%
% vetor estados inicial
Xs = [0 0 0 0];
F = 0;

% Inicialização de variáveis
tfinal = 5; % Tempo total de simulação
ref = 1; % Referência
passo = 0;
tempo = 0;
dT = 0.01; % tempo de amostragem

while (tempo < tfinal)
     passo = passo + 1;
     ts = passo*dT;

     % simulação da equação diferencial
     [T,X] = ode45('modelo_linear',[tempo ts], Xs);

     % Armazenamento de variaveis
     n = length(T);
     tempo = ts;
     Xs = X(n,:);
     if(op == 2);
         F = ref - Xs(1);
     end
     if(op == 1);
         F = ref-(Xs(1) + 0.02*randn(1));
     end
     vetout(passo,:) = X(n,1);
     vettime(passo) = T(n,:);
     vetvref(passo,:) = ref;

end
```

```
% Plotagem dos gráficos
figure
plot(vettime, vetvref,'b');
hold on;
plot(vettime,vetout,'r'); %multiplica por p1 para reduzir o erro
xlabel('Tempo(s)');title('Direção');
legend('Referencia');
grid;
```

.4.3 1.3 (Atraso-Avanço)

```
% ========================================================================
%
%         SEGUNDO TRABALHO COMPUTACIONAL DE SISTEMAS REALIMENTADOS
%
% Alaf do Nascimento Santos - 20171007181
% Felipe Antonio Moreira Silva - 2018205316
% ========================================================================

clear all, close all, clc;
global F A B;
load('C_avanco_atraso');
load('Gma');
Gma_c = Gma*C_avanco_atraso;

Gmf=feedback(Gma_c,1);

% Definição das Matrizes A, B, C e D
a0 = 14156.72;
a1 = 9697.37;
a2 = 1933.58;
a3 = 102.40;
b0 = 286602.32;
b1 = 87812.86;
b2 = 4037.30;
b3 = 0;

A = [-a3 1 0 0;
```

```
     -a2 0 1 0;
     -a1 0 0 1;
     -a0 0 0 0];

B = [b3 b2 b1 b0]';
C = [1 0 0 0];
D = 0;

%%%%%% Agora simular o veículo lunar no espaço de estados %%%%%%
% vetor estados inicial
Xs = [0 0 0 0];
F = 0;

% Inicialização de variáveis
tfinal = 5; % Tempo total de simulação
ref = 1; % Referência
passo = 0;
tempo = 0;
dT = 0.01; % tempo de amostragem

while (tempo < tfinal)
     passo = passo + 1;
     ts = passo*dT;

     % simulação da equação diferencial
     [T,X] = ode45('modelo_linear',[tempo ts], Xs);

     % Armazenamento de variaveis
     n = length(T);
     tempo = ts;
     Xs = X(n,:);
     F = ref - Xs(1);
     vetout(passo,:) = X(n,1);
     vettime(passo) = T(n,:);
     vetvref(passo,:) = ref;
end

% Plotagem dos gráficos
```

```
t=0.0:0.01:5;
y3=step(Gmf,t);

figure;
plot(vettime, vetout,'r');
hold on;
plot(t,y3,'g');
plot(vettime,vetvref,'b');
xlabel('Tempo(s)');title('Direção');
legend('Referencia');
grid;

title('Comparação das respostas dos itens 1.1 e 1.2 para controlador
    Atraso-Avanço');
grid on;
legend('Item 1.1','Item 1.2', 'ref');
hold off;
```

.5 Item 1.5

```
% ========================================================================
%
%         SEGUNDO TRABALHO COMPUTACIONAL DE SISTEMAS REALIMENTADOS
%
% Alaf do Nascimento Santos - 20171007181
% Felipe Antonio Moreira Silva - 2018205316
% ========================================================================

clear all, close all, clc;
global F A B;
load('PID');
load('C_avanco_atraso');
load('Gma');
Gpid = Gma*PID2;
Gma_c = Gma*C_avanco_atraso;

GMF1=feedback(Gpid,1);
GMF2=feedback(Gma_c,1);
```

```
t=0.0:0.01:5;
y1=step(GMF1,t);
y2=step(GMF2,t);

figure;
plot(t,y1,'r');
hold on;
plot(t,y2,'b');
xlabel('Tempo(s)');title('Direção');
legend('Referencia');
title('Comparação das respostas dos itens 1.1 e 1.4');
grid on;
legend('Item 1.1','Item 1.4');
hold off;
```

.6 Item 1.6

```
% ========================================================================
%
%         SEGUNDO TRABALHO COMPUTACIONAL DE SISTEMAS REALIMENTADOS
%
% Alaf do Nascimento Santos - 20171007181
% Felipe Antonio Moreira Silva - 2018205316
% ========================================================================

%%
clear all, close all, clc;
global F A B;
load('PID');
load('Gma');
Gpid = Gma*PID2;

GMF=feedback(Gpid,1);
polos_mf = pole(GMF);

%% Projeto do controlador via realimentação de estados
A = [18.78 1;
```

```
     44.44 0];
B = [0 900]';
C = [1 0];
D = 0;

polos_mf_ordem2 = [polos_mf(2), polos_mf(3)]; %eliminando o polo menos dominante

K = place(A, B, polos_mf_ordem2);
Ak = A-B*K;
m=ss(Ak,B,C,D); % FT de MF

figure;
step(m);title('Resposta da realimentação de estados ');
grid

k0 = freqresp(m,0);
p1 = 1/k0;

Grs = p1*m;

figure
step(Grs);
title('Resposta da realimentação de estados ajustando erro em regime');
grid

save('Grs','Grs');
%% Simulação da resposta em malha fechada com realimentação de estados
A = [18.78 1;
     44.44 0];
B = [0 900]';
C = [1 0];
D = 0;

polos_mf_ordem2 = [polos_mf(2), polos_mf(3)]; %eliminando o polo menos dominante
K = place(A,B,polos_mf_ordem2);

%%%%%% Agora simular o veículo lunar com realimentação de estados %%%%%%
ts = 0.4; qsi = 0.8; %definição de parâmetros
```

```
m = ss(A-B*K,B,C,D); % FT da G esperada representada em espaço de estados
k0 = freqresp(m,0); % calculo do p1 para corrigir o erro em regime
p1 = 1/k0;

% vetor estados inicial
Xs = [0 0];
F = 0;

% Inicialização de variáveis
tfinal = 1.5; % Tempo total de simulação
ref = 1; % Referência
passo = 0;
tempo = 0;
dT = 0.01; % tempo de amostragem

while (tempo < tfinal)

     passo = passo + 1;
     ts = passo*dT;

     % simulação da equação diferencial
     [T,X] = ode45('modelo_linear2',[tempo ts], Xs);

     % Armazenamento de variaveis
     n = length(T);
     tempo = ts;
     Xs = X(n,:);
     F = ref-K*(Xs)';

     vetout(passo,:) = [X(n,1) X(n,2)];
     vettime(passo) = T(n,:);
     vetvref(passo,:) = ref;
end

dir = vetout*C';

% Plotagem dos gráficos
figure
plot(vettime, vetvref,'b');
```

```
hold on;
plot(vettime,dir*p1,'r'); %multiplica por p1 para reduzir o erro
xlabel('Tempo(s)');
ylabel('Amplitude');
title('Simulação da realimentação de estados');
legend('Referencia');
grid;
```

.7 Item 1.7

```
% ========================================================================
%
%         SEGUNDO TRABALHO COMPUTACIONAL DE SISTEMAS REALIMENTADOS
%
% Alaf do Nascimento Santos - 20171007181
% Felipe Antonio Moreira Silva - 2018205316
% ========================================================================

clear all, close all, clc;
load('PID');
load('C_avanco_atraso');
load('Grs');
load('Gma');
Gpid = Gma*PID2;
Gma_c = Gma*C_avanco_atraso;

GMF1=feedback(Gpid,1);
GMF2=feedback(Gma_c,1);
GMF3 = Grs;

t=0.0:0.01:5;
y1=step(GMF1,t);
y2=step(GMF2,t);
y3=step(GMF3,t);

figure;
plot(t,y1,'r');
hold on;
```

```
plot(t,y2,'b');
plot(t,y3,'g');
xlabel('Tempo(s)');title('Direção');
legend('Referencia');
title('Comparação das respostas dos itens 1.1, 1.4 e 1.6');
grid on;
legend('Item 1.1','Item 1.4','Item 1.6');
hold off;
```

.8 Item 1.8

```
% ========================================================================
%
%         SEGUNDO TRABALHO COMPUTACIONAL DE SISTEMAS REALIMENTADOS
%
% Alaf do Nascimento Santos - 20171007181
% Felipe Antonio Moreira Silva - 2018205316
% ========================================================================

%%
clear all, close all, clc;
load('PID');
load('Gma');
Gpid = Gma*PID2;

GMF=feedback(Gpid,1);
polos_mf = pole(GMF);

%% Projeto do controlador via realimentação de estados
A = [18.78 1;
     44.44 0];
B = [0 900]';
C = [1 0];
D = 0;

fator = 5;
polos_mf_ordem2 = [polos_mf(2), polos_mf(3)]; %eliminando o polo menos dominante
polos_obs=fator*polos_mf_ordem2;
```

```
K = place(A, B, polos_mf_ordem2);

% Projeto do observador de estados
L=place(A',C',polos_obs);
L=L';

[numC1, denC1] = ss2tf(A-L*C-B*K, B, -K, 1);
[numC2, denC2] = ss2tf(A-L*C-B*K, L, K, D);
C1 = tf(numC1,denC1)
C2 = tf(numC2,denC2)

m_s = (C1*feedback(Gma,C2))
m=minreal(m_s)

figure;
step(m);title('Resposta da realimentação de estados + observador ');
grid

k0 = freqresp(m,0);
p1 = 1/k0; %inverso do ganho de regime de m

Grs2 = p1*m;

figure
step(Grs2);
title('Ajustando erro em regime - Realimentação de estados + observador');
grid
save('Grs2','Grs2');

%% Compara com 1.7
clear all, close all, clc;

load('PID');
load('C_avanco_atraso');
load('Grs');
load('Grs2');
load('Gma');
Gpid = Gma*PID2;
```

```
Gma_c = Gma*C_avanco_atraso;

GMF1=feedback(Gpid,1);
GMF2=feedback(Gma_c,1);
GMF3 = Grs;
GMF4 = Grs2;

t=0.0:0.01:4;
y1=step(GMF1,t);
y2=step(GMF2,t);
y3=step(GMF3,t);
y4=step(GMF4,t);

figure;
plot(t,y1,'r');
hold on;
plot(t,y2,'b');
plot(t,y3,'g');
plot(t,y4,'black');
xlabel('Tempo(s)');title('Direção');
legend('Referencia');
title('Comparação das respostas dos itens 1.1, 1.4, 1.6 e 1.8');
grid on;
legend('Item 1.1','Item 1.4','Item 1.6', 'Item 1.8');
hold off;
```

.9 Funções de Modelo Linear

.9.1 modelo_linear.m

```
    function d = modelo_linear(t,x);
    global F  A B

    d(1) = A(1,1)*x(1) + A(1,2)*x(2) + A(1,3)*x(3) + A(1,4)*x(4) + B(1)*F;
    d(2) = A(2,1)*x(1) + A(2,2)*x(2) + A(2,3)*x(3) + A(2,4)*x(4) + B(2)*F;
    d(3) = A(3,1)*x(1) + A(3,2)*x(2) + A(3,3)*x(3) + A(3,4)*x(4) + B(3)*F;;
    d(4) = A(4,1)*x(1) + A(4,2)*x(2) + A(4,3)*x(3) + A(4,4)*x(4) + B(4)*F;
```

```
d = d';
```

.9.2 modelo_linearVeiculo.m

```
function d = modelo_linearVeiculo(t,x);
global F A B

d(1) = A(1,1)*x(1) + A(1,2)*x(2) + A(1,3)*x(3) + B(1)*F;
d(2) = A(2,1)*x(1) + A(2,2)*x(2) + A(2,3)*x(3) + B(2)*F;
d(3) = A(3,1)*x(1) + A(3,2)*x(2) + A(3,3)*x(3) + B(3)*F;

d = d';
```

.9.3 modelo_linear2.m

```
function d = modelo_linearVeiculo(t,x);
global F A B

d(1) = A(1,1)*x(1) + A(1,2)*x(2) + B(1)*F;
d(2) = A(2,1)*x(1) + A(2,2)*x(2) + B(2)*F;
d = d';
```