UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
CENTRO TECNOLÓGICO
DEPARTAMENTO DE INFORMÁTICA

Mini-curso de Simulink

Controle Automático I

Agosto de 2009

PET Engenharia de Computação
www.inf.ufes.br/~pet

Sumário

1 Introdução **4**

2 Acessando o Simulink **4**

3 Componentes de um modelo Simulink **6**
3.1 Fontes . . . 6
3.1.1 Sinal de entrada personalizado . . . 6
3.1.2 Combinando Entradas . . . 8
3.2 Diagrama de blocos . . . 8
3.3 Saídas . . . 8

4 Criando Simulações **9**
4.1 Gerador de Sinais . . . 9
4.2 Problema com uma Equação Diferencial . . . 11
4.3 Crescimento de Bactérias . . . 13

5 Utilizando o Workspace **15**

6 Sistemas Contínuos no Tempo **19**
6.1 Sistemas Escalares Lineares . . . 19
6.1.1 Bloco Integrador . . . 19
6.1.2 Exemplo . . . 21
6.1.3 Bloco Função de Transferência . . . 23
6.1.4 Exemplo . . . 23
6.2 Sistemas Contínuos Não-Lineares . . . 25

7 Sistemas Discretos no Tempo **28**
7.1 Sistemas Discretos no Tempo Lineares . . . 29
7.1.1 Atraso Unitário (Unit Delay) . . . 29
7.1.2 Integrador no tempo discreto (Discrete-Time Integrador) . . . 30
7.1.3 Função de Transferência Discreta . . . 31

8 Aplicação **33**
8.1 Especificações dos parâmetros . . . 34
8.2 Equações do Sistema dinâmico . . . 34
8.2.1 Do potenciômetro detector de erros . . . 34
8.2.2 Do Amplificador . . . 35
8.2.3 Da armadura controlada do motor . . . 35
8.3 Função de Transferência do Motor (Planta) . . . 35
8.3.1 Força contra eletromotriz no motor . . . 35
8.3.2 Do circuito da armadura podemos escrever . . . 35
8.3.3 Torque desenvolvido . . . 35
8.3.4 Torque resultante na armadura do motor . . . 36
8.4 Função de transferência do motor (FTMF) . . . 36
8.5 Diagrama de Blocos do Sistema . . . 37
8.6 Ajuste de Kp . . . 37
8.7 Análise do efeito de uma não linearidade - Saturação . . . 38
8.7.1 Análise . . . 38
8.7.2 Conclusões . . . 39

8.8 Análise do efeito de uma não linearidade - Atraso 40
8.8.1 Análise 40
8.8.2 Conclusões 41
8.9 Análise do efeito do período de amostragem e do erro de quantificação na resposta ao degrau unitário 41
8.9.1 Modificações 41
8.9.2 Análise 42
8.9.3 Conclusões 46

1 Introdução

O SIMULINK é uma ferramenta utilizada para Modelagem, Simulação e Análise de Sistemas Dinâmicos. O programa se aplica a sistemas lineares e não lineares, discretos e contínuos no tempo.

Ao contrário do MATLAB, que utiliza linha de comando, o Simulink utiliza uma interface gráfica amigável, representando o sistema por **diagramas de blocos**, no qual cada bloco representa uma operação matemática de entrada e saída que chama-se função de transferência do bloco. Nos sistemas contínuos, estas relações são obtidas utilizando-se a transformada de Laplace nas equações. Não podemos deixar de enfatizar que apesar do simulink ser uma aplicação específica, este não trabalha independentemente do MATLAB.

Curiosidade

O Simulink é um programa para resolver sistemas dinâmicos, ele faz uso de algoritmos de integração para resolver as equações numericamente. Dos diversos algoritmos de integração, você provavelmente fará uso do Runge-Kutta de 4ª e 5ª ordens ou do algoritmo de Euller. Mais detalhes sobre algoritmos de integração podem ser obtidos no "Simulink User's Guide".

2 Acessando o Simulink

Na linha de comando do MATLAB deve-se digitar:

Exemplo 1: *Acessando o Simulink*

```
1 >> simulink
```

Ou pode-se clicar no seguinte ícone da barra de ferramentas toolbar:

Figura 1: Ícone do Simulink

A janela da biblioteca de blocos se abrirá como na Figura 2.

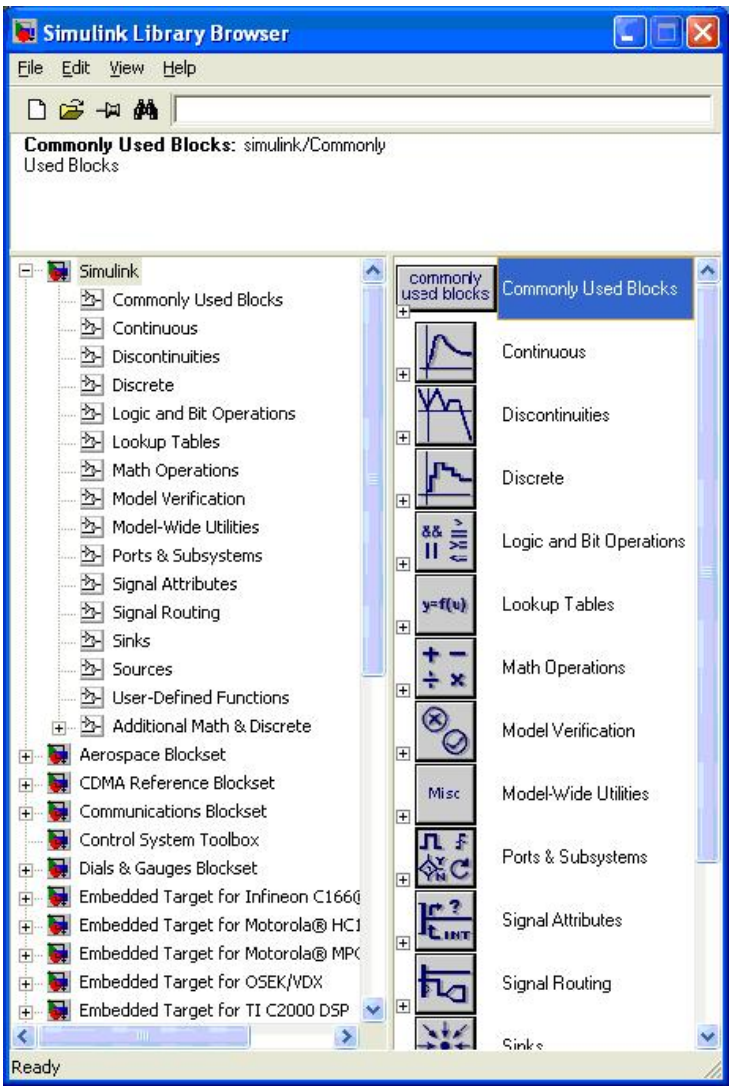

Figura 2: Biblioteca de Blocos

Para realizar uma construção da modelagem do sistema selecione FileNewModel, ou se preferir, utilize a tecla de atalho CTRL + N. Uma janela nomeada untitled como na Figura 3 se abrirá.

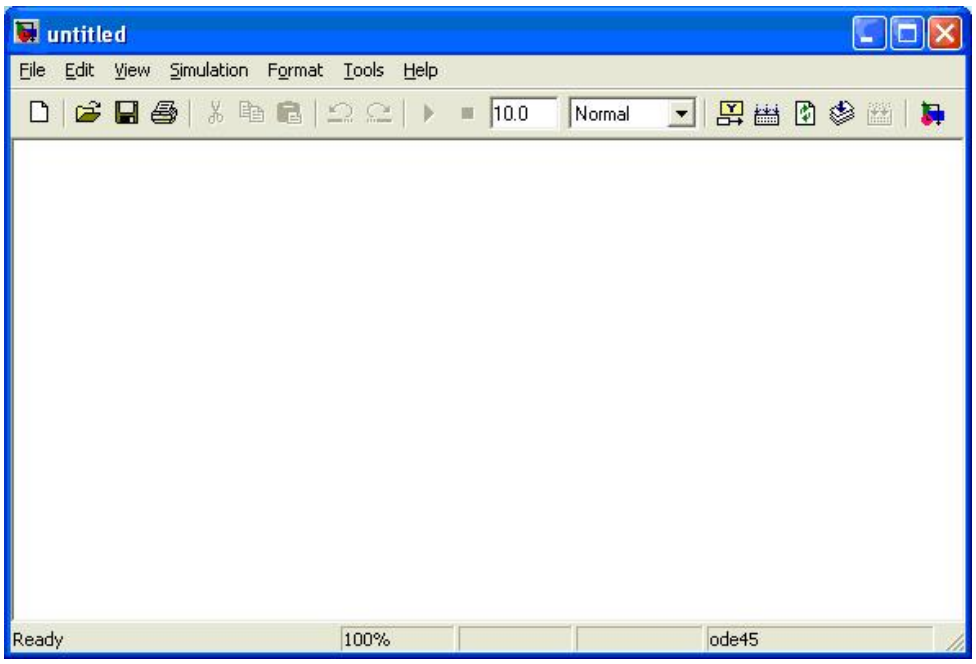

Figura 3: Área de trabalho

Para adicionar os blocos, devem-se arrastar os componentes da Biblioteca de Blocos (Library Browser) para a área de trabalho.

3 Componentes de um modelo Simulink

Um modelo no Simulink consiste em três componentes:

3.1 Fontes

São as entradas do sistema e estão presentes na biblioteca de fontes (sources). A seguir, apresentamos as fontes mais comuns:

- O bloco Constante (Constant) produz um sinal uniforme. A magnitude pode ser escolhida com um duplo clique sobre o bloco.
- O bloco Degrau (Step) produz uma função degrau. Pode-se configurar o instante em que se aplica o degrau, assim como sua magnitude antes e depois da transição.
- O bloco de Onda Senoidal (Sine Wave) gera uma senóide com os seguintes parâmetros a serem configurados: amplitude, fase e freqüência da onda senoidal.
- O Gerador de Sinais (Signal Generator) pode produzir ondas senoidais, quadradas, dente de serra ou sinais aleatórios.
- Outros sinais podem ser gerados a partir de combinações destes blocos apresentados.

3.1.1 Sinal de entrada personalizado

Para criar um sinal de entrada personalizado, devemos inicialmente definir os pares coordenados em uma matriz, e importá-lo para o Simulink. A seguir, apresentamos duas formas de se obter um sinal personalizado.

From Workspace Block

Deve-se definir os pares coordenados em uma matriz no MATLAB, e importá-lo a partir do workspace através do bloco *From Workspace Block*. Este é o bloco que permite ao usuário criar seus próprios sinais de entrada.

Nas propriedades deste bloco, devem-se definir quais serão as matrizes utilizadas como fonte de sinal. Estas matrizes devem ser previamente definidas no MATLAB antes da execução da simulação, a primeira coluna da matriz deve ser preenchida com os valores da variável independente, que corresponde ao tempo da simulação. As colunas seguintes são variáveis referentes à variável independente.

Exemplo

Considere o sinal definido por:

$$u(t) = 2 \times t;$$

Para gerar a matriz de pares coordenados, devemos digitar os seguintes comandos na área de trabalho do MATLAB:

Exemplo 2: *Matriz de pares coordenados*

```
>> t = 0:0.1:100;
>> u = 2*t;
>> A = [t', u'];
```

Definida a matriz a ser usada, deve-se adicionar o bloco *From Workspace Block* da biblioteca *Sources* ao modelo. Com um duplo clique sobre o bloco deve-se digitar o nome da matriz definida no MATLAB no campo *Data*, neste caso "A". Pronto, o bloco já está configurado e pode ser usado.

Caso o sinal de entrada tenha mais que uma dimensão, deve-se definir os valores de entrada usando-se uma struct. O tempo deve ser definido como um vetor no campo *time*, enquanto que os valores referentes à variável independente devem ser definidos como colunas no campo *signals.values*. Ainda deve ser informado o número de dimensões no campo *signals.dimensions*. Usando-se os mesmos valores do exemplo 2, pode-se definir uma struct "a" com os comandos apresentados no exemplo 3.

Exemplo 3: *Matriz de pares coordenados*

```
>> a.time = t;
>> a.signals.values = u';
>> a.signals.dimensions = 1;
```

From File Input Block

A matriz é agora carregada a partir de um arquivo gerado pelo MATLAB, assim o sinal de entrada pode ser salvo. Uma diferença importante é que é que os sinais devem agora ser carregados em linhas.

Exemplo

A partir do mesmo exemplo, deve-se salvar a matriz gerada em um arquivo com extensão *.mat* (arquivo usado pelo MAtlab).

Exemplo 4: *Salvando em um arquivo*

```
>> B = A';
>> save exemplo.mat B;
```

No simulink, deve-se adicionar o bloco From File Input Block da biblioteca Sources. Com um duplo clique sobre o bloco deve-se digitar no campo *File Name* o nome do arquivo, neste caso “exemplo.mat”.

3.1.2 Combinando Entradas

Através de combinações de blocos da biblioteca pode-se obter sinais personalizados.

Exemplo

Para criar uma simulação de um impulso unitário, podemos gerar um degrau crescente em um instante de tempo t_o to seguido de um sinal degrau decrescente e com mesma magnitude em um instante posterior t_f. Logo, devemos utilizar duas fontes de função degrau.

3.2 Diagrama de blocos

É a modelagem por meio de blocos utilizando-se a transformada de Laplace nas equações do sistema.

3.3 Saídas

Os dispositivos de saída são os blocos que permitem verificar o comportamento do sistema, estes blocos são encontrados na biblioteca de dispositivos de saída (*Sinks*).

Scope

O osciloscópio produz gráficos a partir de dados do modelo. Não existem parâmetros a serem configurados.

XY Graph

O bloco de XY Graph produz um gráfico idêntico ao gráfico produzido pelo comando plot do MATLAB. Para isso, devem-se configurar os valores de mínimos e máximos, da horizontal e vertical.

Display

O bloco Display produz uma amostragem digital do valor de sua entrada.

To File

Pode-se ainda armazenar os dados em arquivos do MATLAB para usos posteriores. Deve-se definir o nome do arquivo a ser criado.

To Workspace

Pode-se ainda enviar os dados para a área de trabalho do MATLAB utilizando o bloco To Workspace Block. Deve-se definir o nome da matriz.

Stop Simulation

O bloco de parada (Stop Simulation) causa a parada da simulação quando a sua entrada for diferente de zero.

4 Criando Simulações

Os Exemplos a seguir mostram os passos para se criar modelos de sistemas dinâmicos no Simulink.

4.1 Gerador de Sinais

1. Carregue o Simulink:
   - Execute o MATLAB;
   - Em seguida chame o Simulink, e uma janela em branco para a construção do sistema;
2. Insira o Signal Generator (Gerador de sinais)
   - Entre a lista de opções do Library Browser (Browser de biblioteca), clique em Simulink, que é uma biblioteca de blocos;
   - Clique em Source, que é um tipo de bloco de fonte de sinal;
   - Selecione *Signal Generator* e arraste este bloco para a área de trabalho do novo documento aberto;
3. Insira os blocos do sistema modelado
   - Qualquer bloco no simulink pode ser pesquisado na linha de comando, basta entrar com o nome do bloco na caixa de edição que fica abaixo da barra de ferramentas e clicar no ícone Find block:
   - O bloco também pode ser adicionado ao modelo clicando-se com o botão direito sobre o bloco e escolhendo-se a opção *Add to Untitled*, sendo que *Untitled* será substituído pelo nome do seu projeto.
4. Insira os dispositivos de saída, por exemplo, o Scope (Osciloscópio)
   - Clique em *Commonly Used Blocks* (blocos usados comumente);
   - E insira o Scope;
5. Faça a conexão entre os blocos
   - Faça um caminho com o mouse ligando os dois componentes;
   - Outra opção para ligar os blocos é clicar no bloco de origem, segurar a tecla ctrl e clicar no bloco destino. Um caminho automático será feito.
6. Propriedades do Gerador de Sinais (SignalGenerator)
   - Dê um duplo clique sobre o *signal generator* ou clique com o botão direito e escolha *SignalGenerator Parameter...*;
   - Escolha a forma de onda (wave form) dente de serra (sawtooth) com frequência de 1 Hz com uma amplitude de 2;
7. Propriedades da Simulação (Scope)

- Na barra de menu, selecione *Simulation* e escolha *Configuration Parameters*. Aqui são definidos, principalmente, o tempo inicial e final da simulação, assim como o passo da simulação. Em *Stop Time*, coloque 20. Clique em OK para confirmar as alterações.
- Clique em *Start Simulation* para realizar a simulação;
- Dê um duplo clique sobre o osciloscópio para visualizar a onda;
- Para um melhor visualização da onda, clique no botão *Autoscale*, simbolizado por um binóculos. Também é possível dar um zoom num local específico usando-se as lupas.

8. Armazenando o trabalho...

   - Para armazenar o trabalho clique em File/Save ou pressione CTRL+S;
   - E entre com o local e nome da simulação;
   - O tipo do arquivo é * mdl.

Ao final dos passos citados acima, o diagrama deverá estar descrito como na Figura 4. E a forma de onda observada em nosso exemplo deverá ser como na Figura 5.

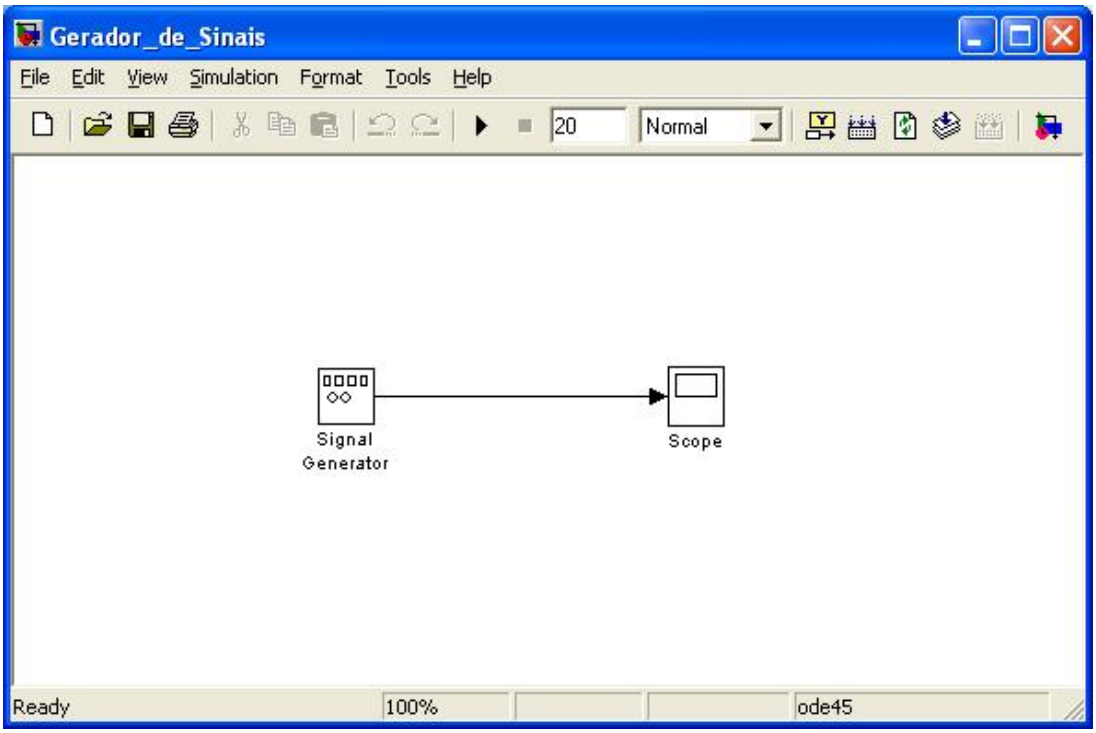

Figura 4: Gerador de Sinais

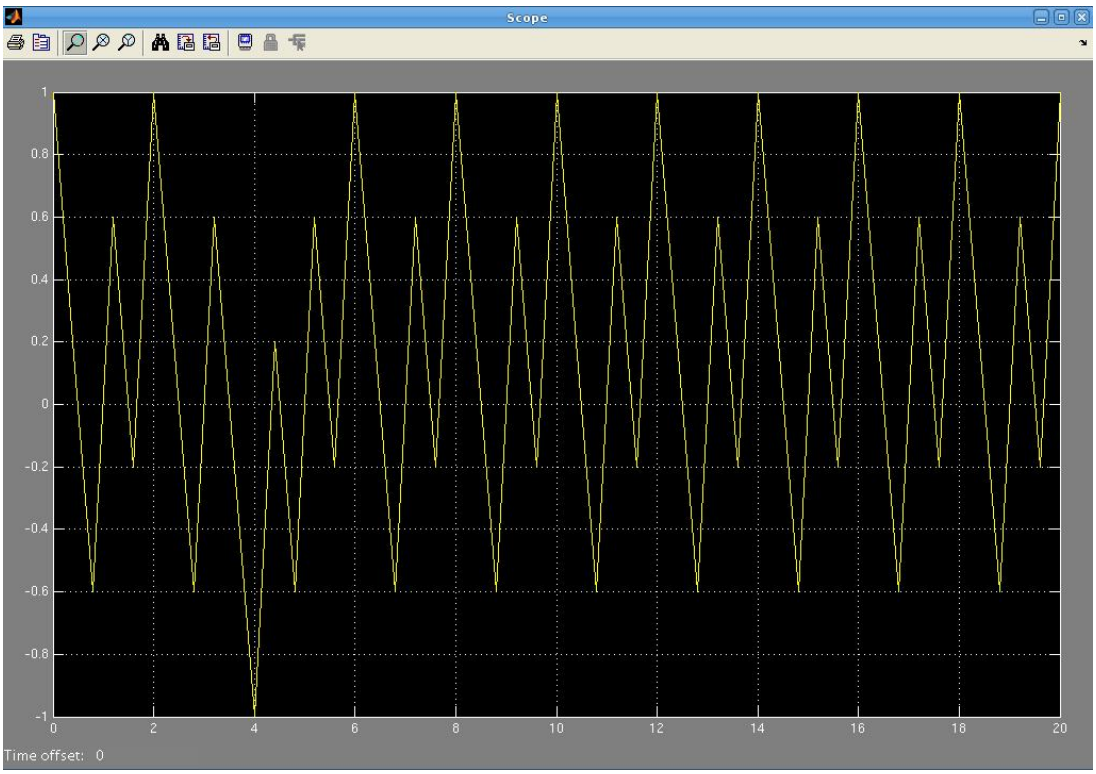

Figura 5: Forma de Onda do Gerador de Sinais

4.2 Problema com uma Equação Diferencial

Vamos modelar a seguinte equação diferencial:

$$\frac{dx}{dt} = sen(t)$$

Com a seguinte condição inicial: $x(0) = 0$.
Logo, teremos:

$$x(t) = \int_0^{\infty} sen(t)dt$$

1. Monte o diagrama de blocos como na Figura 6.

   - Insira o bloco Sine Wave (onda senoidal) da biblioteca Source;
   - Insira o bloco Integrator (integrador) da biblioteca Continuous (blocos lineares);
   - Insira o Scope (Osciloscópio) da biblioteca Commonly Used Blocks;

O figura 6 mostra o diagrama de blocos criados no simulink.

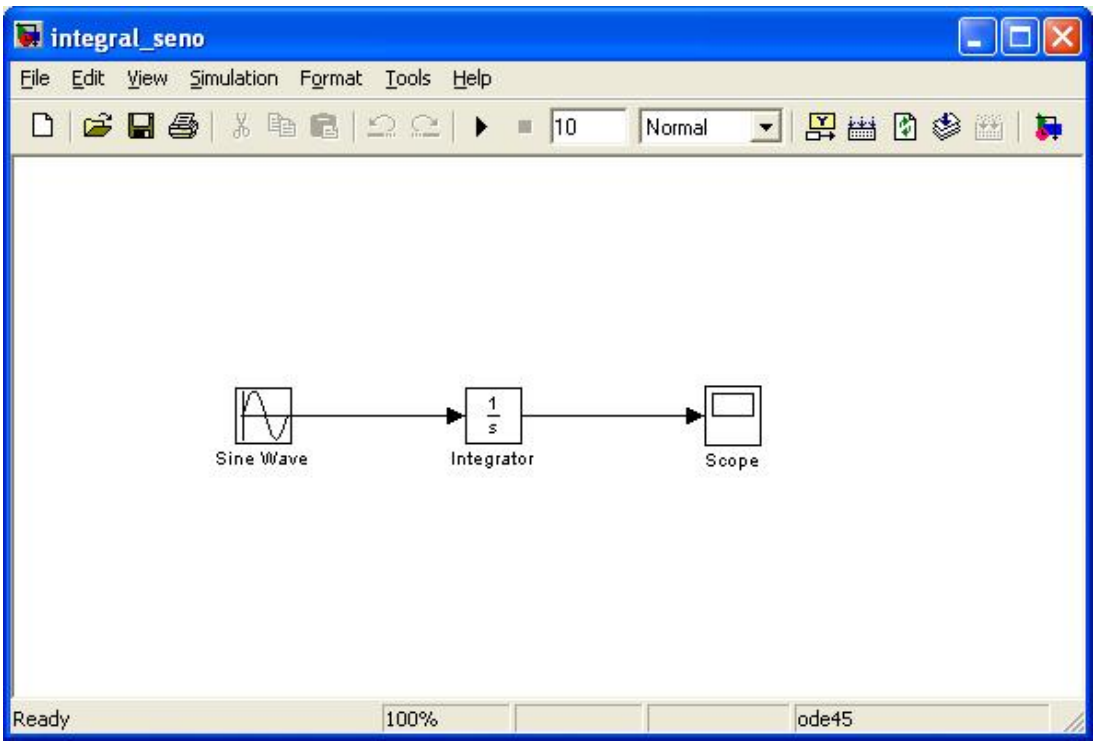

Figura 6: Diagrama de Blocos

Sabemos da prática de estudantes de Engenharia que a solução analítica para a equação diferencial do problema é dada por:

$$x(t) = 1 - \cos t$$

Executando a simulação, obtém-se o gráfico gerado pelo osciloscópio mostrado na figura 7. Pode-se perceber que a resposta obtida correspondente ao gráfico da equação esperada. Lembre-se que o parâmetro de entrada das funções seno e cosseno deve estar em radianos.

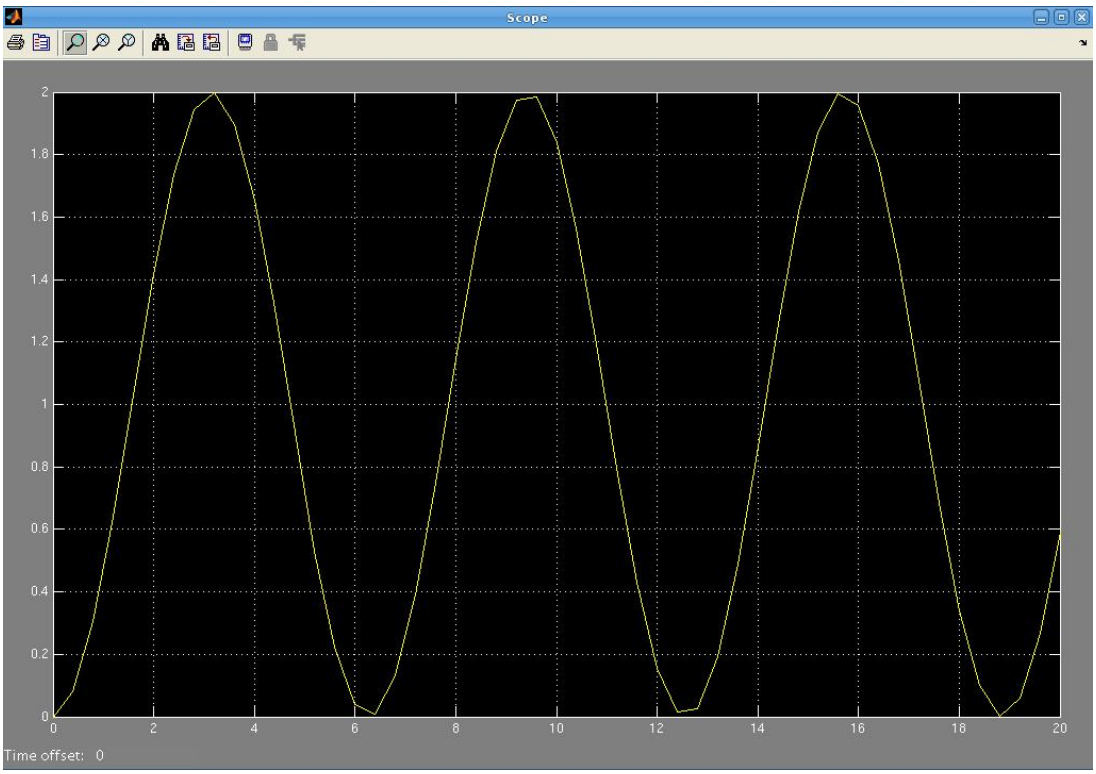

Figura 7: Forma de Onda da Integral do Seno

Abaixo outro exemplo para que o leitor se familiarize mais com o simulink.

4.3 Crescimento de Bactérias

Scheinerman descreveu um modelo simples do crescimento de bactérias isoladas do ambiente externo num pote. Admite-se que as bactérias nascem numa taxa proporcional ao número de bactérias presentes e que elas morrem a uma taxa proporcional ao quadrado do número de bactérias presentes. Sendo x o número de bactérias presentes no pote, a taxa em que as bactérias nascem é definida por:

$$Taxa\ de\ Natalidade = n \times x$$

E a taxa em que elas morrem:

$$Taxa\ de\ Mortalidade = m \times x^2$$

A taxa total de mudança na população de bactérias é a diferença entre a natalidade e a mortalidade de bactérias. O sistema pode ser então descrito pela equação diferencial a seguir:

$$\frac{dx}{dt} = n \times x - m \times x^2$$

Partindo disto será então construído o modelo do sistema dinâmico supondo que $n = 1\ bacteria/hora$ e $m = 0,5\ bacteria/hora$. Será determinado o números de bactérias contidas no pote após 1 hora, admitindo que inicialmente existiam 100 bactérias presentes. O diagrama de blocos que representa essa modelo é mostrado na figura 8.

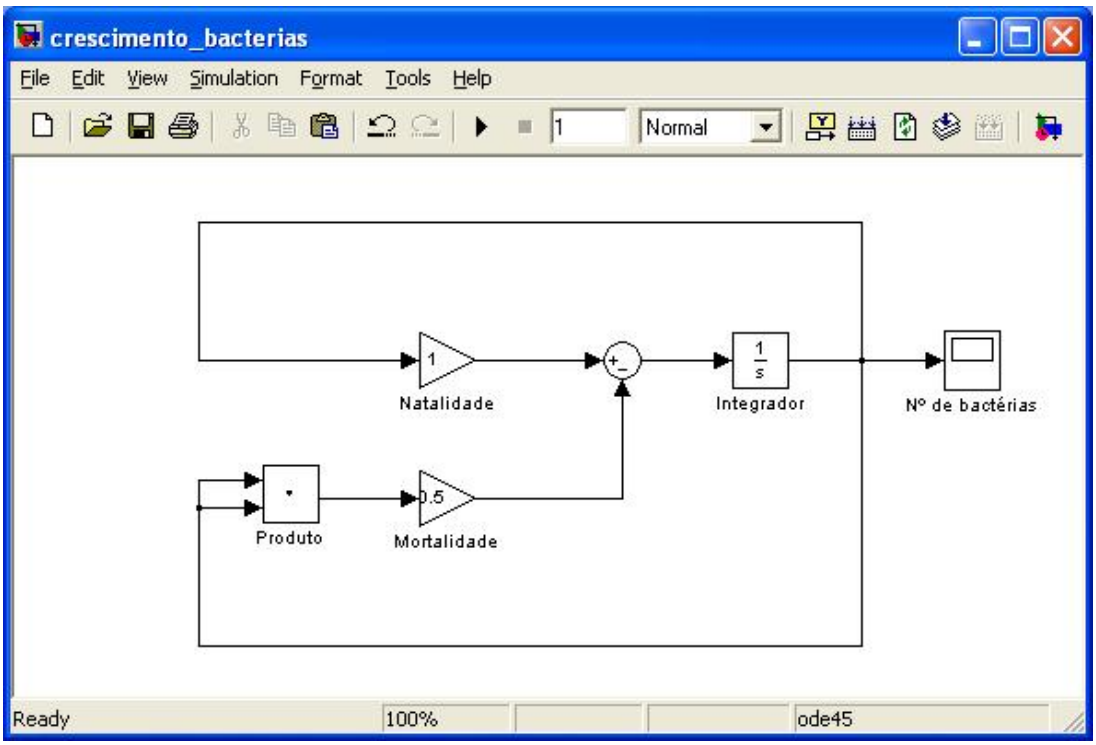

Figura 8: Diagrama de Blocos

Assim, siga os seguintes passos para montar o modelo da Figura 8.

1. Novo Modelo
   - Clique na barra de Menu, escolhendo FILE: NEW;
2. Abra a biblioteca linear *Commonly Used Blocks*
   - Arraste o integrador para a janela do modelo;
   - Insira dois blocos *Gain* (Ganho)
   - O Simulink não permite que exista mais de um bloco com o mesmo nome, renomeie um dos blocos *Gain*
   - Arraste ainda um bloco de *Sum* (Soma);
3. Abra a biblioteca linear *Commonly Used Blocks*
   - Arraste o integrador para a janela do modelo;
   - Insira dois blocos *Gain* (Ganho);
   - E insira um bloco *scope* (osciloscópio);
4. Abra a biblioteca *Math Operations* (Operações Matemáticas)
   - Insira o bloco *Dot Product* (Produto);
   - Insira o bloco *Sum* (Soma);
5. Conexão dos blocos
   - Por fim, conecte os blocos.

O decaimento do número de bactérias pode ser observado na Figura 9.

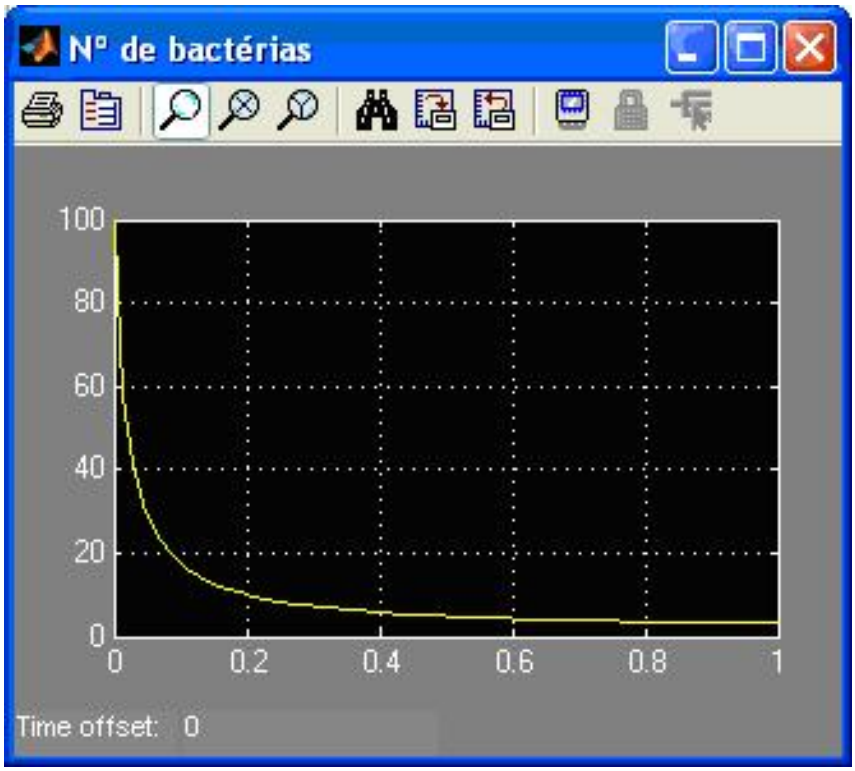

Figura 9: Gráfico do Crescimento de Bactérias

5 Utilizando o Workspace

Existe passagem de parâmetros entre o workspace do Matlab (workspace ou espaço de trabalho pode ser entendido como o conjunto de variáveis que pode ser observado com o comando *whos* na linha de comando) com o simulink. Sobretudo, qualquer bloco do simulink aceita variáveis do workspace do MATLAB como argumentos de entrada.

O exemplo a seguir mostra passo a passo como utilizar as variáveis do workspace do MATLAB. Neste exemplo duas variáveis, u e y, são criadas no workspace a partir da simulação do Simulink.

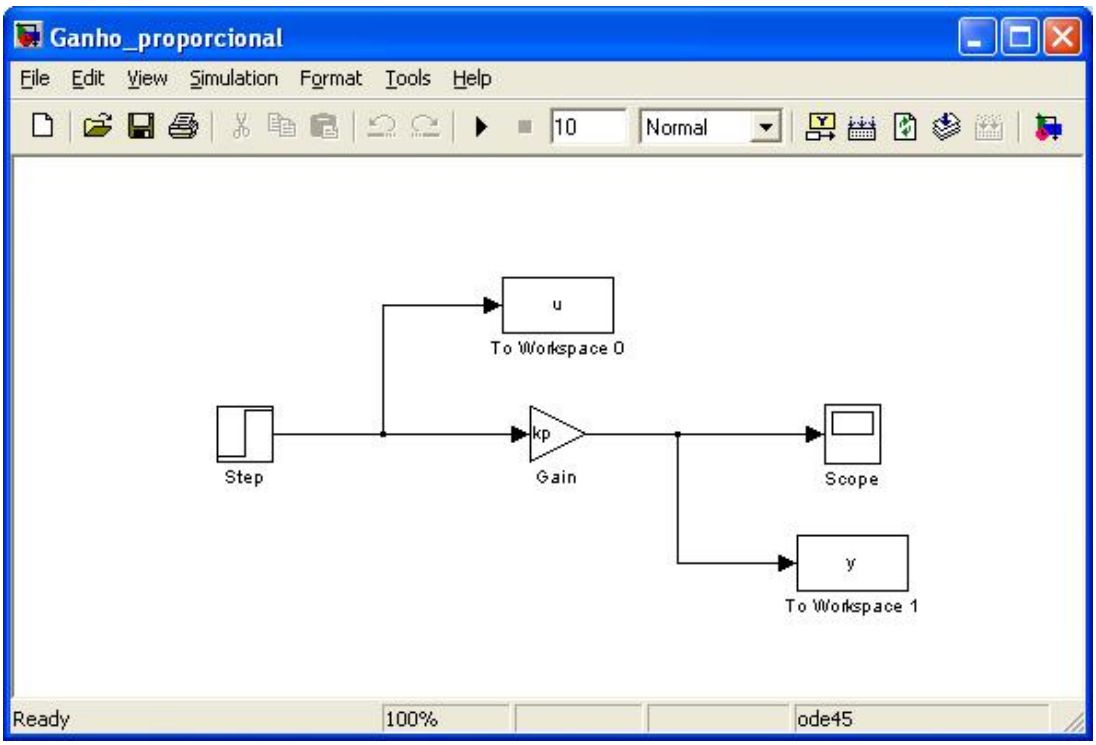

Figura 10: Modelo do Ganho Proporcional

1. Monte o diagrama de blocos como na Figura 10.

   - Insira o bloco *Step* (saída a um degrau) da biblioteca *Source*;
   - Insira o bloco *Gain* (elemento de ganho) da biblioteca *Commonly Used Blocks*;
   - Insira o *Scope* (Osciloscópio) da biblioteca *Commonly Used Blocks*;
   - Insira o *To Workspace* (Bloco de transferência entre simulação e workspace) da biblioteca *Sinks* (dispositivos de saída).

2. Save format

   - Modifique a propriedade *save format* dos blocos *To workspace* para *array*. Por padrão o Matlab exportar os dados em structs ao invés de vetores.

3. A variável K_p (ganho proporcional) utilizada no MATLAB pode ser utilizada como argumento do bloco de ganho. Ajuste o ganho K_p e verifique o conteúdo das variáveis u e y do MATLAB (variáveis referidas nos blocos To Workspace).

4. Tome a seguinte situação, digite Kp = 3 na linha de comando do MATLAB:

Exemplo 5: *Atribuição de Kp*

```
1 >> kp = 3;
```

5. Execute a simulação no Simulink.

6. Volte para o MATLAB e trace o gráfico das variáveis u e y, estas variáveis foram criadas através do Simulink e estão presentes no workspace através dos comandos mostrados no exemplo 6. O gráfico gerado é mostrado na figura 11.

Exemplo 6: *Plotando variáveis*

```
>> plot (u)
>> hold on
>> plot (y)
>> grid
>> axis ([0 50 -0.5 4])
```

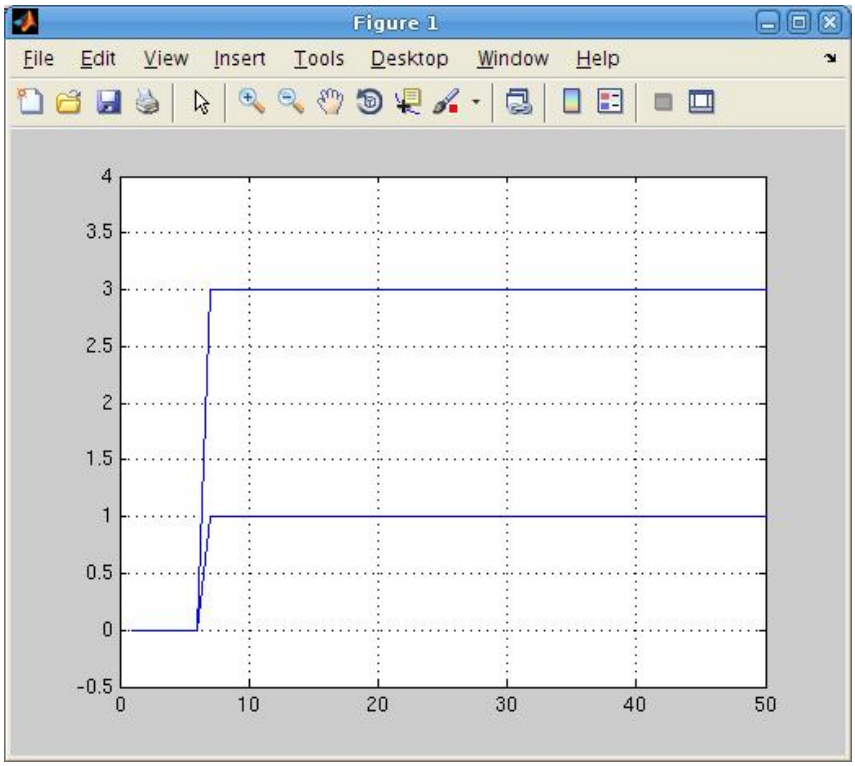

Figura 11: Gráfico das variáveis u e y

A forma de onda observada no osciloscópio é verificada na Figura 12. Percebe-se que é a mesma forma de onda da Figura 11.

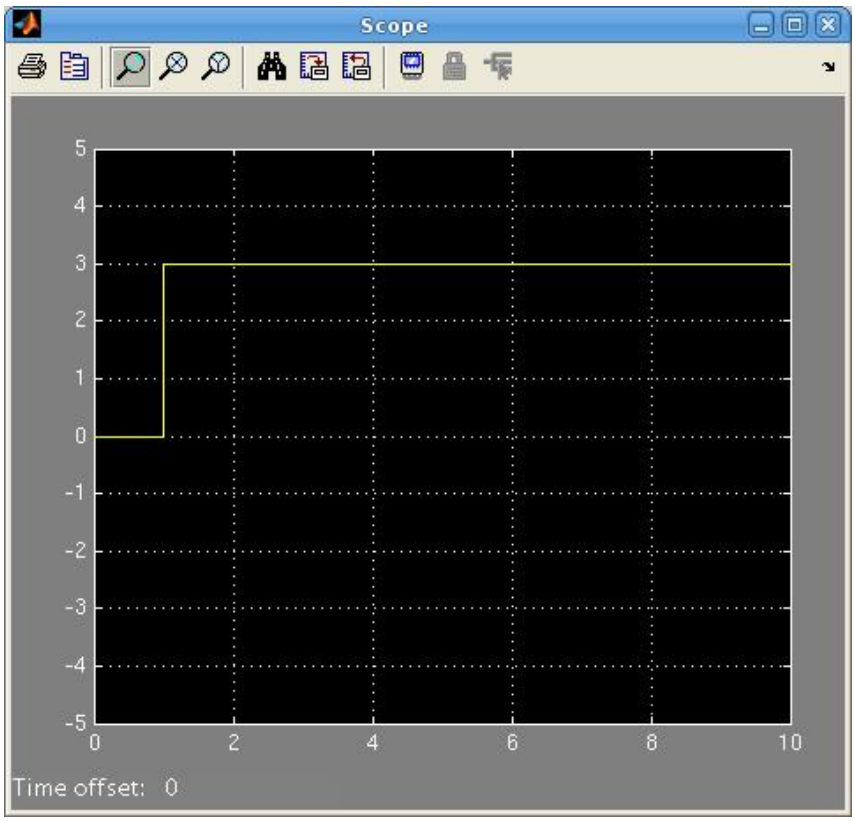

Figura 12: Resposta verificada no osciloscópio

Observação

Note que as variáveis u e y guardaram apenas as informações dos valores coletados, não guardando qualquer informação do tempo de simulação. Isso fez com que a função *plot* usasse a opção padrão para valores do eixo x: números inteiros correspondentes à posição de cada valor do vetor passado como parâmetro. Como foram coletadas 53 valores, os valores do eixo x ficaram na faixa de 1 à 53, enquanto que no osciloscópio esses valores ficaram na faixa de 0 a 10.

Para contornar esse problema, pode-se usar structs com tempo ao invés de vetores. Para tanto, clique duas vezes no bloco *To Workspace* de cada variável exportada e em *save format* escolha *Structure with time*. Execute a simulação novamente para que as variáveis sejam atualizadas. O exemplo 7 mostra como usar o plot com essas variáveis e a figura 13 mostra o resultado gerado.

Exemplo 7: *Plotando structs with time*

```
>> plot (y.time, y.signals.values, u.time, u.signals.values)
>> grid
>> axis ([0 10 -0.5 4])
```

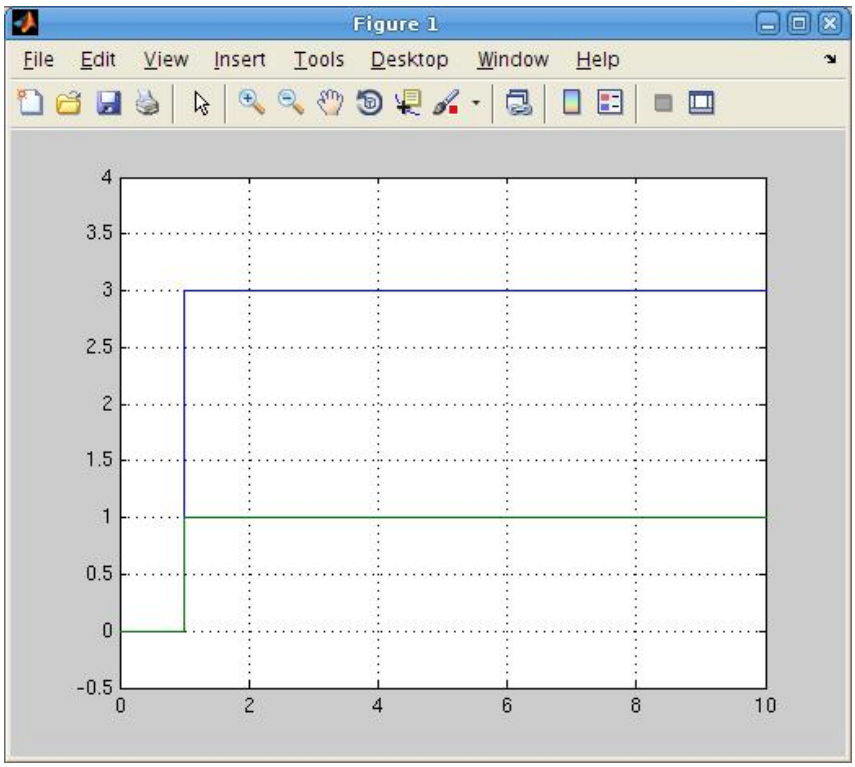

Figura 13: Resposta verificada no osciloscópio

6 Sistemas Contínuos no Tempo

Sistemas contínuos são aqueles que são modelados por equações diferenciais. Os sistemas mais simples são escalares e pode-se dizer que são lineares e invariantes no tempo.

6.1 Sistemas Escalares Lineares

São modelados através de blocos da biblioteca linear.

6.1.1 Bloco Integrador

Os parametros do bloco integrador podem ser observados na Figura 14.

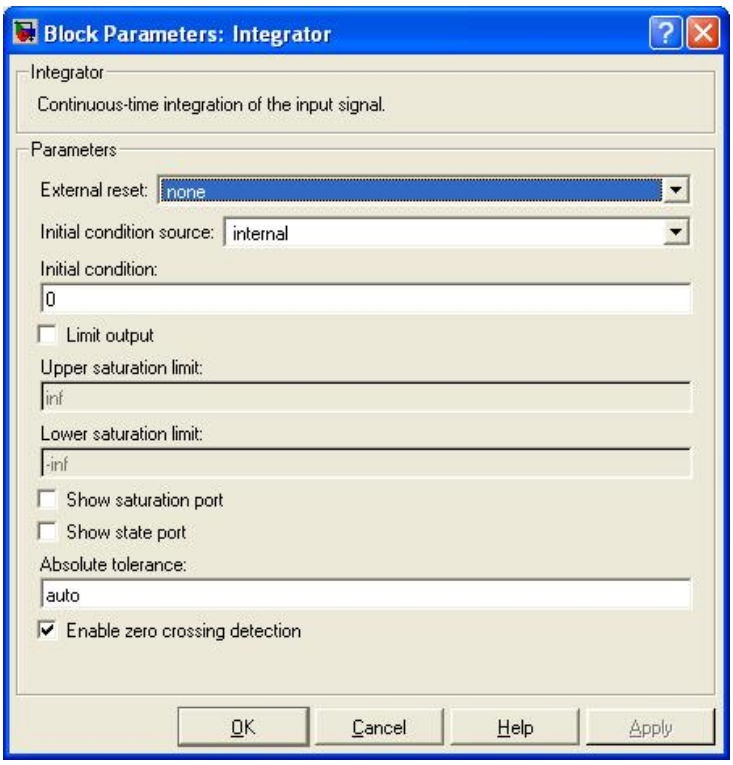

Figura 14: Parametros do Bloco Integrador

O bloco integrador pode ser utilizado com um sinal de reset, a saída sempre retorna à condição inicial toda vez que o sinal de reset dispara. As opções de reset externo (External reset) são:

- None: O reset é desabilitado;
- Rising: A saída é levada à condição inicial quando o sinal de reset passa pelo zero no sentido crescente;
- Falling: A saída é levada à condição inicial quando o sinal de reset passa pelo zero no sentido decrescente;
- Either: A saída é levada à condição inicial quando o sinal de reset passa pelo zero;

Quando o reset externo é ativado, aparece uma segunda entrada no bloco integrador (entrada de baixo). Conecte a essa entrada o sinal que irá controlar o reset.

A saída do integrador pode ser limitada. Para tanto, deve-se habilitar o *Limit output* (limite de saída) e escolher o limite de saturação superior e inferior. Pode-se também configurar a saída inicial. Para isso deve-se escolher como fonte da condição inicial o parâmetro *Internal* (interno) e definir a condição inicial. A opção *Show saturation port* (mostrar porta de saturação) habilita uma saída adicional no bloco (saída de baixo) que indica o estado de saturação. A opção *Show state port* (mostrar porta de estado) cria uma saída adicional no bloco. Esta saída contém o mesmo sinal de saída do integrador. Deve-se então usar a saída de estado para este acionamento em duas ocasiões: se a saída do bloco integrador realimenta (feedback) o mesmo bloco como reset ou como condição inicial, e se deseja utilizar a saída do integrador como acionador de um subsistema com execução condicionada a este bloco.

A Figura 15 ilustra um modelo de um bloco integrador com sinal de reset configurado para **falling** e limite de saturação superior igual a 5.

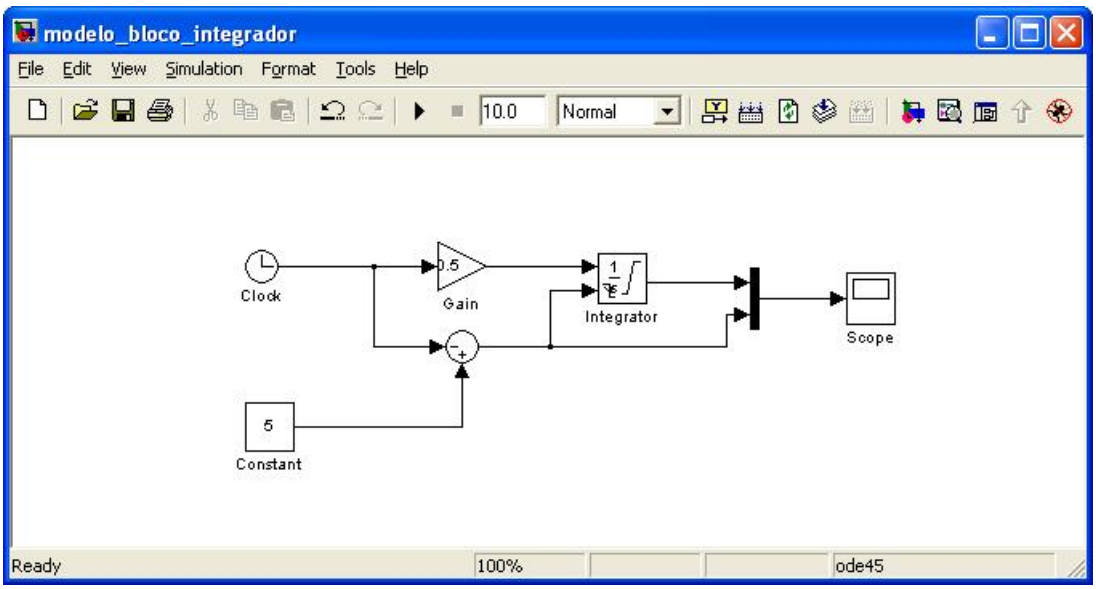

Figura 15: Uso do Bloco Integrador

A resposta do bloco integrador junto com o sinal de reset pode ser observado na Figura 16. Vemos que quando o sinal de reset cruza o zero, a saída do integrador retorna ao seu valor inicial e a simulação continua. Configurando a Fonte de condição inicial para externa, surgirá uma entrada adicional no integrador. O valor contido neste entrada será utilizado pelo integrador como condição inicial quando a simulação for iniciada ou quando o integrador for resetado.

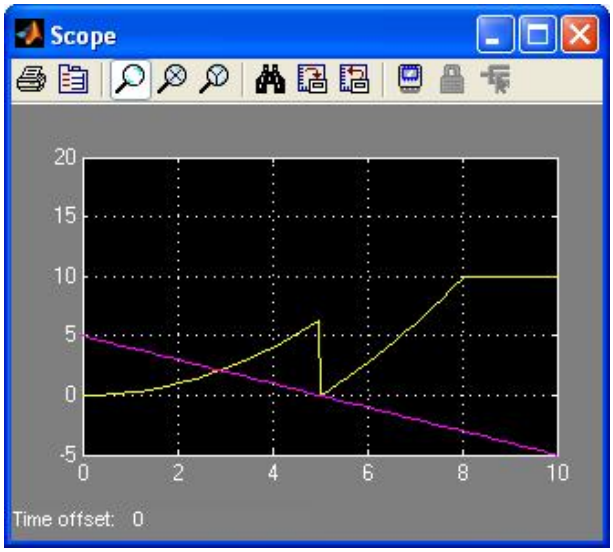

Figura 16: Resposta do Bloco Integrador

6.1.2 Exemplo

Considere o sistema amortecido de segunda ordem da Figura 17.

Sendo a massa do corpo $m = 5\ kg$, admitindo o coeficiente de amortecimento $c = 1\ N \times s/m$ e a constante elástica da mola $k = 2\ N/m$, considerando-se a deflexão inicial $x_o = 1\ m$. Não há entradas no sistema.

Considerando as forças no sistema dinâmico notamos a presença a força da mola e a força de amortecimento. A força da mola é $-k \times x$ e a força de amortecimento é $-c \times \dot{x}$. Desde que não haja forças externas, a soma das forças deve ser a força resultante $m \times \ddot{x}$. Logo podemos escrever a seguinte equação:

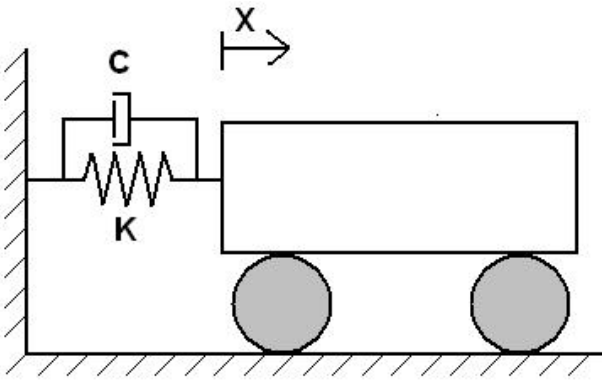

Figura 17: Sistema Amortecido de Segunda Ordem

$$-c \times \dot{x} - k \times x = m \times \ddot{x}$$

Sendo um sistema de segunda ordem, necessitamos de dois integradores para modelar seu comportamento.

Reescrevendo a equação, teremos:

$$\ddot{x} = -\frac{c}{m} \times \dot{x} - \frac{k}{m} \times x$$

Substituindo os valores, teremos:

$$\ddot{x} = -0.2 \times \dot{x} - 0.4 \times x$$

As condições iniciais são: $x(0) = 1$ e $\dot{x}(0) = 0$.

A saída da aceleração é $\ddot{x}$, da velocidade é $\dot{x}$ e da posição é x.

Assim, teremos o sistema conforme a Figura 18.

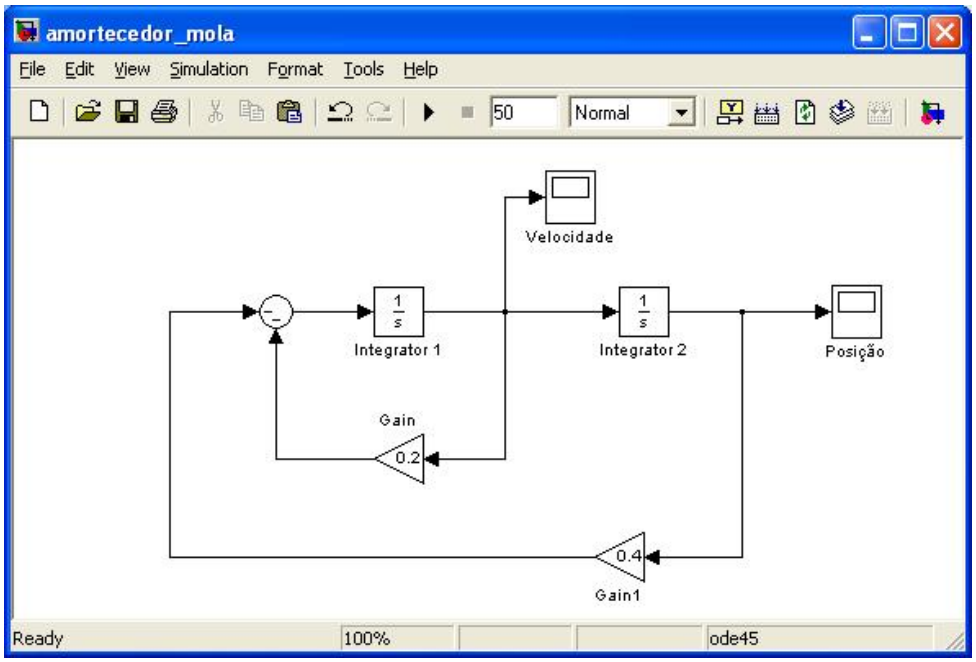

Figura 18: Modelo de Sistema Amortecido de Segunda Ordem

Deve-se configurar a condição inicial do bloco integrador que corresponde à velocidade para 0 e a condição inicial do bloco integrador que corresponde à posição para 1.

A resposta da posição será como na Figura 19.

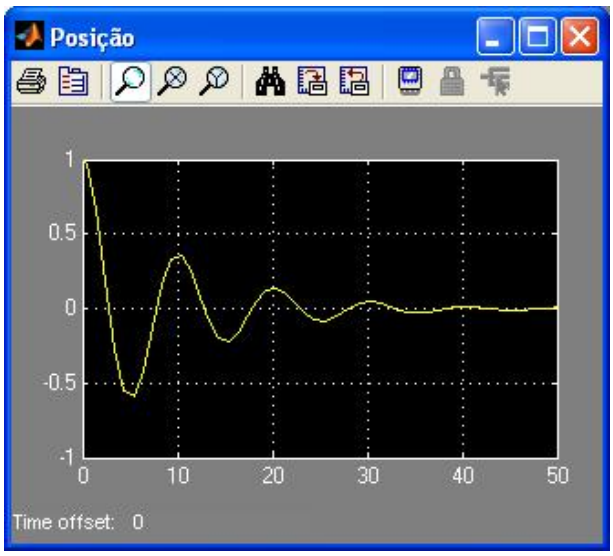

Figura 19: Posição do carro

6.1.3 Bloco Função de Transferência

A Função de Transferência pode ser definida como a razão da transformada de Laplace da entrada de um sistema pela transformada de Laplace da saída, considerando as condições iniciais nulas.

O bloco Função de Transferência tem em sua caixa de diálogo dois campos: *Numerator* (Numerador) e *Denominator* (denominador). O numerador contém os coeficientes do numerador da função de transferência em ordem decrescente de potências de s. O denominador contém os coeficientes do denominador de forma semelhante ao numerador.

6.1.4 Exemplo

Considere o sistema massa-mola amortecido de segunda ordem do exemplo anterior excitado por uma força F, conforme a Figura 20.

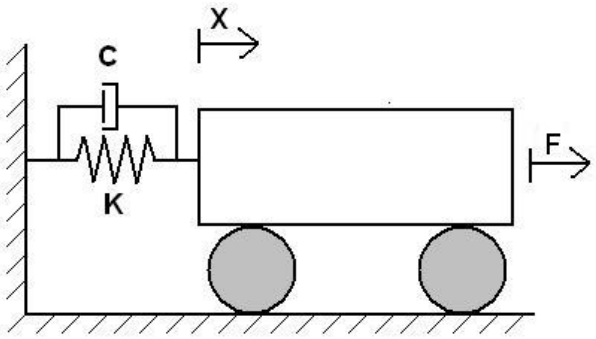

Figura 20: Sistema Amortecido de Segunda Ordem excitado por uma força

A força aplicada ao sistema pode ser modelada por um degrau. Assim, teremos a seguinte modelagem para o problema:

$$F - c \times \dot{x} - k \times x = m \times \ddot{x}$$

Considerando as condições iniciais: $x(0) = 0$ e $\dot{x}(0) = 0$.

Realizando a transformada de Laplace, teremos:

$$F(s) - c \times s \times X(s) - k \times X(s) = m \times s^2 \times X(s)$$

Logo, a função de transferência do sistema será dada por:

$$G(s) = \frac{X(s)}{F(s)} = \frac{\frac{1}{m}}{s^2 + \frac{c}{m} \times s + \frac{k}{m}}$$

Utilizando o bloco função de transferência o numerador deve conter o vetor [0.2] e o denominador o vetor [1 0.2 0.4]. O modelo segue na Figura 21.

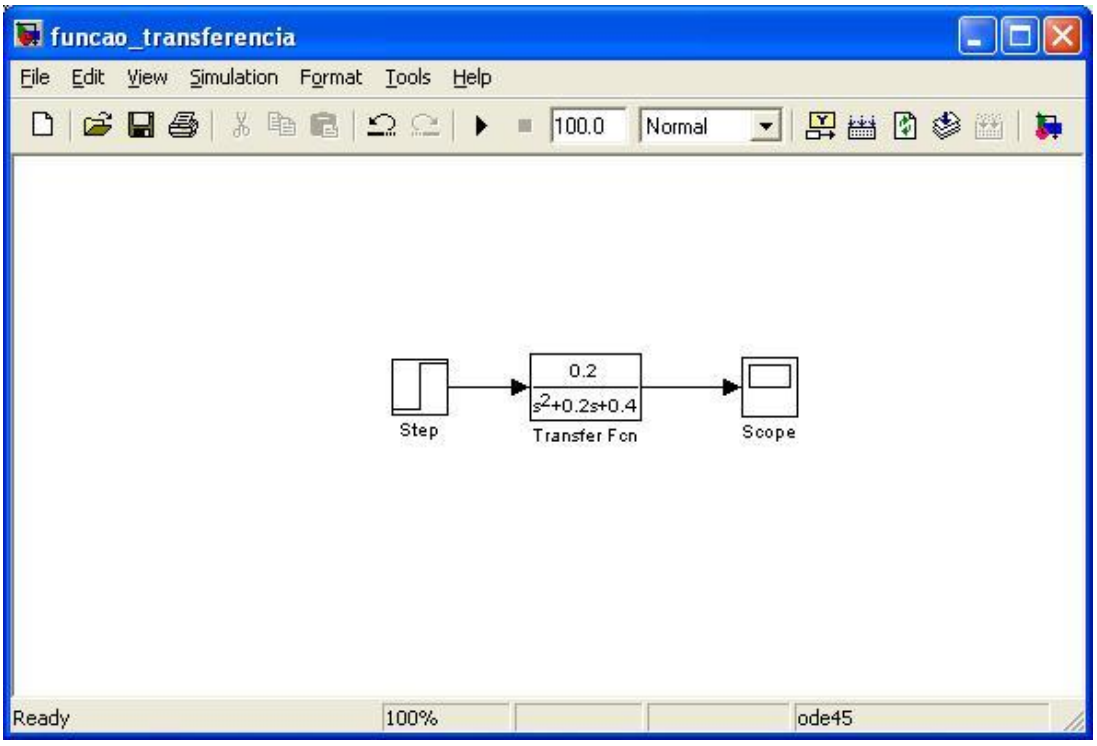

Figura 21: Modelo do Sistema Amortecido de Segunda Ordem excitado por uma força

A resposta do sistema segue na Figura 22.

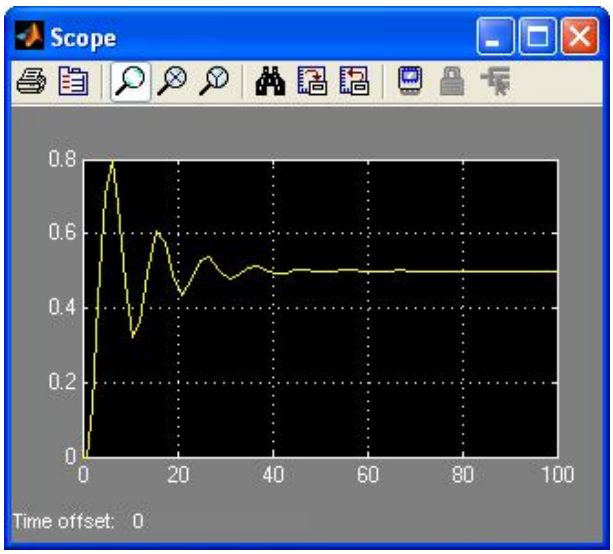

Figura 22: Resposta do Sistema Amortecido de Segunda Ordem excitado por uma força

6.2 Sistemas Contínuos Não-Lineares

O SIMULINK fornece uma gama de blocos para sistemas não lineares. Esses blocos estão presentes na biblioteca *Nonlinear*. O exemplo a seguir ilustrará melhor esta classe de problemas e ainda apresentará alguns componentes para a modelagem não-linear.

Exemplo

Considere um carrinho de massa $M = 5\ kg$ que parte do repouso e que acelera e freia através de uma força $F = 1\ N$. O carrinho acelera durante 10 s e nos próximos 10 segundos freia até parar. Vamos determinar a resposta das grandezas físicas relacionadas ao problema. Considerando a massa do carro e a força de propulsão do motor, pela segunda Lei de Newton, teremos:

$$\ddot{x} = \frac{F}{m}$$

A velocidade e a posição podem ser obtidas através de dois integradores sobre a aceleração. Durante o movimento acelerado a força F é positiva e no movimento retardado a força F é negativa. A simulação deve ser parada quando o automóvel chegar ao repouso.

O componente *Sign* da biblioteca *Math Operations* retorna 1 para uma entrada positiva, 0 para entrada nula e -1 para entrada negativa. Assim, a força muda de direção quando o clock ao ponto de quebra de 10 s. O componente *clock* pode ser obtido da biblioteca *Sources*. Este componente apresenta o tempo corrente da simulação no sistema. O valor do clock é observado em um display.

O ponto de quebra ainda é multiplicado por dois, que é o tempo total de movimento, assim obtemos o carrinho parado no final de seu movimento. O tempo total é comparado com o clock para finalizar a simulação.

Deve-se ligar na entrada X do componente XY Graph a posição e na entrada Y a velocidade, com um duplo clique sobre este componente deve-se alterar o intervalo de variação de X para 0 a 20 e o intervalo de variação de Y para 0 a 2.

A massa é representada no bloco *constant* da biblioteca *Sources*, o valor da constante é alterado para 5.

As condições iniciais são: $x(0) = 0$ e $\dot{x}(0) = 0$.

Assim, teremos o modelo da Figura 23:

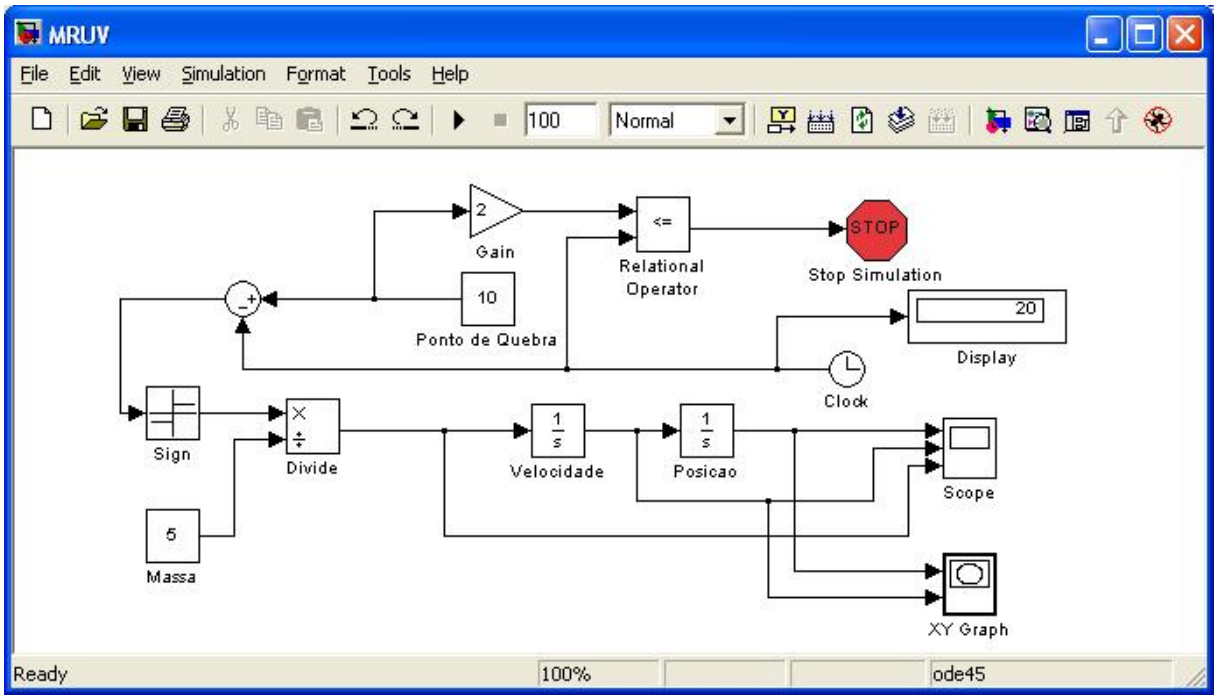

Figura 23: Modelo do problema do carrinho em MRUV

Siga os seguintes passos para verificar em um único scope, como a posição, a velocidade e a aceleração variam:

- Dê um duplo clique no scope;
- clique no ícone parameters (Parametros);
- Digite 3 para o número de axes;
- Ligue as respostas do sistema nas 3 entrados do scope.

As respostas do sistema aparecem na Figura 24.

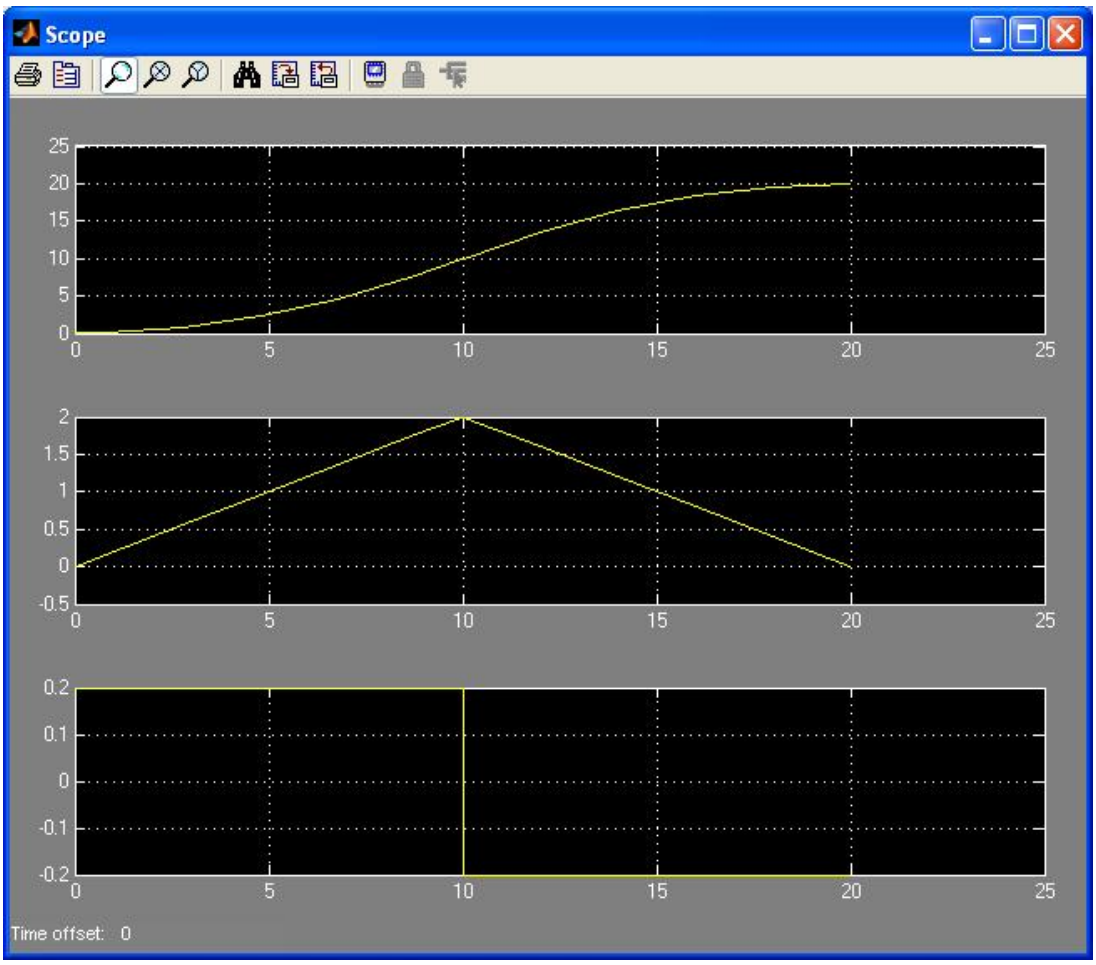

Figura 24: Respostas do carrinho em MRUV

A velocidade em função da posição pode ser verificada através do gráfico da Figura 25.

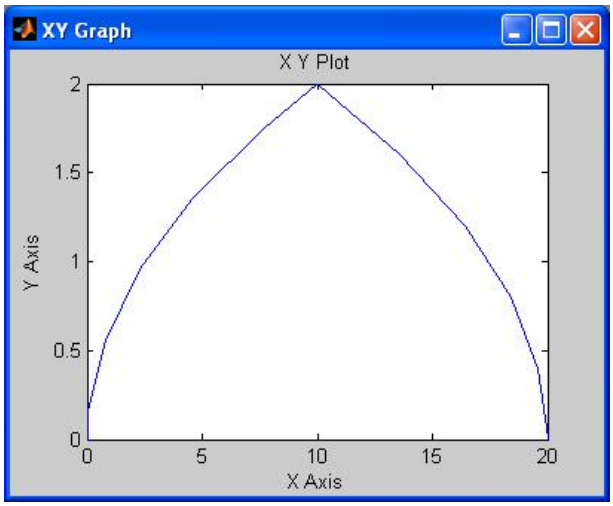

Figura 25: Velocidade em função da posição

7 Sistemas Discretos no Tempo

Um sistema discreto é um sistema que pode ser representado utilizando equações a diferenças. Na maioria das vezes, obtemos o sistema discreto de um sinal contínuo no tempo. Este processo é denominado *amostragem*. Um exemplo de um sinal discreto obtido através de amostragem é apresentado na Figura 26.

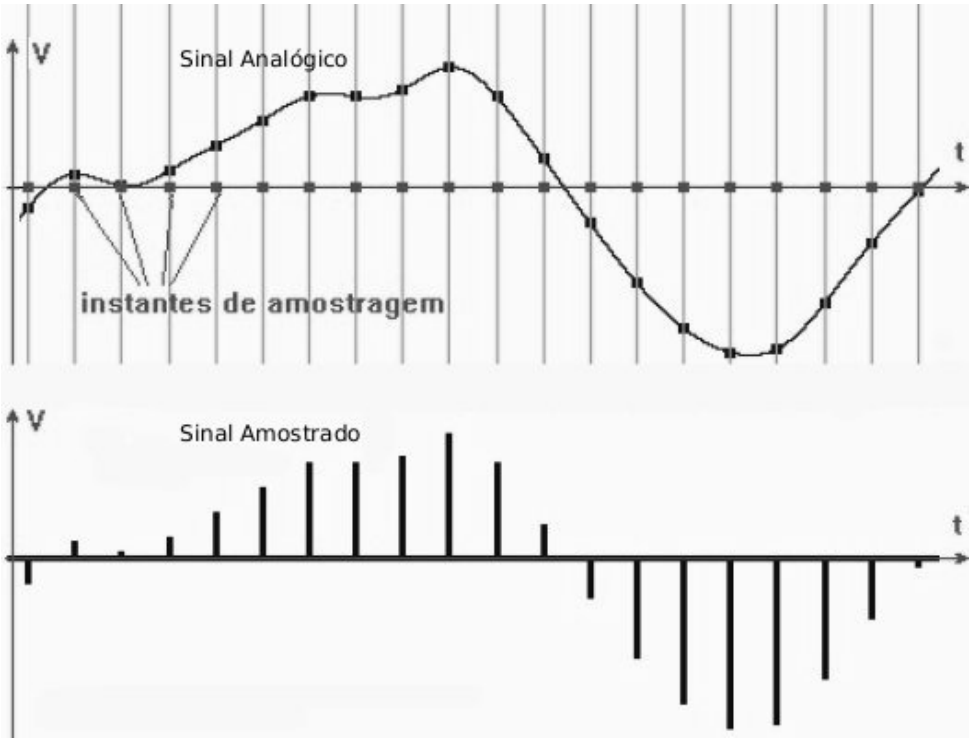

Figura 26: Amostragem

A Figura 27 ilustra o processo de amostragem.

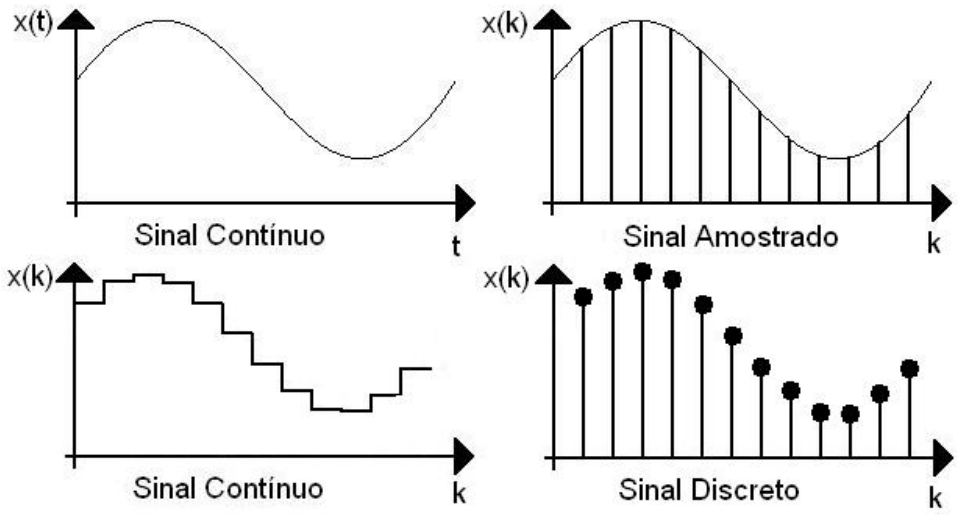

Figura 27: Processo de Amostragem

Este tipo de amostragem utiliza um amostrador, que fecha uma chave periodicamente, e um extrapolador de ordem zero (*zero-order hold*), que produz um sinal na forma de um degrau. A notação para o sinal discreto obtido é x(k), onde k é o número do pulso. O mapeamento para o sinal contínuo é feito por:

$$x(t) = x(kT)$$

Onde T é o período amostral.

A combinação do amostrador e do extrapolador pode ser obtida pelo bloco *Zero-Order Hold* da biblioteca *discrete* (discreta).

7.1 Sistemas Discretos no Tempo Lineares

A modelagem discreta é similar à modelagem contínua. A biblioteca Discreta contém blocos equivales aos blocos contínuos. Todos os blocos discretos possuem um parâmetro sample time que define um intervalo de amostragem.

7.1.1 Atraso Unitário (Unit Delay)

A saída do bloco atraso unitário é a entrada no instante de amostragem anterior. O atraso unitário representa a equação à diferença:

$$y(k) = x(k-1)$$

A caixa de diálogo contém dois campos: um é a condição inicial, que contém o valor da saída, ou outro é o sample time.

Exemplo

Considere o sistema de amortização do pagamento de um dívida. No final de cada mês, é realizado o pagamento da dívida. A dívida no final de cada mês pode ser modelada por:

$$S(k) = (1+i) \times S(k-1) - p(k)$$

Onde i é a taxa de juros mensal.

Sabendo que o valor do financiamento é de R$ 1.800,00, a taxa de juros é 3% ao mês (36 % ao ano) e a mensalidade é de R$ 250,00, determine o valor restante do financiamento após 8 parcelas.

A Figura 28 apresenta o modelo do SIMULINK.

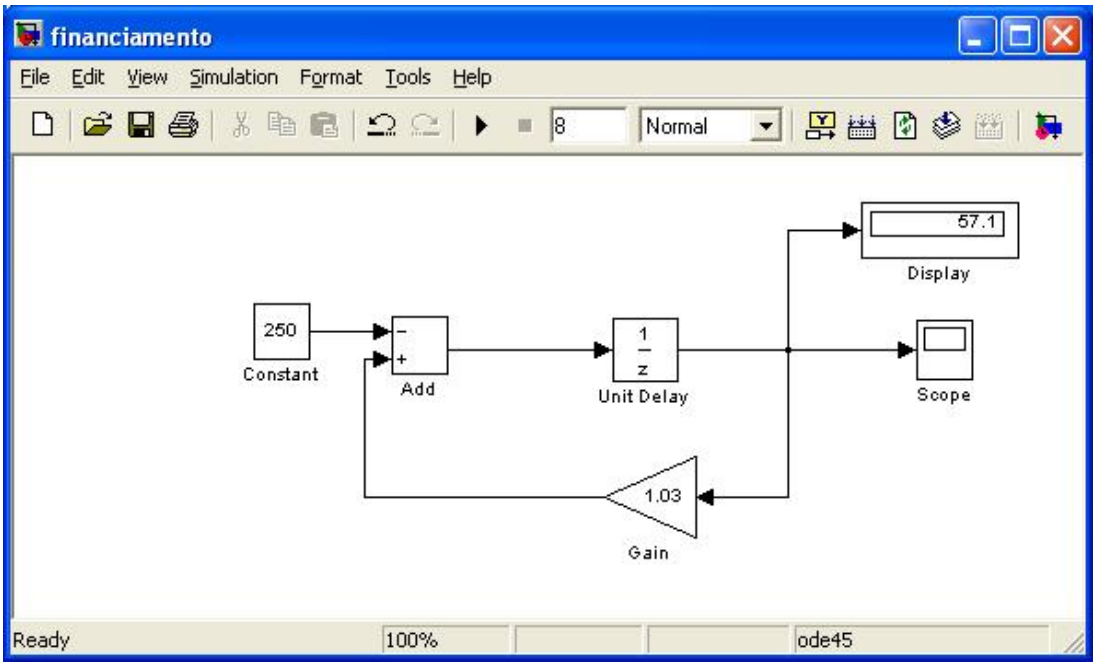

Figura 28: Financiamento

A condição inicial do bloco atraso unitário deve ser 1800 ao passo de 1. A constante é a parcela paga a cada mês, no caso 250. Executamos a simulação para um tempo de 8 e poderemos observar o valor no display.

7.1.2 Integrador no tempo discreto (Discrete-Time Integrador)

Este bloco é uma aproximação discreta do integrador contínuo.

$$y(k) = y(k-1) + \int_{T(k-1}^{T(k)} u(t)dt$$

Onde $u(t)$ é a entrado do integrador, $y(k)$ é a saída e T o período de amostragem.

O bloco pode realizar a integração por três métodos: Euller adiantado (Forward Euller), Euller atrasado (Backward Euller) e trapezoidal.

Exemplo

Considerando o financiamento do exemplo anterior, o valor do financiamento pode ser determinado por:

$$S(k) = S(k-1) + \int_{T(k-1}^{T(k)} (i \times S(k-1) - p)dt$$

Analisando o integrado, pode-se ver que o problema pode ser resolvido utilizando a integração de Euller adiantado (Forward Euller).

O novo modelo é apresentado na Figura 29. Note que, neste caso, o ganho de retroação é $i = 3\%$.

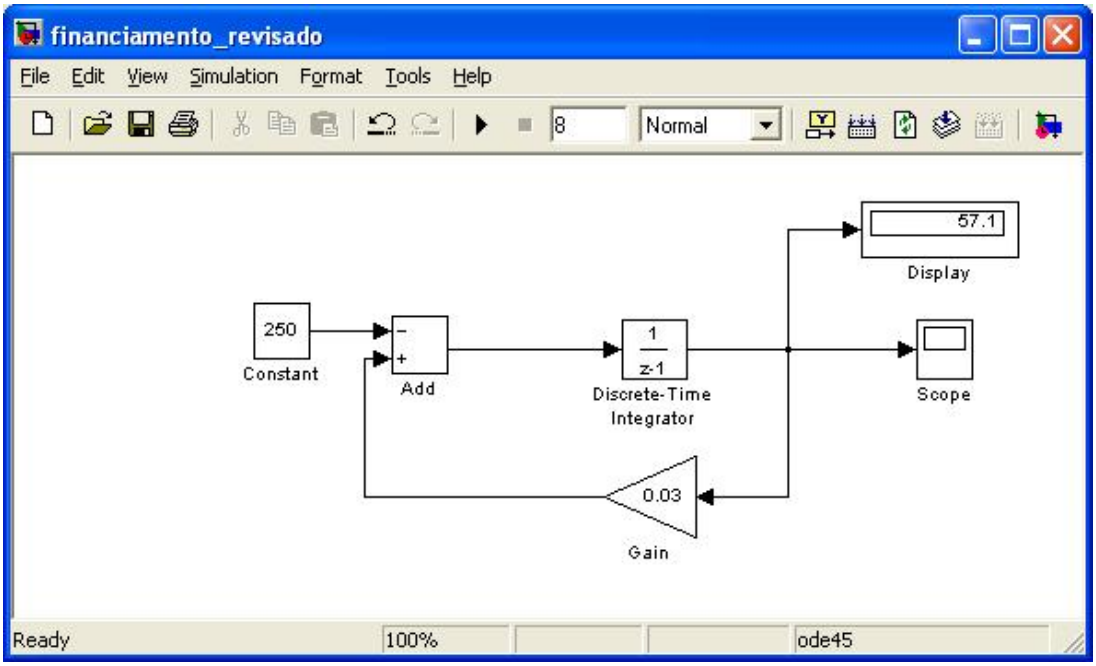

Figura 29: Financiamento revisado

7.1.3 Função de Transferência Discreta

A função de transferência é a razão entre a transformada Z do sinal de saída pela transformada Z do sinal de entrada. A biblioteca Discreta fornece 3 blocos que implementam funções de transferência discreta: *Discrete Filter* (Filtro Discreto), *Discrete Transfer Fcn* (Função de Transferência Discreta) e *Discrete Zero-Pole* (Zeros-Pólos Discretos). Todos estes blocos são equivalentes, diferindo somente nos coeficientes dos polinômios do numerador e do denominador.

O bloco Filtro Discreto necessita de coeficientes dos polinômios em potências ascendentes de Z^{-1}. O bloco Função de Transferência Discreta requer coeficientes dos polinômios em potências descendentes de Z. Os dois blocos são equivalentes, diferindo somente no modo de representar a função de transferência. O bloco Pólos-Zeros Discreto requerem vetores de zeros (numerador) e pólos (denominador) da função de transferência e o ganho. Note que esses blocos possuem os seus equivalentes para sistema contínuos e podem ser encontrados na biblioteca *Continuos*.

Exemplo

Geralmente, filtros são empregados para remover ruídos em altas frequências dos sinais de entrada. O MATLAB possui uma Toolbox de processamento de sinais que fornece uma grande variedade de algoritmos de projetos de filtros. O projetista pode simular um filtro projetado no MATLAB em um modelo do SIMULINK utilizando um bloco *Discrete Transfer Fcn* (Função de Transferência Discreta).

Suponha que se deseja projetar um sistema de controle com um ruído senoidal na entrada e precisa-se filtrar este ruído. O período de amostragem (Sampling Period) é 0.1 segundos e se deseja remover sinais com uma frequência superior a 0.2 Hz.

Os comandos apresentados no Exemplo 8 fornecem os coeficientes da função de transferência do filtro Butterworth de quarta ordem com frequência de corte em 0.2 Hz.

Exemplo 8: *Numerador e denominador do polinômio da função de transferência de Butterworth*

```
>> [B, A] = butter(4, 0.2)
B =
    0.0048    0.0193    0.0289    0.0193    0.0048
A =
    1.0000   -2.3695    2.3140   -1.0547    0.1874
```

Os vetores B e A devem ser configurados, respectivamente, como numerador e denominador do bloco Função de Transferência Discreta. O campo Numerador da caixa de diálogo do bloco Função de Transferência deve conter o vetor [0.0048 0.0193 0.0289 0.0193 0.0048] e o denominador [1.0000 − 2.3695 2.3140 − 1.0547 0.1874]. Como as variáveis *A* e *B* estão presentes no workspace, pode-se também usar diretamente essas variáveis na caixa de diálogo do bloco Função de Transferência para definir o vetor do numerador e o vetor do denominador da FT. O *Sample Time* (Período de amostragem) desse bloco bloco foi definido em 0.1.

O modelo do filtro segue na Figura 30.

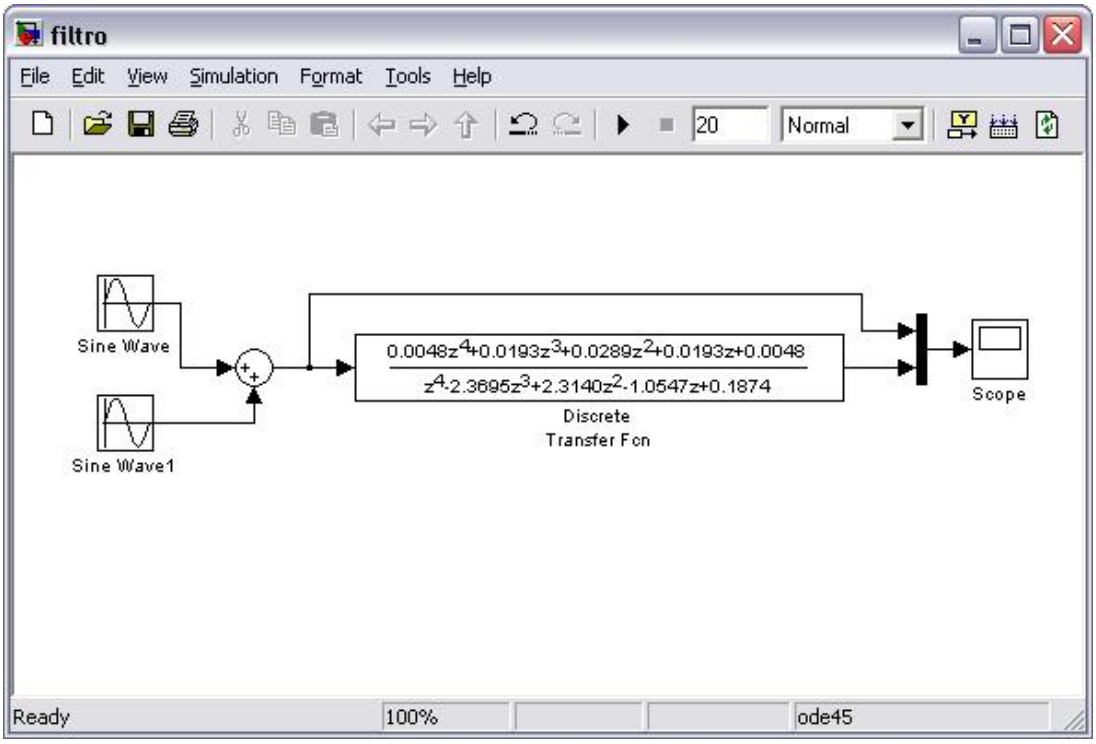

Figura 30: Modelo do filtro

O bloco de onda senoidal superior representa o sinal sem o ruído e está configurado para uma frequência de 0.5 rad/s com uma amplitude de 1. O bloco inferior representa o ruído de alta frequência de 10 rad/s e amplitude 0.4. A simulação foi configurada com o tempo final de 20.

Neste exemplo usou-se um novo bloco, *Mux*, disponível da biblioteca *Commonly Used Blocks*. Este bloco, acoplado ao bloco *Scope*, faz com que o *Scope* exiba os dois sinais em uma mesma área, diferentemente de quando se cria duas entradas no *scope*, fazendo com que cada sinal seja mostrado em um gráfico diferente. A resposta do sistema pode ser visualizado na Figura 31. O sinal com ruído é mostrado de amarelo, enquanto o sinal filtrado é mostrado em rosa.

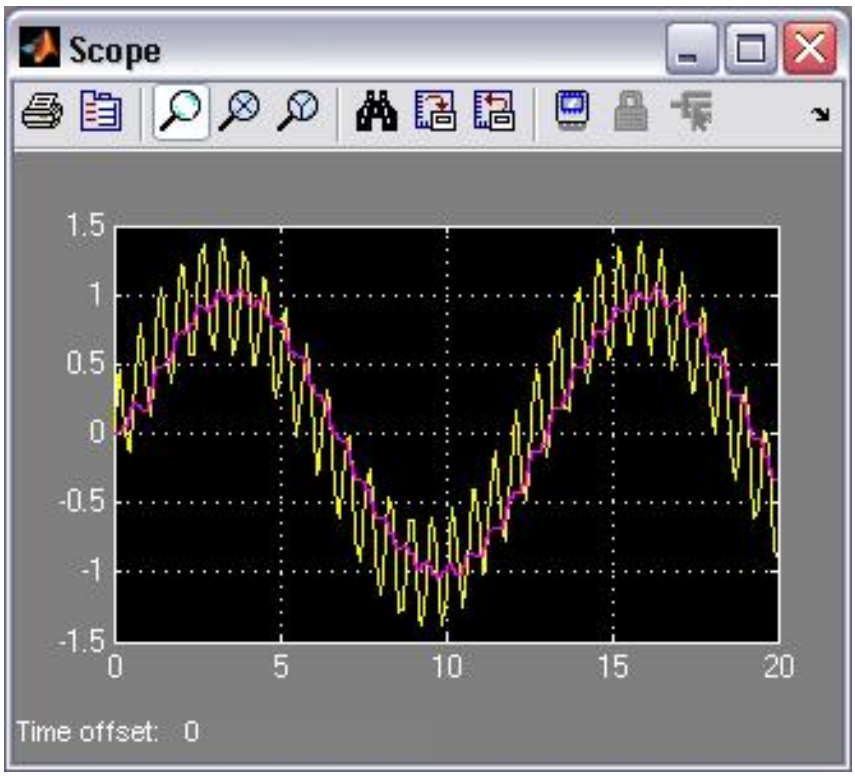

Figura 31: Onda com ruído e filtrada

8 Aplicação

Nesta seção, usou-se o trabalho de Controle Automático I proposto pelo professor Edson de Paula Ferreira (Dept° de Eng. Elétrica - UFES) e feito pelo aluno Rodrigo Lopes Batista (Engenharia de Computação e então bolsista PET Eng Comp), em 2006 / 2. Neste trabalho, usou-se o Simulink para projetor um servo posicionador e estudar o efeito da variação de alguns componentes do modelo.

Enunciado:

Um servomotor constitui um modelo de um sistema eletromecânico através de um motor dc projetado especificamente para ser usado em um sistema de controle de malha fechada. O diagrama do circuito de um servomotor é apresentado na Figura 32.

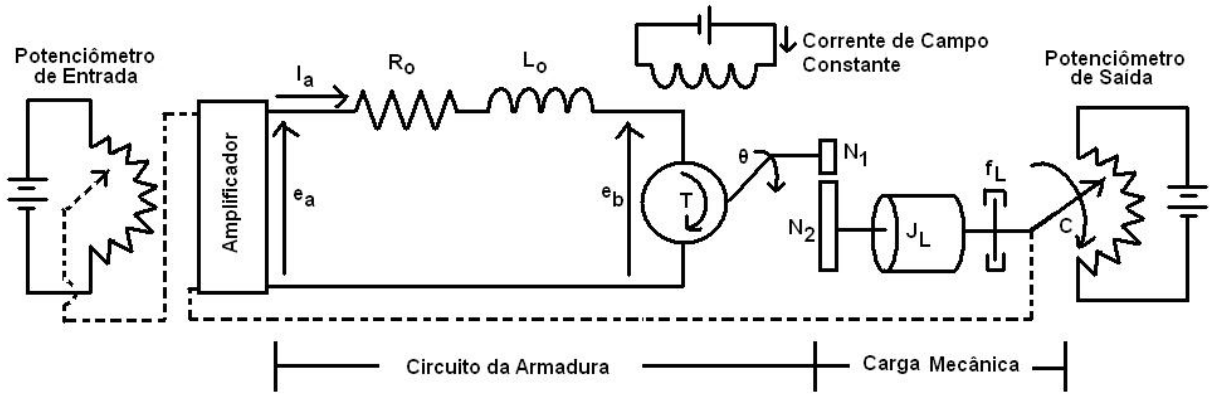

Figura 32: Mecanismo servo motor

Um exemplo prático para uma aplicação do mecanismo servomotor é o sistema de rastreamento por radar. O objetivo deste sistema é rastrear uma aeronave ou algum alvo equivalente

automaticamente, onde o ângulo de saída (c) é corrigido para que ele esteja orientado com a aeronave.

8.1 Especificações dos parâmetros

Segue abaixo a lista dos valores nominais dos parâmetros fornecidos, no S.I. (Sistema Internacional).

- r: Ângulo do eixo da entrada de referência (em radianos).
- c: Ângulo do eixo de saída (em radianos).
- θ: Ângulo do eixo do motor (em radianos).
- k_V: Ganho do detector de erro do potenciômetro, $k_V = \frac{24}{\pi}\ V/rad$.
- k_p: Amplificação de 10 V/V. Este é o ganho proporcional (ajustável).
- e_a: Tensão na armadura, é considerada entrada no sistema (em Volts).
- e_b: é a tensão gerada na armadura devido ao momento de sua bobina no campo magnético do motor e normalmente é chamada de força contra-eletromotriz, FCEM (em Volts).
- R_b: A resistência da armadura é $R_b = 0,2\Omega$.
- L_a: A indutância da armadura é L_a, uma aproximação que pode ser feita é ignorar a indutância da armadura quando esta é pequena o suficiente, $L_a = 0$.
- i_a: é a corrente na armadura, que é dada em ampéres.
- k_b: Constante da FCEM, $k_b = 5,5 \times 10^{-2}\ V \times s/rad$.
- k: Constante do torque do motor, $k = 8,13 \times 10^{-5}\ N \times m/A$.
- J_m: Momento de inércia do motor, $J_m = 1,36 \times 10^{-5}\ N \times m \times s^2$.
- f_m: Coeficiente de atrito viscoso do motor, que é desprezível, $f_m = 0$.
- J_L: Momento de inércia da carga, $J_L = 5,96 \times 10^{-3}\ N \times m \times s^2$.
- f_L: Coeficiente de atrito viscoso da carga, $f_L = 5,42 \times 10^{-2}\ N \times m/rad/s$
- n: fator de redução de trem de engrenagens, $n = \dfrac{N1}{N2} = \dfrac{1}{33}$.

8.2 Equações do Sistema dinâmico

As equações que descrevem o sistema dinâmico são as seguintes:

8.2.1 Do potenciômetro detector de erros

A tensão de entrada do amplificador é o produto da constante de ganho do detector de erro do potenciômetro pela diferença entre os ângulos dos ponteiros do potenciômetro.

$$E(s) = k_V \times [R(s) - C(s)]$$

8.2.2 Do Amplificador

O amplificador amplifica a tensão de entrada E(s) para a tensão de entrada do motor dc $E_a(s)$.

$$E_a(s) = k_P \times E(s)$$

8.2.3 Da armadura controlada do motor

Equivalente Momento de Inércia (J)

$$J = J_m + n^2 \times J_L$$

Atrito viscoso referente ao eixo do motor (f)

$$f = f_m + n^2 \times f_L$$

8.3 Função de Transferência do Motor (Planta)

8.3.1 Força contra eletromotriz no motor

$$e_b(t) = K \times \phi \times \frac{d\theta}{dt} = K_m \times \frac{d\theta}{dt}$$

Tomando a transformada de Laplace da equação acima:

$$E_b(s) = k_m \times s \times \theta(s)$$

8.3.2 Do circuito da armadura podemos escrever

$$E_a(s) = (L_a s + R_a) \times I_a(s) + E_b(s)$$

Que resolvendo para $I_a(s)$ e desprezando L_a, teremos:

$$I_a(s) = \frac{E_a(s) - E_b(s)}{R_a}$$

8.3.3 Torque desenvolvido

$$\tau(t) = k_1 \times \phi \times i_a(t) = k_\tau \times i_a(t)$$

Tomando a transformada de Laplace da equação acima, temos:

$$T(s) = k_\tau \times I_a(s)$$

8.3.4 Torque resultante na armadura do motor

Pela 2ª Lei de Newton, na forma angular, temos:

$$\tau(t) - f \times \frac{d\theta}{dt} = J \times \frac{d^2\theta}{dt^2}$$

Tomando a transformada de Laplace da equação acima e reescrevendo-a, obtemos:

$$\theta(s) = \frac{T(s)}{J \times s^2 + f \times s}$$

Assim, obtemos o diagrama de blocos da Figura 33.

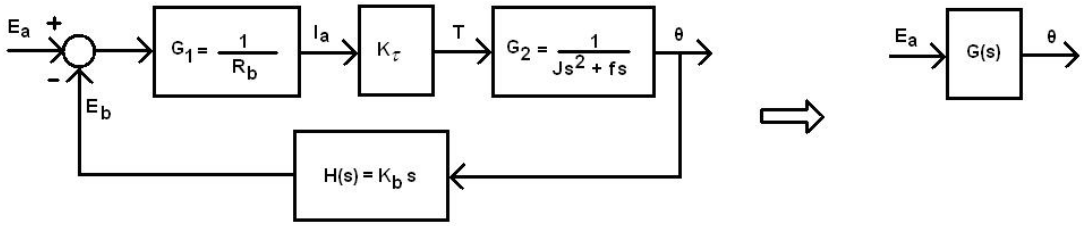

Figura 33: Diagrama de blocos da planta

8.4 Função de transferência do motor (FTMF)

Da fórmula de ganho de Mason, obtemos G(s) que é a função de transferência do motor.

$$G(s) = \frac{\Theta(s)}{E_a(s)} = \frac{G_1(s) \times k_\tau \times G_2(s)}{1 + G_1(s) \times k_\tau \times G_2(s) \times H(s)}$$

Assim:

$$G(s) = \frac{k}{s^2 \times R_b \times J + s \times (R_b \times f + k_\tau \times k_b)}$$

Simplificando, teremos:

$$G(s) = \frac{k_m}{s \times (s \times T_m + 1)}$$

Onde:

$$K_m = \frac{k}{R_b \times f + k \times k_b}$$

$$T_m = \frac{R_b \times J}{R_b \times f + k \times k_b}$$

Realizando os devidos cálculos, obtemos a FTMF da planta:

$$G(s) = \frac{5,63}{s \times (s \times 0,26 + 1)}$$

8.5 Diagrama de Blocos do Sistema

Assim, escrevemos o diagrama de blocos de todo o sistema na Figura 34.

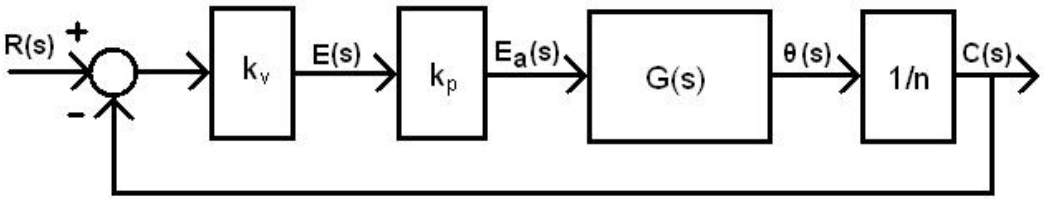

Figura 34: Diagrama de blocos do sistema

8.6 Ajuste de Kp

Enunciado:

Ajuste o ganho Kp para que a resposta c(t) a um degrau unitário r(t) = 1 ou R(s) = 1/S, seja oscilatória com o primeiro pico (sobrelevação máxima ou overshoot) 15% acima do seu valor final c(∞). Verifique se o valor de saída confere com o valor calculado pelo teorema do valor final:

$$c(\infty) = \lim_{t\to\infty} C(t) = \lim_{t\to 0} S \times C(s)$$

Para que o primeiro pico (sobrelevação máxima ou overshoot) seja 15% acima de seu valor final o overshoot (M_p) deve ser de 0,15. Alterando-se o valor de k_p até que se atingisse este overshoot encontramos, aproximadamente, $k_p = 2.8$. A saída do sistema, c(t), é mostrada na Figura 35. Variando-se o ganho K_p, foi possível observar que Mp é maior tanto quanto maior for o ganho proporcional (K_p).

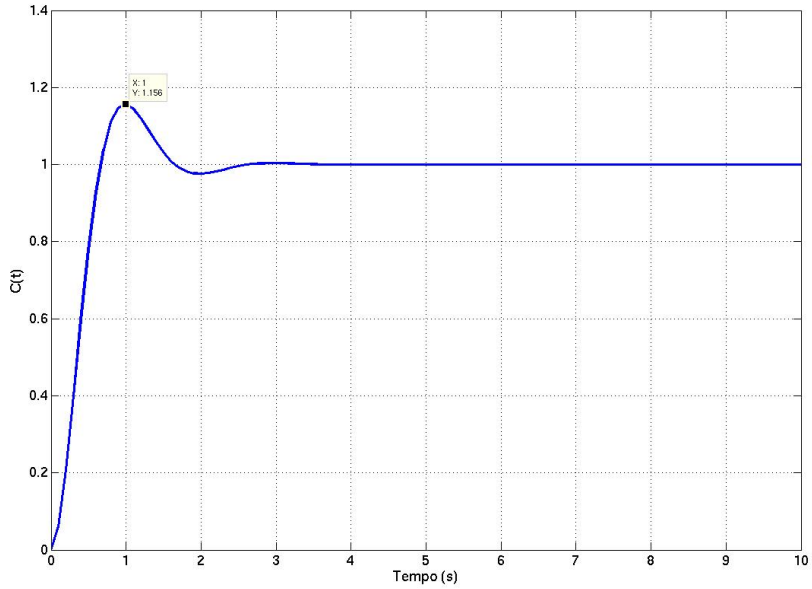

Figura 35: Resposta a um Degrau para overshoot de 15%

O diagrama de blocos modelado no Simulink é mostrado na Figura 36.

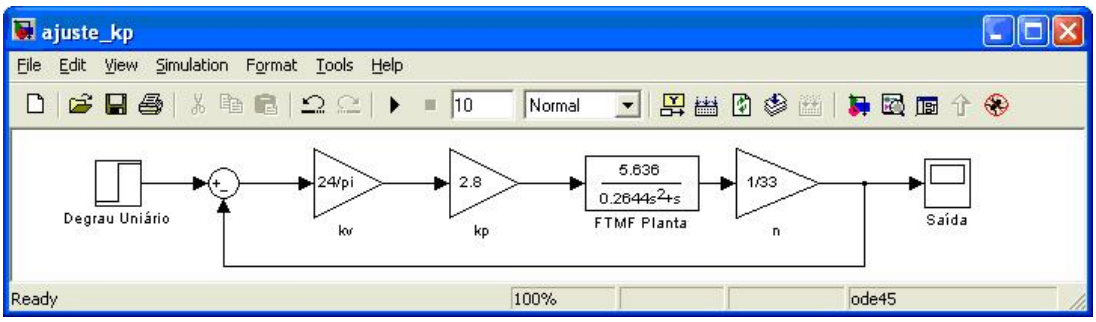

Figura 36: Diagrama de Blocos do modelo

8.7 Análise do efeito de uma não linearidade - Saturação

Enunciado:

Considere uma saturação na saída do controlador, limitado por um valor máximo (u_{max}), variando adequadamente este máximo, conforme a Figura 37. Para que valor de u_{max}, você considera que a resposta se tornou muito ruim?

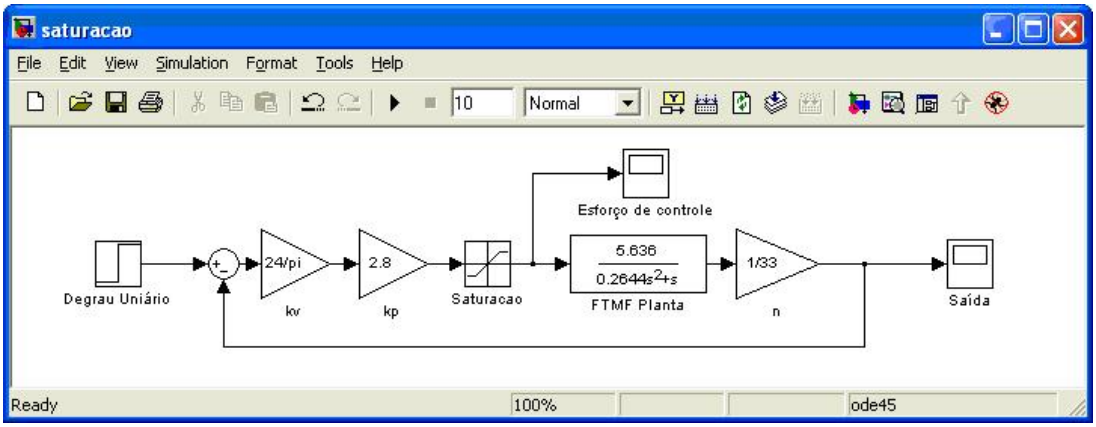

Figura 37: Diagrama de Blocos do modelo com uma saturação

Para modelar a saturação, usou-se o bloco *Saturation* da biblioteca *Discontinuities*. Nas propriedades do bloco, deve-se ajustar os valores de *Upper limit* (limite superior) e *Lower limit* (Limite inferior).

8.7.1 Análise

Testes:

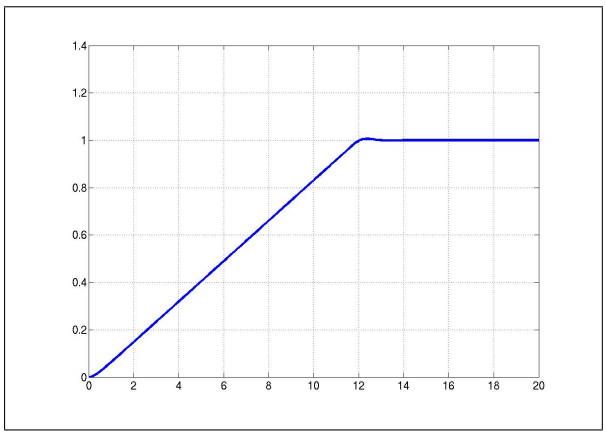

Figura 38: **Resposta (u_{max} = 0.5)**

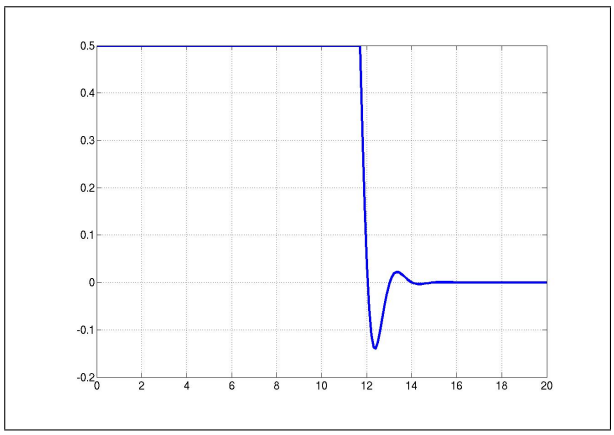

Figura 39: **Esforço de Controle (u_{max} = 0.5)**

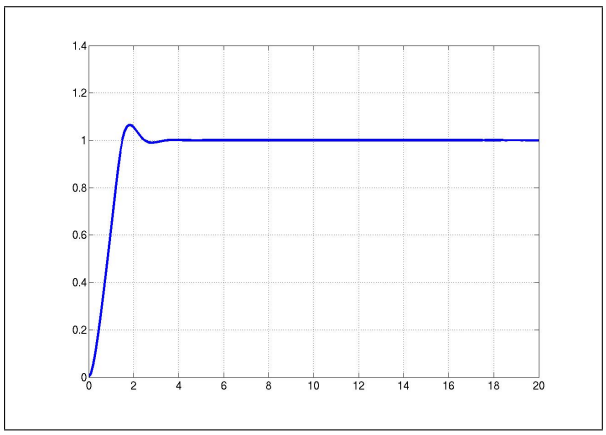

Figura 40: **Resposta (u_{max} = 5)**

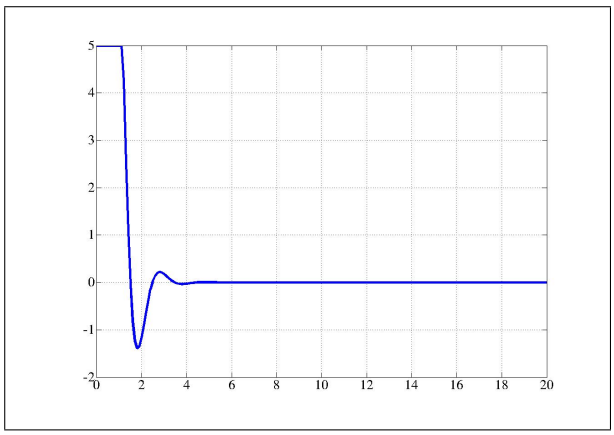

Figura 41: **Esforço de Controle (u_{max} = 5)**

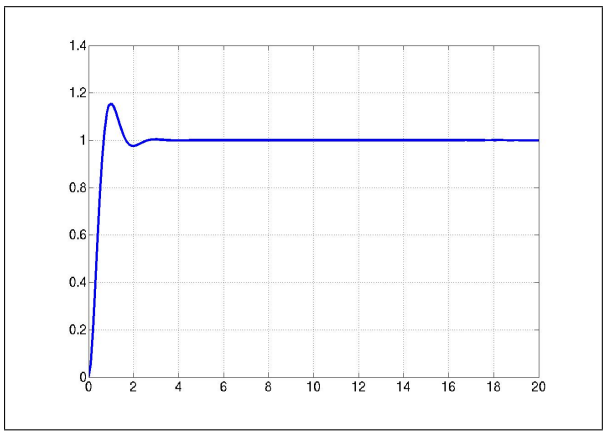

Figura 42: **Resposta (u_{max} = 20)**

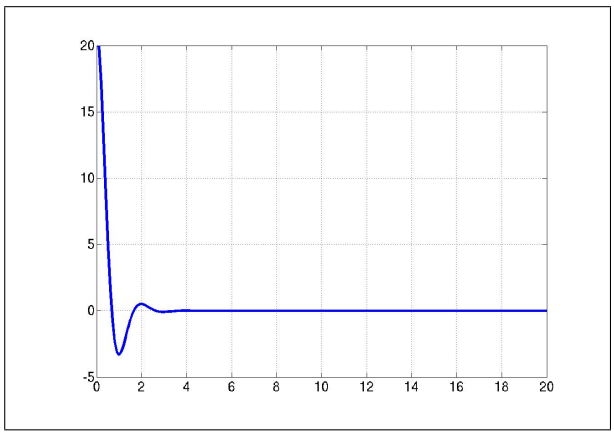

Figura 43: **Esforço de Controle (u_{max} = 20)**

8.7.2 Conclusões

- A função da saturação é a de modelar um sistema real. Contudo, para valores muitíssimos pequenos de u_max o tempo de estabilização é tão grande quanto menor for u_max e o overshoot é desprezível (muitíssimo próximo de zero). Aumentando u_max o tempo de estabilização diminui e o overshoot cresce.
- Para valores maiores que 20, o sistema deixa de saturar e para valores muito pequenos de u_max o tempo de estabilização é muito grande.

8.8 Análise do efeito de uma não linearidade - Atraso

Enunciado:

Considere um atraso em caminho direto (e^{-sT} em série com a FT da planta), de T segundos, variando T, conforme a Figura 44. Para que valor de T você considera que a resposta se tornou muito ruim?

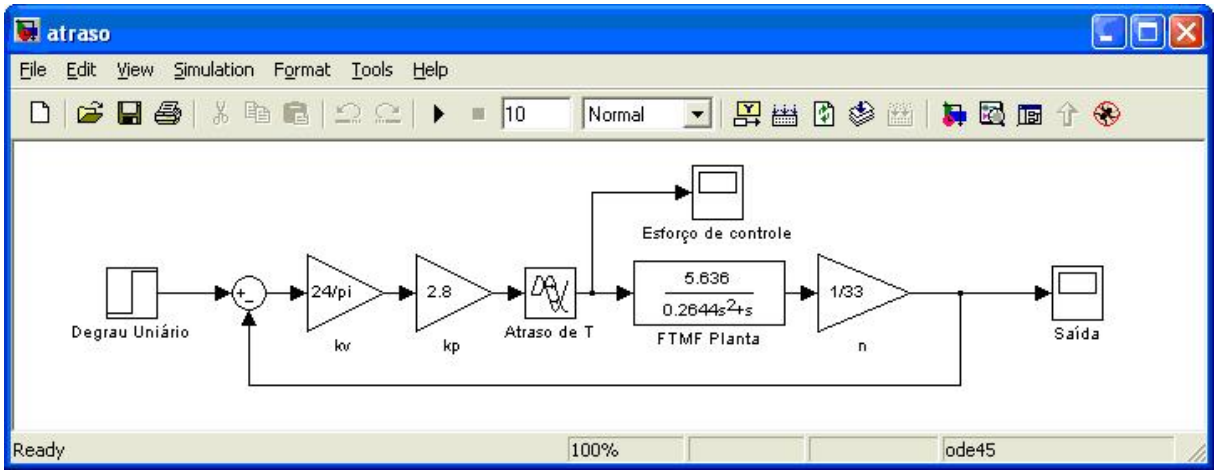

Figura 44: Diagrama de Blocos do modelo com atraso

Para modelar o atraso, usou-se o bloco *Transport Delay* da biblioteca *Continuous*. Nas propriedades do bloco, deve-se ajustar o valor do *Time Delay* (Tempo de atraso).

8.8.1 Análise

Testes:

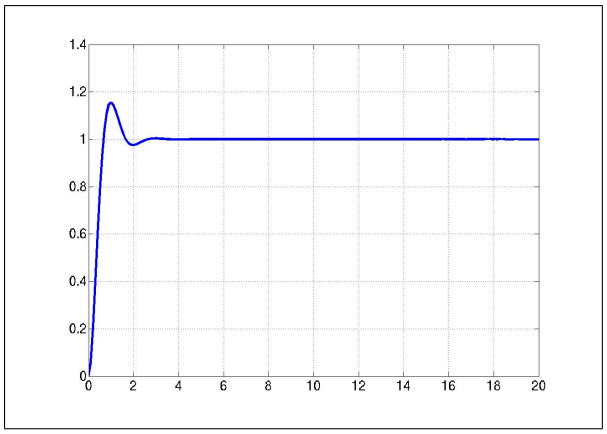

Figura 45: **Resposta (T = 0)**

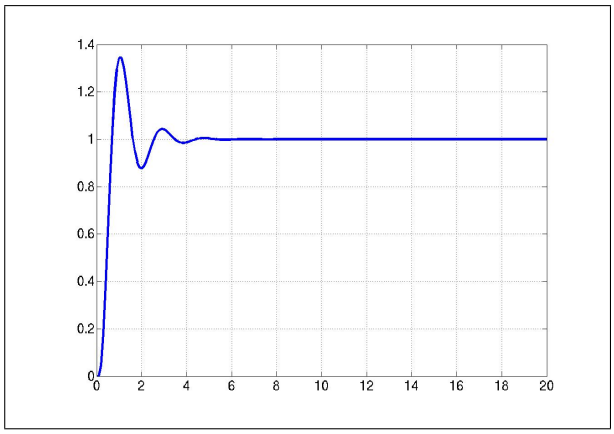

Figura 46: **Resposta (T = 0.1**

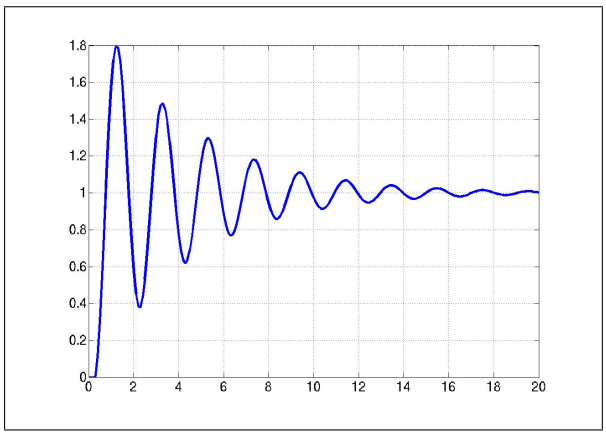

Figura 47: **Resposta (T = 0.25)**

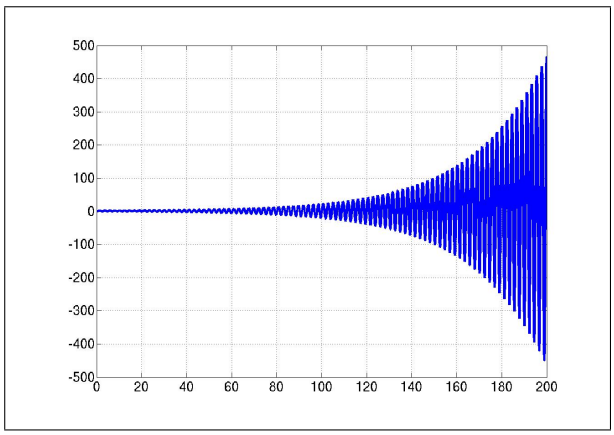

Figura 48: **Resposta (T = 0.33**

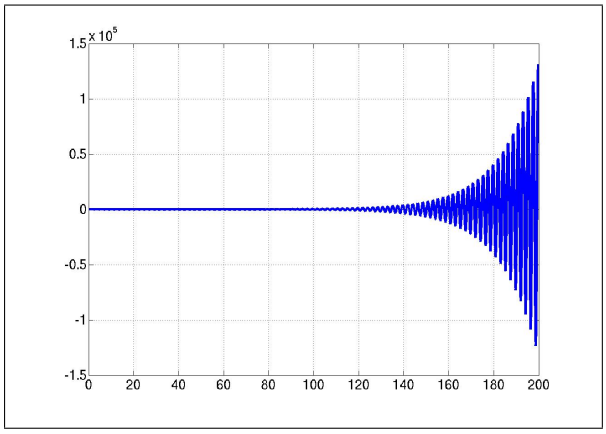

Figura 49: **Resposta (T = 0.34)**

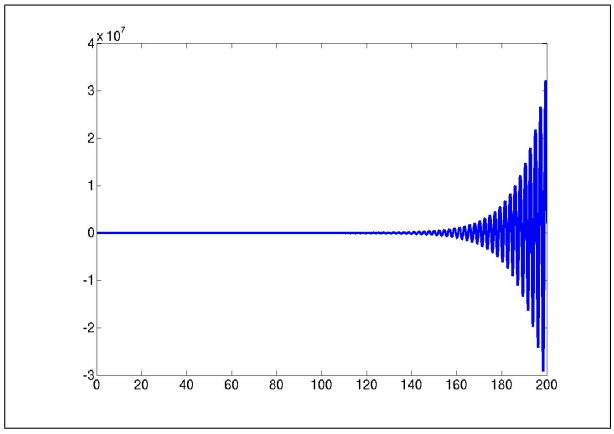

Figura 50: **Resposta (T = 0.35**

8.8.2 Conclusões

Verificamos que quando o atraso cresce a partir de T = 0, o seu overshoot cresce juntamente com o tempo de estabilização, assim o sistema oscila mais. Quanto maior for o delay, maior será a oscilação e maior o overshoot. No entanto, a partir de, aproximadamente, T = 0,33 o sistema se torna instável, divergindo para todo atraso maior que 0,33.

8.9 Análise do efeito do período de amostragem e do erro de quantificação na resposta ao degrau unitário

Enunciado:

Considerando o controle digital sobre um motor, que é a planta do sistema de controle, inclua um amostrador, com período de amostragem T e um hold de ordem zero na saída do controlador, como conversor DA. Inclua também um ruído, devido à codificação do sinal no conversor AD, na malha de realimentação da posição c(t).

8.9.1 Modificações

- Na saída do controlador foi colocado um Hold de ordem zero, como conversor D/A.
- Na malha de realimentação de c(t) um ruído foi somado.

- O período de amostragem foi alterado no hold e no componente do ruído para manter sincronização.
- O poder do ruído foi alterado no componente de ruído para verificar o erro de quantificação na resposta do sistema.
- Alguns valores de ruídos foram fixados, e a partir de um determinado valor o período de amostragem variou para obter a resposta.

O diagrama de blocos do modelo de controle digital segue na Figura 51.

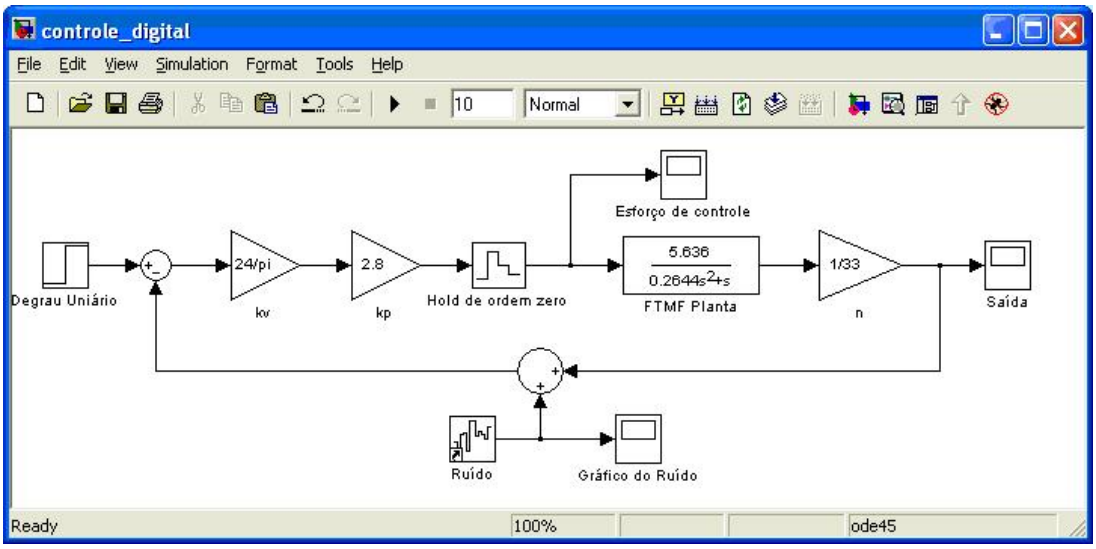

Figura 51: Diagrama de Blocos do modelo de Controle Digital

8.9.2 Análise

A saída do sistema como definido no ajuste, sem a influência do ruído, é verifica na Figura 35. Logo o erro máximo do sistema é de **1,4**; e o Noise Power pode ser definido por:

$$Noise\ Power = \frac{1,4}{2^{n+1}}$$

Onde 2^n é o número de bits da palavra.

Palavra de 4 bits

$$Noise\ Power = \frac{1,4}{2^{4+1}} = 0,04375$$

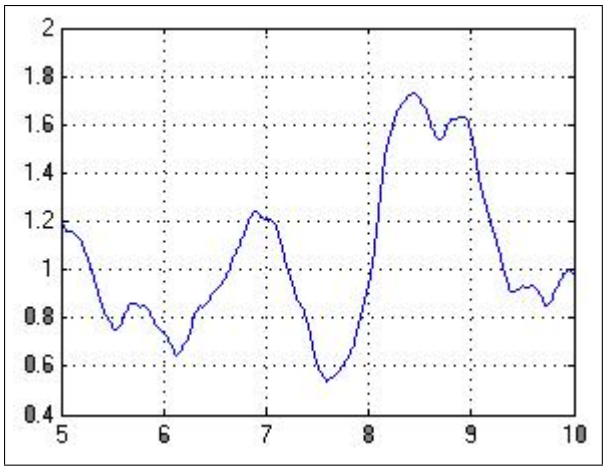

Figura 52: **Resposta (T = 0.001)**

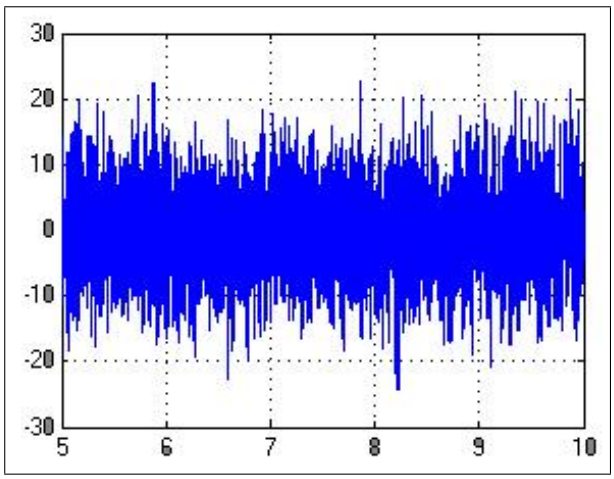

Figura 53: **Ruído (T = 0.001)**

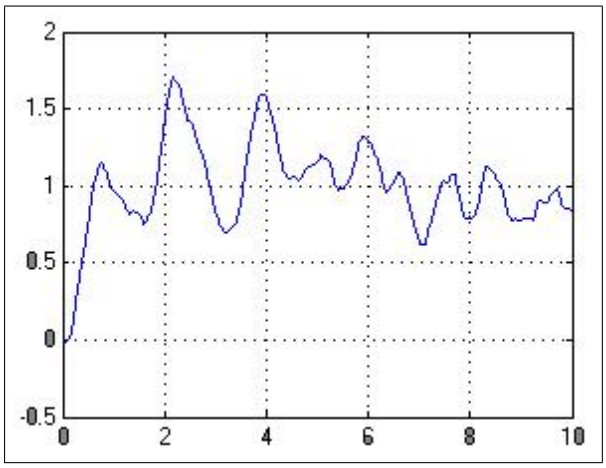

Figura 54: **Resposta (T = 0.01)**

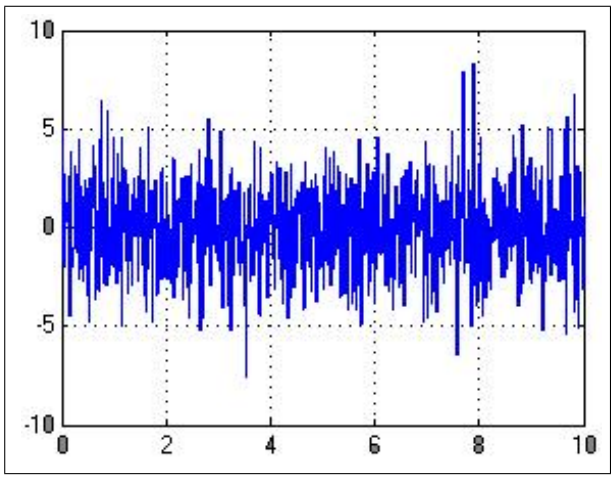

Figura 55: **Ruído (T = 0.01)**

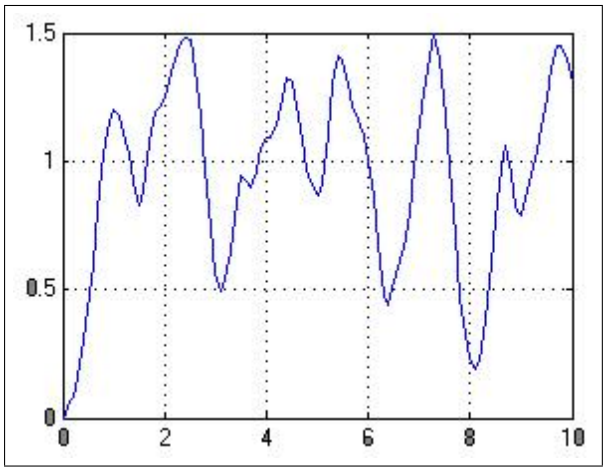

Figura 56: **Resposta (T = 0.1)**

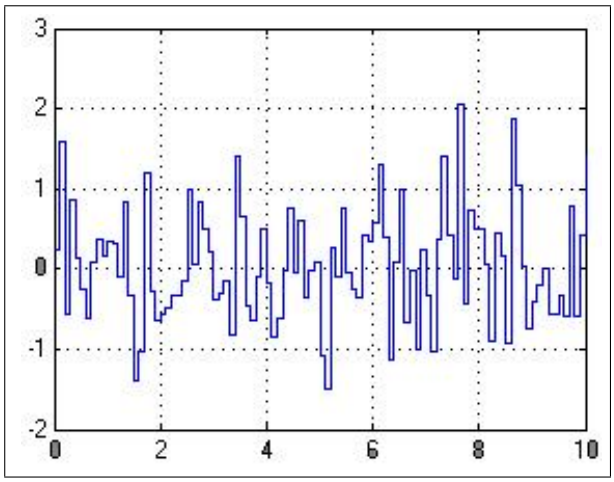

Figura 57: **Ruído (T = 0.1)**

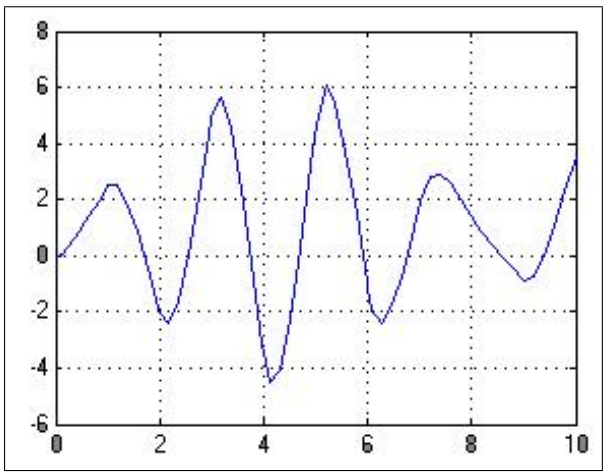

Figura 58: **Resposta (T = 1)**

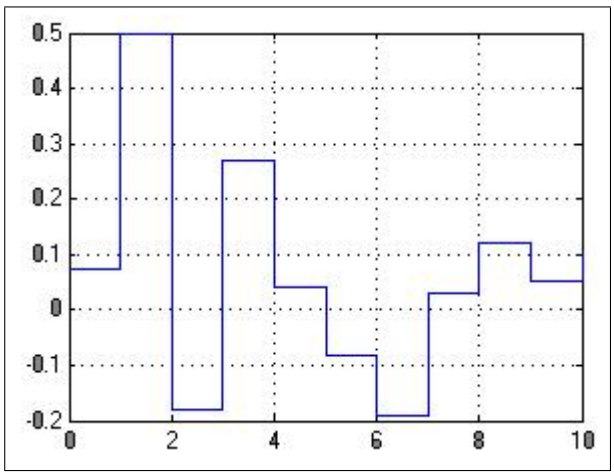

Figura 59: **Ruído (T = 1)**

Palavra de 16 bits

$$Noise\ Power = \frac{1,4}{2^{16+1}} = 0,00001068115234$$

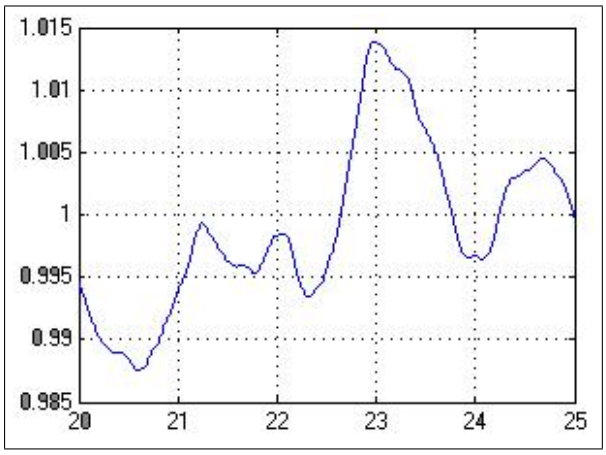

Figura 60: **Resposta (T = 0,001)**

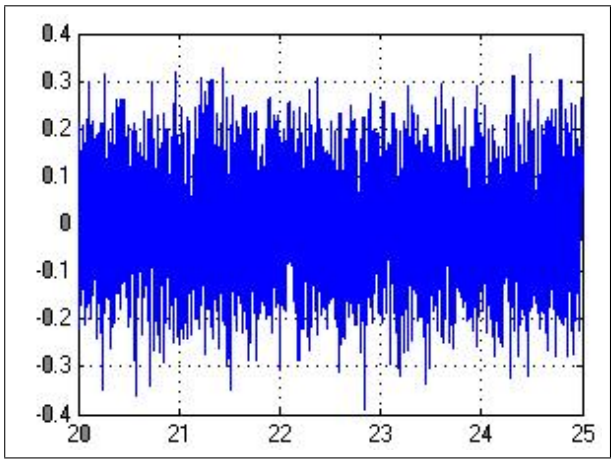

Figura 61: **Ruído (T = 0,001)**

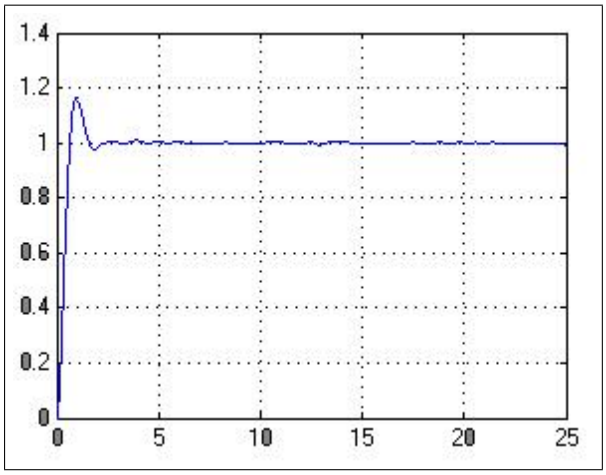

Figura 62: **Resposta (T = 0,01)**

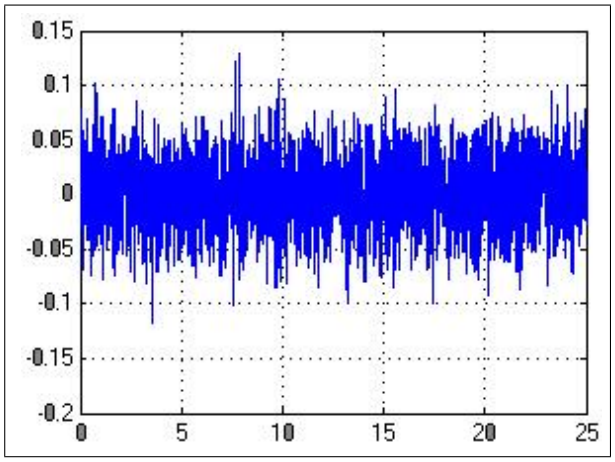

Figura 63: **Ruído (T = 0,01)**

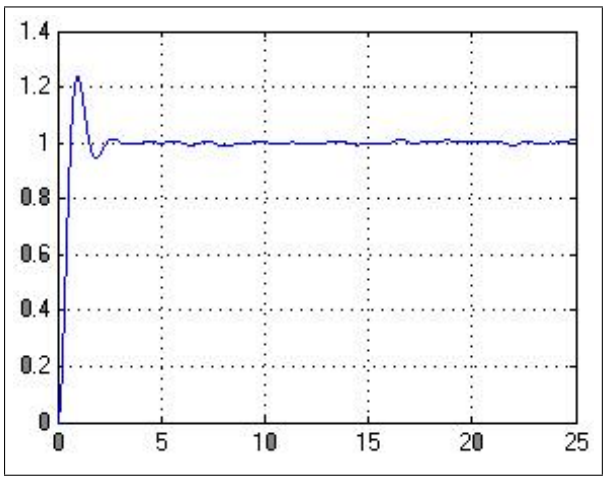

Figura 64: **Resposta (T = 0,1)**

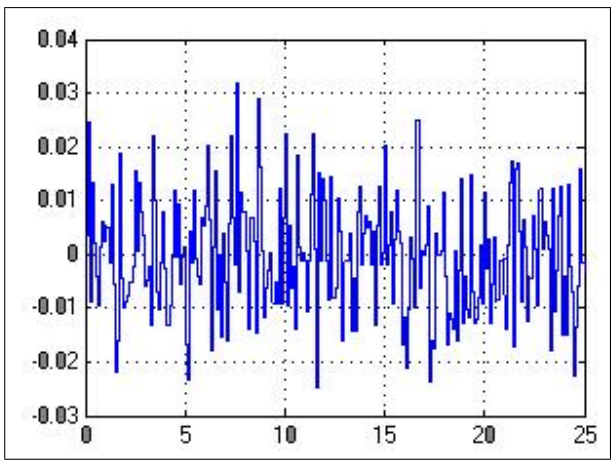

Figura 65: **Ruído (T = 0,1)**

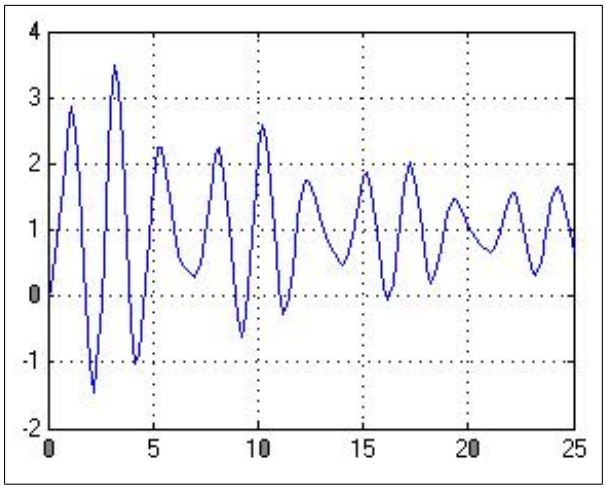

Figura 66: **Resposta (T = 1)**

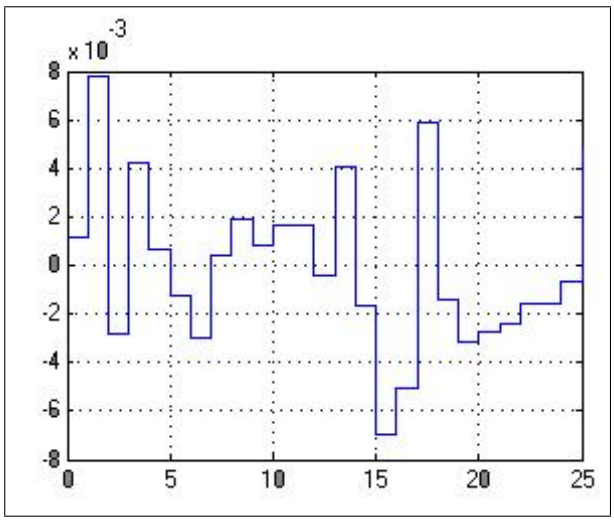

Figura 67: **Ruído (T = 1)**

8.9.3 Conclusões

- Observa-se que quanto maior o número de bits da palavra (menor o noise power) menor será a amplitude do ruído.
- Observa-se que para uma mesma palavra, ao aumentar o período, a amplitude do ruído diminui.
- Observa-se que para palavras de 4 bits, o sistema é instável.
- Para palavras de 16 bits, os gráficos para T = 0,01 e T = 0,1 apresentam uma ótima estabilizada.
- Conclui-se que o sistema será estável quão grande for a palavra.