

UNIVERSIDADE DE SÃO PAULO–USP
ESCOLA DE ENGENHARIA DE SÃO CARLOS
DEPARTAMENTO DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO
SEL0328 - LABORATÓRIO DE CONTROLE DE SISTEMAS

**Diego S. Carneiro, Rafael F. Q. Magossi,
Vilma A. Oliveira**

**Guia de identificação de parâmetros e função
de transferência de um motor CC**

Lista de ilustrações

Figura 2.1	Modelo do motor CC	5
Figura 4.1	Diagrama de blocos de um motor de corrente contínua	16
Figura 4.2	Diagrama do motor no Simulink	19
Figura 4.3	Tela do parameter estimation	20
Figura 4.4	Seleção de parâmetros	20
Figura 4.5	Configuração dos parâmetros	21
Figura 4.6	Definição do experimento	22
Figura 4.7	Leitura das variáveis do experimento	23
Figura 4.8	Resultados da estimação de parâmetros	23

Sumário

1	Introdução	3
2	Função de transferência do motor	5
3	Interface ident	7
3.1	Obtendo os dados a partir do arquivo .csv	7
3.1.1	Descobrimo o sample time	10
3.1.2	Obtendo o sinal de entrada	10
3.2	Utilizando o ident	11
3.3	Utilizando um objeto	14
4	Interface parameter estimation	16
4.1	Passo 1 - Extração dos dados e criação das variáveis	17
4.1.1	Importação das planilhas	17
4.1.2	Tratamento dos dados	18
4.2	Passo 2 - Definição dos parâmetros iniciais	18
4.3	Passo 3 - Elaboração do modelo em Simulink	19
4.4	Passo 4 - Estimação de parâmetros	20
4.4.1	Seleção de parâmetros	20
4.4.2	Seleção de entradas e saídas	21
4.4.3	O experimento	22
	Referências	24

Introdução

No contexto de identificação de sistemas, duas ferramentas importantes dentro dos pacotes disponíveis no software MATLAB, são o Toolbox de identificação de sistemas e o de estimação de parâmetros, utilizando diagramas de bloco do Simulink (disponível à partir da versão R2016a do MATLAB). Para a devida utilização de ambas, deve-se ter disponível arquivos com dados experimentais de entrada e saída de um sistema, a fim de obter o modelo correspondente.

O Toolbox de Identificação de Sistemas™, doravante denominado Ident, fornece funções MATLAB, blocos Simulink® , e um aplicativo para a construção de modelos matemáticos de sistemas dinâmicos a partir de dados de entrada e saída medidos. Ele permite que você criar e usar modelos de sistemas dinâmicos não são facilmente modelados a partir de dados a priori ou especificações. É possível usar os dados no domínio do tempo e no domínio da frequência de entrada-saída para identificar modelos de espaço de estados, funções de transferência de tempo discreto e/ou tempo contínuo, etc. A caixa de ferramentas também fornece algoritmos para estimativa de parâmetros online.

A caixa de ferramentas fornece técnicas de identificação, tais como a máxima verossimilhança, minimização de erros de previsão (PEM), e sistema de identificação de subespaço. Para representar a dinâmica do sistema não-linear, você pode estimar modelos Hammerstein-Wiener e modelos ARX não-lineares com a rede wavelet, partição árvore, e não-linearidades de rede sigmóide. A caixa de ferramentas executa sistema de identificação caixa-cinza para estimar parâmetros de um modelo definido pelo usuário. É possível usar o modelo identificado para a previsão de resposta do sistema e modelagem fábrica em Simulink. A caixa de ferramentas também suporta modelagem de dados de séries temporais e previsão de séries temporais.

A estimação de parâmetros (*Parameter Estimation*) é uma ferramenta do SIMULINK que permite estimar valores dos parâmetros de um sistema físico, conhecendo-se o modelo matemático. A simulação se dá por meio do diagrama de blocos, e um problema de otimização é resolvido utilizando dados de entrada e saída. No caso de um motor CC, podem ser estimados parâmetros como momento de inércia, coeficiente de atrito viscoso,

constante de torque, resistência de armadura e reatância de armadura. Neste manual, foi feito um processo de identificação de parâmetros e função de transferência de um motor CC, da marca Eletrocraft. São dadas as instruções de como se obter as funções de transferência de primeira e segunda ordem da saída em velocidade em relação a um degrau de tensão na entrada. Na obtenção dos parâmetros, foi considerada a presença do coeficiente de atrito viscoso, além da perda indutiva. Os dados experimentais de corrente, tensão e velocidade foram obtidos em laboratório e exportados por meio de um osciloscópio digital.

A obtenção da função de transferência teórica está descrita no Capítulo 2 , enquanto no Capítulo 3 pode se observar o procedimento de obter a função de transferência por meio dos dados de entrada e saída. No Capítulo 4 há a instrução de uso da ferramenta de estimação de parâmetros. Os algoritmos, dados e simulações utilizados como exemplo podem ser baixados de Carneiro e Oliveira (2021).

Função de transferência do motor

As equações dinâmicas que descrevem o funcionamento do motor CC são dadas pela Lei de Kirchoff das malhas e pela Segunda Lei de Newton. Os detalhes sobre o princípio de funcionamento do motor podem ser apreciados em (OLIVEIRA; AGUIAR; VARGAS, 2017). Denotando $\omega = \dot{\theta}$, tem-se:

$$\begin{aligned} v_a(t) &= K_e \omega(t) + R_a i_a(t) + L_a \frac{di_a(t)}{dt} \\ T_e &= K_t i_a(t) = B \omega(t) + J \frac{d\omega(t)}{dt} - F \end{aligned} \quad (2.1)$$

sendo ω a velocidade angular do motor, F uma força aplicada ao eixo do motor, $v_a(t)$, R_a e $i_a(t)$ a tensão, resistência e corrente de armadura, respectivamente, T_e o torque no eixo do motor, K_e a constante de tensão ou constante de força contra-eletromotriz (fcem), K_t a constante de torque, B o coeficiente de atrito viscoso e J o momento de inércia. O modelo esquemático do motor é esquematizado na Figura 2.1.

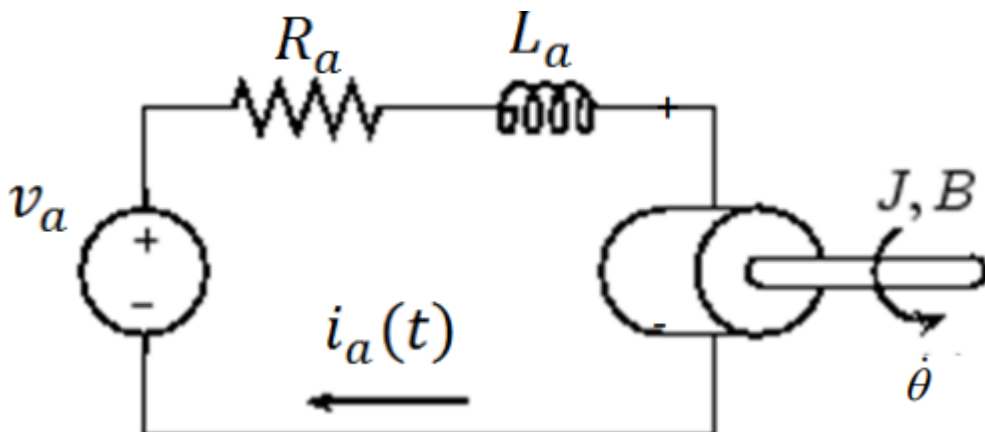


Figura 2.1: Modelo do motor CC

A função de transferência que a ser obtida é a relação entre a entrada em tensão e a saída em velocidade angular, ou seja $G(s) = \frac{\omega(s)}{V_a(s)}$. Admitindo-se $F = 0$, e aplicando-se a transformada de Laplace no sistema (2.1), o seguinte resultado é obtido:

$$V_a(s) = R_a I_a(s) + sL_a I_a(s) + K_e \omega(s) \quad (2.2)$$

$$T_e(s) = K_t I_a(s) = B\omega(s) + sJ\omega(s) \quad (2.3)$$

usando (2.3),

$$I_a = \frac{(B + sJ)\omega(s)}{K_t}$$

e substituindo (2.3) em (2.2), tem-se:

$$V_a(s) = (R_a + sL_a) \frac{(B + sJ)\omega(s)}{K_t} + K_e \omega(s)$$

fazendo-se as devidas manipulações algébricas, pode-se obter $G(s)$, de modo que:

$$G(s) = \frac{\omega(s)}{V_a(s)} = \frac{K_t}{JL_a s^2 + (JR_a + BL_a)s + (BR_a + K_e K_t)} \quad (2.4)$$

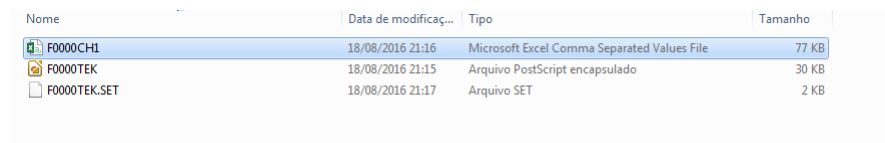
Interface ident

Neste capítulo é descrito o método de obtenção da função de transferência de um motor de corrente contínua por meio de dados de entrada e saída. O método apresentado de inserção de informações no Matlab é uma das maneiras possíveis de se trabalhar com dados em arquivo .csv. No Capítulo 4 é apresentada uma outra maneira.

As respostas obtidas neste capítulo são **ilustrativas** e não correspondem aos mesmos dados utilizados na etapa de estimação de parâmetros. Uma boa prática é repetir ambos procedimentos com o mesmo conjunto de dados e comparar as respostas.

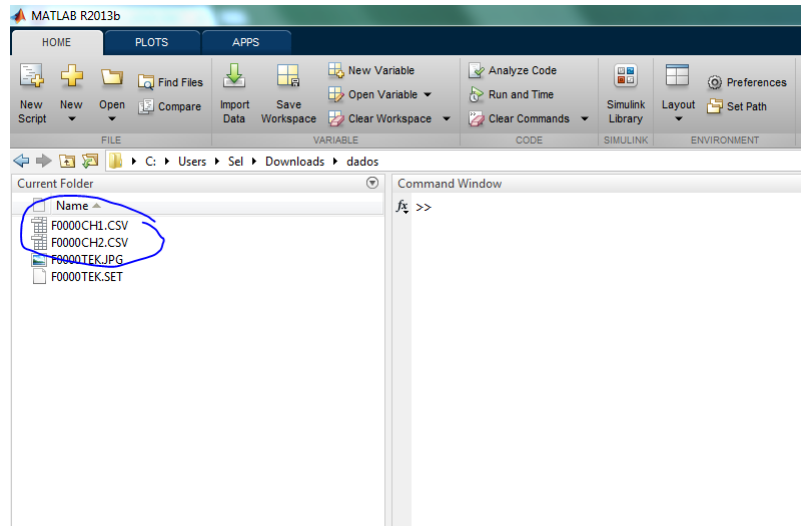
3.1 Obtendo os dados a partir do arquivo .csv

O osciloscópio Tektronix quando salva os arquivos em mídia móvel, usualmente salva: um arquivo .csv por canal (note a terminação CH1 significa canal 1), uma figura da tela do osciloscópio e um arquivo .set. O que nos interessa são os arquivos .csv. Os arquivos Comma-separated values, também conhecido como CSV, são arquivos de formato regulamentado pelo RFC 4180 que faz uma ordenação de bytes ou um formato de terminador de linha. Ele comumente usado em softwares como o Microsoft Excel.

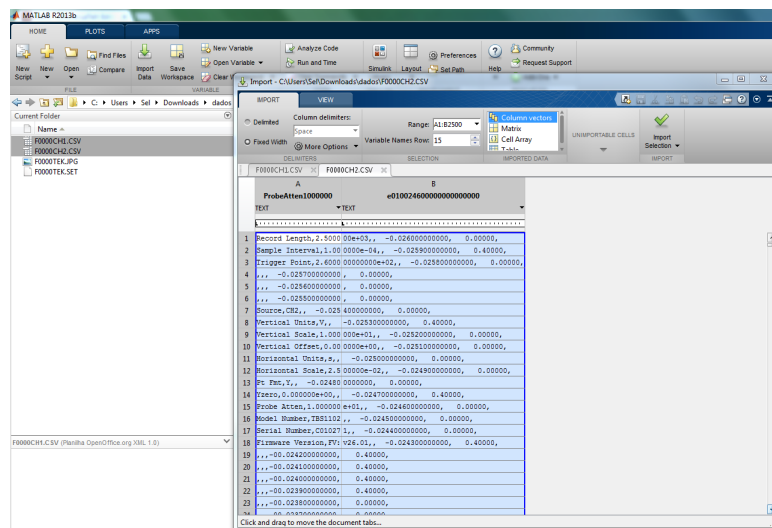


Nome	Data de modificaç...	Tipo	Tamanho
F0000CH1	18/08/2016 21:16	Microsoft Excel Comma Separated Values File	77 KB
F0000TEK	18/08/2016 21:15	Arquivo PostScript encapsulado	30 KB
F0000TEK.SET	18/08/2016 21:17	Arquivo SET	2 KB

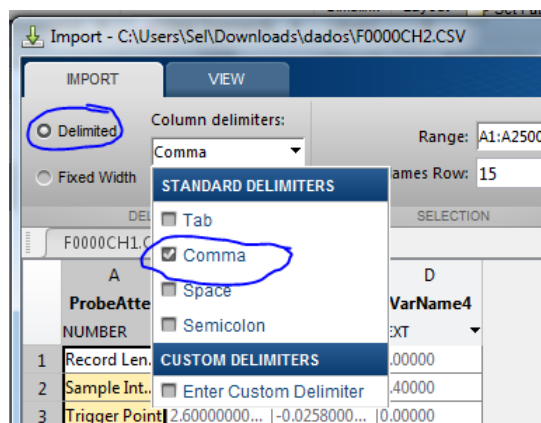
O Matlab conta com um GUI para importar arquivos .csv. Primeiramente, copie e cole seus arquivos para dentro da pasta de trabalho do Matlab.



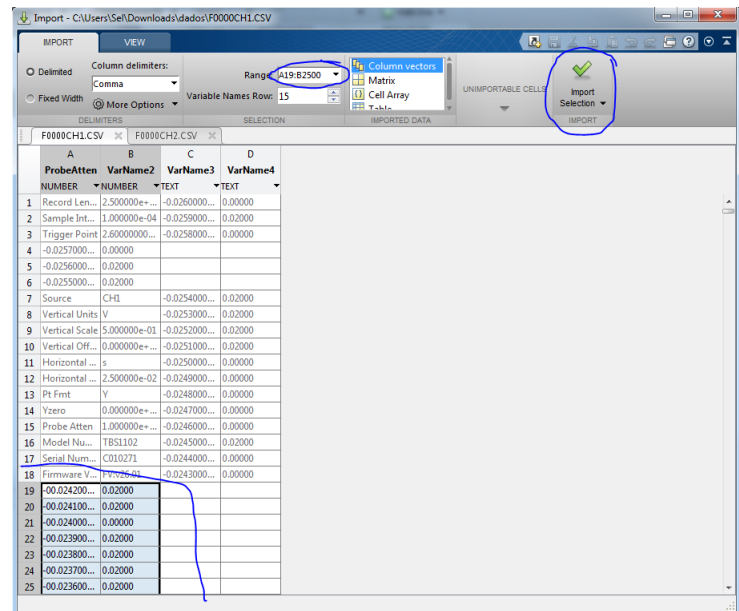
Para usá-lo, clique e arraste o arquivo .csv desejado para o workspace do Matlab. O GUI de importação será iniciado automaticamente.



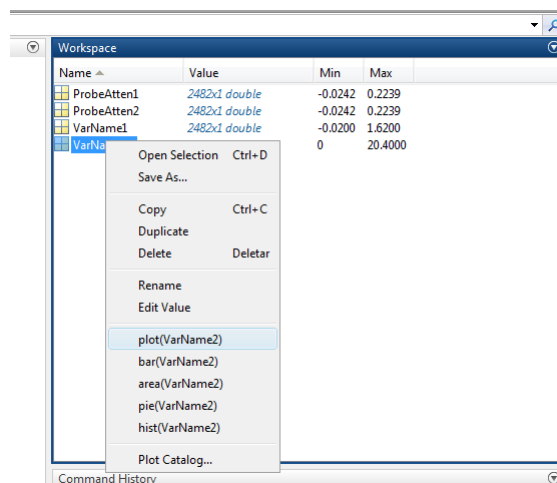
O osciloscópio gera os dados em forma de texto. Para realizar a importação correta, vamos delimitar o arquivo por vírgulas.



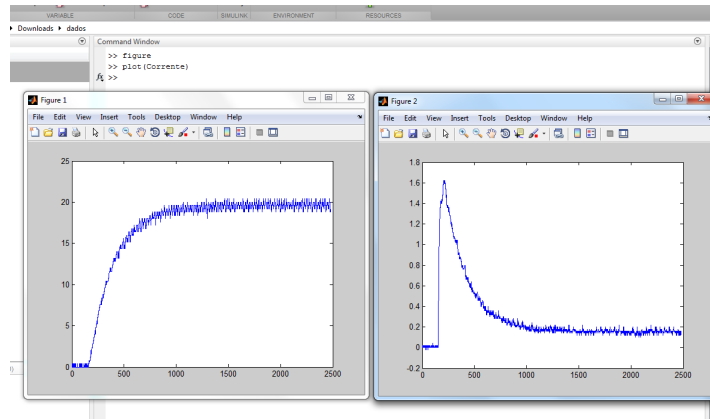
As colunas A e B representam o vetor de tempo e sinal medido. Encontre o início dos dados e selecione os dados dessas colunas até o fim (como no Excel). Caso seja necessário é possível mudar o nome da variável (VarName2 para Canal, por exemplo) clicando no nome abaixo da letra da coluna e renomeando como desejado. Quando finalizar clique em *ImportSelection*. Repita o mesmo procedimento para quantos arquivos .csv você possuir.



Note que os dados agora se encontram no seu *workspace* do Matlab. Se você não tem ideia de quem são as variáveis que estão ali, clique com o botão direito do mouse em cima da variável que deseja descobrir o que é e clique em plot.

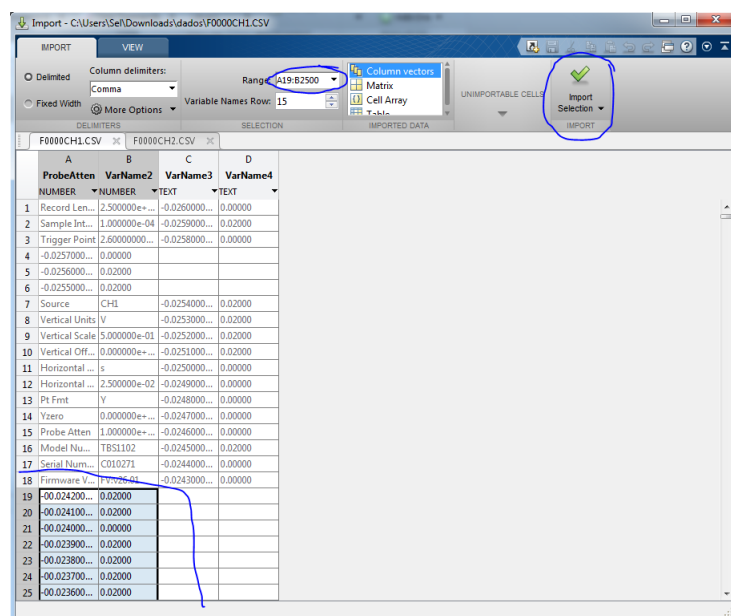


Faça isso para quantas variáveis forem necessárias. E verifique quem é o vetor de tempo e dados. Renomeie as variáveis após descobrir.



3.1.1 Descobindo o sample time

Os dados estão digitalizados, portanto, é necessário descobrir qual é o período de amostragem do seu sinal. Para isso, você pode utilizar o GUI de importação do Matlab. Na segunda linha encontra-se a informação sobre o sample time, 1e-4 segundos.



Esse valor também pode ser descoberto utilizando a diferença entre quaisquer dois pontos do seu vetor de tempo.

```
>> SampleTime = Tempo(1001)-Tempo(1000)

SampleTime =

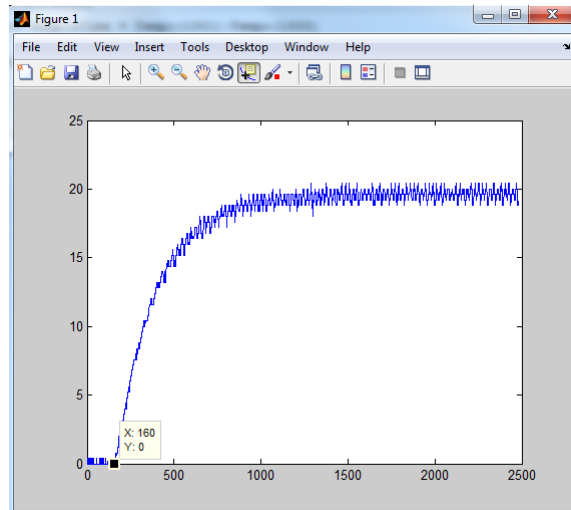
    1.0000e-04

>> |
```

3.1.2 Obtendo o sinal de entrada

O ident necessitará do vetor de entrada que gerou a resposta de saída. Caso não tenha salvo o vetor de entrada no osciloscópio, será necessário gerar um sinal de entrada

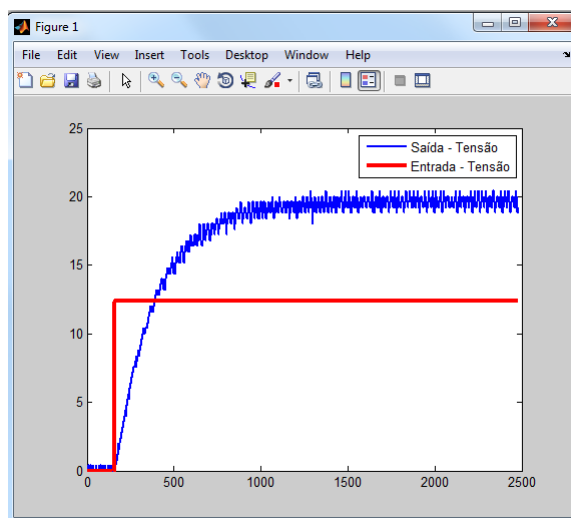
aproximado. Primeiramente, com o vetor de dados plotado, adicione um marcador de dados na imagem e tente identificar o início do degrau. Note que no eixo X não é tempo, mas sim posição do dado no vetor. No exemplo abaixo, o degrau se inicia próximo ao dado 160.



Uma vez descoberto isso, vamos gerar um sinal do tipo degrau, com amplitude igual a usada para colher os dados, utilizando a linha de comando abaixo.

```
>> Entrada = [zeros(160,1); 12.4*ones(length(Tensao)-160,1)];
```

Agora plote no mesmo gráfico para ver como ficou a sua entrada com a sua saída de dados.

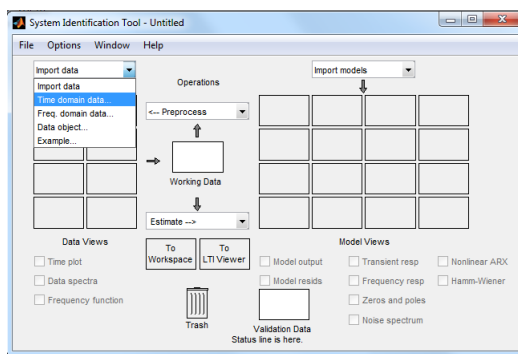


3.2 Utilizando o ident

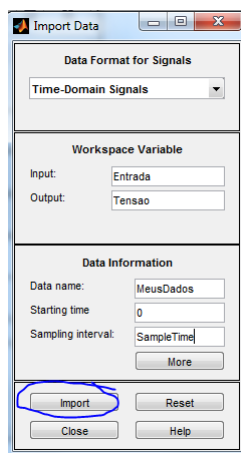
Para começar a utilizar o Ident, vamos primeiramente digitar na linha de comando do Matlab:

```
fx >> ident
```

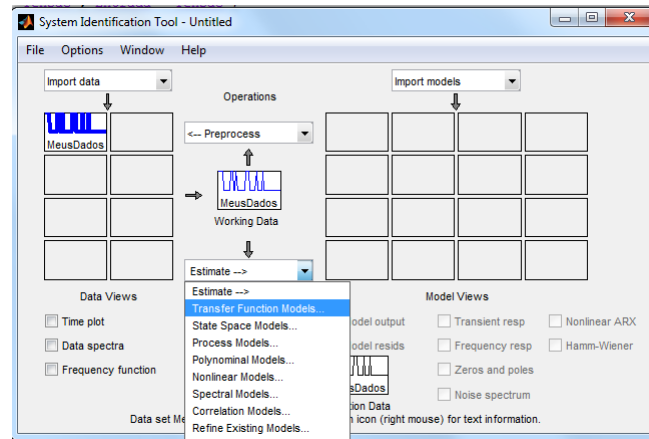
A seguinte tela aparecerá. Vamos importar os dados para trabalhar. Para isso vá em *import data* e depois em *Time domain data*, pois estamos com os dados todos no domínio do tempo.



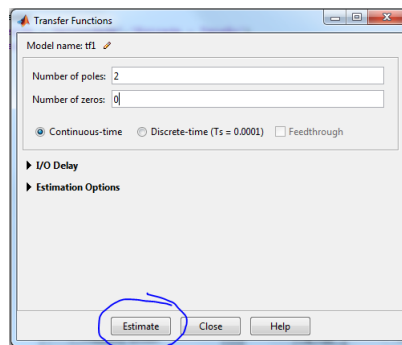
Uma nova janela para importação do dados aparecerá. Configure-a como mostrado. Em input será o vetor de entrada, obtido pelo osciloscópio ou criado como na seção anterior. Em output o vetor de dados de saída (nesse exemplo a tensão de saída do taco gerador de um motor CC). Se desejar troque o nome do seu conjunto de dados. O starting time é 0, pois nesse sistema não há delays. O sampling interval é o sample time que o seu osciloscópio forneceu. Clique em import para finalizar a importação dos dados.



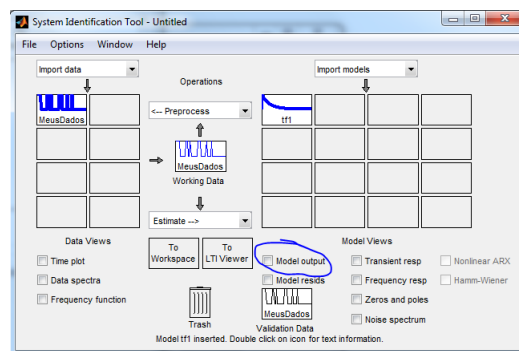
Agora em *Estimate* e clicar em *Transfer Function Models*. Outro modelo que pode ser usado é o *Process Models*.



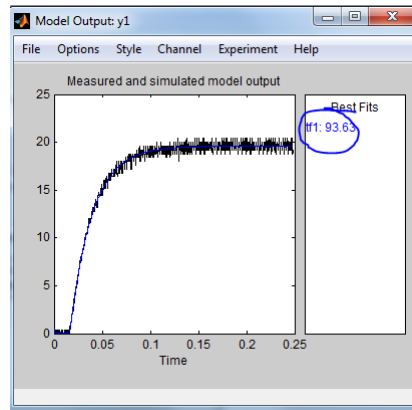
Agora configure quantos polos e quantos zeros você acredita que tenha sua planta. Na opção Continuous-time será estimada uma planta em s . Em Discrete-time será estimada uma planta em z . Nesse caso estamos interessados numa planta em s . Clique em *estimate*.



Depois de concluído a estimação, verifique quão boa foi resposta do modelo. Para isso clique em *model output*.



Obteve-se uma estimação com aproximadamente 94% de confiança. Se não estiver satisfeito, repita o processo para diferentes combinações de zeros e polos. Lembre que o aumento da ordem de um sistema desnecessariamente pode dificultar desnecessariamente o projeto de um controlador.



Para salvar o resultado clique e arraste *tf1* para *To Workspace*. Note que o resultado no seu *workspace* será um dado do tipo *idtf*.

```
>> tf1

tf1 =

    From input "u1" to output "y1":
    3.264e05
    -----
    s^2 + 4859 s + 2.066e05

Name: tf1
Continuous-time identified transfer function.

Parameterization:
Number of poles: 2   Number of zeros: 0
Number of free coefficients: 3
Use "tfdata", "getpvec", "getcov" for parameters and their uncertainties.

Status:
Estimated using TFEST on time domain data "MeusDados".
Fit to estimation data: 93.63% (simulation focus)
FFE: 0.1337, MSE: 0.1352
```

Para trabalhar somente como função de transferência, utilize o comando abaixo.

```
>> G = tf(tf1)

G =

    From input "u1" to output "y1":
    3.264e05
    -----+---
    s^2 + 4859 s + 2.066e05

Name: tf1
Continuous-time transfer function.
```

Salve e use de acordo com seu interesse. Para salvar use “save G”. Pode usar G como entrada do bloco LTI system no simulink.

3.3 Utilizando um objeto

Nesse método, parte-se do pressuposto que todos os dados de entrada e saída e as informações do osciloscópio estão disponíveis no workspace do Matlab. A partir disso, o seguinte algoritmo pode ser utilizado.

```
1 %ident
2 motor = iddata;
3 motor.Tstart = 0;
4 motor.Ts = Ts;
```

```
5 motor.InputData = Va;
6 motor.OutputData = Omega;
7
8 %tfest para fazer a estimacao da funcao de transferencia
9 tfmotor1pole = tfest(motor,1,0);
10 tfmotor2pole = tfest(motor,2,0);
11 figure
12 step(Vstep*tfmotor1pole);
13 hold
14 plot(time,Omega);
15 figure
16 step(Vstep*tfmotor2poles);
17 hold
18 plot(time,Omega);
```

Na rotina apresentada, é definido um objeto "motor" do tipo "iddata", que contém as informações necessárias para utilizar o `ident`. Define-se o tempo de amostragem, os dados de entrada e os dados de saída por meio dos atributos do objeto `motor`. Dessa maneira, é possível estimar a função de transferência sem utilizar a interface gráfica, deixando o processo mais automatizado. A função `tfest()` requer um objeto `iddata` e a quantidade de polos e zeros que a função de transferência de saída é desejável ter. Para um polo e dois polos, a função de transferência resultante é armazenada nas variáveis "tfmotor1pole" e "tfmotor2pole", respectivamente. A quantidade de zero é nula para ambas. A resposta em degrau pode ser comparada com os valores reais do motor.

Interface parameter estimation

Para utilizar a interface, deve se definir qual o modelo dinâmico a ser utilizado, definido-se os parâmetros do sistema a serem encontrados e as respectivas entradas e saídas. Para o motor CC, o diagrama de blocos do modelo dinâmico pode ser visto na Figura 4.1, os parâmetros são:

- J : Momento de Inércia do rotor [Kgm^2];
- B : Coeficiente de atrito viscoso [Nms^2/rad];
- K : Constante de torque [Nm/A];
- R_a : Resistência de armadura [Ω];
- L_a : Reatância de Armadura [Ω];

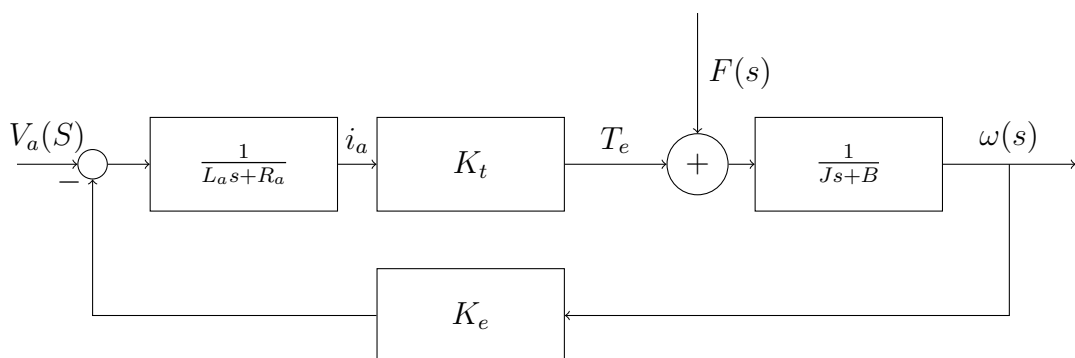


Figura 4.1: Diagrama de blocos de um motor de corrente contínua

Sendo V_a, i_a e ω a tensão de armadura, corrente de armadura e velocidade angular do motor, respectivamente. Esses valores são obtidos em experimento prático e devem ser inseridos no no *Parameter Estimation*.

Uma vez definidos o modelo, os parâmetros e em posse dos dados experimentais (preferencialmente em formato csv), os procedimentos para realizar a estimação devem ser

feitos como a seguir, podendo-se alterar algumas variáveis ou maneiras de carregar os dados conforme o tipo de problema.

4.1 Passo 1 - Extração dos dados e criação das variáveis

Nessa etapa, deve se carregar todos os dados experimentais necessários e armazená-los no Workspace do MATLAB. Os arquivos utilizados para estimação de parâmetros são "MotorLoadData.m", "MotorParameters.m" e "SimulacaoMotorCC_r2016a". Abra-os e execute o primeiro, referente à aquisição de dados. As etapas serão explicadas e os trechos de código mostrados a seguir.

4.1.1 Importação das planilhas

Primeiramente, é interessante se definir o caminho de arquivo onde estão as planilhas com os dados do osciloscópio. Uma outra maneira é colocar esses arquivos na mesma pasta do programa principal. Feito isso, o comando `readtable()` é o padrão utilizado para abrir planilhas. São criadas variáveis em forma de tabela do MATLAB, e o seu conteúdo não deve ser utilizado para operações matemáticas. Portanto, é utilizado o comando `table2array()`. Uma vez obtidas todas as informações necessárias, os dados devem ser tratados, uma vez que o osciloscópio fornece informações em tensão.

```
1 close all;
2 clc;
3 %%Aquisicao dos dados
4 filePath = 'G:\Drives compartilhados\Disciplina Fundamentos de
           Controle - SEL0417\Aula1\DadosMotor\'; %Colocar caminho dos
           arquivos .csv
5 ch1Table = readtable([filePath 'F0001CH1.csv']); %Corrente
6 ch2Table = readtable([filePath 'F0001CH2.csv']); %Velocidade
7 currentAtten = table2array(ch1Table(15,2));
8 velocityAtten = table2array(ch2Table(15,2));
9
10 Ts = table2array(ch1Table(2,2)); %Sample Time, presente no arquivo
           .csv importado do osciloscopio.
11
12 %Obtenção dos valores de corrente e velocidade (não tratados)
13 currentValue = table2array(ch1Table(1:end,5));
14 currentTime = table2array(ch1Table(1:end,4));
15 velocityValue = table2array(ch2Table(1:end,5));
16 velocityTime = currentTime;
```

4.1.2 Tratamento dos dados

Como os dados estão em formato de tensão, deve se utilizar a lei de Ohm ($V = R \cdot i$) para encontrar os valores de corrente. Para os valores de tensão, utiliza a relação do tacogerador utilizado, em que $V(t) = K_{tg}(t)\omega$. O valor da constante do gerador K_{tg} é obtido empiricamente e calculado por meio de uma regressão linear. Há ainda um fator de atenuação utilizado no osciloscópio. Esse valor também deve ser considerado no tratamento. Para a estimação de parâmetros deve ser inserida uma matriz com o vetor de tempo e o valor da variável desejada. O tempo amostral é fornecido pelo osciloscópio. Por fim, deve se fornece um vetor de entrada, que nesse caso corresponde a um degrau de tensão de 12.28 V.

```

1 %Parametros iniciais
2 Ktg = 0.1504; %Valor da constante que relaciona a tensão de
   saída do tacogerador e a velocidade do eixo.
3 Rext = 0.47; %Resistor utilizado para medir corrente
4 Vstep = 12.28; %Degrau de tensão aplicado na entrada.
5 current = currentAtten*currentValue/Rext; %A
6 velocity = velocityAtten*velocityValue/Ktg; %rad/s
7
8 %Tratando tamanho dos vetores para o degrau
9 Ia = current(193:end);
10 Omega = velocity(193:end);
11 Va = [zeros(0,1);Vstep*ones(length(Omega),1)];
12 %Adequa tempo às amostras
13 time = linspace(Ts,Ts*size(Omega,1),size(Omega,1));
14 time = time';
15 Wmed=[time Omega]; %Velocidade medida do degrau rad/s
16 Vamed=[time Va]; %Tensao de armadura Volts
17 Iamed=[time Ia]; %Corrente de armadura Volts

```

4.2 Passo 2 - Definição dos parâmetros iniciais

A estimação de parâmetros requer valores iniciais para poder executar o algoritmo de otimização. Assim, deve-se colocar alguns valores próximos aos reais de todas as variáveis. Essas informações podem ser obtidas pelo manual do fabricante ou por meio de alguns testes práticos. O trecho a seguir contém uma estimativa inicial e logo a seguir é inicializado o arquivo em Simulink. A função de transferência é obtida por G_m , e pode ser calculada antes da simulação a fim de comparação. O código a seguir é executado pelo arquivo "MotorParameters.m".

```

1 %Chute inicial dos parametros do motor

```

```

2 Ra = 0.1501; % Resistencia de armadura [Ohm]
3 La = 0.000150; % Indutancia de armadura [H]
4 K = 0.087; % Kt=Ke para este motor: Constante de torque [N-m/A]
   ou Constante f.c.e.m [V-s/rad]
5 J = 0.00050804; % Momento de inercia [N-m-s^2/rad]
6 B = 0.0006; % Coeficiente de atrito viscoso [N-m-s/rad]
7 Kt = K;
8 Ke = K;
9 s = tf('s');
10 Gm = @(Kt,Ke,Ra,La,J,B,s) Kt/(La*J*s^2+(Ra*J+La*B)*s+(Ra*B+Ke*Kt))
   ;
11 Gm0 = Gm(Kt,Ke,Ra,La,J,B,s);
12
13 %Execucao do modelo em Simulink
14 run('SimulacaoMotorCC_r2016a');

```

4.3 Passo 3 - Elaboração do modelo em Simulink

Na Figura 4.2, pode se ver a configuração do modelo contido no arquivo "Simulacao-MotorCC_r2016a.slx". Foi elaborado na versão *R2016a* e é possível executá-lo em versões superiores.

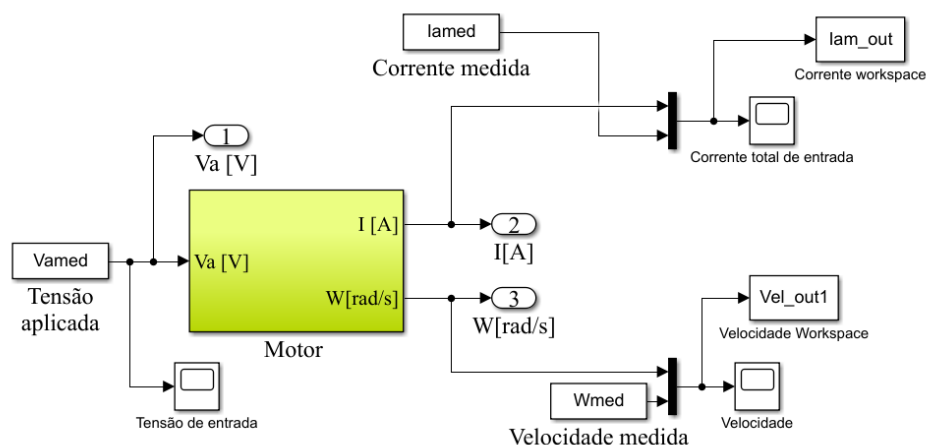


Figura 4.2: Diagrama do motor no Simulink

São inseridos blocos de conexão ao workspace para estabelecer a integração das informações dos arquivos de dados com o modelo do motor. Após a estimação de parâmetros, a corrente a velocidade medidos com os novos parâmetros podem ser exportadas e comparadas aos dados reais. O bloco motor (em verde), contém a dinâmica do motor, tal como no diagrama de blocos da Figura 4.1.

4.4 Passo 4 - Estimação de parâmetros

A partir da versão R2016a, há uma aplicação no ambiente interno do simulink denominada "Parameter Estimation". Para acessá-la, procure pela aba **Analysis** (ou **APPs** na versão R2020a) e vá em **Parameter Estimation**. A tela vista na Figura 4.3 será aberta.

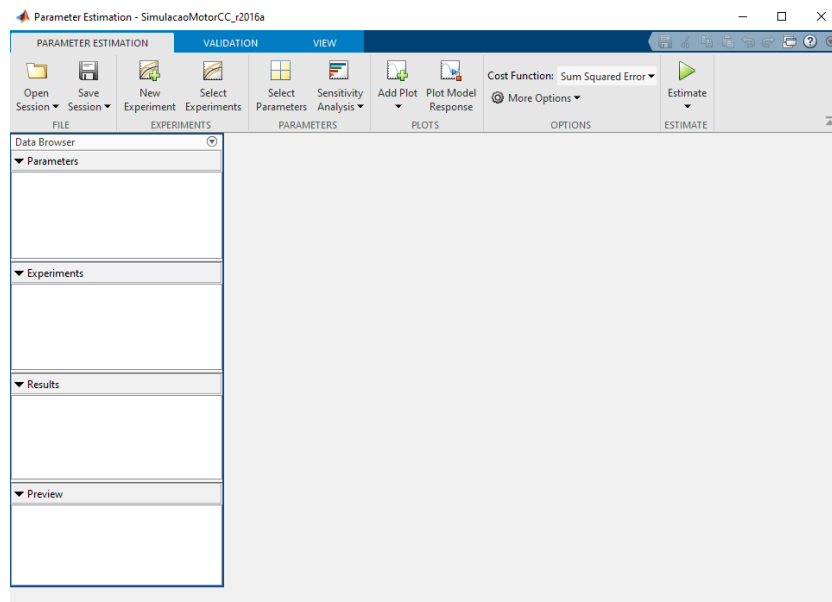


Figura 4.3: Tela do parameter estimation

4.4.1 Seleção de parâmetros

Na janela aberta, vá no ícone **Select Parameters**, e novamente em **Select Parameters**. Selecione os parâmetros do motor, tal como na Figura 4.4.

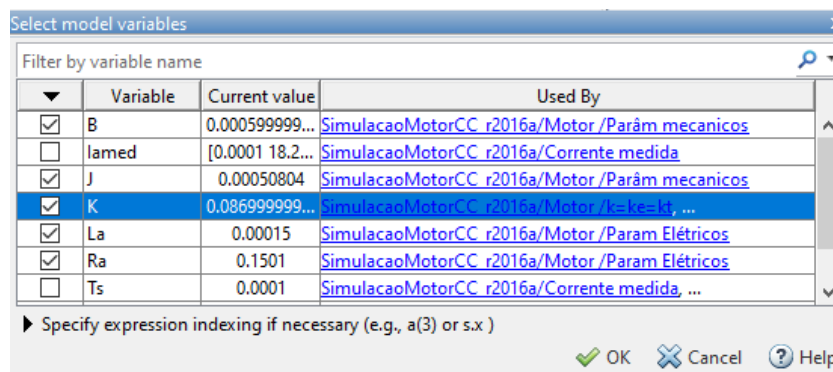


Figura 4.4: Seleção de parâmetros

Será aberta a janela vista na Figura 4.5. Abra as setas que estão à esquerda dos valores dos parâmetros iniciais e defina o limite mínimo de todos parâmetros como 0. Essa restrição é necessária pois não há sentido físico em parâmetros negativos no modelo do motor. Clique em "Ok".

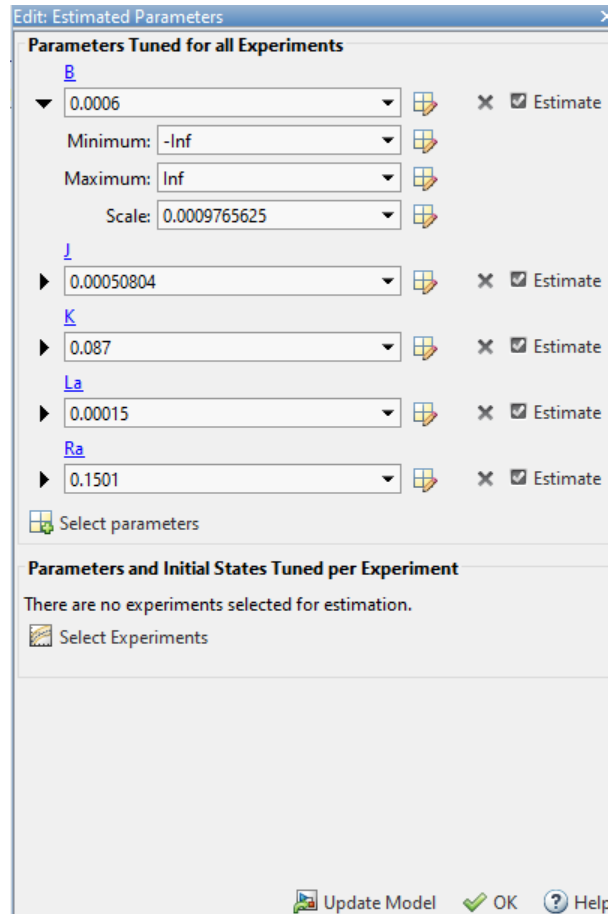


Figura 4.5: Configuração dos parâmetros

De acordo com a experiência de alguns usuários, em algumas versões é necessário primeiramente realizar o experimento com somente os parâmetros R_a e L_a . São os componentes da malha interna de corrente e, em alguns casos, é necessário estimá-los antes dos demais. Uma vez obtido tais valores, os valores restantes podem ser estimados com os valores atualizados de resistência e corrente de armadura. Os erros podem ocorrer pela forma como a otimização realizada trata os parâmetros iniciais. Na versão R2020a esse procedimento não é necessário, portanto podem ser estimados todos os parâmetros em uma única iteração.

4.4.2 Seleção de entradas e saídas

Nessa próxima etapa, será definido o experimento a ser realizado para a estimação. Clique no ícone **New Experiment** e será aberta uma nova janela (Figura 4.6).

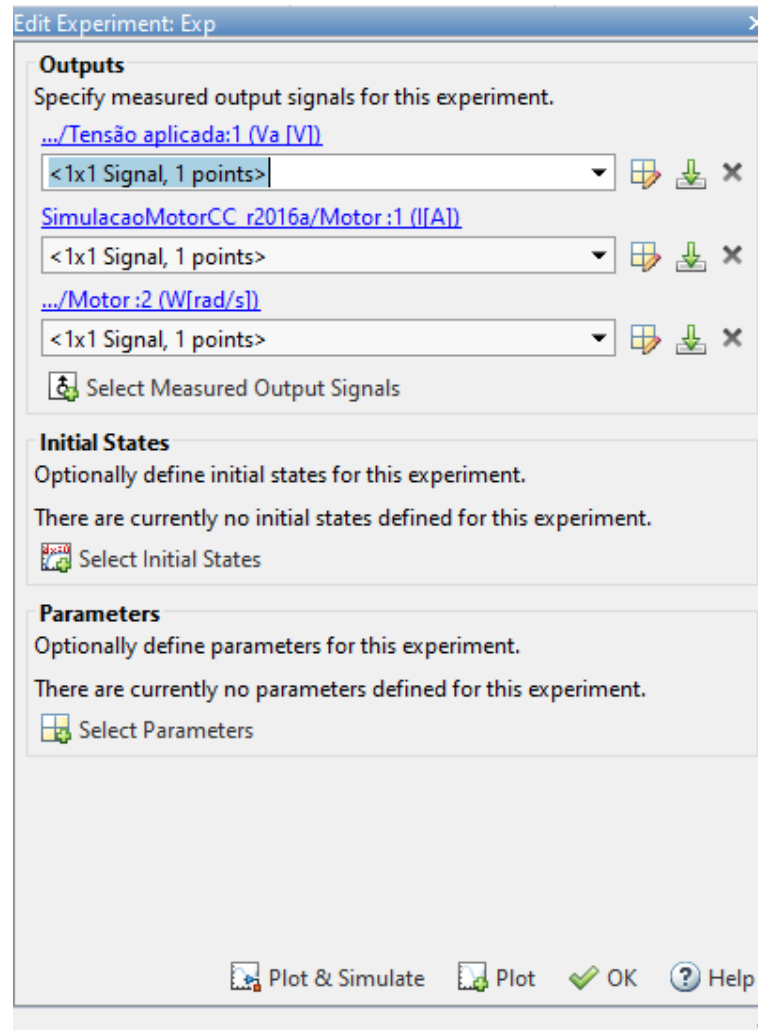


Figura 4.6: Definição do experimento

Nos campos vazios devem ser colocados os nomes das respectivas variáveis, ou seja, tensão de entrada, corrente de armadura e velocidade angular do motor. Essas informações devem ser inseridas em forma de vetores de tempo pelo valor da variável. As variáveis do workspace "Vamed", "Iamed", "Wmed" são inseridas nessa etapa. Clique em "OK".

4.4.3 O experimento

Para realizar a visualização da variação dos valores dos parâmetros e a resposta do sistema com os parâmetros otimizados, vá em **Add Plot** e, em seguida, em **Exp**, como mostra a Figura 4.7.

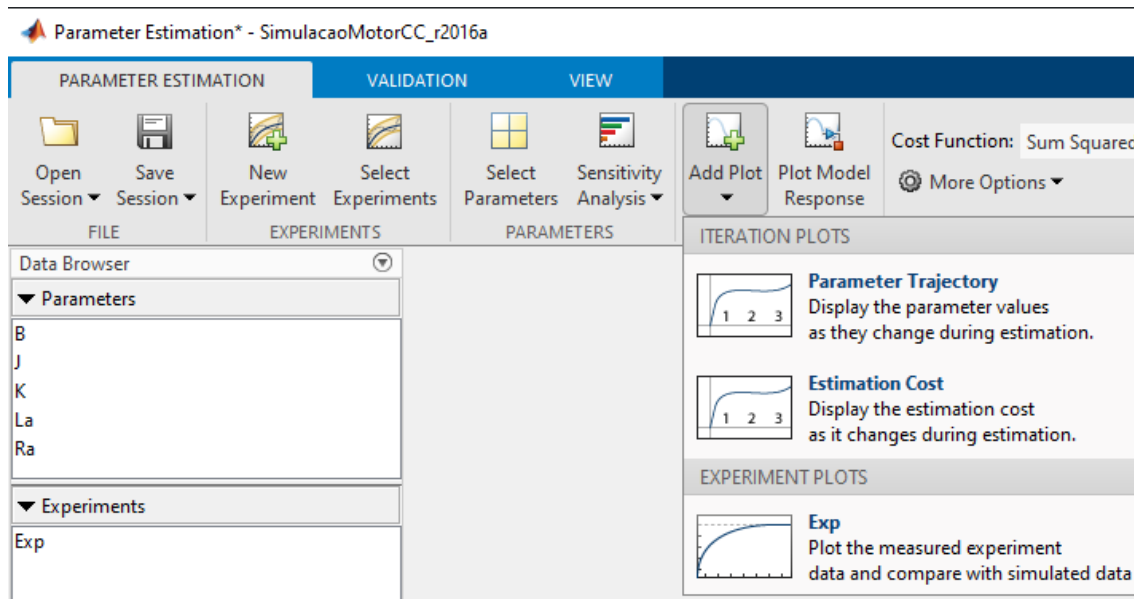


Figura 4.7: Leitura das variáveis do experimento

Clique agora no botão verde **Estimate**. Os parâmetros serão calculados e as respostas serão exibidas, tais como na Figura 4.8. As curvas em azul são os dados de entrada e as em vermelho correspondem às respostas obtidas simulando o modelo do motor com os parâmetros estimados. É esperado um comportamento muito próximo entre o modelo e os valores reais medidos.

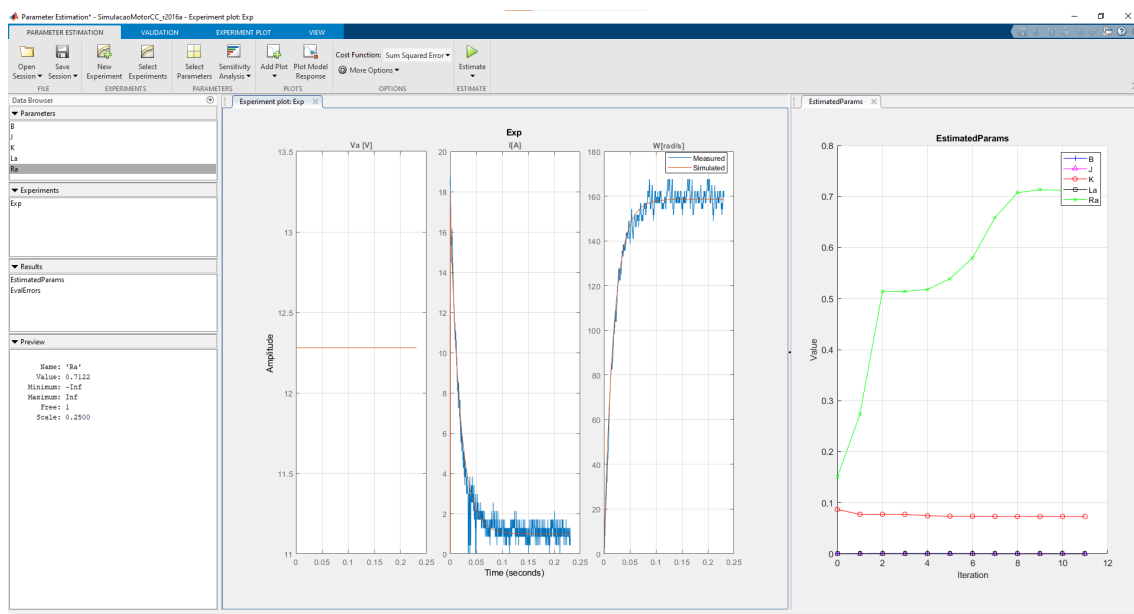


Figura 4.8: Resultados da estimação de parâmetros

Referências

CARNEIRO, D.; OLIVEIRA, V. **Simulações em MATLAB e SIMULINK de métodos de identificação de parâmetros e função de transferência de um motor CC**. 2021. Disponível em: <https://drive.google.com/drive/folders/1XeY_OxwXkYXilYavTgWVgGoVoBFkg7Hi?usp=sharing>.

OLIVEIRA, V.; AGUIAR, M.; VARGAS, J. **Engenharia de Controle: Fundamentos e Aulas de Laboratório**. [S.l.]: Elsevier Brasil, 2017.