

自动控制原理 II：线性系统分析与设计

课内实验

系统模型部分

概述：本部分实验主要内容

- MATLAB简单介绍
- 线性系统各类数学模型的表示
- 线性系统各种数学模型时间的相互转化
- 子系统的连接合并

概述：重点&预备知识

➤ 重点

- ❑ 线性系统各类数学模型的表示与相互转化
- ❑ 多个子系统的不同连接合并

➤ 预备知识

- ❑ MATLAB基本知识：界面、常用操作、等
- ❑ 线性系统理论知识：传递函数（阵）、状态空间表达式、坐标变换、等

课内实验：模型部分

1.1 MATLAB使用简介

1.2 线性系统数学模型表示

1.3 状态空间模型与其它模型的相互转化

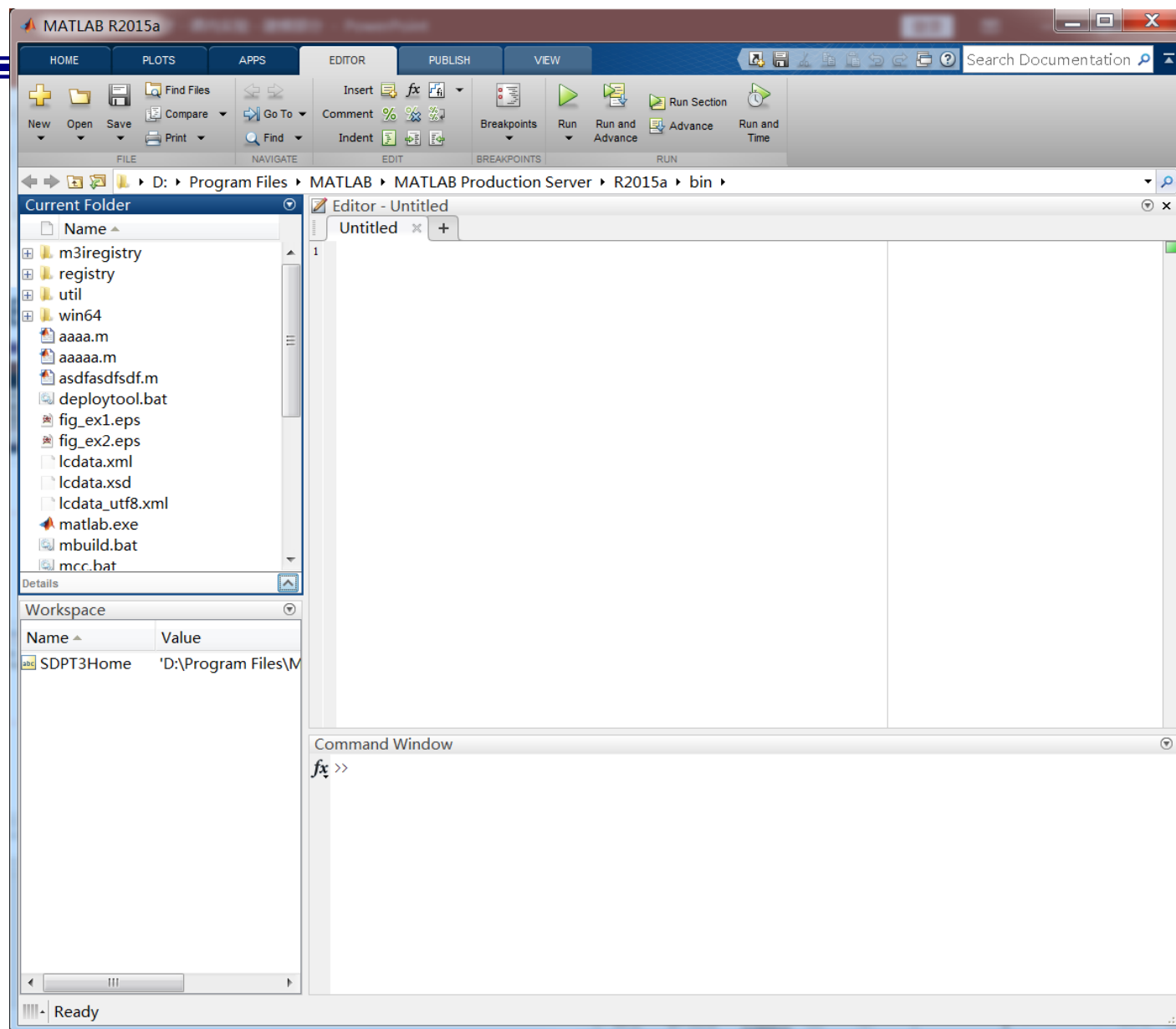
1.4 状态空间模型与状态空间模型的转化（坐标变换）

1.5 多个子系统不同连接下的整体系统模型

MATLAB使用简介

□ 各部分功能

- 在哪里写代码？
- 如何进行调试？
- 在哪里看结果？
- 相关文件保存在哪里？



MATLAB使用简介

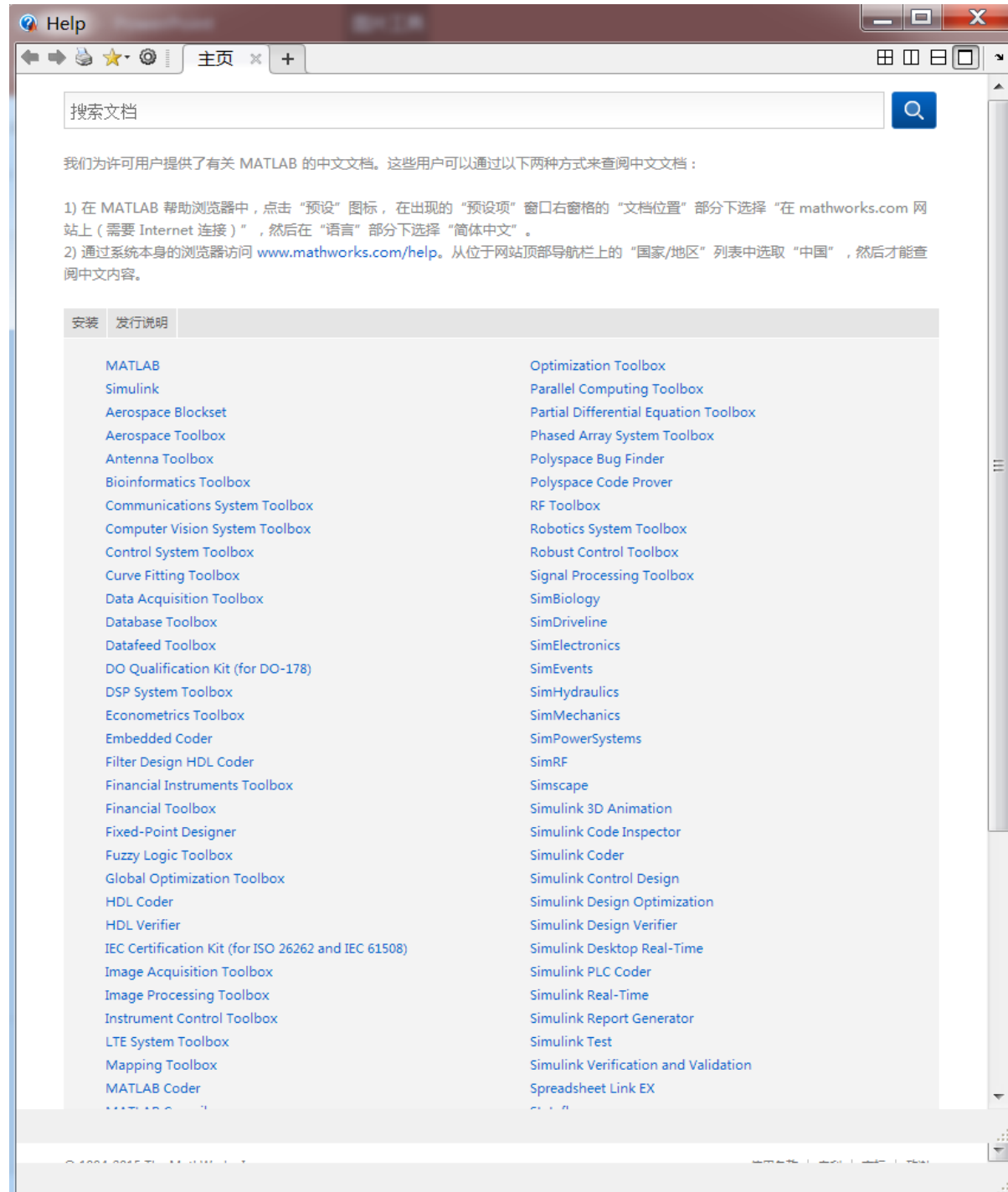
□ 学习工具



Google 搜索

手气不错

Google 提供：English 日本語



课内实验：模型部分

1.1 MATLAB使用简介

1.2 线性系统数学模型表示

1.3 状态空间模型与其它模型的相互转化

1.4 状态空间模型与状态空间模型的转化（坐标变换）

1.5 多个子系统不同连接下的整体系统模型



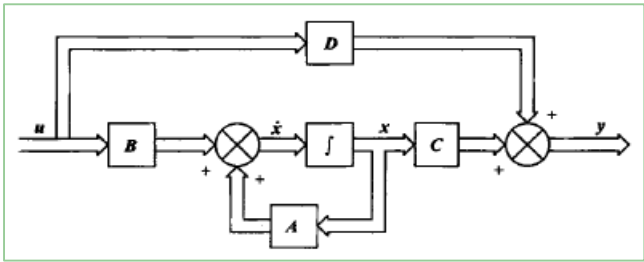
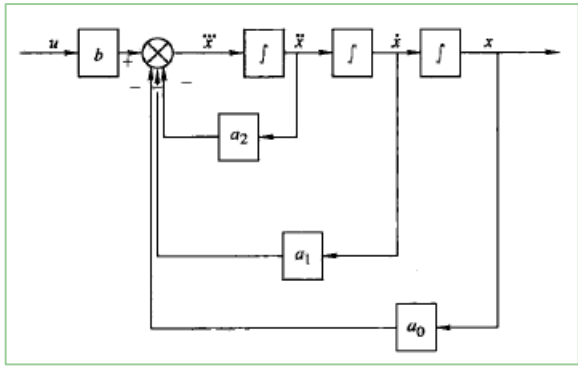
传递函数模型

$$W(s) = \frac{b_m s^m + b_{m-1} s^{m-1} + \dots + b_1 s + b_0}{s^n + a_{n-1} s^{n-1} + \dots + a_1 s + a_0}$$

$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \\ \mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t) \end{cases}$$

状态空间模型

结构图模型



线性系统数学模型表示

$$W(s) = \frac{b_m s^m + b_{m-1} s^{m-1} + \cdots + b_1 s + b_0}{s^n + a_{n-1} s^{n-1} + \cdots + a_1 s + a_0}$$

tf()函数

功能：建立系统的传递函数模型

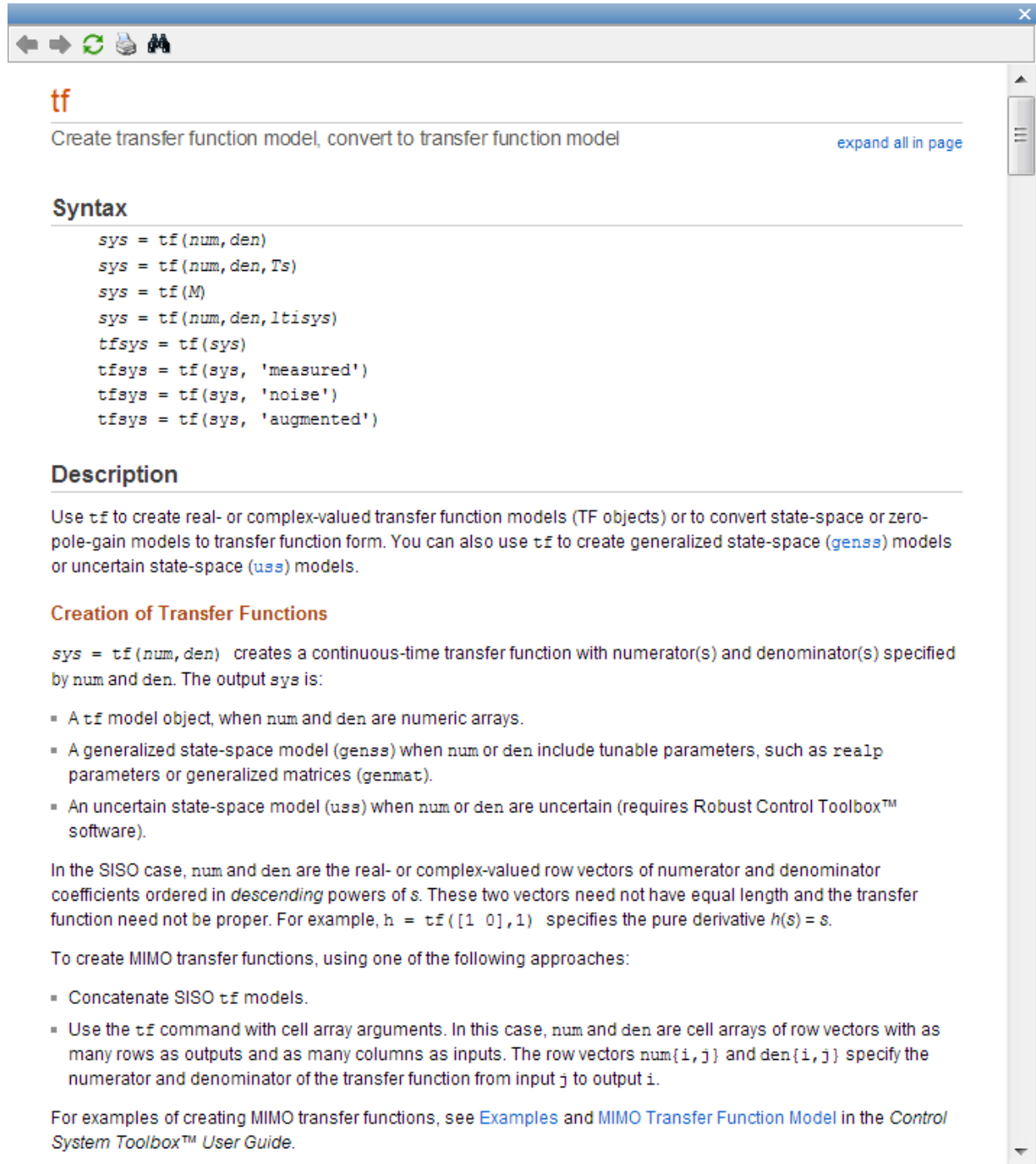
格式：

$$\text{sys} = \text{tf}(\text{num}, \text{den})$$

其中，

$$\text{num} = [b_m, b_{m-1}, \cdots, b_1, b_0]$$

$$\text{den} = [1, a_{n-1}, a_{n-2}, \cdots, a_1, a_0]$$



The screenshot shows the MATLAB documentation for the `tf` function. At the top, there's a navigation bar with icons for back, forward, search, and other functions. Below the navigation bar, the title `tf` is displayed in a large font. Underneath the title, a brief description reads: "Create transfer function model, convert to transfer function model". To the right of this description is a link that says "expand all in page". Below the description, there's a section titled "Syntax" which lists several ways to use the `tf` function: `sys = tf(num, den)`, `sys = tf(num, den, Ts)`, `sys = tf(M)`, `sys = tf(num, den, ltisys)`, `tfsys = tf(sys)`, `tfsys = tf(sys, 'measured')`, `tfsys = tf(sys, 'noise')`, and `tfsys = tf(sys, 'augmented')`. Below the syntax section is a "Description" section which explains that `tf` is used to create real- or complex-valued transfer function models (TF objects) or to convert state-space or zero-pole-gain models to transfer function form. It also mentions that `tf` can be used to create generalized state-space (`genss`) models or uncertain state-space (`uss`) models. Below the description is a section titled "Creation of Transfer Functions" which provides a detailed explanation of the `sys = tf(num, den)` command, stating that it creates a continuous-time transfer function with numerator(s) and denominator(s) specified by `num` and `den`. The output `sys` is then listed as:

- A `tf` model object, when `num` and `den` are numeric arrays.
- A generalized state-space model (`genss`) when `num` or `den` include tunable parameters, such as `realp` parameters or generalized matrices (`genmat`).
- An uncertain state-space model (`uss`) when `num` or `den` are uncertain (requires Robust Control Toolbox™ software).

Below this list, it explains that in the SISO case, `num` and `den` are the real- or complex-valued row vectors of numerator and denominator coefficients ordered in descending powers of `s`. It also provides an example: `h = tf([1 0], 1)` specifies the pure derivative $h(s) = s$. Finally, it mentions that to create MIMO transfer functions, one of the following approaches can be used:

- Concatenate SISO `tf` models.
- Use the `tf` command with cell array arguments. In this case, `num` and `den` are cell arrays of row vectors with as many rows as outputs and as many columns as inputs. The row vectors `num{i, j}` and `den{i, j}` specify the numerator and denominator of the transfer function from input `j` to output `i`.

At the bottom, it provides a reference to the MATLAB documentation for creating MIMO transfer functions, pointing to the "Examples" and "MIMO Transfer Function Model" sections in the "Control System Toolbox™ User Guide".

线性系统数学模型表示

例1-1：试用Matlab描述如下系统模型

$$W(s) = \frac{s^2 + 3s + 1}{s^3 + 2s^2 + 4s + 6}$$

```
ex1_1.m x +
1      %%%%%%%%%%
2      %%% Chuan-Ke Zhang
3      %%% 2021-10-06
4      %%% Example 1-1
5      %%% tf function
6      %%%%%%%%%%
7
8 -    clc
9 -    clear
10
11 -    num = [1 3 1];    % 分子多项式系数
12 -    den = [1 2 4 6]; % 分母多项式系数
13 -    G1 = tf(num, den)
14 -    G2 = tf([1 3 1], [1 2 4 6])
15
```

Command Window

G1 =

$$\frac{s^2 + 3s + 1}{s^3 + 2s^2 + 4s + 6}$$

Continuous-time transfer function.

G2 =

$$\frac{s^2 + 3s + 1}{s^3 + 2s^2 + 4s + 6}$$

Continuous-time transfer function.

fx >>

线性系统数学模型表示

$$W(s) = k \frac{(s - z_1)(s - z_2) \cdots (s - z_m)}{(s - p_1)(s - p_2) \cdots (s - p_n)}$$

zpk()函数

功能：建立系统的传递函数模型

格式：

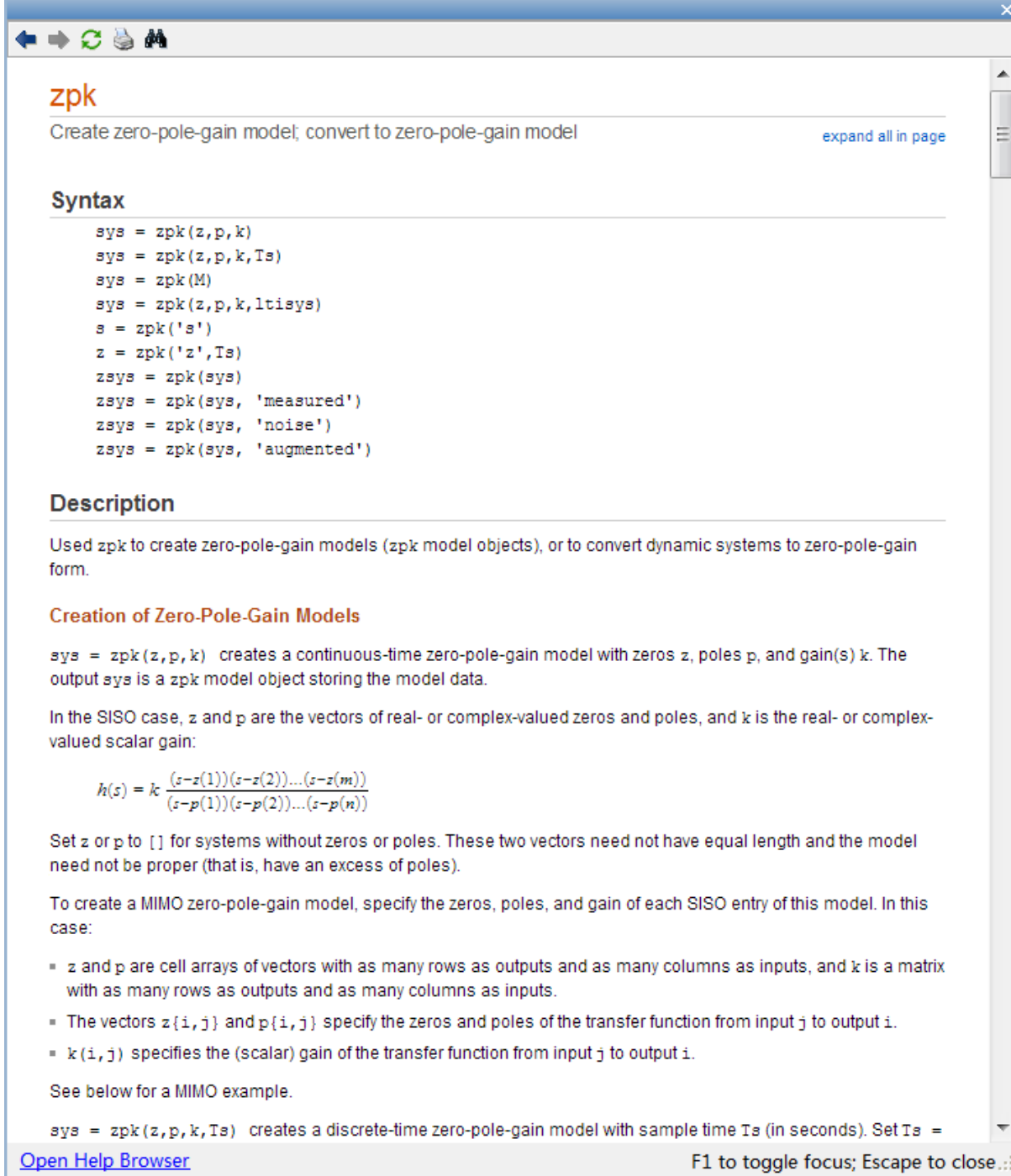
$$\text{sys} = \text{zpk}(z, p, k)$$

其中，

$$z = [z_1, z_2, \cdots, z_m]$$

$$p = [p_1, p_2, \cdots, p_n]$$

$$k = k$$



The screenshot shows the MATLAB documentation for the `zpk` function. At the top, it says "zpk" and "Create zero-pole-gain model; convert to zero-pole-gain model" with a link to "expand all in page". Below this is the "Syntax" section, which lists several ways to use the function: `sys = zpk(z,p,k)`, `sys = zpk(z,p,k,Ts)`, `sys = zpk(M)`, `sys = zpk(z,p,k,ltisys)`, `s = zpk('s')`, `z = zpk('z',Ts)`, `zsys = zpk(sys)`, `zsys = zpk(sys, 'measured')`, `zsys = zpk(sys, 'noise')`, and `zsys = zpk(sys, 'augmented')`. The "Description" section explains that `zpk` is used to create zero-pole-gain models or convert dynamic systems to zero-pole-gain form. It then details the "Creation of Zero-Pole-Gain Models" with the command `sys = zpk(z,p,k)` and describes the inputs: `z` and `p` are vectors of real- or complex-valued zeros and poles, and `k` is a real- or complex-valued scalar gain. A mathematical formula for $h(s)$ is provided: $h(s) = k \frac{(s-z(1))(s-z(2)) \cdots (s-z(m))}{(s-p(1))(s-p(2)) \cdots (s-p(n))}$. It also notes that `z` and `p` can be cell arrays for MIMO systems. A list of three bullet points further clarifies the inputs:

- `z` and `p` are cell arrays of vectors with as many rows as outputs and as many columns as inputs, and `k` is a matrix with as many rows as outputs and as many columns as inputs.
- The vectors `z{i,j}` and `p{i,j}` specify the zeros and poles of the transfer function from input `j` to output `i`.
- `k(i,j)` specifies the (scalar) gain of the transfer function from input `j` to output `i`.

 It then says "See below for a MIMO example." and provides the command `sys = zpk(z,p,k,Ts)` for discrete-time models. At the bottom, there is a link to "Open Help Browser" and a note "F1 to toggle focus; Escape to close..".

线性系统数学模型表示

例1-2：试用Matlab描述如下系统模型

$$W(s) = \frac{7(s-3)}{(s-1)(s+2)(s+4)}$$

```
Editor - C:\Users\hp\Desktop\ex1_2.m*
ex1_1.m x ex1_2.m* +
1 %%%%%%%%%%%%%%
2 %% Chuan-Ke Zhang
3 %% 2021-10-06
4 %% Example 1-2
5 %% zpk function
6 %%%%%%%%%%%%%%
7
8 - clc
9 - clear
10
11 - z = [3]; % 零点
12 - p = [1, -2, -4]; % 极点
13 - k = 7; % 增益
14 - G1 = zpk(z, p, k)
15 - G2 = zpk([3], [1, -2, -4], 7)
```

```
Command Window

G1 =

      7 (s-3)
-----
(s-1) (s+2) (s+4)

Continuous-time zero/pole/gain model.

G2 =

      7 (s-3)
-----
(s-1) (s+2) (s+4)

Continuous-time zero/pole/gain model.

fx >>
```

线性系统数学模型表示

$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \\ \mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t) \end{cases}$$

ss()函数

功能：建立系统的状态空间模型

格式：

$$\text{sys} = \text{ss}(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D})$$

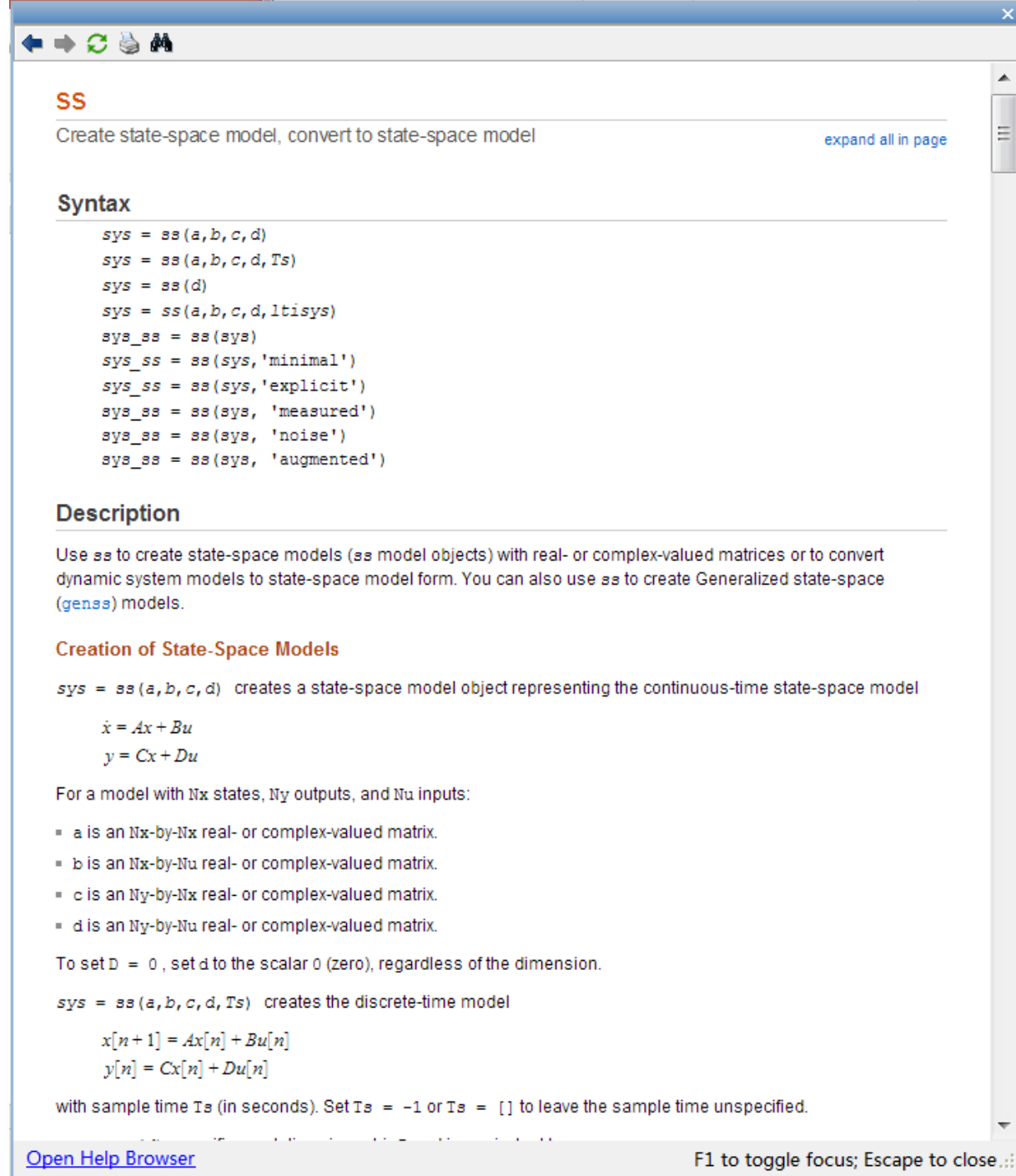
其中，

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} & a_{21} & a_{22} & \cdots & a_{2n} & \cdots & a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1n} & b_{21} & b_{22} & \cdots & b_{2n} & \cdots & b_{n1} & b_{n2} & \cdots & b_{nr} \end{bmatrix}$$

$$\mathbf{C} = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} & c_{21} & c_{22} & \cdots & c_{2n} & \cdots & c_{m1} & c_{m2} & \cdots & c_{mn} \end{bmatrix}$$

$$\mathbf{D} = \begin{bmatrix} d_{11} & d_{12} & \cdots & d_{1r} & d_{21} & d_{22} & \cdots & d_{2r} & \cdots & d_{m1} & d_{m2} & \cdots & d_{mr} \end{bmatrix}$$



The screenshot shows the MATLAB documentation for the `ss` function. At the top, it says "SS" and "Create state-space model, convert to state-space model" with a link to "expand all in page". Below this is the "Syntax" section, which lists various ways to call the function: `sys = ss(a,b,c,d)`, `sys = ss(a,b,c,d,Ts)`, `sys = ss(d)`, `sys = ss(a,b,c,d,ltisys)`, `sys_ss = ss(sys)`, `sys_ss = ss(sys,'minimal')`, `sys_ss = ss(sys,'explicit')`, `sys_ss = ss(sys, 'measured')`, `sys_ss = ss(sys, 'noise')`, and `sys_ss = ss(sys, 'augmented')`. The "Description" section explains that `ss` is used to create state-space models or convert dynamic system models to state-space form. It also mentions that `ss` can be used to create Generalized state-space (`genss`) models. The "Creation of State-Space Models" section provides the syntax `sys = ss(a,b,c,d)` for creating a continuous-time model and the equations $\dot{x} = Ax + Bu$ and $y = Cx + Du$. It also mentions that for a model with N_x states, N_y outputs, and N_u inputs, `a` is an N_x -by- N_x matrix, `b` is an N_x -by- N_u matrix, `c` is an N_y -by- N_x matrix, and `d` is an N_y -by- N_u matrix. It also mentions that to set `D = 0`, set `d` to the scalar 0 (zero), regardless of the dimension. The "Discrete-time model" section provides the syntax `sys = ss(a,b,c,d,Ts)` and the equations $x[n+1] = Ax[n] + Bu[n]$ and $y[n] = Cx[n] + Du[n]$. It also mentions that `Ts` is the sample time in seconds, and that `Ts = -1` or `Ts = []` leaves the sample time unspecified. At the bottom, there is a link to "Open Help Browser" and a note "F1 to toggle focus; Escape to close..".

线性系统数学模型表示

例1-3：试用Matlab描述如下系统模型

$$\begin{cases} \dot{x} = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix} x + \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} u \\ y = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 2 \end{bmatrix} x + \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} u \end{cases}$$

```
Editor - C:\Users\hp\Desktop\ex1_3.m
ex1_1.m x ex1_2.m x ex1_3.m x +
1 %%%%%%%%%%%%%%
2 %%% Chuan-Ke Zhang
3 %%% 2021-10-06
4 %%% Example 1-3
5 %%% state-space equation
6 %%%%%%%%%%%%%%
7
8 - clc
9 - clear
10
11 - A = [0 1; -2 -3];
12 - B = [1 0; 1 1];
13 - C = [1 0; 1 1; 0 2];
14 - D = [0 0; 1 0; 0 1];
15
16 - G1 = ss(A,B,C,D)
17 - G2 = ss([0 1; -2 -3], [1 0; 1 1], [1 0; 1 1; 0 2], [0 0; 1 0; 0 1])
```

```
Command Window

G1 =

a =
      x1  x2
x1    0    1
x2   -2   -3

b =
      u1  u2
x1     1    0
x2     1    1

c =
      x1  x2
y1     1    0
y2     1    1
y3     0    2

d =
      u1  u2
y1     0    0
y2     1    0
y3     0    1

Continuous-time state-space model.
```

```
Command Window

G2 =

a =
      x1  x2
x1     0    1
x2    -2   -3

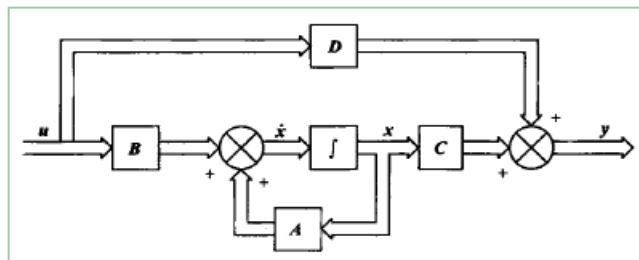
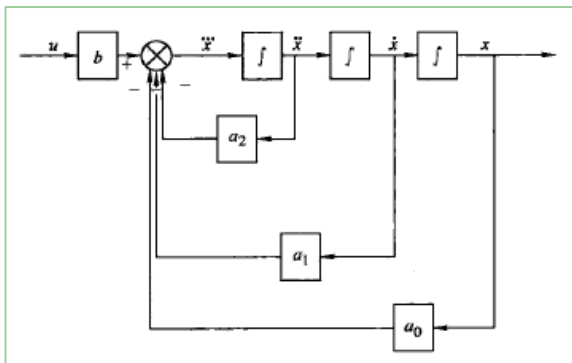
b =
      u1  u2
x1     1    0
x2     1    1

c =
      x1  x2
y1     1    0
y2     1    1
y3     0    2

d =
      u1  u2
y1     0    0
y2     1    0
y3     0    1

Continuous-time state-space model.
```

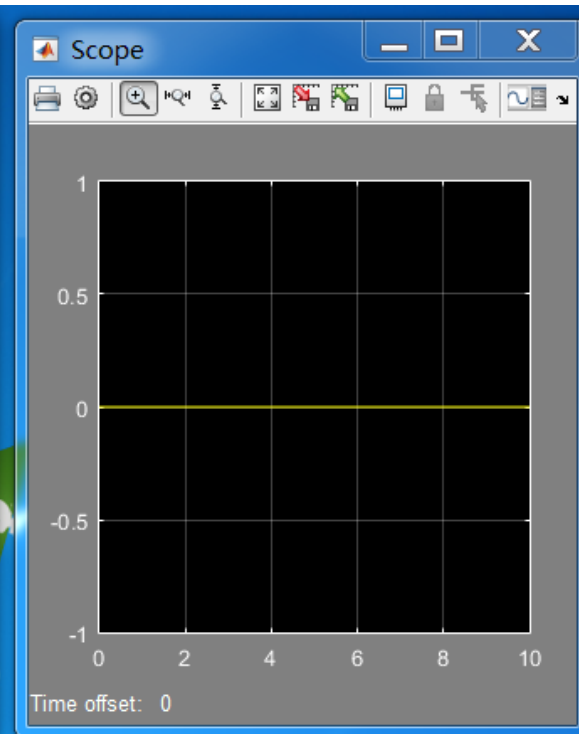
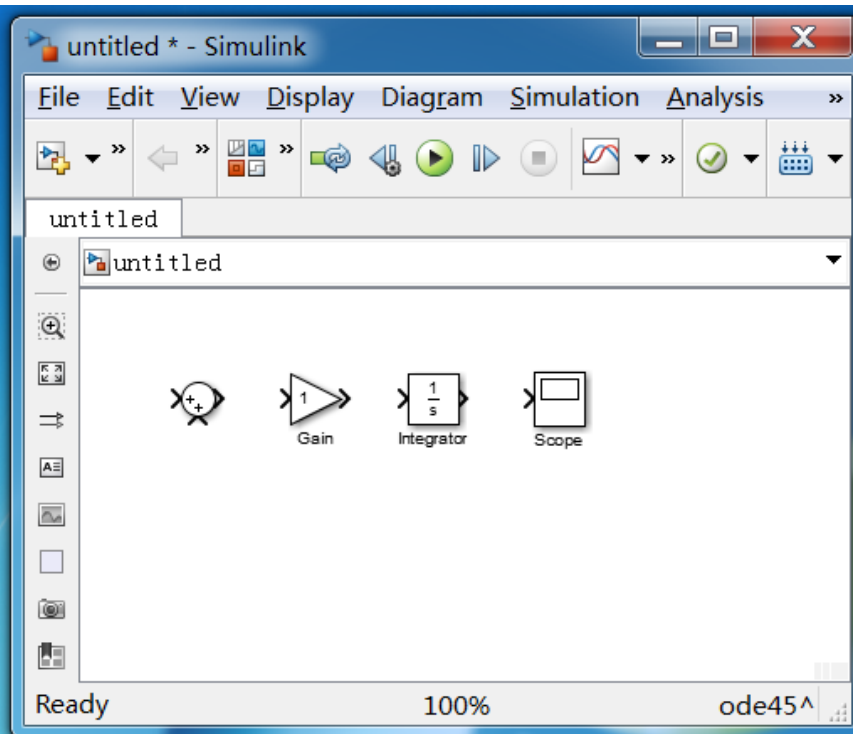
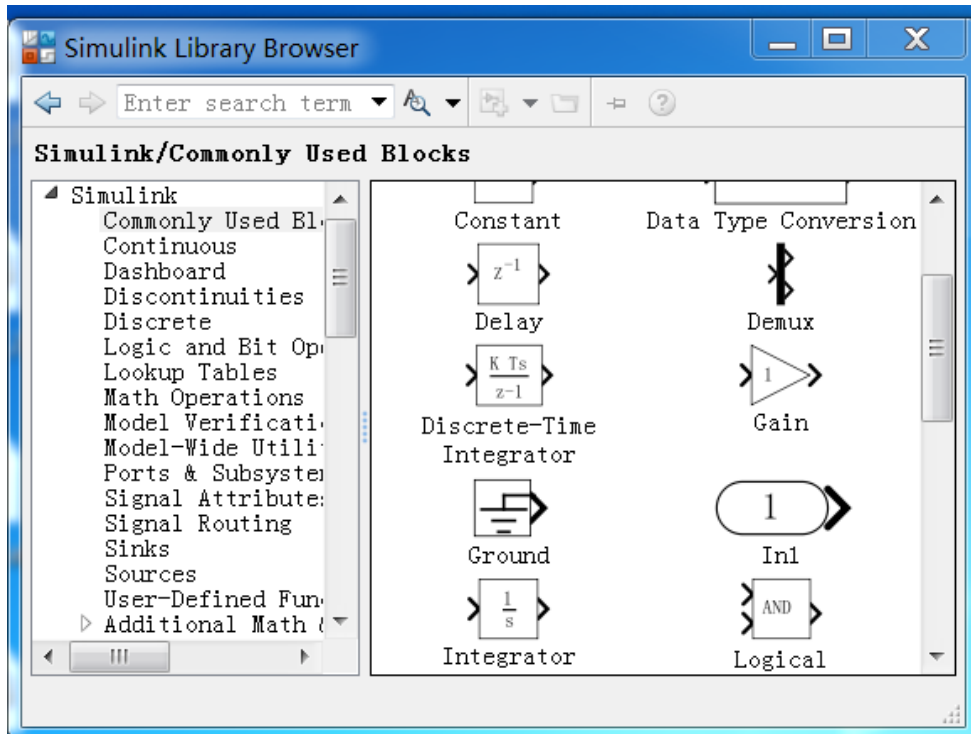
线性系统数学模型表示



Simulink 模块

功能:

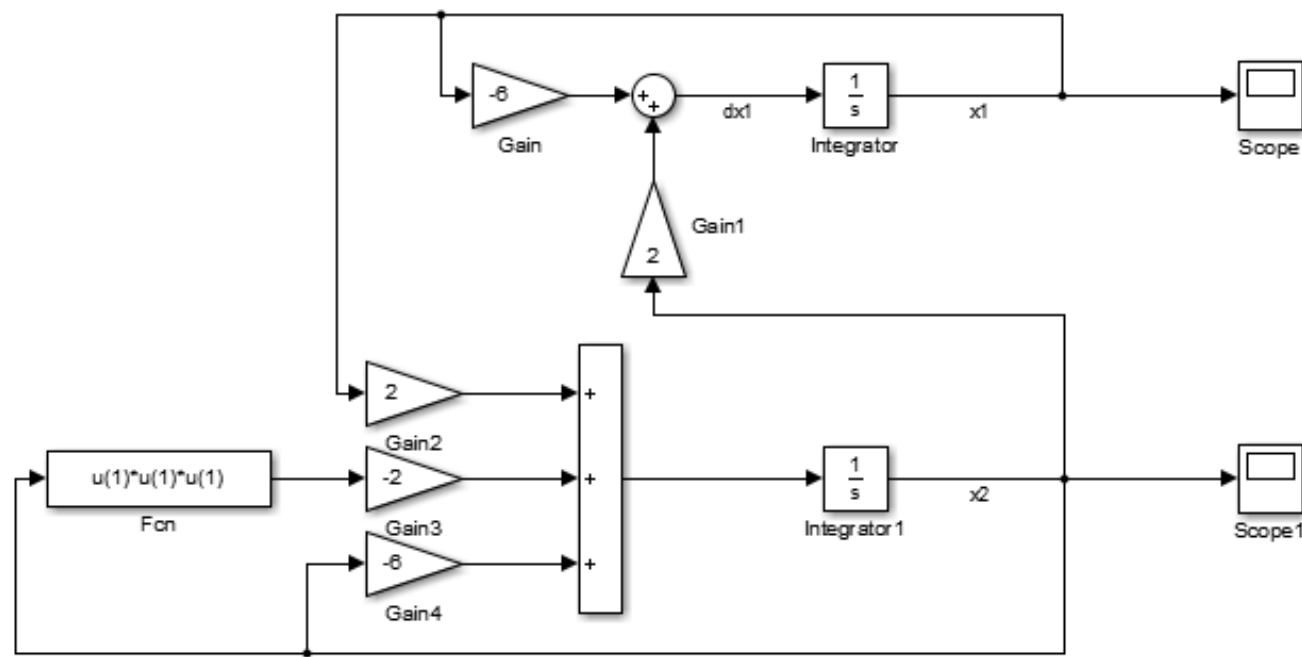
- ✓ 搭建系统结构框图模型
- ✓ 运行系统获取响应曲线



线性系统数学模型表示

例1-4：试用Simulink描述如下系统模型

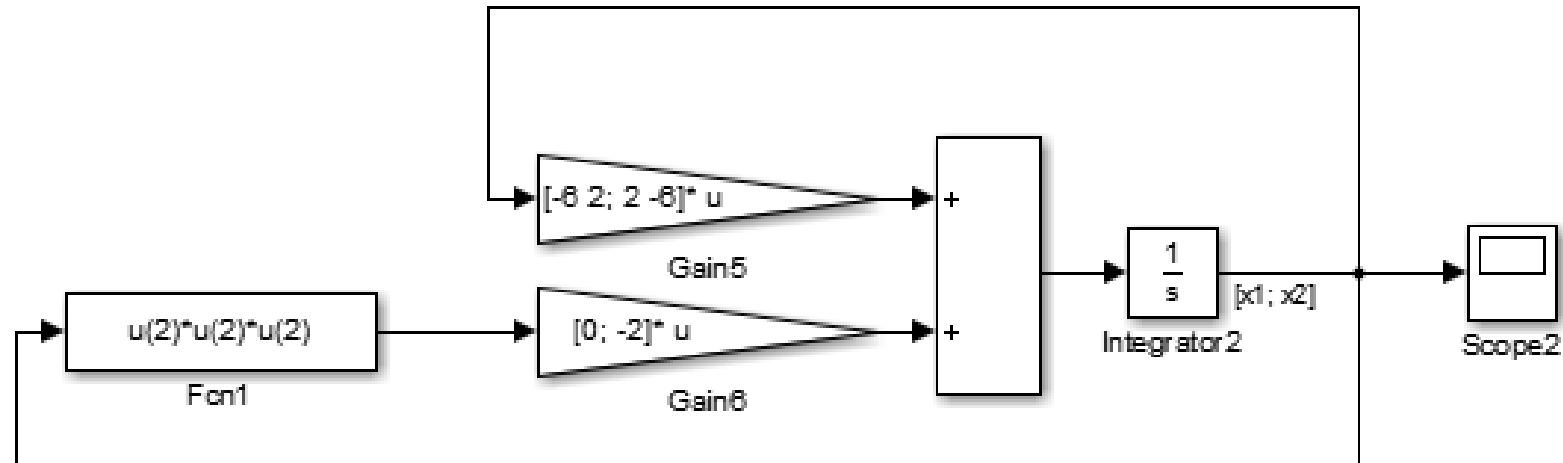
$$\begin{cases} \dot{x}_1 = -6x_1 + 2x_2 \\ \dot{x}_2 = 2x_1 - 6x_2 - 2x_2^3 \end{cases}$$



线性系统数学模型表示

例1-4：试用Simulink描述如下系统模型

$$\begin{cases} \dot{x}_1 = -6x_1 + 2x_2 \\ \dot{x}_2 = 2x_1 - 6x_2 - 2x_2^3 \end{cases} \quad \Rightarrow \quad \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -6 & 2 \\ 2 & -6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ -2 \end{bmatrix} x_2^3$$



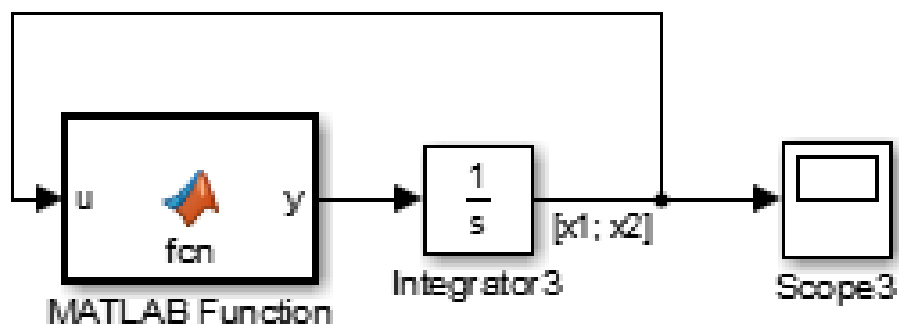
线性系统数学模型表示

例1-4：试用Simulink描述如下系统模型

$$\begin{cases} \dot{x}_1 = -6x_1 + 2x_2 \\ \dot{x}_2 = 2x_1 - 6x_2 - 2x_2^3 \end{cases}$$



$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -6 & 2 \\ 2 & -6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ -2 \end{bmatrix} x_2^3$$



```
MATLAB Function  x  +
1  function y = fcn(u)
2      %#codegen
3
4      % x1 = u(1);
5      % x2 = u(2);
6      % dx1 = -6*x1 + 2*x2;
7      % dx2 = 2*x1 - 6*x2 - 2*x2^3;
8      % y = [dx1; dx2];
9
10 - y = [-6 2; 2 -6]*u + [0; -2]*u(2)^3;
```

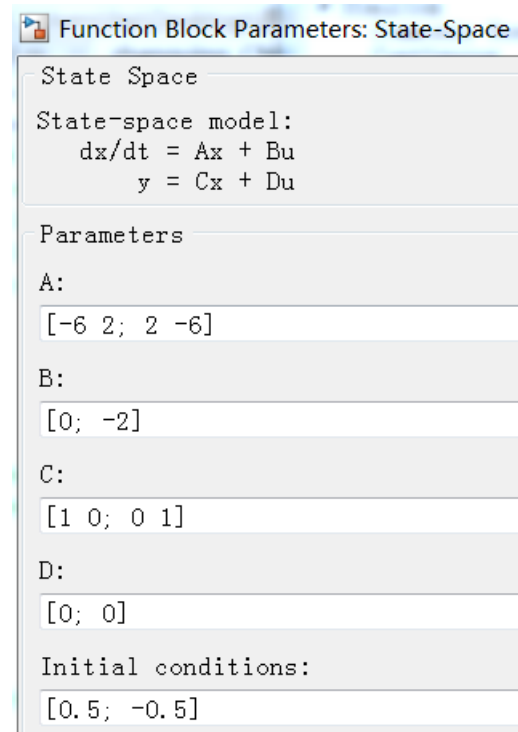
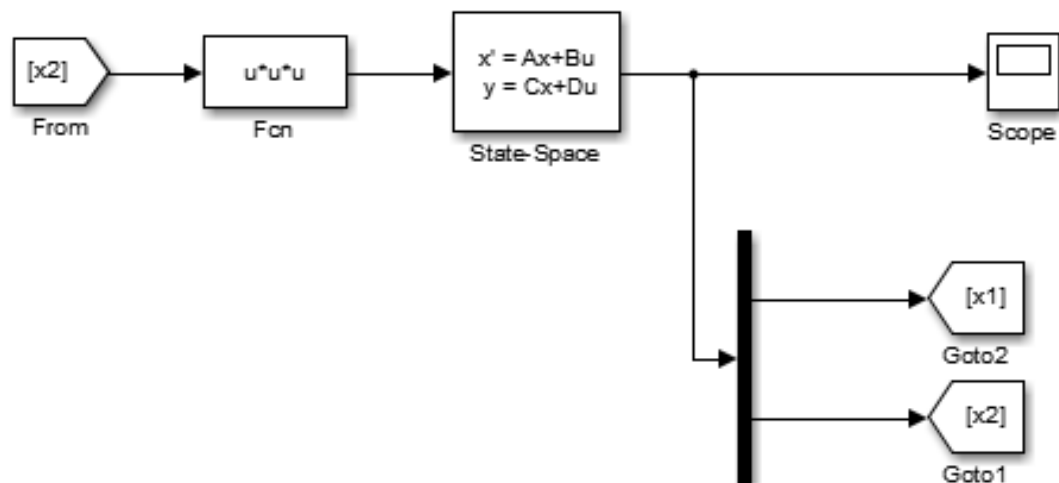
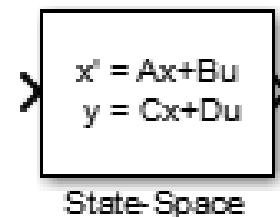
线性系统数学模型表示

例1-4：试用Simulink描述如下系统模型

$$\begin{cases} \dot{x}_1 = -6x_1 + 2x_2 \\ \dot{x}_2 = 2x_1 - 6x_2 - 2x_2^3 \end{cases}$$



$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -6 & 2 \\ 2 & -6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ -2 \end{bmatrix} x_2^3$$



课内实验：模型部分

1.1 MATLAB使用简介

1.2 线性系统数学模型表示

1.3 状态空间模型与其它模型的相互转化

1.4 状态空间模型与状态空间模型的转化（坐标变换）

1.5 多个子系统不同连接下的整体系统模型

状态空间模型与其它模型的相互转化

传递函数模型

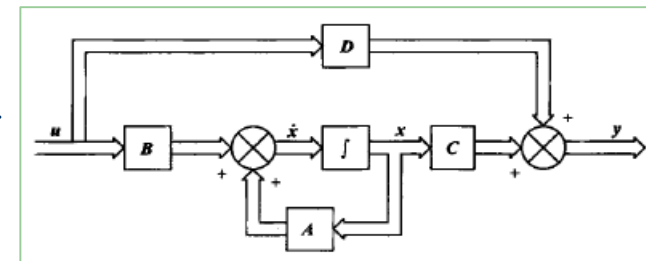
$$W(s) = \frac{b_m s^m + b_{m-1} s^{m-1} + \cdots + b_1 s + b_0}{s^n + a_{n-1} s^{n-1} + \cdots + a_1 s + a_0}$$

$$W(s) = k \frac{(s - z_1)(s - z_2) \cdots (s - z_m)}{(s - p_1)(s - p_2) \cdots (s - p_n)}$$

状态空间模型

$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}u(t) \\ \mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}u(t) \end{cases}$$

结构图模型



状态空间模型与其它模型的相互转化

$$W(s) = \frac{b_m s^m + b_{m-1} s^{m-1} + \cdots + b_1 s + b_0}{s^n + a_{n-1} s^{n-1} + \cdots + a_1 s + a_0}$$

$$W(s) = k \frac{(s - z_1)(s - z_2) \cdots (s - z_m)}{(s - p_1)(s - p_2) \cdots (s - p_n)}$$



$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}u(t) \\ \mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}u(t) \end{cases}$$

tf2ss()函数、zp2ss() 函数

功能： 传函模型转化为状态空间模型
格式：

$$[\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}] = \text{tf2ss}(\text{num}, \text{den})$$

$$[\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}] = \text{zp2ss}(\text{num}, \text{den})$$

tf2ss

Convert transfer function filter parameters to state-space form [expand all in page](#)

Syntax

```
[A,B,C,D] = tf2ss(b,a)
```

Description

tf2ss converts the parameters of a transfer function representation of a given system to those of an equivalent state-space representation.

[A,B,C,D] = tf2ss(b,a) returns the A, B, C, and D matrices of a state space representation for the single-input transfer function

$$H(s) = \frac{B(s)}{A(s)} = \frac{b_1 s^{n-1} + \cdots + b_{n-1} s + b_n}{a_1 s^{m-1} + \cdots + a_{m-1} s + a_m} = C(sI - A)^{-1}B + D$$

in controller canonical form

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{B}u \\ \mathbf{y} &= \mathbf{C}\mathbf{x} + \mathbf{D}u \end{aligned}$$

The input vector a contains the denominator coefficients in descending powers of s. The rows of the matrix b contain the vectors of numerator coefficients (each row corresponds to an output). In the discrete-time case, you must supply b and a to correspond to the numerator and denominator polynomials with coefficients in descending powers of z.

For discrete-time systems you must make b have the same number of columns as the length of a. You can do this by padding each numerator represented in b (and possibly the denominator represented in the vector a) with trailing zeros. You can use the function `eqtlength` to accomplish this if b and a are vectors of unequal lengths.

zp2ss

Convert zero-pole-gain filter parameters to state-space form

Syntax

```
[A,B,C,D] = zp2ss(z,p,k)
```

Description

zp2ss converts a zero-pole-gain representation of a given system to an equivalent state-space representation.

[A,B,C,D] = zp2ss(z,p,k) finds a single input, multiple output, state-space representation

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{B}u \\ \mathbf{y} &= \mathbf{C}\mathbf{x} + \mathbf{D}u \end{aligned}$$

given a system in factored transfer function form.

$$H(s) = \frac{Z(s)}{P(s)} = k \frac{(s - z_1)(s - z_2) \cdots (s - z_n)}{(s - p_1)(s - p_2) \cdots (s - p_n)}$$

Column vector p specifies the pole locations, and matrix z the zero locations with as many columns as there are outputs. The gains for each numerator transfer function are in vector k. The A, B, C, and D matrices are returned in controller canonical form.

Inf values may be used as place holders in z if some columns have fewer zeros than others.

More About

[expand all](#)

► Algorithms

状态空间模型与其它模型的相互转化

例1-5：将以下传递函数模型变为状态空间模型

$$W(s) = \frac{s^2 + 3s + 1}{s^3 + 2s^2 + 4s + 6}$$
$$\begin{cases} \dot{x} = \begin{bmatrix} -2 & -4 & -6 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} x + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} u \\ y = [1 \quad 3 \quad 1]x \end{cases}$$

```
Editor - C:\Users\hp\Desktop\ex1_5.m
ex1_1.m x ex1_2.m x ex1_3.m x ex1_5.m x
1 %%%%%%%%%%%%%%
2 %% Chuan-Ke Zhang
3 %% 2021-10-06
4 %% Example 1-5
5 %% IF 2 SS
6 %%%%%%%%%%%%%%
7
8 - clc
9 - clear
10
11 - num = [1 3 1]; % 分子多项式系数
12 - den = [1 2 4 6]; % 分母多项式系数
13
14 - [A1,B1,C1,D1] = tf2ss(num,den)
15 - [A2,B2,C2,D2] = tf2ss([1 3 1],[1 2 4 6])
```

Command Window

```
A1 =
    -2    -4    -6
     1     0     0
     0     1     0

B1 =
     1
     0
     0

C1 =
     1     3     1

D1 =
     0
```

Command Window

```
A2 =
    -2    -4    -6
     1     0     0
     0     1     0

B2 =
     1
     0
     0

C2 =
     1     3     1

D2 =
     0
```

状态空间模型与其它模型的相互转化

$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \\ \mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t) \end{cases} \rightarrow \begin{cases} W(s) = \frac{b_ms^m + b_{m-1}s^{m-1} + \dots + b_1s + b_0}{s^n + a_{n-1}s^{n-1} + \dots + a_1s + a_0} \\ W(s) = k \frac{(s-z_1)(s-z_2)\dots(s-z_m)}{(s-p_1)(s-p_2)\dots(s-p_n)} \end{cases}$$

ss2tf()函数、ss2zp() 函数

功能：状态空间模型转化为传函模型

格式：

SISO:

$$[\text{num}, \text{den}] = \text{ss2tf}(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D})$$

$$[\mathbf{z}, \mathbf{p}, \mathbf{k}] = \text{ss2zp}(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D})$$

MIMO:

$$[\text{num}, \text{den}] = \text{ss2tf}(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}, \text{iu})$$

$$[\mathbf{z}, \mathbf{p}, \mathbf{k}] = \text{ss2zp}(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}, \text{iu})$$

ss2tf
Convert state-space representation to transfer function [expand all in page](#)

Syntax

```
[b,a] = ss2tf(A,B,C,D)  
[b,a] = ss2tf(A,B,C,D,ni)
```

[example](#) [example](#)

Description

[example](#)

`[b,a] = ss2tf(A,B,C,D)` converts a state-space representation of a system into an equivalent transfer function. `ss2tf` returns the Laplace-transform transfer function for continuous-time systems and the Z-transform transfer function for discrete-time systems.

[example](#)

`[b,a] = ss2tf(A,B,C,D,ni)` returns the transfer function that results when the *ni*th input of a system with multiple inputs is excited by a unit impulse.

Examples [expand all](#)

- ▶ Mass-Spring System
- ▶ Two-Body Oscillator

Input Arguments [expand all](#)

- ▶ **A** — State matrix matrix
- ▶ **B** — Input-to-state matrix matrix

[Open Help Browser](#) F1 to toggle focus; Escape to close...

ss2zp
Convert state-space filter parameters to zero-pole-gain form [expand all in page](#)

Syntax

```
[z,p,k] = ss2zp(A,B,C,D,i)
```

Description

`ss2zp` converts a state-space representation of a given system to an equivalent zero-pole-gain representation. The zeros, poles, and gains of state-space systems represent the transfer function in factored form.

`[z,p,k] = ss2zp(A,B,C,D,i)` calculates the transfer function in factored form

$$H(s) = \frac{Z(s)}{P(s)} = k \frac{(s-z_1)(s-z_2)\dots(s-z_n)}{(s-p_1)(s-p_2)\dots(s-p_n)}$$

of the continuous-time system

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \\ \mathbf{y} &= \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u} \end{aligned}$$

from the *i*th input (using the *i*th columns of **B** and **D**). The column vector **p** contains the pole locations of the denominator coefficients of the transfer function. The matrix **z** contains the numerator zeros in its columns, with as many columns as there are outputs **y** (rows in **C**). The column vector **k** contains the gains for each numerator transfer function.

`ss2zp` also works for discrete time systems. The input state-space system must be real.

The `ss2zp` function is part of the standard MATLAB® language.

Examples [expand all](#)

[Open Help Browser](#) F1 to toggle focus; Escape to close...

状态空间模型与其它模型的相互转化

例1-6: 求以下状态空间模型对应的传递函数阵

$$\begin{cases} \dot{x} = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix} x + \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} u \\ y = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 2 \end{bmatrix} x + \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} u \end{cases}$$

$$W(s) = \frac{\begin{bmatrix} s+4 & 1 \\ s^2+5s+4 & s+1 \\ 2s-4 & s^2+5s+2 \end{bmatrix}}{s^2+3s+2}$$

```
Editor - C:\Users\hp\Desktop\ex1_6.m
ex1_3.m  ex1_5.m  ex1_6.m
1  %%%%%%%%%%
2  %%% Chuan-Ke Zhang
3  %%% 2021-10-06
4  %%% Example 1-6
5  %%% SS 2 IF
6  %%%%%%%%%%
7
8  -  clc
9  -  clear
10
11 -  A = [0 1; -2 -3];
12 -  B = [1 0; 1 1];
13 -  C = [1 0; 1 1; 0 2];
14 -  D = [0 0; 1 0; 0 1];
15
16 -  [num1, den1] = ss2tf(A, B, C, D, 1)
17 -  [num2, den2] = ss2tf(A, B, C, D, 2)
```

Command Window

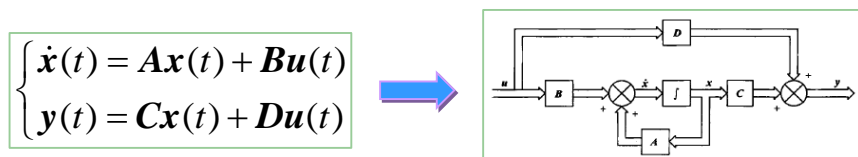
```
num1 =
      0      1.0000      4.0000
      1.0000      5.0000      4.0000
      0      2.0000     -4.0000

den1 =
      1      3      2

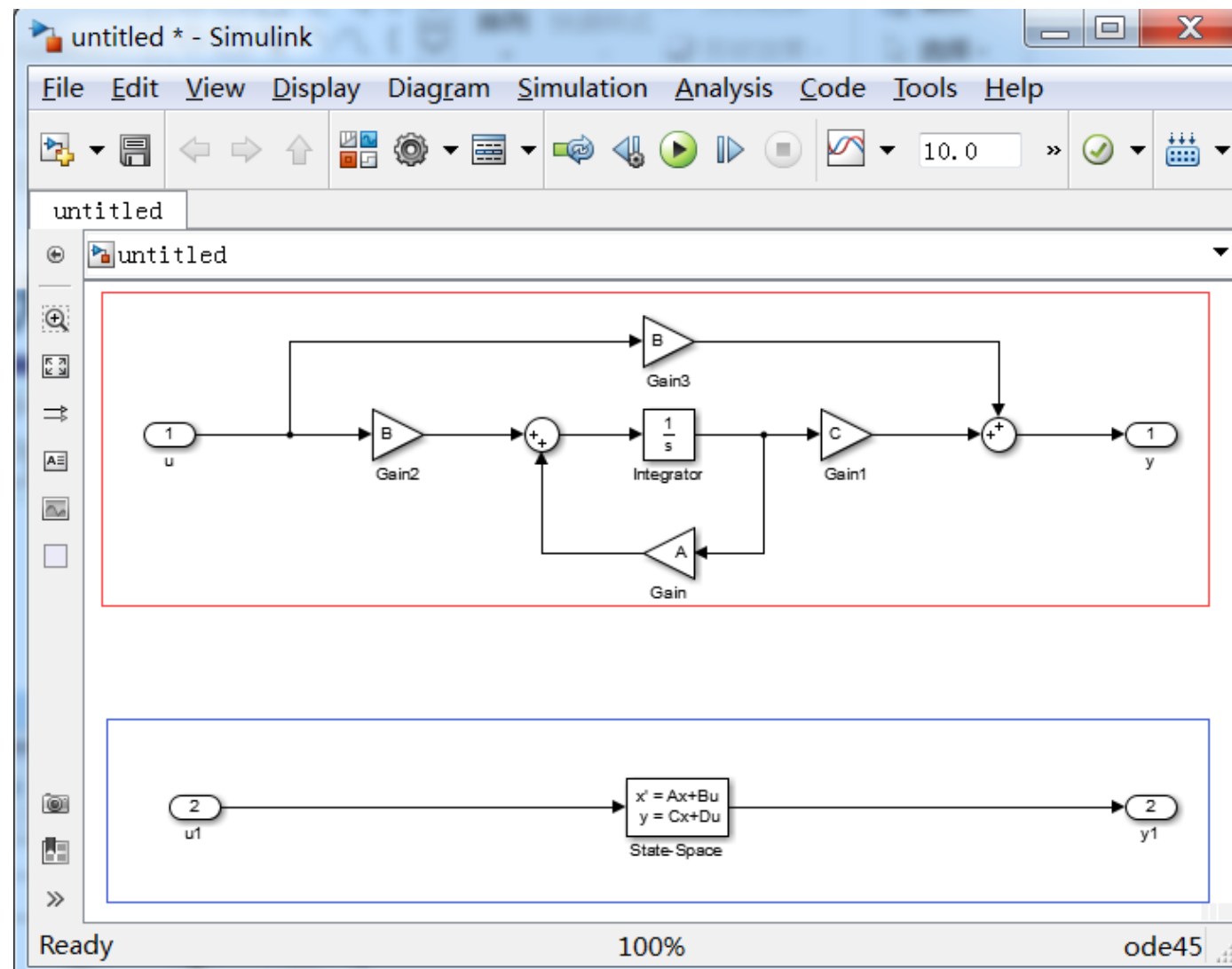
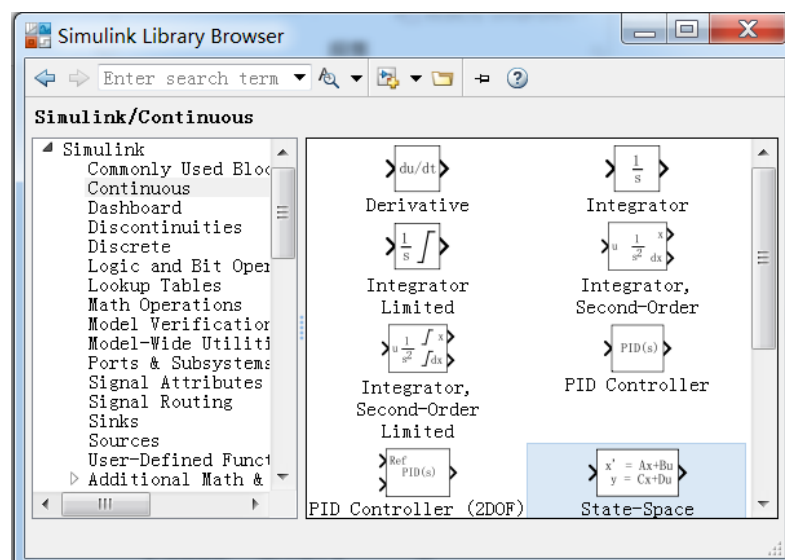
num2 =
      0      0      1.0000
      0      1.0000      1.0000
      1.0000      5.0000      2.0000

den2 =
      1      3      2
```

状态空间模型与其它模型的相互转化

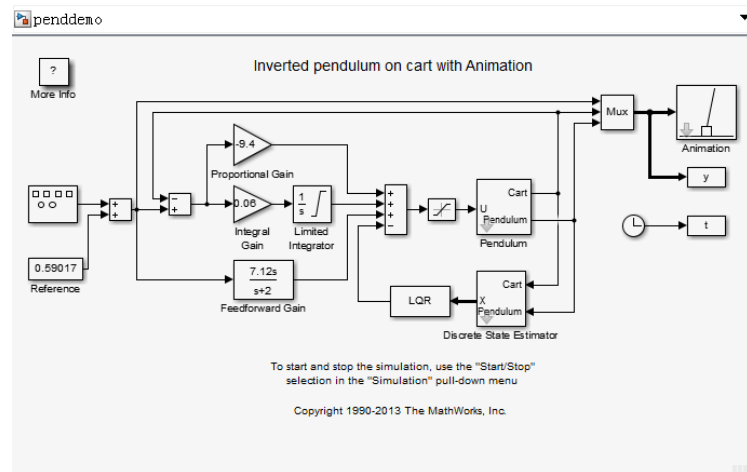


Simulink



例1-4: Pages 15-18

状态空间模型与其它模型的相互转化



$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}u(t) \\ \mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}u(t) \end{cases}$$

Simulink: Linear Analysis Tool (自学内容)

功能：从结构图获取线性状态空间模型

simulink 线性 工具



百度一下

Q 网页 资讯 视频 图片 知道 文库 贴吧 地图 采购 更多

百度为您找到相关结果约11,400,000个

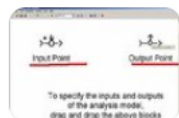
搜索工具

[利用MATLAB 自带的工具提取系统线性化总模型 - 百度经验](#)

2014年8月5日 利用MATLAB 自带的工具提取系统线性化总模型,利用MATLAB自带的liearaal yi工具提取系统线性化总模型,将复杂系统线性化,更容易分析其时域和频域的关系了

百度经验 百度快照

[Simulink下的线性系统分析工具在哪找到? - MATLAB中文论坛](#)



2012年2月12日 书上说“选中Tools|Linear System 菜单项,则打开线性系统分析工具,用户选择其中的Get Linear Model 菜单...

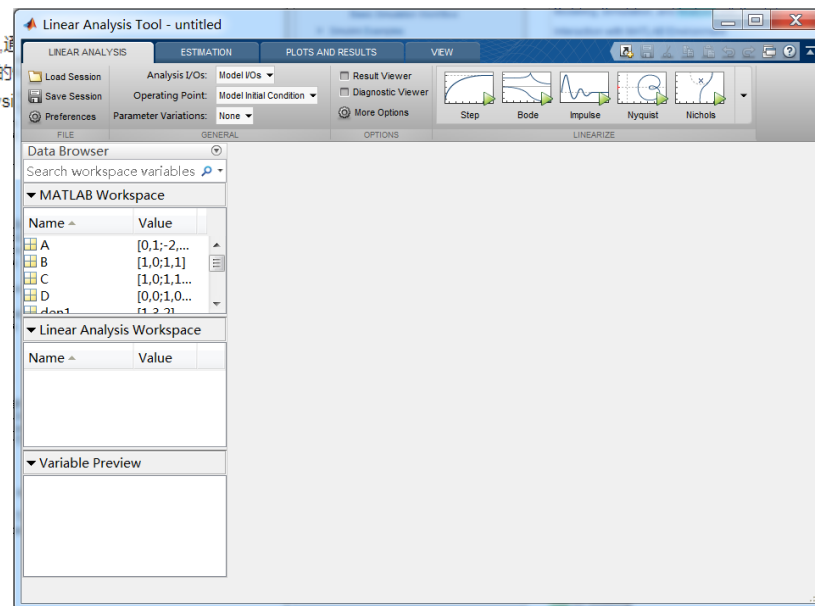
www.ilovematlab.cn/thread-1619... 百度快照

[simulink之线性系统分析 - 知乎](#)



2018年9月13日 在利用matlab 进行控制系统设计时,通过 step, rlocus, nichols, step等指令来分析线性系统的,可以进行相同分析,且更加直观,性能更强。 Analysis

知乎 百度快照



课内实验：模型部分

1.1 MATLAB使用简介

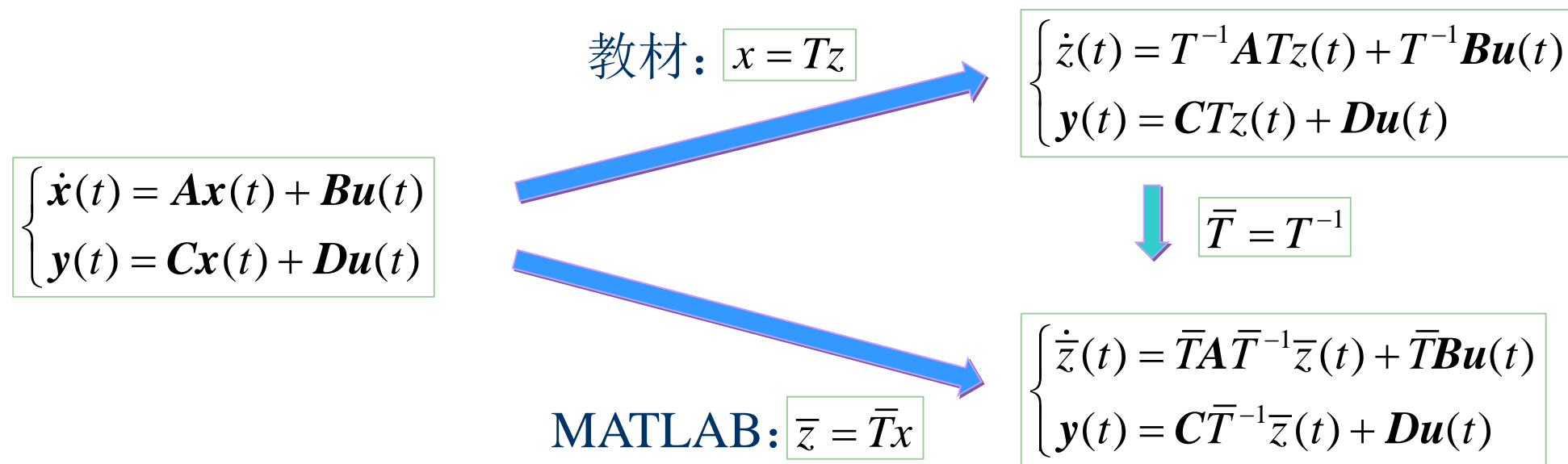
1.2 线性系统数学模型表示

1.3 状态空间模型与其它模型的相互转化

1.4 状态空间模型与状态空间模型的转化（坐标变换）

1.5 多个子系统不同连接下的整体系统模型

状态空间模型与状态空间模型的转化



实现状态空间模型的规范化，以便于系统分析与设计

- ✓ 对角线标准型、约旦标准型
- ✓ 能控标准型、能观标准型（下次实验）
- ✓ 能控结构分解、能观结构分解（下次实验）

状态空间模型与状态空间模型的转化

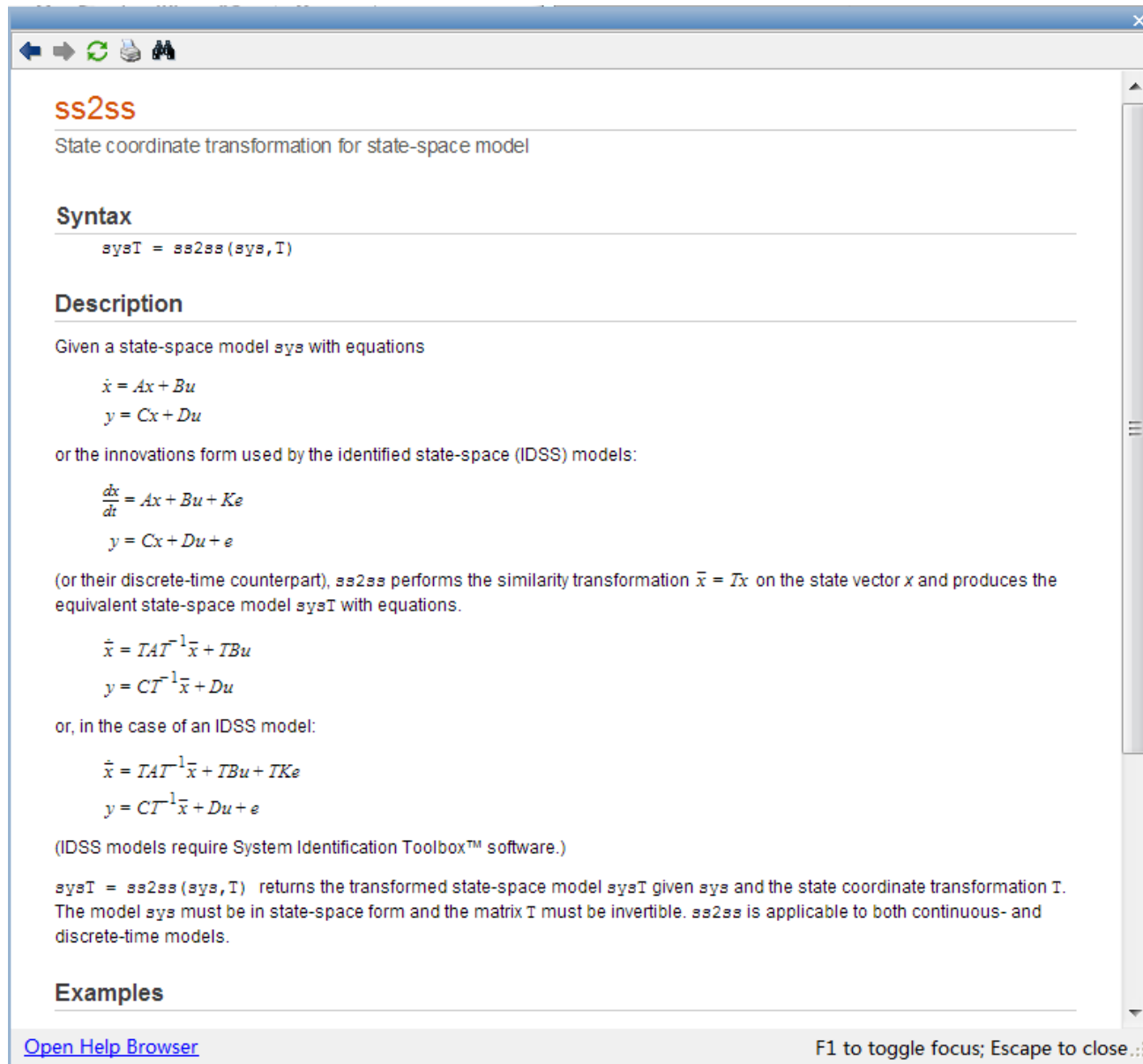
$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \\ \mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t) \end{cases} \quad \rightarrow \quad \begin{cases} \dot{\bar{\mathbf{z}}}(t) = \bar{\mathbf{T}}\mathbf{A}\bar{\mathbf{T}}^{-1}\bar{\mathbf{z}}(t) + \bar{\mathbf{T}}\mathbf{B}\mathbf{u}(t) \\ \mathbf{y}(t) = \mathbf{C}\bar{\mathbf{T}}^{-1}\bar{\mathbf{z}}(t) + \mathbf{D}\mathbf{u}(t) \end{cases}$$

ss2ss()函数

功能：实现状态空间模型的线性变换格式：

$$\begin{aligned} \text{sys_new} &= \text{ss2ss}(\text{sys_original}, \mathbf{T}) \\ [\mathbf{A}_n, \mathbf{B}_n, \mathbf{C}_n, \mathbf{D}_n] &= \text{ss2ss}(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}, \mathbf{T}) \end{aligned}$$

其中， \mathbf{T} 为非奇异变换矩阵



The screenshot shows the MATLAB help window for the `ss2ss` function. The window title is "ss2ss". The subtitle is "State coordinate transformation for state-space model". The "Syntax" section shows the function call: `sysT = ss2ss(sys,T)`. The "Description" section explains that given a state-space model `sys` with equations $\dot{x} = Ax + Bu$ and $y = Cx + Du$, or the innovations form used by the identified state-space (IDSS) models: $\frac{dx}{dt} = Ax + Bu + Ke$ and $y = Cx + Du + e$, the function performs a similarity transformation $\bar{x} = Tx$ on the state vector x and produces the equivalent state-space model `sysT` with equations: $\dot{\bar{x}} = \bar{T}A\bar{T}^{-1}\bar{x} + \bar{T}Bu$ and $y = C\bar{T}^{-1}\bar{x} + Du$. It also mentions that for IDSS models, the equations are $\dot{\bar{x}} = \bar{T}A\bar{T}^{-1}\bar{x} + \bar{T}Bu + \bar{T}Ke$ and $y = C\bar{T}^{-1}\bar{x} + Du + e$. The text notes that IDSS models require System Identification Toolbox™ software. The "Examples" section shows the function call: `sysT = ss2ss(sys,T)` returns the transformed state-space model `sysT` given `sys` and the state coordinate transformation `T`. The model `sys` must be in state-space form and the matrix `T` must be invertible. `ss2ss` is applicable to both continuous- and discrete-time models. At the bottom, there is a link "Open Help Browser" and a note "F1 to toggle focus; Escape to close..."

状态空间模型与状态空间模型的转化

$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \\ \mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t) \end{cases} \rightarrow \begin{cases} \dot{\bar{\mathbf{z}}}(t) = \bar{\mathbf{T}}\mathbf{A}\bar{\mathbf{T}}^{-1}\bar{\mathbf{z}}(t) + \bar{\mathbf{T}}\mathbf{B}\mathbf{u}(t) \\ \mathbf{y}(t) = \mathbf{C}\bar{\mathbf{T}}^{-1}\bar{\mathbf{z}}(t) + \mathbf{D}\mathbf{u}(t) \end{cases}$$

对角线标准型

要求：系统矩阵均为单根

步骤：先求变换矩阵T，再求标准型

$$\text{Lambda} = \text{eig}(\mathbf{A})$$

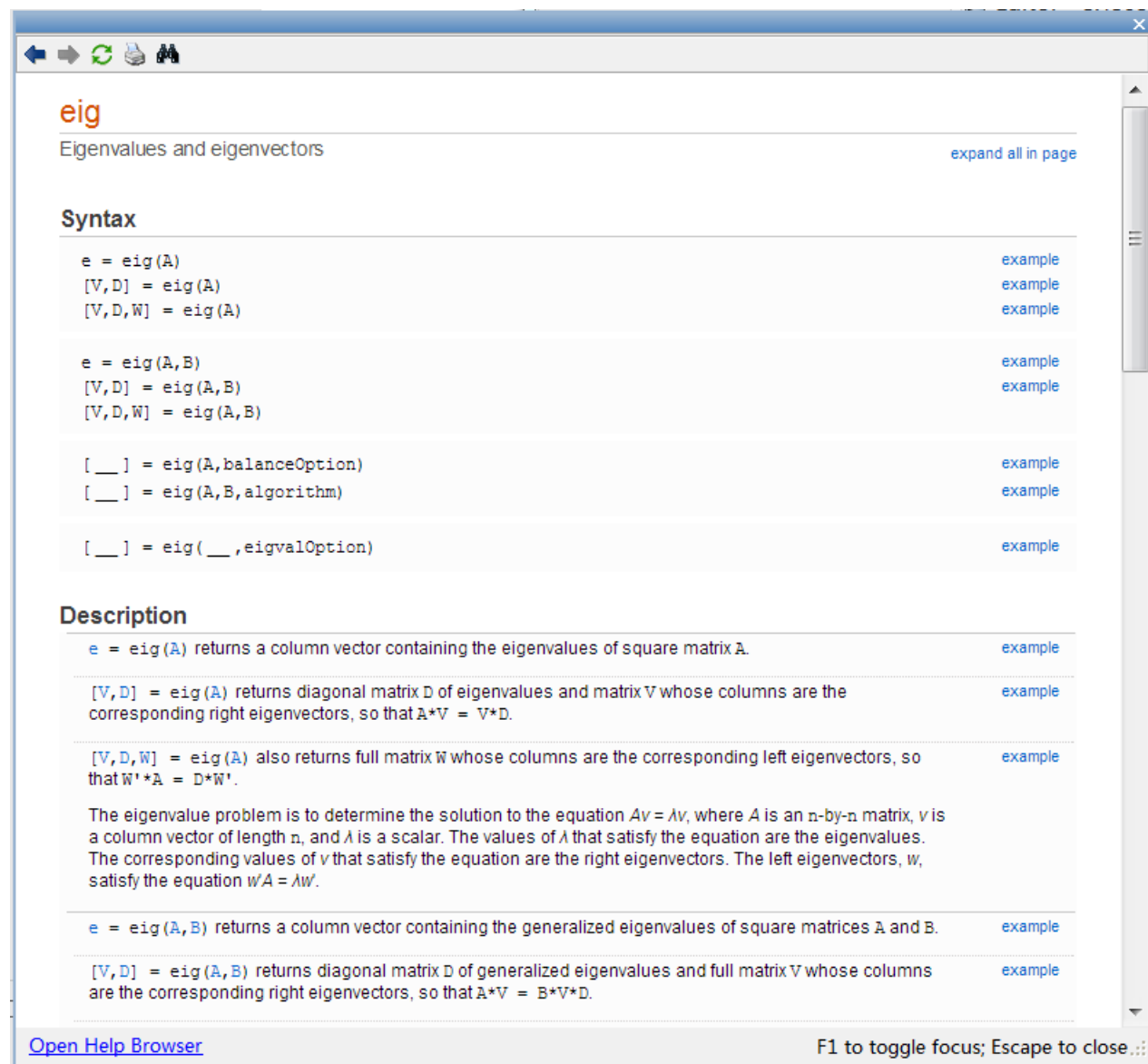
$$[\mathbf{V}, \mathbf{L}] = \text{eig}(\mathbf{A})$$

$$\mathbf{T} = \text{inv}(\mathbf{V})$$

$$\text{sys_new} = \text{ss2ss}(\text{sys_original}, \mathbf{T})$$

$$[\mathbf{A}_n, \mathbf{B}_n, \mathbf{C}_n, \mathbf{D}_n] = \text{ss2ss}(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}, \mathbf{T})$$

其中，Lambda为特征值，V为对应特征向量，即为所需变换阵T的逆



The screenshot shows the MATLAB help browser for the `eig` function. The title is "eig" with the subtitle "Eigenvalues and eigenvectors". There is a link "expand all in page".

Syntax

- `e = eig(A)` (example)
- `[V,D] = eig(A)` (example)
- `[V,D,W] = eig(A)` (example)
- `e = eig(A,B)` (example)
- `[V,D] = eig(A,B)` (example)
- `[V,D,W] = eig(A,B)` (example)
- `[__] = eig(A,balanceOption)` (example)
- `[__] = eig(A,B,algorithm)` (example)
- `[__] = eig(__,eigvalOption)` (example)

Description

- `e = eig(A)` returns a column vector containing the eigenvalues of square matrix A. (example)
- `[V,D] = eig(A)` returns diagonal matrix D of eigenvalues and matrix V whose columns are the corresponding right eigenvectors, so that $A*V = V*D$. (example)
- `[V,D,W] = eig(A)` also returns full matrix W whose columns are the corresponding left eigenvectors, so that $W'*A = D*W'$. (example)
- The eigenvalue problem is to determine the solution to the equation $Av = \lambda v$, where A is an n-by-n matrix, v is a column vector of length n, and λ is a scalar. The values of λ that satisfy the equation are the eigenvalues. The corresponding values of v that satisfy the equation are the right eigenvectors. The left eigenvectors, w, satisfy the equation $wA = \lambda w'$.
- `e = eig(A,B)` returns a column vector containing the generalized eigenvalues of square matrices A and B. (example)
- `[V,D] = eig(A,B)` returns diagonal matrix D of generalized eigenvalues and full matrix V whose columns are the corresponding right eigenvectors, so that $A*V = B*V*D$. (example)

Open Help Browser F1 to toggle focus; Escape to close...

状态空间模型与状态空间模型的转化

例1-7：将以下状态空间模型化为对角线标准型

$$\begin{cases} \dot{x} = \begin{bmatrix} 2 & -1 & -1 \\ 0 & -1 & 0 \\ 0 & 2 & 1 \end{bmatrix} x + \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} u \\ y = [1 \quad 0 \quad 1] x + u \end{cases}$$

$$\begin{cases} \dot{x} = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1.4142 \\ 1.4142 \end{bmatrix} u \\ y = [1 \quad 1.4142 \quad -0.7071] x + u \end{cases}$$

```
Editor - C:\Users\hp\Desktop\ex1_7.m
ex1_6.m x ex1_7.m x +
1 %%%%%%%%%%%%%%
2 %%% Chuan-Ke Zhang
3 %%% 2021-10-06
4 %%% Example 1-7
5 %%% SS 2 SS, 对角线标准型
6 %%%%%%%%%%%%%%
7
8 - clc
9 - clear
10
11 - A = [2 -1 -1; 0 -1 0; 0 2 1];
12 - B = [1; 1; 0];
13 - C = [1 0 1];
14 - D = 1;
15
16 - Lambda1 = eig(A) % 只求特征值
17 - [V, Lambda2] = eig(A); % 同时求特征值及其对应特征向量
18 - T = inv(V);
19 - [An, Bn, Cn, Dn] = ss2ss(A, B, C, D, T)
```

Command Window

Lambda1 =

2
1
-1

Command Window

An =

2 0 0
0 1 0
0 0 -1

Bn =

0
1.4142
1.4142

Cn =

1.0000 1.4142 -0.7071

Dn =

1

fx >> |

状态空间模型与状态空间模型的转化

$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \\ \mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t) \end{cases} \quad \Rightarrow \quad \begin{cases} \dot{\bar{\mathbf{z}}}(t) = \bar{\mathbf{T}}\mathbf{A}\bar{\mathbf{T}}^{-1}\bar{\mathbf{z}}(t) + \bar{\mathbf{T}}\mathbf{B}\mathbf{u}(t) \\ \mathbf{y}(t) = \mathbf{C}\bar{\mathbf{T}}^{-1}\bar{\mathbf{z}}(t) + \mathbf{D}\mathbf{u}(t) \end{cases}$$

约旦标准型

要求：系统矩阵存在重根

步骤：先求变换矩阵T，再求标准型

$$\mathbf{J} = \text{jordan}(\mathbf{A})$$

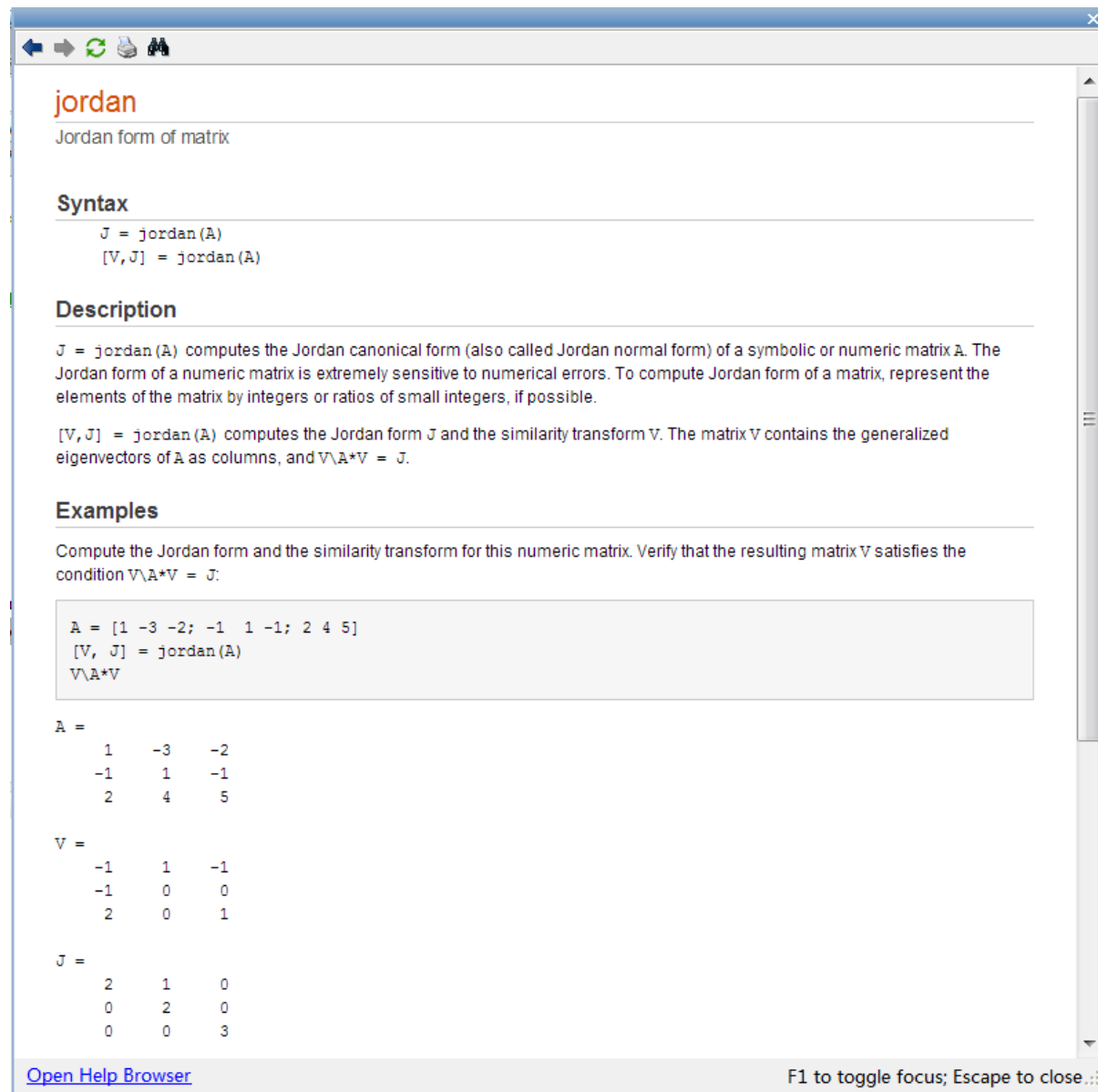
$$[\mathbf{V}, \mathbf{J}] = \text{jordan}(\mathbf{A})$$

$$\mathbf{T} = \text{inv}(\mathbf{V})$$

$$\text{sys_new} = \text{ss2ss}(\text{sys_original}, \mathbf{T})$$

$$[\mathbf{A}_n, \mathbf{B}_n, \mathbf{C}_n, \mathbf{D}_n] = \text{ss2ss}(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}, \mathbf{T})$$

其中，J为约旦矩阵，V为对应广义特征向量，即为所需变换阵T的逆



The image shows a MATLAB Help Browser window titled "jordan". The window displays the following information:

- Header:** "jordan" (in orange), "Jordan form of matrix"
- Syntax:**
 - `J = jordan(A)`
 - `[V, J] = jordan(A)`
- Description:**
 - `J = jordan(A)` computes the Jordan canonical form (also called Jordan normal form) of a symbolic or numeric matrix A. The Jordan form of a numeric matrix is extremely sensitive to numerical errors. To compute Jordan form of a matrix, represent the elements of the matrix by integers or ratios of small integers, if possible.
 - `[V, J] = jordan(A)` computes the Jordan form J and the similarity transform V. The matrix V contains the generalized eigenvectors of A as columns, and $V \backslash A * V = J$.
- Examples:**
 - Compute the Jordan form and the similarity transform for this numeric matrix. Verify that the resulting matrix V satisfies the condition $V \backslash A * V = J$:
 - Code snippet:

```
A = [1 -3 -2; -1 1 -1; 2 4 5];
[V, J] = jordan(A);
V \ A * V
```
 - Output:

```
A =
     1     -3     -2
    -1      1     -1
     2      4      5

V =
    -1      1     -1
    -1      0      0
     2      0      1

J =
     2      1      0
     0      2      0
     0      0      3
```

At the bottom, there is a link "Open Help Browser" and a status bar indicating "F1 to toggle focus; Escape to close..."

状态空间模型与状态空间模型的转化

例1-8：将以下状态空间模型化为对角线标准型

$$\begin{cases} \dot{x} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 8 & -12 & 6 \end{bmatrix} x + \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} u \\ y = [1 \ 0 \ 1] x + u \end{cases}$$

$$\begin{cases} \dot{x} = \begin{bmatrix} 2 & 1 & 0 \\ 0 & 2 & 1 \\ 0 & 0 & 2 \end{bmatrix} x + \begin{bmatrix} 0.125 \\ -0.25 \\ 0 \end{bmatrix} u \\ y = [20 \ 6 \ 1] x + u \end{cases}$$

```
Editor - C:\Users\hp\Desktop\ex1_8.m
ex1_6.m x ex1_7.m x ex1_8.m x +
1 %%%%%%%%%%%%%%
2 %%% Chuan-Ke Zhang
3 %%% 2021-10-06
4 %%% Example 1-8
5 %%% SS 2 SS, 约旦标准型
6 %%%%%%%%%%%%%%
7
8 - clc
9 - clear
10
11 - A = [0 1 0; 0 0 1; 8 -12 6];
12 - B = [1; 1; 0];
13 - C = [1 0 1];
14 - D = 1;
15
16 - J = jordan(A) % 只求广义特征值构成的约旦阵
17 - [V, J] = jordan(A); % 同时求约旦阵及其对应特征向量
18 - I = inv(V);
19 - [An, Bn, Cn, Dn] = ss2ss(A, B, C, D, I)
```

Command Window

J =

```
2    1    0
0    2    1
0    0    2
```

Command Window

An =

```
2    1    0
0    2    1
0    0    2
```

Bn =

```
0.1250
-0.2500
0
```

Cn =

```
20    6    1
```

Dn =

```
1
```

fx >>

课内实验：模型部分

1.1 MATLAB使用简介

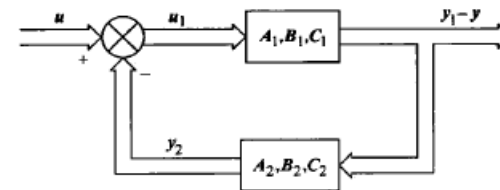
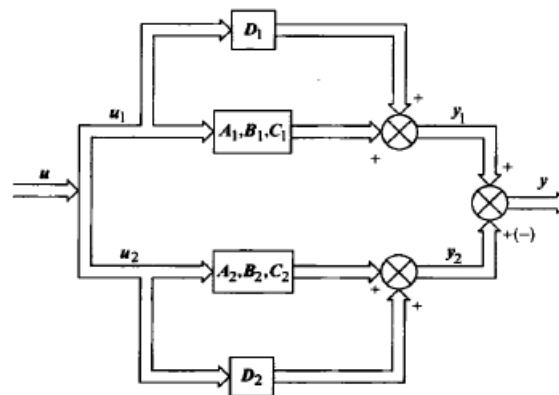
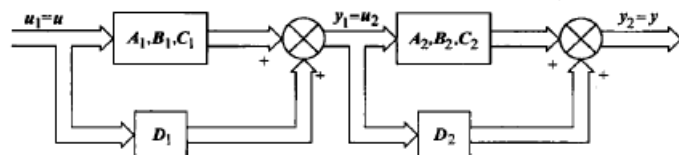
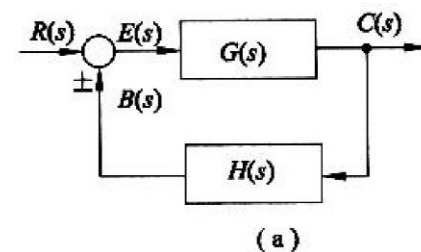
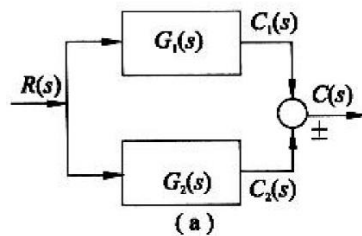
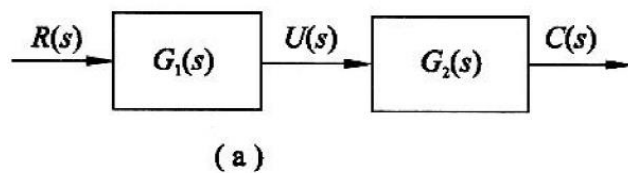
1.2 线性系统数学模型表示

1.3 状态空间模型与其它模型的相互转化

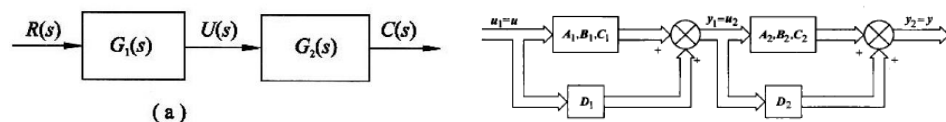
1.4 状态空间模型与状态空间模型的转化（坐标变换）

1.5 多个子系统不同连接下的整体系统模型

多子系统连接下的整体系统模型



多子系统连接下的整体系统模型



series()函数

功能：实现两个子系统的串联
格式：

$\text{sys_new} = \text{series}(\text{sys1}, \text{sys2})$

$[A, B, C, D] = \text{series}(A1, B1, C1, D1, A2, B2, C2, D2)$

$[\text{num}, \text{den}] = \text{series}(\text{num1}, \text{den1}, \text{num2}, \text{den2})$

其中，sys1和sys2为同类表示子系统

series
Series connection of two models

Syntax

```
series  
sys = series(sys1, sys2)  
sys = series(sys1, sys2, outputs1, inputs2)
```

Description

`series` connects two model objects in series. This function accepts any type of model. The two systems must be either both continuous or both discrete with identical sample time. Static gains are neutral and can be specified as regular matrices.

`sys = series(sys1, sys2)` forms the basic series connection shown below.

This command is equivalent to the direct multiplication

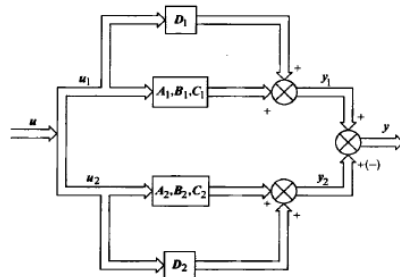
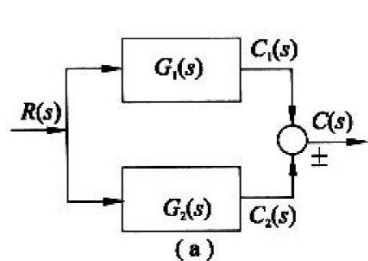
```
sys = sys2 * sys1
```

`sys = series(sys1, sys2, outputs1, inputs2)` forms the more general series connection.

The index vectors `outputs1` and `inputs2` indicate which outputs y_1 of `sys1` and which inputs u_2 of `sys2` should be connected. The resulting model `sys` has u as input and y as output.

[Open Help Browser](#) F1 to toggle focus; Escape to close

多子系统连接下的整体系统模型



parallel()函数

功能：实现两个子系统的串联
格式：

$\text{sys_new} = \text{parallel}(\text{sys1}, \text{sys2})$

$[\text{A}, \text{B}, \text{C}, \text{D}] = \text{parallel}(\text{A1}, \text{B1}, \text{C1}, \text{D1}, \text{A2}, \text{B2}, \text{C2}, \text{D2})$

$[\text{num}, \text{den}] = \text{parallel}(\text{num1}, \text{den1}, \text{num2}, \text{den2})$

其中，sys1和sys2为同类表示子系统

parallel

Parallel connection of two models

Syntax

```
parallel
sys = parallel(sys1,sys2)
sys = parallel(sys1,sys2,inp1,inp2,out1,out2)
sys = parallel(sys1,sys2,'name')
```

Description

parallel connects two model objects in parallel. This function accepts any type of model. The two systems must be either both continuous or both discrete with identical sample time. Static gains are neutral and can be specified as regular matrices.

`sys = parallel(sys1,sys2)` forms the basic parallel connection shown in the following figure.

This command equals the direct addition

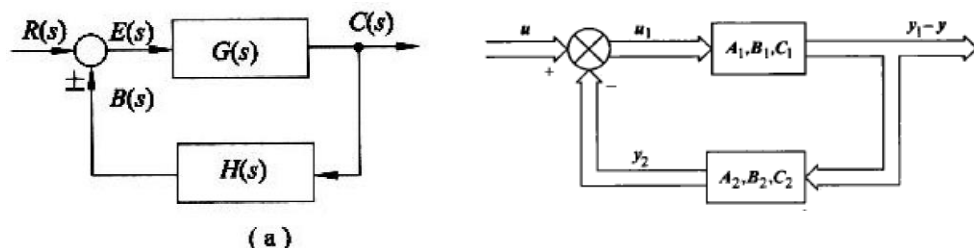
```
sys = sys1 + sys2
```

`sys = parallel(sys1,sys2,inp1,inp2,out1,out2)` forms the more general parallel connection shown in the following figure.

[Open Help Browser](#)

F1 to toggle focus; Escape to close...

多子系统连接下的整体系统模型



feedback()函数

功能：实现两个子系统的反馈连接
格式：

$\text{sys_new} = \text{feedback}(\text{sys1}, \text{sys2}, \text{sign})$

$[\text{A}, \text{B}, \text{C}, \text{D}] = \text{feedback}(\text{A1}, \text{B1}, \text{C1}, \text{D1}, \text{A2}, \text{B2}, \text{C2}, \text{D2}, \text{sign})$

$[\text{num}, \text{den}] = \text{feedback}(\text{num1}, \text{den1}, \text{num2}, \text{den2}, \text{sign})$

其中， sys1 和 sys2 为同类表示子系统， sign 为反馈类型（1~正反馈、-1~负反馈）

feedback
Feedback connection of two models [expand all in page](#)

Syntax
`sys = feedback(sys1, sys2)`

Description
`sys = feedback(sys1, sys2)` returns a model object `sys` for the negative feedback interconnection of model objects `sys1` and `sys2`.

The closed-loop model `sys` has `u` as input vector and `y` as output vector. The models `sys1` and `sys2` must be both continuous or both discrete with identical sample times. Precedence rules are used to determine the resulting model type (see [Rules That Determine Model Type](#)).

To apply positive feedback, use the syntax
`sys = feedback(sys1, sys2, +1)`

By default, `feedback(sys1, sys2)` assumes negative feedback and is equivalent to `feedback(sys1, sys2, -1)`.

Finally,
`sys = feedback(sys1, sys2, feedin, feedout)`
computes a closed-loop model `sys` for the more general feedback loop.

The vector `feedin` contains indices into the input vector of `sys1` and specifies which inputs `u` are involved in the feedback loop. Similarly,

[Open Help Browser](#) F1 to toggle focus; Escape to close

多子系统连接下的整体系统模型

例1-9：给定如下两个子系统，求其串联、并联、或负反馈连接后的新系统模型

$$W_1(s) = \frac{3s + 1}{s^2 + 3s + 2}, \quad W_2(s) = \frac{s + 4}{s + 2}$$

```
Editor - C:\Users\hp\Desktop\ex1_9.m
ex1_9.m x Untitled9* x +
1      %%%%%%%%%%%%%%
2      %%% Chuan-Ke Zhang
3      %%% 2021-10-06
4      %%% Example 1-9
5      %%% 子系统连接
6      %%%%%%%%%%%%%%
7
8 -    clc
9 -    clear
10
11 -   num1 = [3 1];
12 -   den1 = [1 3 2];
13 -   num2 = [1 4];
14 -   den2 = [1 2];
15
16 -   [num_s, den_s] = series(num1, den1, num2, den2)
17 -   [num_p, den_p] = parallel(num1, den1, num2, den2)
18 -   [num_f, den_f] = feedback(num1, den1, num2, den2, -1)
```

```
Command Window
num_s =
    0     3    13     4

den_s =
    1     5     8     4

num_p =
    1    10    21    10

den_p =
    1     5     8     4

num_f =
    0     3     7     2

den_f =
    1     8    21     8
```

$$W_s(s) = \frac{3s^2 + 13s + 4}{s^3 + 5s^2 + 8s + 4}$$

$$W_p(s) = \frac{s^3 + 10s^2 + 21s + 10}{s^3 + 5s^2 + 8s + 4}$$

$$W_f(s) = \frac{3s^2 + 7s + 2}{s^3 + 8s^2 + 21s + 8}$$

多子系统连接下的整体系统模型

例1-9：给定如下两个子系统，求其串联、并联、或负反馈连接后的新系统模型

$$\begin{cases} \dot{x}_1 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -4 & -8 & -5 \end{bmatrix} x_1 + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u_1 \\ y_1 = [1 \ 0 \ 0] x_1 \end{cases} \quad \begin{cases} \dot{x}_2 = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix} x_2 + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_2 \\ y_2 = [1 \ 0] x_2 \end{cases}$$

```
Editor - C:\Users\hp\Desktop\ex1_10.m
ex1_9.m  ex1_10.m  +
3      %%% 2021-10-06
4      %%% Example 1-10
5      %%% 子系统连接
6      %%%%%%%%%%%%%%%%%%%%%%%%%
7
8      clc
9      clear
10
11     A1 = [0 1 0; 0 0 1; -4 -8 -5];
12     B1 = [0; 0; 1];
13     C1 = [1 0 0];
14     D1 = 0;
15     A2 = [0 1; -2 -3];
16     B2 = [0; 1];
17     C2 = [1 0];
18     D2 = 0;
19
20     [A_s,B_s,C_s,D_s] = series(A1,B1,C1,D1,A2,B2,C2,D2)
21     [A_p,B_p,C_p,D_p] = parallel(A1,B1,C1,D1,A2,B2,C2,D2)
22     [A_f,B_f,C_f,D_f] = feedback(A1,B1,C1,D1,A2,B2,C2,D2,-1)
```

Command Window

A_s =

0	1	0	0	0
-2	-3	1	0	0
0	0	0	1	0
0	0	0	0	1
0	0	-4	-8	-5

B_s =

0
0
0
0
1

C_s =

1	0	0	0	0
---	---	---	---	---

D_s =

0

fx

Command Window

A_p =

0	1	0	0	0
0	0	1	0	0
-4	-8	-5	0	0
0	0	0	0	1
0	0	0	-2	-3

B_p =

0
0
1
0
1

C_p =

1	0	0	1	0
---	---	---	---	---

D_p =

0

fx

Command Window

A_f =

0	1	0	0	0
0	0	1	0	0
-4	-8	-5	-1	0
0	0	0	0	1
1	0	0	-2	-3

B_f =

0
0
1
0
0

C_f =

1	0	0	0	0
---	---	---	---	---

D_f =

0

fx

本部分实验小结

□ MATLAB使用

- 如何在MATLAB中 编写代码、调试运行、查看结果、等常规操作
- 如何利用simulink搭建系统结构框图

□ 系统模型

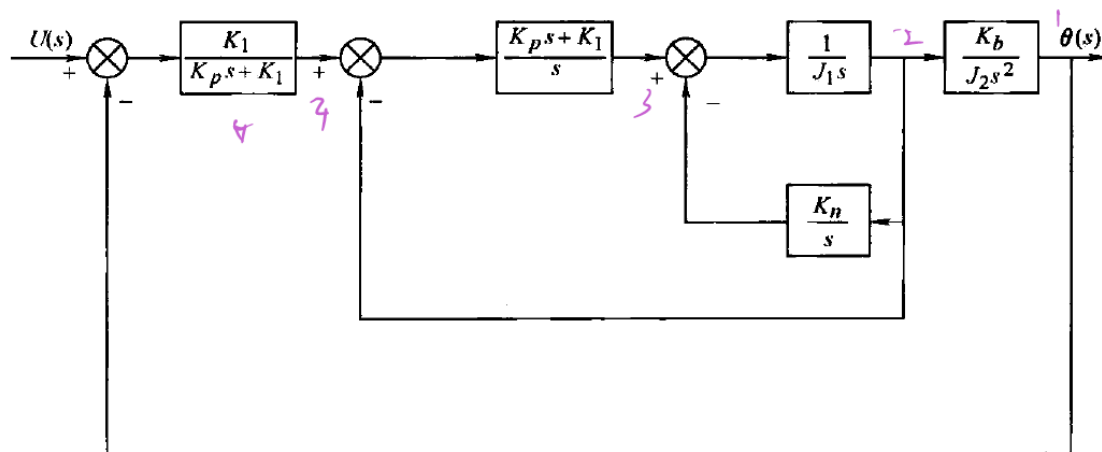
- 如何用MATLAB表示各类系统数学模型
- 如何用MATLAB实现各类模型之间的转化
- 如何用MATLAB合并多个子系统生成整体系统模型

本部分实验练习题

作业一：

- 利用MATLAB给出如下系统的 传递函数模型、状态空间模型
- 利用simulink搭建各类模型，并观察其单位阶跃响应

（相关参数自由取值）



- 利用simulink搭建如下系统结构框图
- 选初始条件[-0.2; 0.3; 0.7]，观察状态响应

the following representation of Chua's circuit systems:

$$\begin{cases} \dot{x}_1(t) = a [x_2(t) - h(x_1(t))] \\ \dot{x}_2(t) = x_1(t) - x_2(t) + x_3(t) \\ \dot{x}_3(t) = -bx_2(t) \\ p(t) = x_1(t) \end{cases} \quad (23)$$

with the nonlinear characteristics of Chua's diode

$$h(x) = m_1 x_1(t) + \frac{1}{2}(m_0 - m_1)(|x_1(t) + c| - |x_1(t) - c|) \quad (24)$$

and parameters $a = 9$, $b = 14.28$, $c = 1$, $m_0 = -(1/7)$, $m_1 = 2/7$, and $c = 1$.

本部分实验练习题

□ 作业二：

- 尝试利用MATLAB完成教材第1章习题，1-5、1-6、1-7、1-9、1-10、1-11
- 至少完成2题

谢谢！