



线性系统分析与设计： 实验PART II

基于LMI与SIMULINK的
系统分析、设计与仿真



主要内容

- 基于LMI的系统稳定性分析
 - 基于LMI的系统稳定判据
 - 基于YALMIP工具箱的LMI条件验证
- 基于LMI的系统镇定设计
 - 基于LMI的系统镇定条件及控制器计算公式
 - 基于YALMIP工具箱的设计
- 基于SIMULINK的系统响应曲线绘制
 - 基于状态空间方程的系统SIMULINK搭建
 - 基于SIMULINK运行数据的曲线绘制

基于LMI的系统稳定性分析

➤ 基于LMI的系统稳定判据

对于如下线性定常系统

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x}$$

系统平衡态 $\mathbf{x}_e = 0$ 为渐近稳定的充要条件是：存在对称

矩阵 \mathbf{P} 满足如下线性矩阵不等式条件：

$$\mathbf{P} > 0, \quad \mathbf{A}^T \mathbf{P} + \mathbf{P} \mathbf{A} < 0$$

$$V(\mathbf{x}) = \mathbf{x}^T \mathbf{P} \mathbf{x}$$

如何求解上述线性矩阵不等式？

基于LMI的系统稳定性分析

➤ 基于YALMIP工具箱的LMI条件求解

□ YALMIP工具箱



网页 资讯 贴吧 知道 视频

百度为您找到相关结果约128,000个

 您可以仅查看：[英文结果](#)

[YALMIP](#)

查看此网页的中文翻译，请点击
A question on the [YALMIP](#) forum
squares solutions which really a
<https://yalmip.github.io/> ▼ - [百度快](#)

 为您推荐：[cplex matlab](#) [matlab trace](#)

[Yalmip使用学习 - 简书](#)

2018年1月23日 - [yalmip](#)学习 0.

□ MATLAB加载工具箱



网页 资讯 贴吧 知道 视频

百度为您找到相关结果约2,720,000个

[给Matlab添加工具箱Toolbox的方法](#)

2013年11月20日 - 虽然庞大的Matlab已经有的要求,常常需要自己添加Toolbox。下面以<https://blog.csdn.net/u0127364...> ▼ - [百度快](#)

[Matlab添加toolbox - CongliYin的博文](#)

2017年8月10日 - 由于科研需要,为matlab添具包,它专门用于简化最先进的黎曼优化算法
https://blog.csdn.net/sinat_20... ▼ - [百度快](#)

[MATLAB2016添加工具箱toolbox方](#)

2016年11月11日 - 自然语言处理入门 - 自然语言处理入门

4

基于LMI的系统稳定性分析

实例分析

```

1  %%%%%%%%%%%%%%
2  %%% Chuan-Ke Zhang
3  %%% 2018-01-19
4  %%% 验证系统的稳定性程序
5  %%%%%%%%%%%%%%
6
7  -   clc; clear
8
9  %%% 系统参数
10 -   A = [1 2 4; 1 1 1; 0 2 1];
11
12 %%% 描述待求的LMI
13 -   P = sdpvar(3,3,'symmetric'); % 给出待求矩阵
14 -   Fcond = [P>0,A'*P+P*A<0]; % 列出所有待求LMI
15
16 %%% 求解LMI
17 -   ops = sdpsettings('verbose',0,'solver','sedumi'); % 设置求解环境
18 -   diagnostics = solvesdp(Fcond,[],ops); % 迭代求解
19 -   [m p] = checkset(Fcond); % 返回求解结果
20 -   tmin = min(m); % 验证是否满足
21
22 -   if tmin > 0
23 -       disp('System is stable') % 结论输出
24 -   else
25 -       disp('System is unstable') % 结论输出
26 -   end
    
```

$$\dot{x}(t) = \begin{bmatrix} 1 & 2 & 4 \\ 1 & 1 & 1 \\ 0 & 2 & 1 \end{bmatrix} x(t)$$

$P = \text{sdpvar}(n,n,'symmetric');$
 % P 为 n 维* n 维对称矩阵
 $Q = \text{sdpvar}(n,m,'full');$
 % Q 为 n 维* m 维矩阵

$$P > 0, \quad A^T P + P A < 0$$

基于LMI的系统稳定性分析

□ 练习题1：利用LMI判据分析系统稳定性

$$\dot{x}(t) = \begin{bmatrix} 1 & 3 & 5 & 9 \\ 6 & 2 & 1 & 1 \\ 3 & 7 & 4 & 3 \\ 4 & 3 & 7 & 7 \end{bmatrix} x(t)$$

□ 练习题2：求解如下LMI

$$A = \begin{bmatrix} 5 & 3 \\ 1 & 2 \end{bmatrix}, B = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} A^T P + PA + Q & PB \\ B^T P & -R \end{bmatrix} < 0, P > 0, Q > 0, R > 0$$

基于LMI的系统镇定设计

➤ 基于LMI的系统镇定条件及控制器计算公式

对于如下线性定常系统及状态反馈控制器

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ u(t) = Kx(t) \end{cases} \quad \longrightarrow \quad \dot{x}(t) = (A + BK)x(t)$$

系统可由状态反馈控制器镇定的条件是：存在对称矩阵 L ，矩阵 V ，满足如下LMI：

$$\begin{cases} L > 0 \\ AL + LA^T + BV + V^T B^T < 0 \end{cases}$$

$$K = VL^{-1}$$

$$\longleftarrow \begin{cases} P > 0 \\ PA + A^T P < 0 \end{cases}$$

基于LMI的系统镇定设计

实例分析

```
1 %%%%%%%%%%%%%%%
2 %%% Chuan-Ke Zhang
3 %%% 2018-01-19
4 %%% 设计系统镇定控制程序
5 %%%%%%%%%%%%%%%
6
7 clc; clear
8
9 %%% 系统参数
10 A = [1 2 4; 1 1 1; 0 2 1];
11 B = [1; 2; 1];
12
13 %%% 描述待求的LMI
14 L = sdpvar(3,3,'symmetric'); % 给出待求矩阵
15 V = sdpvar(1,3,'full'); % 给出待求矩阵
16 Fcond = [L>0,A*L+L*A'+B*V+V'*B'<0]; % 列出所有待求LMI
17
18 %%% 求解LMI
19 ops = sdpsettings('verbose',0,'solver','sedumi'); % 设置求解环境
20 diagnostics = solvesdp(Fcond,[],ops); % 迭代求解
21 [m p] = checkset(Fcond); % 返回求解结果
22 tmin = min(m); % 验证是否满足
23
24 if tmin > 0
25     Vh = double(V);
26     Lh = double(L);
27     disp('System can be stabilized, and the control gain is given as') % 结论输出
28     K = Vh*inv(Lh)
29 else
30     disp('System cannot be stabilized using state-feedback controller') % 结论输出
31 end
```

$$\begin{cases} \dot{x}(t) = \begin{bmatrix} 1 & 2 & 4 \\ 1 & 1 & 1 \\ 0 & 2 & 1 \end{bmatrix} x(t) + \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} u(t) \\ u(t) = Kx(t) \end{cases}$$

$$\begin{cases} L > 0 \\ AL + LA^T + BV + V^T B^T < 0 \end{cases}$$

$$K = VL^{-1}$$



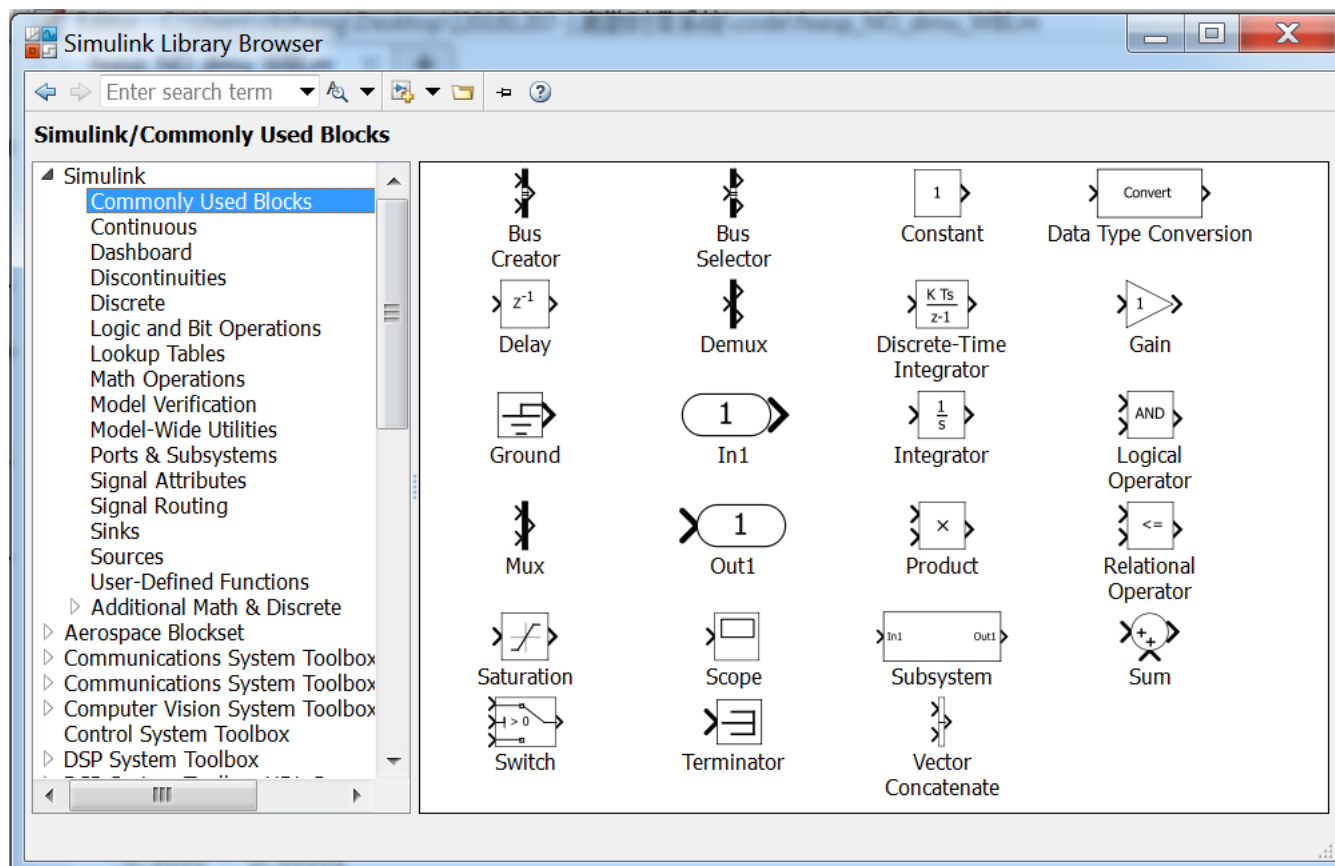
基于LMI的系统镇定设计

□ 练习题3：设计状态反馈控制器镇定如下系统

$$\begin{cases} \dot{x}(t) = \begin{bmatrix} 2 & 1 \\ 0 & 1 \end{bmatrix} x(t) + \begin{bmatrix} 1 & 3 \\ 1 & 2 \end{bmatrix} u(t) \\ u(t) = Kx(t) \end{cases}$$

基于SIMULINK的系统响应曲线绘制

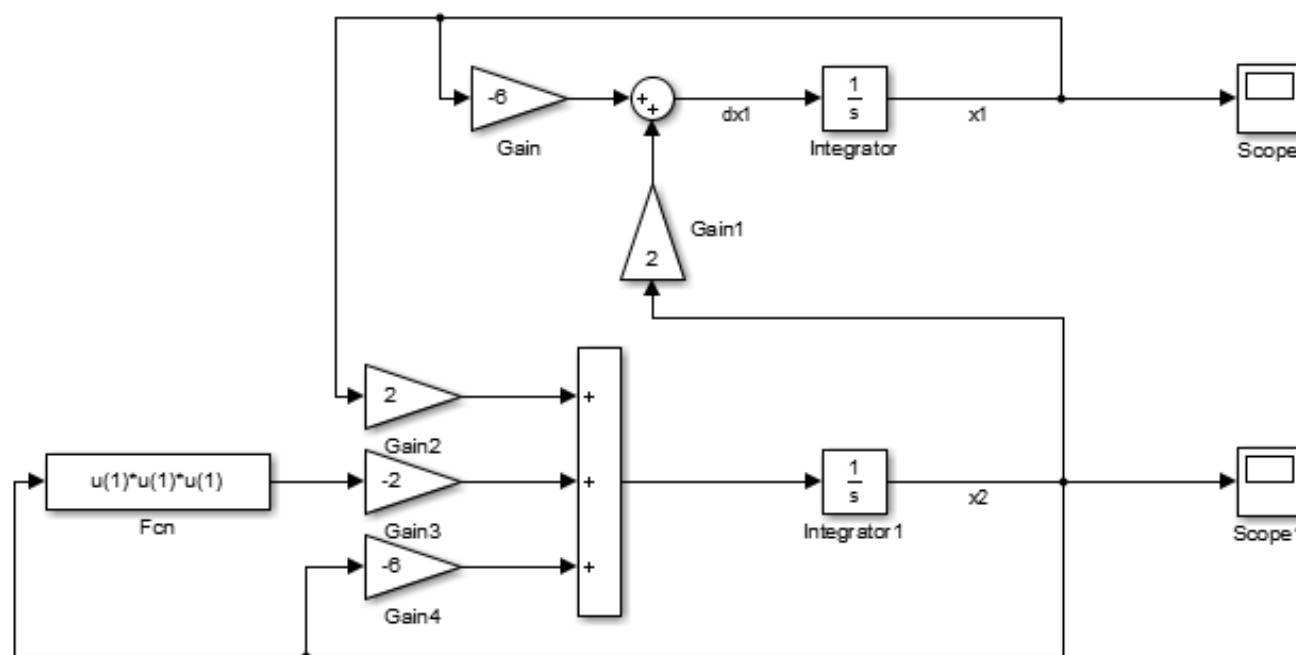
➤ SIMULINK LAB



基于SIMULINK的系统响应曲线绘制

➤ 基于状态空间方程的系统SIMULINK搭建(1)

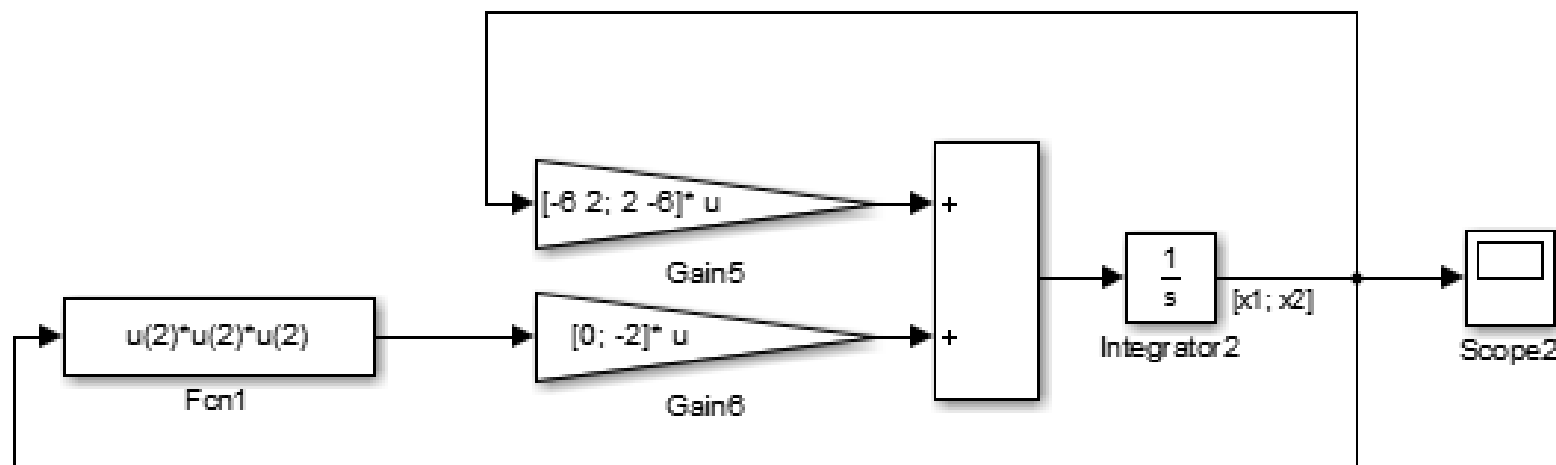
$$\begin{cases} \dot{x}_1 = -6x_1 + 2x_2 \\ \dot{x}_2 = 2x_1 - 6x_2 - 2x_2^3 \end{cases}$$



基于SIMULINK的系统响应曲线绘制

➤ 基于状态空间方程的系统SIMULINK搭建 (2)

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



基于SIMULINK的系统响应曲线绘制

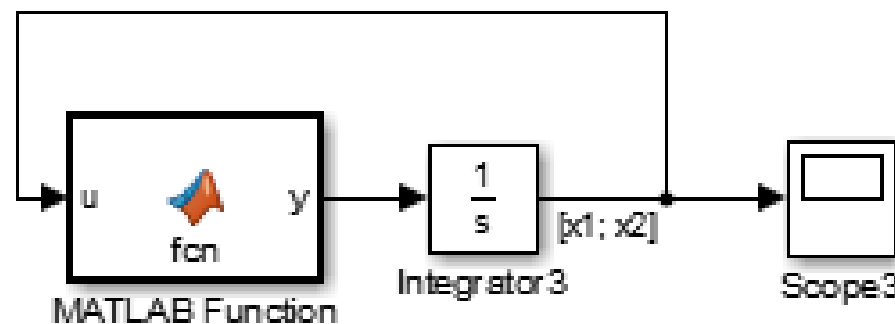
➤ 基于状态空间方程的系统SIMULINK搭建 (3)

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

MATLAB Function

```

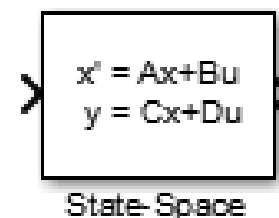
1 function y = fcn(u)
2     %#codegen
3
4     % x1 = u(1);
5     % x2 = u(2);
6     % dx1 = -6*x1 + 2*x2;
7     % dx2 = 2*x1 - 6*x2 - 2*x2^3;
8     % y = [dx1; dx2];
9
10 y = [-6 2; 2 -6]*u + [0; -2]*u(2)^3;
```



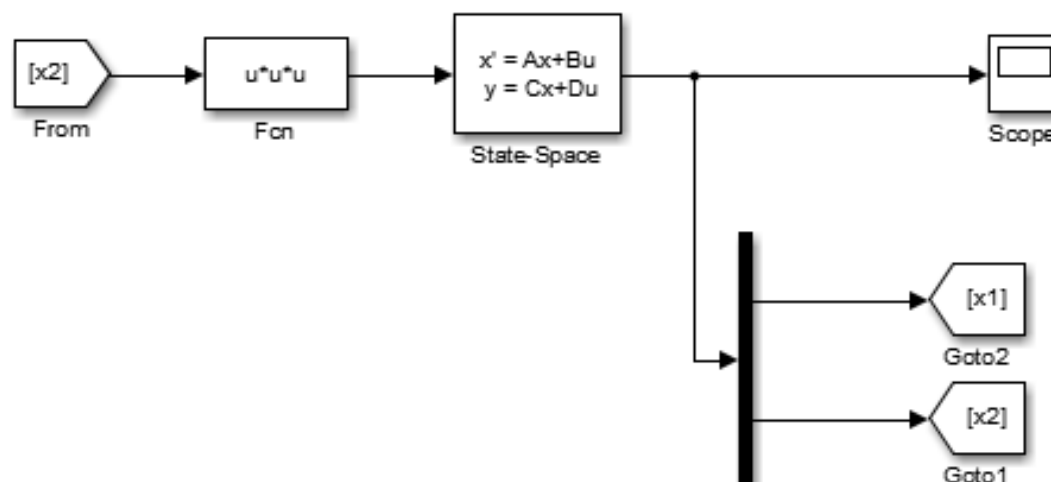
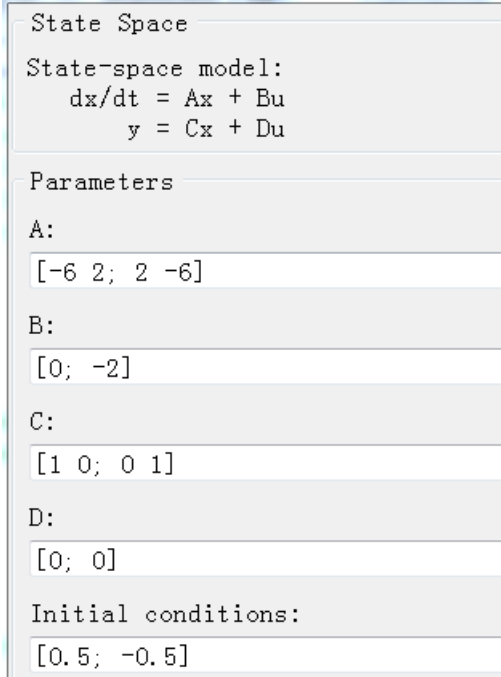
基于SIMULINK的系统响应曲线绘制

➤ 基于状态空间方程的系统SIMULINK搭建 (3)

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



Function Block Parameters: State-Space





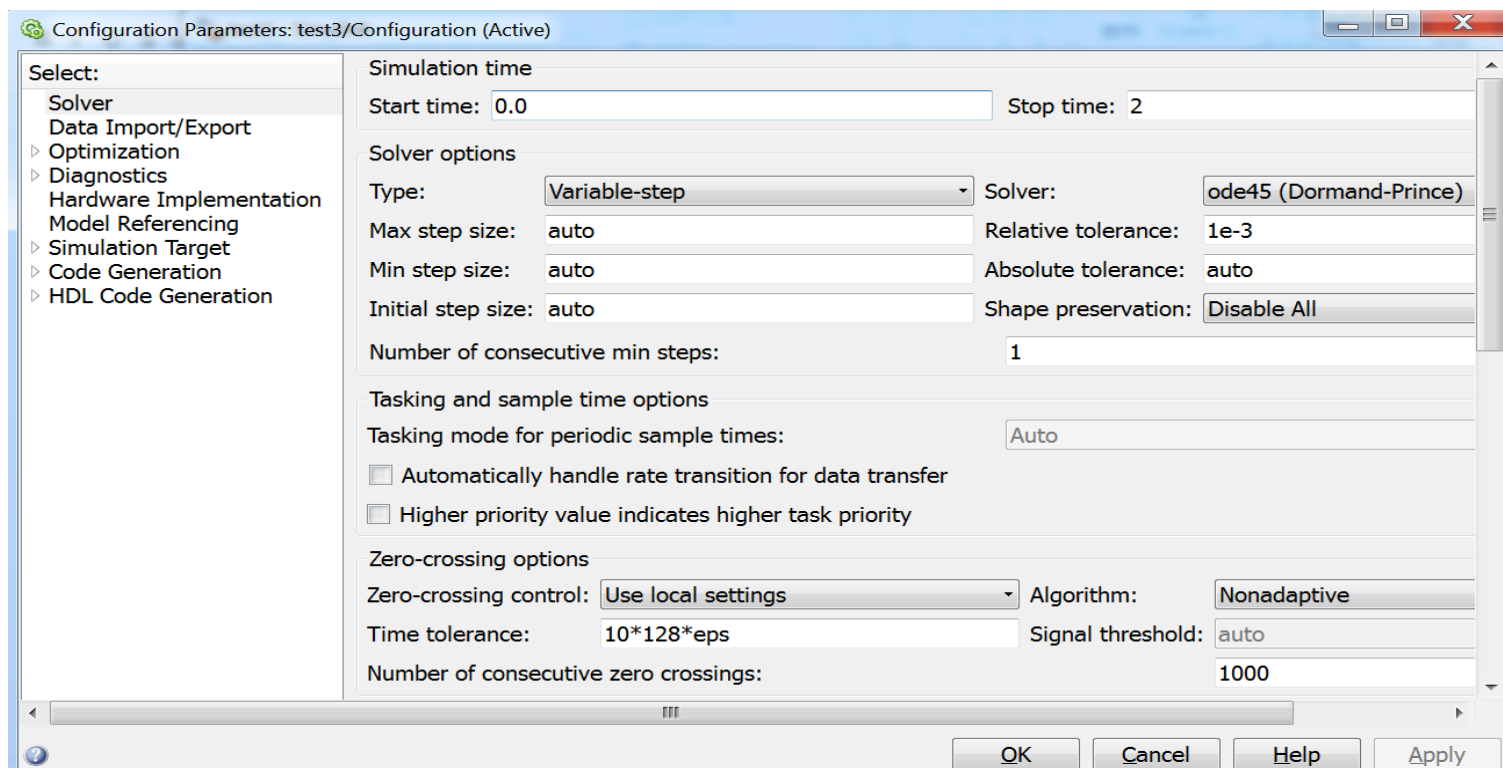
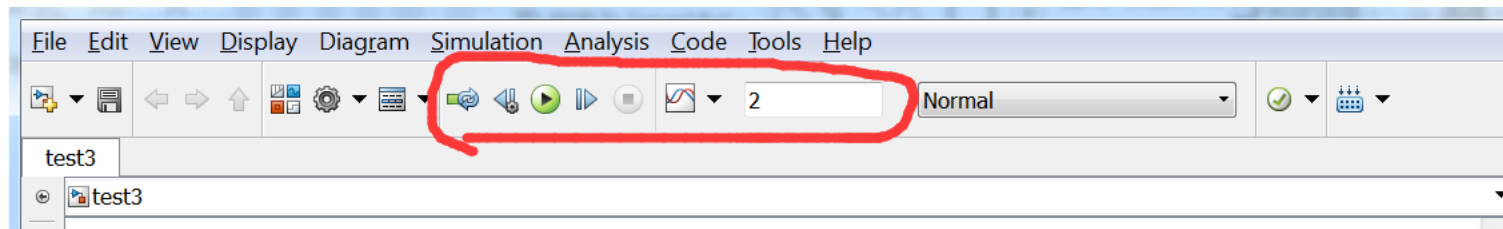
基于SIMULINK的系统响应曲线绘制

➤ 基于状态空间方程的系统SIMULINK搭建

- ❑ S-函数
- ❑ 创建子系统
- ❑ 封装子系统
- ❑ 模型库
- ❑

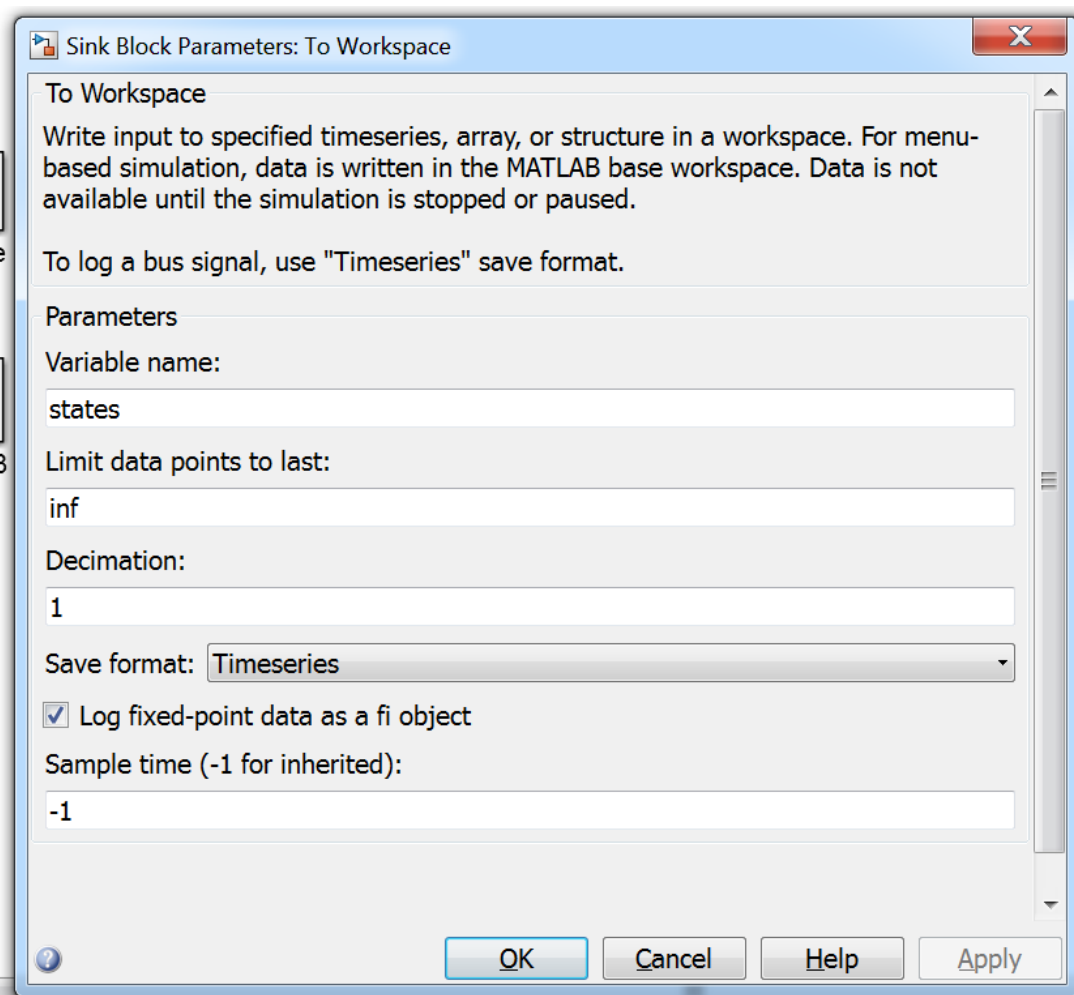
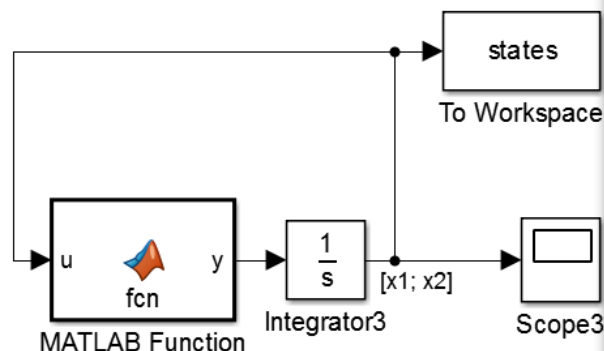
基于SIMULINK的系统响应曲线绘制

➤ 基于SIMULINK运行数据的曲线绘制：运行simulink模块



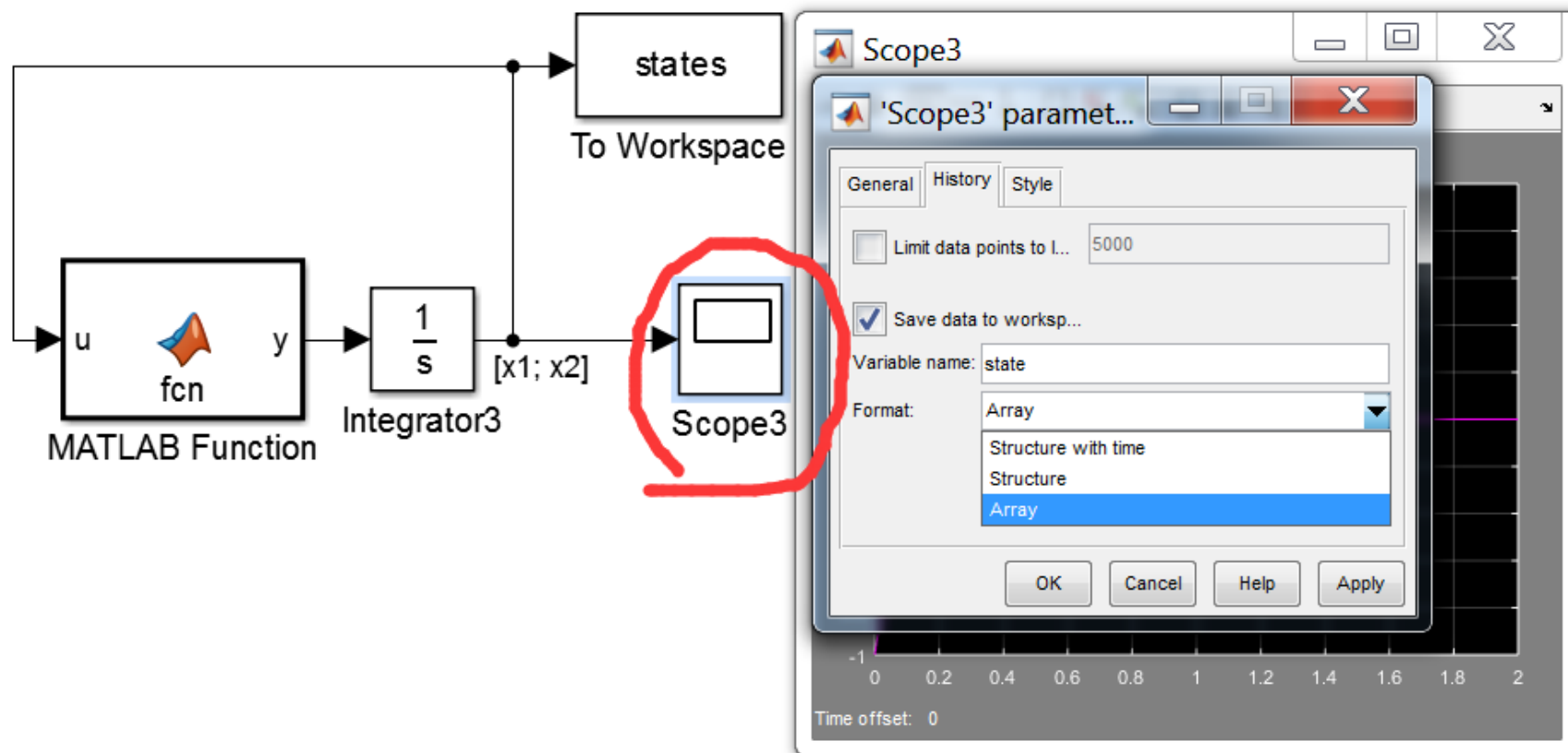
基于SIMULINK的系统响应曲线绘制

➤ 基于SIMULINK运行数据的曲线绘制：导入WORKSPACE（1）



基于SIMULINK的系统响应曲线绘制

➤ 基于SIMULINK运行数据的曲线绘制：导入WORKSPACE（2）



基于SIMULINK的系统响应曲线绘制

➤ 基于SIMULINK运行数据的曲线绘制：WORKSPACE中数据

The image displays the MATLAB environment during a simulation. The top-left window shows the **Workspace** with variables `states` (1x1 struct) and `tout` (52x1 double). The top-right window is the **Editor - MATLAB Function**, showing the `states.signals` tab with a 1x1 struct containing `time` (52x1 double), `signals` (1x1 struct), and `blockName` ('test3/To Workspace'). The bottom-left window shows the `states.signals.values` tab with a 52x2 double matrix. The bottom-right window shows a preview of the `values` data as a 52x2 double matrix.

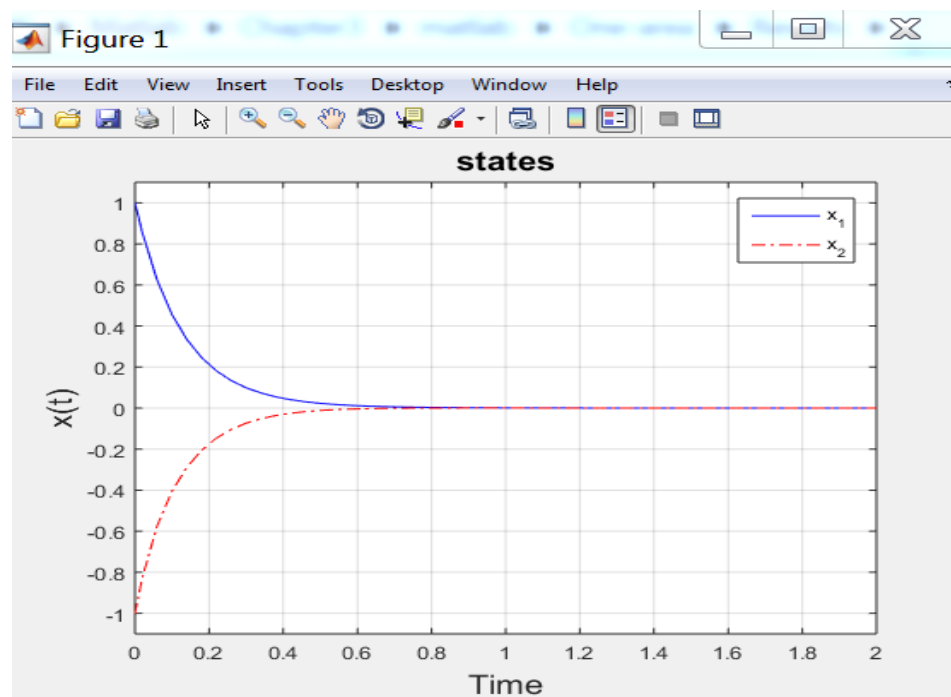
	1	2	3	4
1	1	-1		
2	0.8521	-0.8231		
3	0.6217	-0.5723		
4	0.4549	-0.4044		
5	0.3336	-0.2876		

```
>> plot(states.time, states.signals.values(:, 1))
```

基于SIMULINK的系统响应曲线绘制

➤ 基于SIMULINK运行数据的曲线绘制：绘图（1）

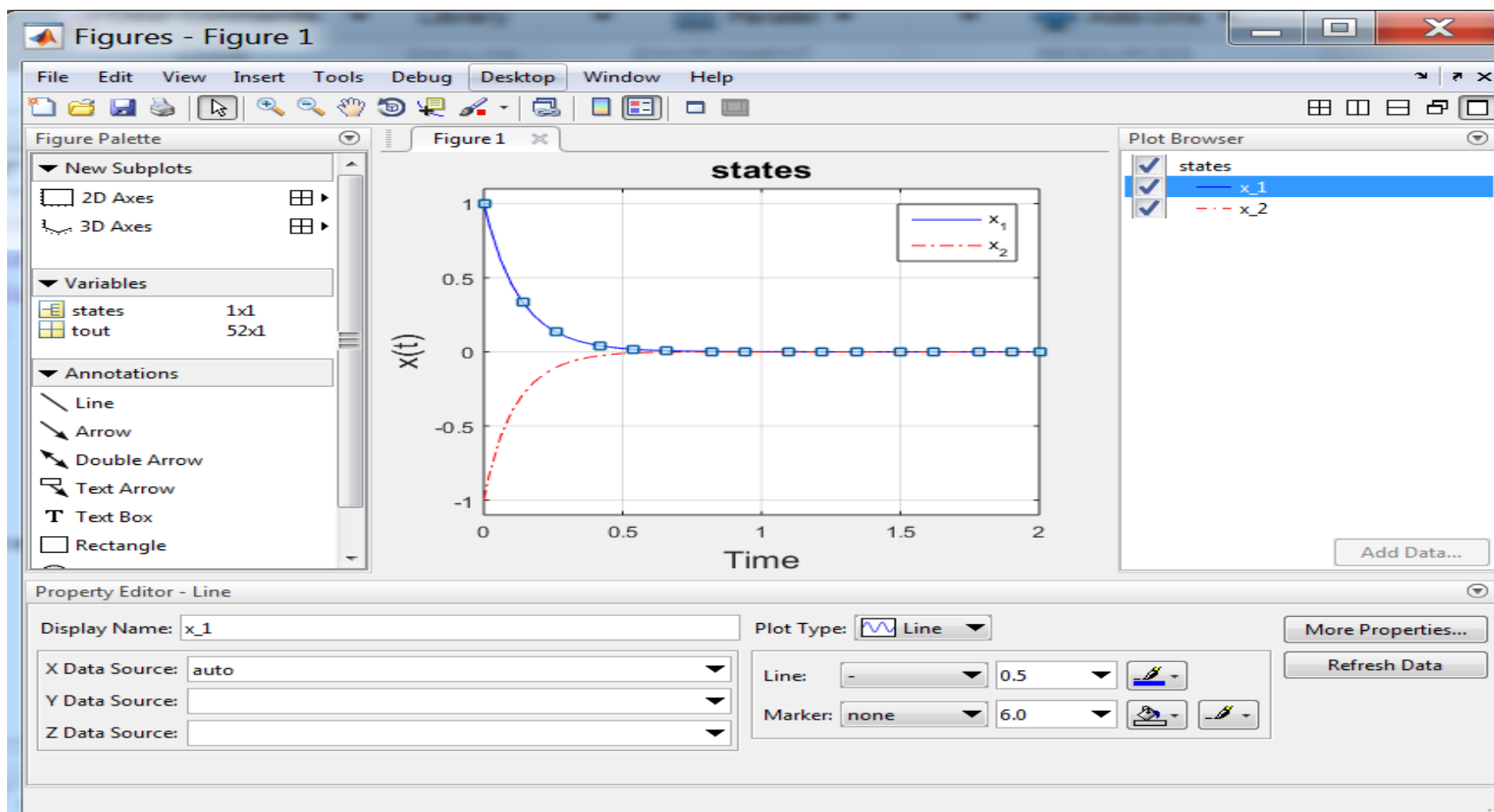
```
figure(1)
set(gca,'FontSize',15,'FontName','Times New Roman')
plot(states.time,states.signals.values(:,1),'b-',...
      states.time,states.signals.values(:,2),'r-.')
xlabel('Time','FontSize',15);
ylabel('x(t)','FontSize',15);
title('states','FontSize',15)
axis([0 2 -1.1 1.1]);
grid on
legend('x_1','x_2')
```



基于SIMULINK的系统响应曲线绘制

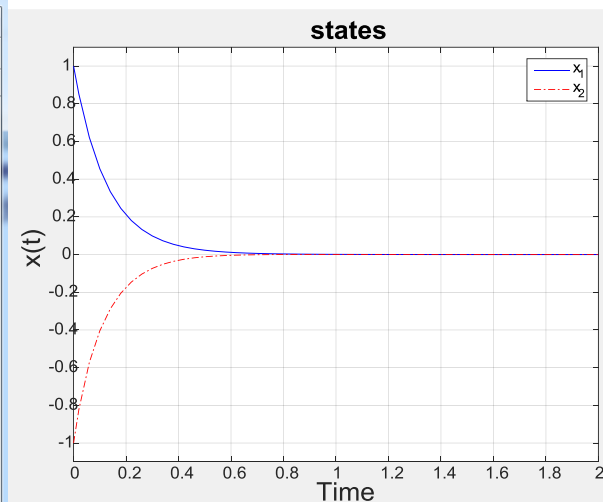
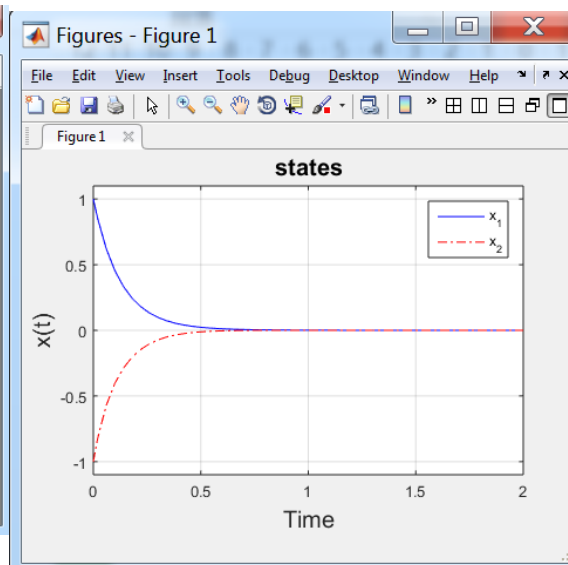
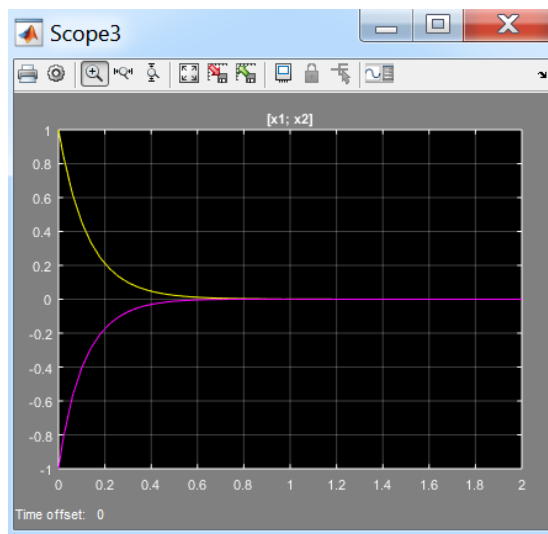
➤ 基于SIMULINK运行数据的曲线绘制：绘图（2）

```
plot(states.time, states.signals.values(:, 1), states.time, states.signals.values(:, 2))
```



基于SIMULINK的系统响应曲线绘制

➤ 基于SIMULINK运行数据的曲线绘制：取出曲线结果



基于SIMULINK的系统响应曲线绘制

- 练习题4：利用SIMULINK仿真如下系统，并绘出其状态响应曲线、相图；初始条件选 $[-0.2; 0.3; 0.7]$

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 $c = 1$. In this case, Chua's system will obtain the

SIMULINK的案例库

Help

Inverted Pendulum with Animation

Contents

Documentation

- Simulink
 - Getting Started with Simulink
 - Simulink Examples
 - Simulink Examples
 - Getting Started With the Simulink Environment (3 min, 27 sec)
 - Loading and Logging Data (3 min, 51 sec)
 - Creating and Masking Subsystems (2 min, 20 sec)
 - Visualizing Results in Simulink (2 min, 52 sec)
 - Using Solvers (2 min, 27 sec)
 - Using MATLAB Function block to incorporate MATLAB Code in Simulink Models (2 min, 11 sec)
 - Creating Libraries (2 min, 18 sec)
 - Using Configurable Subsystems (1 min, 4 sec)
 - Simulation of a Bouncing Ball
 - Single Hydraulic Cylinder Simulation
 - Thermal Model of a House
 - Approximating Nonlinear Relationships: Type S Thermocouple
 - Digital Waveform Generation: Approximating a Sine Wave
 - Accurate Zero-Crossing Detection
 - Spiral Galaxy Formation Simulation Using MATLAB Function Blocks
 - Modulo-4 Counter Using Flip-Flops
 - Counters Using Conditionally Executed Subsystems
 - Friction Model with Hard Stops
 - State Events
 - Bang-Bang Control Using Temporal Logic
 - Inverted Pendulum with Animation**
 - Double Spring Mass System
 - Tank Fill and Empty with Animation
 - Simulating Systems with Variable Transport Delay Phenomena
 - Modeling a Foucault Pendulum

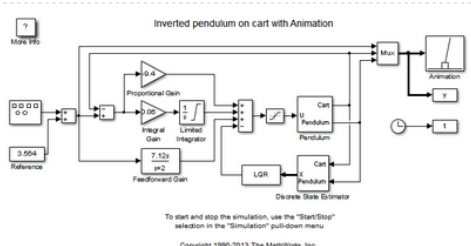
Search Documentation

Simulink Simulink Examples General Applications

Inverted Pendulum with Animation

This example shows how to model an inverted pendulum. The animation is created using MATLAB® Handle Graphics®. The animation block is a masked S-function. For more information, use the context menu to look under the Animation block's mask and open the S-function for editing.

Inverted pendulum on cart with Animation

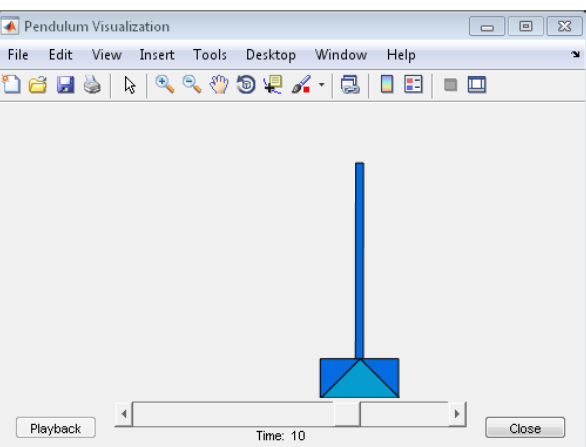


To start and stop the simulation, use the "Start/Stop" selection in the "Simulation" pull-down menu.

Copyright 1990-2013 The MathWorks, Inc.

Pendulum Visualization

File Edit View Insert Tools Desktop Window Help



24