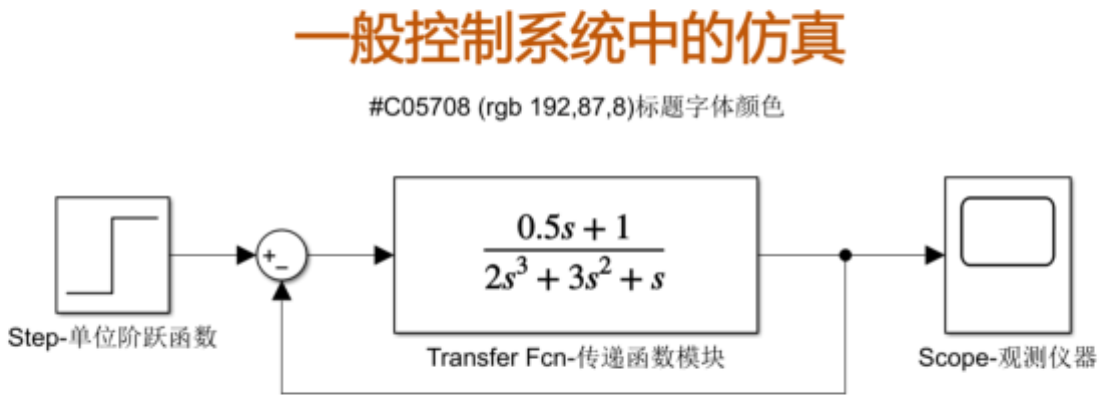
一般控制系统仿真



简单系统仿真

对于满足下列条件的系统，我们称之为**简单系统**：

（1）系统某一时刻的输出直接且唯一依赖于该时刻的输入量；

（2）系统对同样的输入，其输出响应不随时间的变化而变化；

（3）系统中不存在输入的状态量，所谓的状态量是指系统输入的微分。

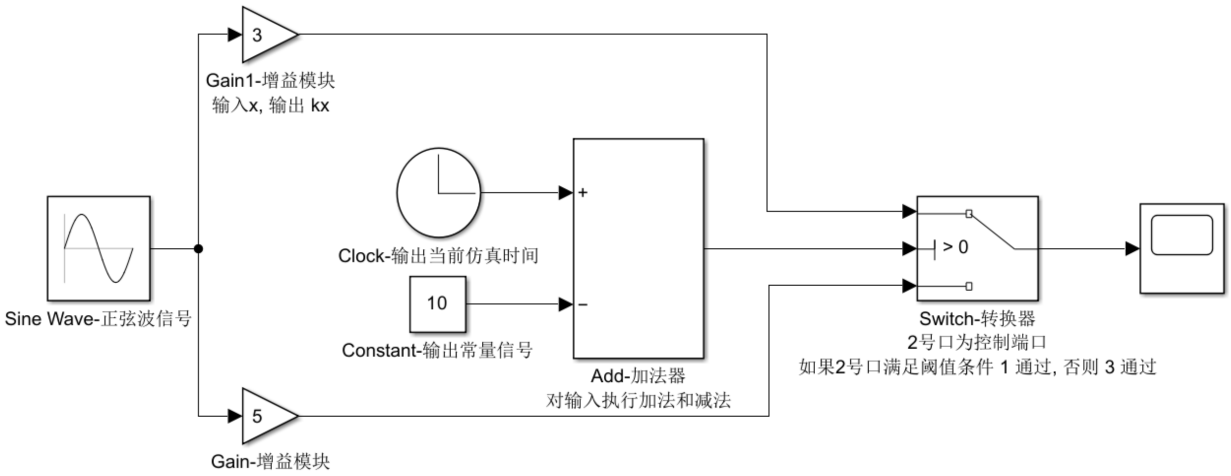
$$y = f(u(t))$$

【例4-1】对于下述的简单系统：

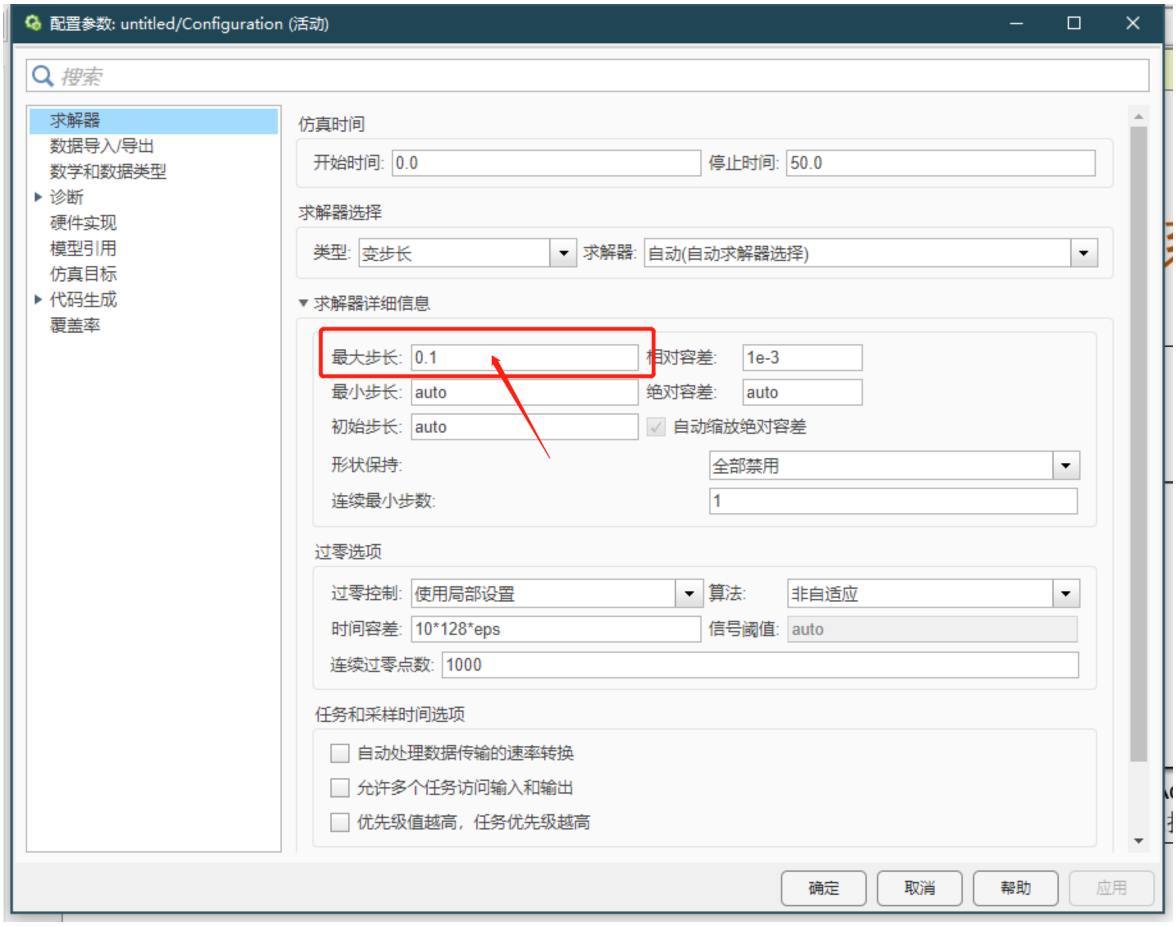
$$y=\begin{cases}3u(t) & , \quad t>10\\ 5u(t) & , \quad t\leqslant 10\end{cases}$$

其中u(t)为系统输入，y(t)为系统输出。建立此简单系统的模型并进行仿真分析。

简单系统仿真



> ❗❗❗
>
> **仿真步长设置问题**
>
> 对于简单系统，由于系统中并不存在状态变量，因此每一次计算都应该是准确的（不考虑数据截断误差）。
>
> 在使用Simulink对简单系统进行仿真时，不论采用何种求解器，Simulink总是在仿真过程中选用最大的仿真步长。如果仿真时间区间较长，而且最大步长设置采用默认取值auto，则会导致系统在仿真时使用较大的步长，因为Simulink的仿真步长是通过下面的式子得到的：
>
> $$h = \frac{t_{fianl} - t_{last}}{50}$$
>
> 对于仿真时间 50s 的仿真，步长为1，可能会导致系统仿真输出曲线的不光滑。
>
> **解决办法** 设置最大步长为 0.1



离散系统仿真

离散系统，是指系统的输入与输出仅在离散的时间上取值，而且离散的时间具有相同的时间间隔。

离散系统 对于满足下列条件的系统，我们称之为离散系统：

（1）系统每隔固定的时间间隔才更新一次，即系统的输入与输出每隔固定的时间间隔便改变一次。固定的时间间隔称为系统的**采样时间；**

（2）**系统的输出依赖于系统当前的输入、以往的输入与输出；**

（3）离散系统具有离散的状态。**状态**指的是系统前一时刻的输出量。

离散系统的数学描述：

输入变量 $u(nT_s), n = 0,1,2,...$ 系统采样时间为 T_s，n 为采样时刻。

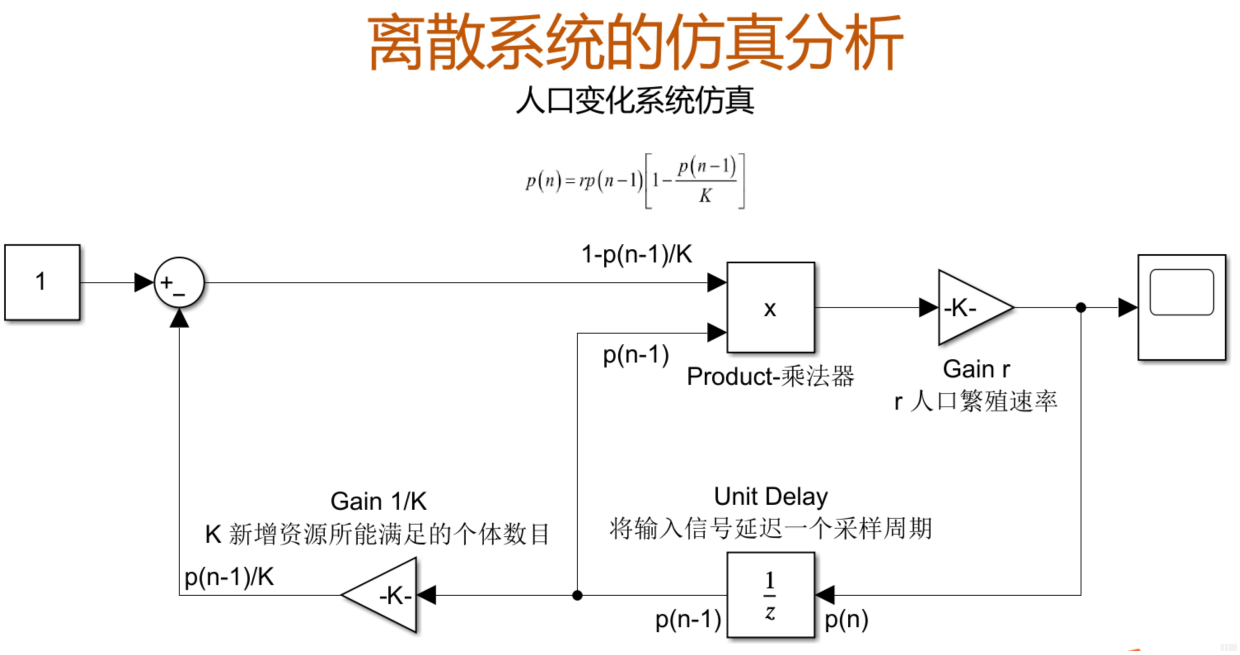
由于 T_s 为一固定值，系统输入简记为 $u(n)$，输入简记为 $y(n)$，因此离散系统的数学模型为：

$$y(n) = f(u(n), u(n-1), \dots; y(n-1), y(n-2), \dots)$$

> !!!
>
> - 对于离散系统，系统的输出不仅和系统当前的输入有关，而且还和系统以往的输入与输出有关；
> - 对于一般的系统而言，系统均是从某一时刻开始运行的。因此，在离散系统中，==系统初始状态的确定是非常关键的。==

仿真参数设置

- 求解器设置：对离散系统进行仿真需要使用离散求解器，即discrete。
- 仿真步长设置：选定定步长求解器，即Fix step。



Simulink 命令行仿真

> 学习目标：
>
> （1）掌握命令行方式建立系统模型；
>
> （2）掌握 Simulink 与 MATLAB 的接口；
>
> （3）熟悉命令行方式进行动态系统仿真的过程。

与图形建模方式操作控制相比，命令行方式仿真具有如下优点：

（1）==自动地重复运行仿真；==

（2）==在仿真过程中动态调整参数；==

（3）==分析不同输入信号下的系统响应；==

（4）==进行快速仿真。==

使用命令行方式建立系统模型

命　令	功　能	命　令	功　能
new_system	建立一个新的 Simulink 系统模型	set_param	设置系统模型中的参数
open_system	打开一个已存在的 Simulink 系统模型	add_param	增加系统模型中的参数
load_system	载入一个已存在的 Simulink 系统模型	delete_param	删除系统模型中的参数
save_system	保存 Simulink 系统模型	gcb	获得当前模块的路径名
close_system	关闭 Simulink 系统模型	gcs	获得当前系统的路径名
find_system	查找 Simulink 系统模型、模块、连线及注释	gcbh	获得当前模块的操作句柄
add_block	在系统模型中加入指定模块	getfullname	获得模块或连线的路径名
replace_block	替代系统模型中的指定模块	dbroot	获得最上层系统模型的名称
delete_block	从系统模型中删除指定模块	dbclose	关闭所有 Simulink 模型窗口
add_line	在系统模型中加入指定连线	dbIsLoaded	检查系统模型是否在内存中
delete_line	从系统模型中删除指定连线	simulink	打开 Simulink 模块库浏览器
get_param	获取系统模型中的参数		

命令行命令

使用语法

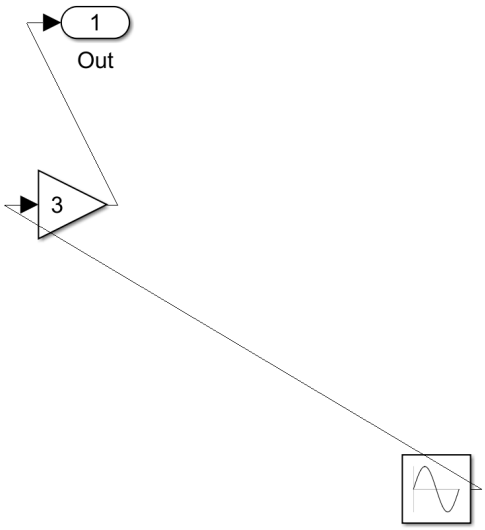
```
%% 系统模型命令
% 新建一个空系统
new_system(sys)
new_system(sys, 'Library')
% 打开一个已存在的Simulink系统模型
open_system('sys')
open_system('blk')
open_system('blk', 'force')
% 保存Simulink系统模型
save_system
save_system(sys)
save_system(sys, newsysname)
save_system(sys, newsysname.slx)
% 载入系统模型
load_system('sys')
% 关闭系统模型
close_system
close_system('sys')
close_system('sys', saveflag)
close_system('sys', 'newname')
% 搜索指定系统或子系统
find_system(sys, 'c1', cv1, 'c2', cv2, ...'p1', v1, 'p2', v2, ...)


%% 模块命令
% 添加模块
add_block('src', 'dest')
add_block('src', 'dest', 'param1', value1, ...)
% 删除模块
delete_block('blk')
```

```
% 替换模块
replace_block('sys', 'old_blk', 'new_blk')
replace_block('sys', 'parameter', 'value', ..., 'blk')

%% 连线命令
h = add_line('sys', 'oport', 'iport') % 在系统模型sys中从输出端口oport添加连线至输入端口
iport,并返回连线的句柄
delete_line('sys', 'outPort', 'inPort')

%% 参数命令
set_param(object, param1, value1, ..., paramN, valueN)
ParameterValue = get_param(Object, ParameterName)
ParameterValues = get_param(ObjectCellArray, ParameterName)
ParameterValue = get_param(ObjectHandle, ParameterName)
ParameterValue = get_param(0, ParameterName)
ParameterStruct = get_param(Object, 'ObjectParameters')
ParameterCellArray = get_param(Object, 'DialogParameters')

add_param('sys', 'parameter1', value1, 'parameter2', value2, ...)
delete_param('sys', 'parameter1', 'parameter2', ...)
```

示例

```
% 模型名称
systemName = 'UsingTheCommandLineForDynamicSystemSimulation001';

% 新建一个新的系统，名为 UsingTheCommandLineForDynamicSystemSimulation001
new_system(systemName)

% 打开系统模型
open_system(systemName)

% 添加需要添加的系统模块
add_block('simulink/Sources/Sine Wave', [systemName, '/Sine Wave'])
add_block('simulink/Math Operations/Gain', [systemName, '/Gain'])
add_block('simulink/Sinks/Out1', [systemName, '/Out'])

% 为模块连线
add_line(systemName, 'Sine Wave/1', 'Gain/1')
add_line(systemName, 'Gain/1', 'Out/1')

% 设置增益模块参数
set_param([systemName, '/Gain'], 'Gain', '3')

% 设置正弦波模块参数
set_param([systemName, '/Sine Wave'], 'Frequency', '4')

% 运行仿真
[t, xout, yout] = sim(systemName);

% 输出结果
```

```
plot(t,yout);

% 保存系统模型
% 使用 save_system(systemName) 保存，文件后缀则为 .slx
save_system(systemName, [systemName, '.mdl'])

% 保存系统模型后，就可以关闭模型了
close_system(systemName)
```



使用命令脚本构建的 Simulink 模型

使用命令行修改模型文件

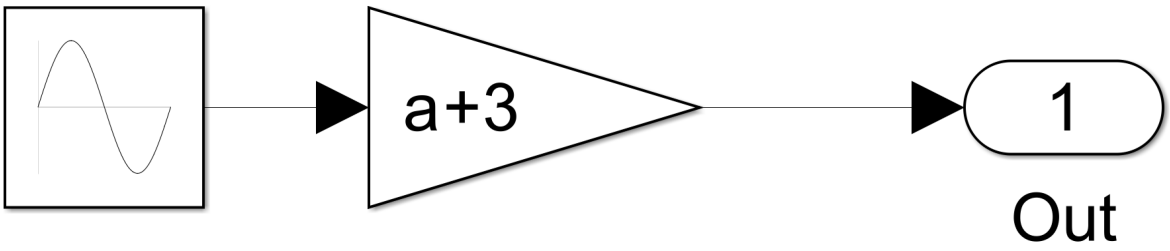
```
% open_system(systemName) 或 load_system(systemName) 打开模型
% 模型文件打开后，才可用使用 Simulink API 命令修改模型
% open_system(systemName) 会打开 Simulink 的模型编辑窗口
% load_system(systemName) 将模型载入到内存中,而无需在 Simulink 编辑器中打开模型
load_system(systemName)% 1. 打开模型

% 2. 修改模型
add_block('simulink/Sinks/Scope', [systemName, '/Scope'])% 添加模块
delete_block([systemName, '/Sine1 Wave'])% 删除模块
replace_block(systemName, 'Scope', 'simulink/Sinks/To Workspace', 'noprompt')% 替换模
型
% ...

% 3.关闭模型
% 保存并关闭模型
% 或 close_system(systemName, 1) 保存并关闭
save_system(systemName)
close_system(systemName)

% 关闭模型不保存
close_system(systemName, 0)
```

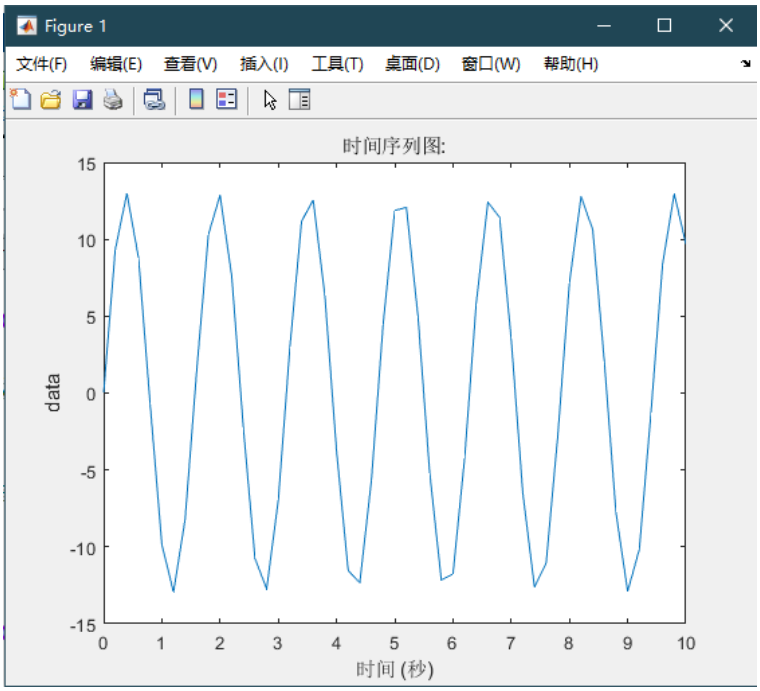
Simulink与MATLAB的接口

由MATLAB工作空间变量设置系统模块参数

- 在前面模型的基础上修改模型

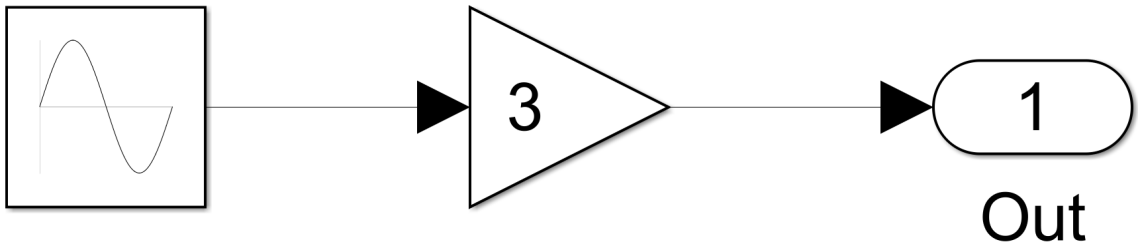
```
%% 由MATLAB工作空间变量设置系统模块参数
clear;clc;
% 要修改的模型名字
systemName = 'UsingTheCommandLineForDynamicSystemSimulation001';
% 新的名字
newSystemName = 'UsingTheCommandLineForDynamicSystemSimulation002';

% 首先在 MATLAB 工作空间建立变量 a
a = 10;

% 载入模型（打开Simulink编辑窗口）
open_system(systemName)

% 修改模块参数
set_param([systemName, '/Gain'], 'Gain', 'a+3')

% 运行仿真
[t, xout, yout] = sim(systemName);

% 打印结果
plot(t, yout);% To WorkSpace 输出的变量名默认为 simout

% 保存模型
save_system(systemName,newSystemName);

% 关闭模型
close_system(newSystemName);
```

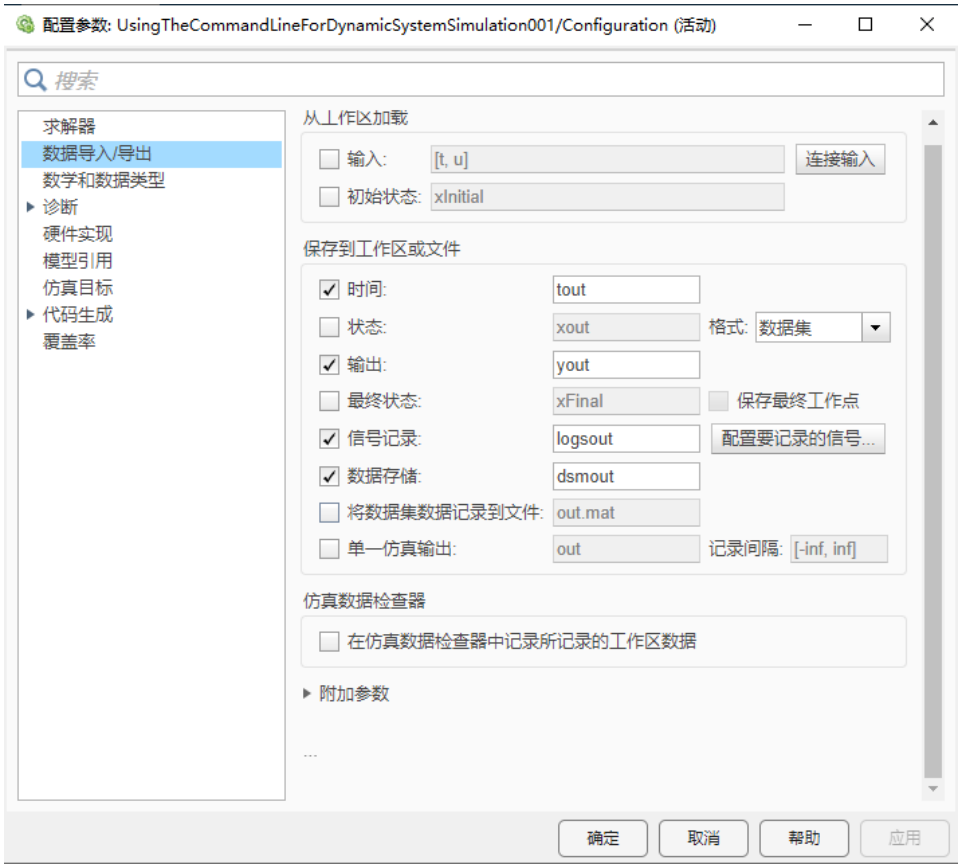




将信号输入到 MATLAB 工作空间中

方法1 使用 Out1 模块






```
systemName = 'UsingTheCommandLineForDynamicSystemSimulation001';
% 载入模型
load_system(systemName)

% 运行仿真
[t, xout, yout] = sim(systemName);

% 输出结果
plot(t,yout);

% 关闭模型
close_system(systemName)
```

方法2 利用 Scope 示波器模块

设置 Scope 模块参数，选中 Save data to workspace

或者使用命令行方式基于前面模型修改：

```
%%  利用 Scope 示波器模块将信号输入到 MATLAB 工作空间中
clear;clc;
% 要修改的模型名字
systemName = 'UsingTheCommandLineForDynamicSystemSimulation001';
% 新的名字
newSystemName = 'UsingTheCommandLineForDynamicSystemSimulation003';

%% 载入模型（不打开Simulink编辑窗口）
open_system(systemName)

%% 修改模型
% 删除与 Out 的连线
delete_line(systemName, 'Gain/1', 'Out/1')
% 删除 Out 模块
delete_block([systemName, '/Out'])
% 添加 Scope 模块
add_block('simulink/Sinks/Scope', [systemName, '/Scope'])
% 添加到 Scope 的连线
add_line(systemName, 'Gain/1', 'Scope/1')

%% 设置 Scope 模块
% 获取 Scope 模块的设置句柄
scopeConfig = get_param([systemName, '/Scope'], 'ScopeConfiguration');
% 设置 Scope 属性
scopeConfig.DataLogging = true;% 使得能够记录数据到工作区
scopeConfig.DataLoggingVariableName = 'Fist1Sim';% 设置保存在工作区中的变量名

% 设置输出数据的格式，以数组形式输出结果
% StructureWithTime -- 带时间的结构体
% Structure -- 结构体
% Array -- 数组
% Dataset -- 数据集
scopeConfig.DataLoggingSaveFormat = 'Array';
```

```
%% 运行仿真
sim(systemName);

%% 输出结果
plot(Fist1Sim(:,1), Fist1Sim(:,2));

%% 保存并关闭
save_system(systemName,newSystemName);
close_system(newSystemName);
```

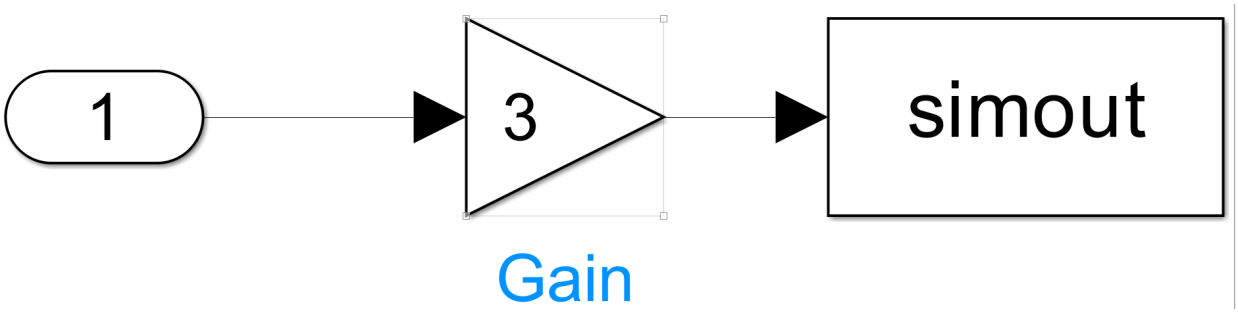
方法3 利用Sink模块中的 To Workspace 模块

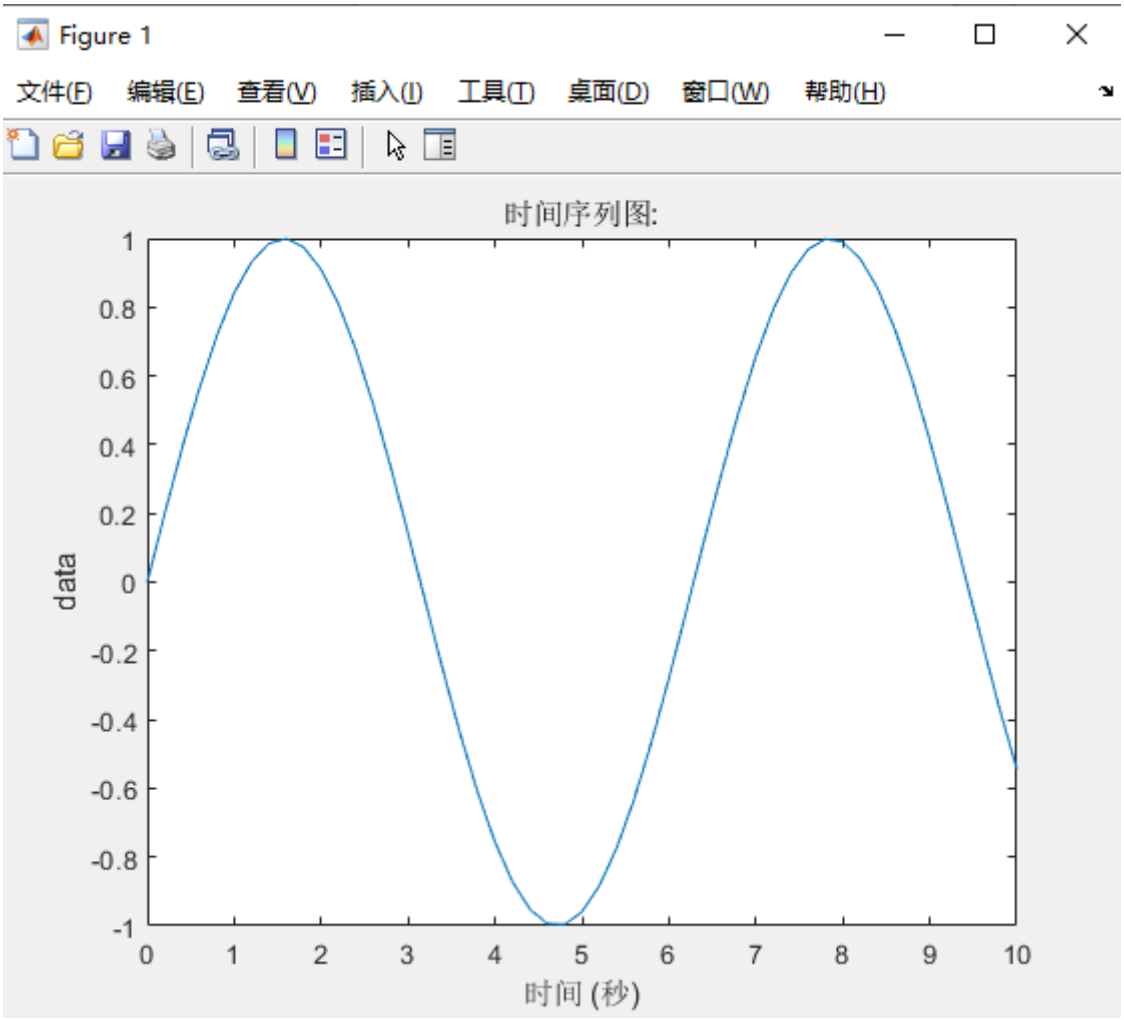
```
%% 利用Sink模块中的 To Workspace 模块将信号输入到 MATLAB 工作空间中

clear;clc;
% 要修改的模型名字
systemName = 'UsingTheCommandLineForDynamicSystemSimulation001';
% 新的名字
newSystemName = 'UsingTheCommandLineForDynamicSystemSimulation004';

%% 载入模型（不打开Simulink编辑窗口）
load_system(systemName)

%% 修改模型
% 删除与 Out 的连线
delete_line(systemName, 'Gain/1', 'Out/1')
% 删除 Out 模块
delete_block([systemName, '/Out'])
% 添加 Scope 模块
add_block('simulink/Sinks/To Workspace', [systemName, '/To Workspace'])
% 添加到 Scope 的连线
add_line(systemName, 'Gain/1', 'To Workspace/1')

%% 运行仿真
sim(systemName);

%% 输出结果
plot(simout);

%% 保存并关闭
save_system(systemName,newSystemName);
close_system(newSystemName);
```

使用工作空间变量作为系统输入信号

方法1 使用 In1

```
%% 使用工作空间中的变量作为系统输入信号 使用 In1
clear;clc;

% 要修改的模型名字
systemName = 'UsingTheCommandLineForDynamicSystemSimulation004';
% 新的名字
newSystemName = 'UsingTheCommandLineForDynamicSystemSimulation006';

%% 载入模型（不打开Simulink编辑窗口）
load_system(systemName)

%% 修改模型
% 删除 Sine Wave 与 Gain 的连线
delete_line(systemName, 'Sine Wave/1', 'Gain/1')
% 删除 Sine Wave
delete_block([systemName, '/Sine Wave'])
% 添加 From Workspace
add_block('simulink/Sources/In1', [systemName, '/Input'])
% From Workspace 连接到系统
add_line(systemName, 'Input/1', 'Gain/1')

%% 设置工作区变量作为系统输入
t = 0:0.1:10;
u = 1/3 * sin(t);
Input = [t', u'];

sim(systemName, 10, [], Input);

% 打印结果
plot(simout);

% 保存模型
save_system(systemName,newSystemName);

% 关闭模型
close_system(newSystemName);
```

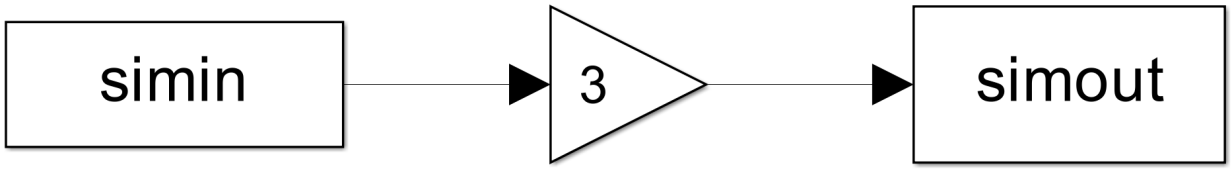


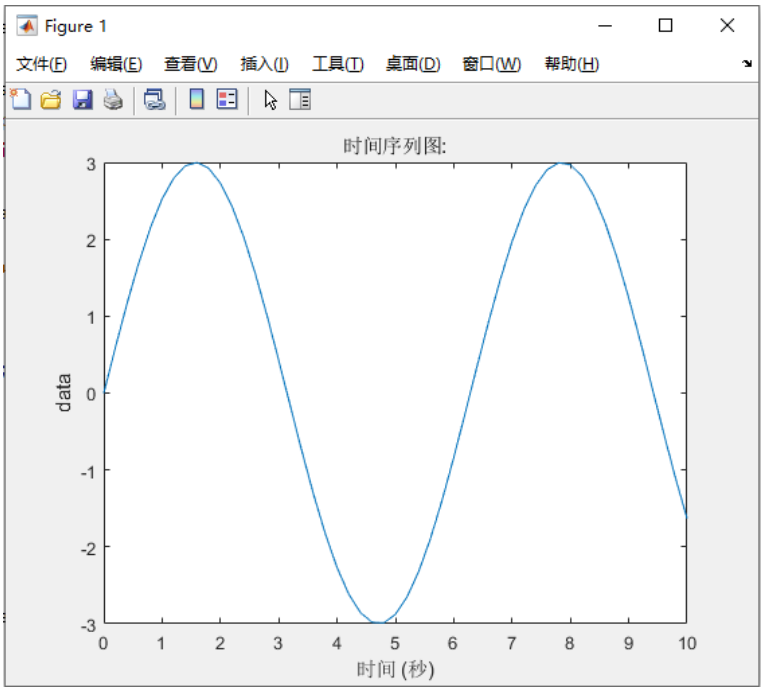


方法2 使用 From Workspace

```
%% 使用工作空间中的变量作为系统输入信号
clear;clc;

% 要修改的模型名字
systemName = 'UsingTheCommandLineForDynamicSystemSimulation004';
% 新的名字
newSystemName = 'UsingTheCommandLineForDynamicSystemSimulation005';

%% 载入模型（不打开Simulink编辑窗口）
load_system(systemName)

%% 修改模型
% 删除 Sine Wave 与 Gain 的连线
delete_line(systemName, 'Sine Wave/1', 'Gain/1')
% 删除 Sine Wave
delete_block([systemName, '/Sine Wave'])
% 添加 From Workspace
add_block('simulink/Sources/From Workspace', [systemName, '/From Workspace'])
% From Workspace 连接到系统
add_line(systemName, 'From Workspace/1', 'Gain/1')

%% 设置工作区变量作为系统输入
```

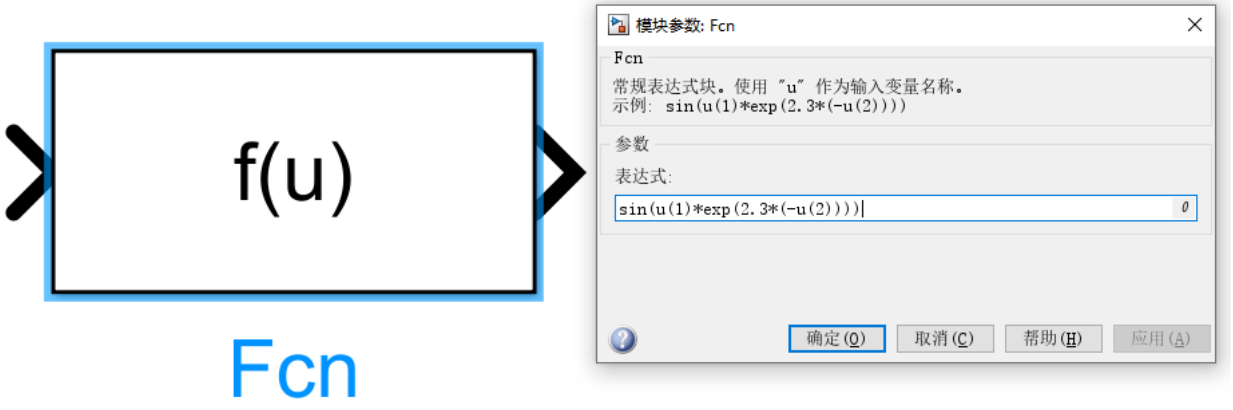
```
t = 0:0.1:10;
u = sin(t);
simin = [t', u'];% simin 为 From Workspace 默认使用的变量名称

sim(systemName);

% 打印结果
plot(simout);

% 保存模型
save_system(systemName,newSystemName);

% 关闭模型
close_system(newSystemName);
```



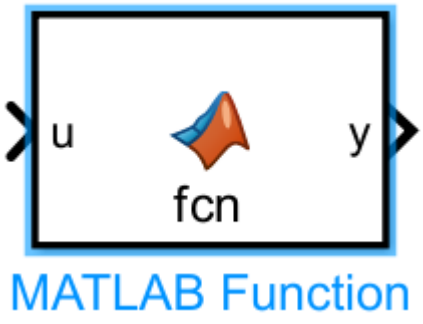


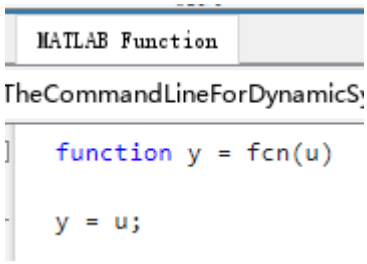
MATLAB Function 与 Function 模块

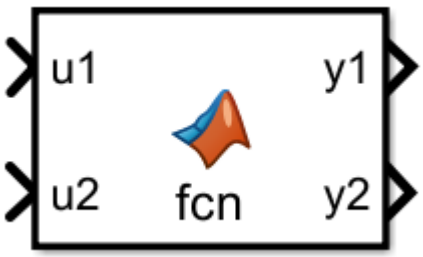
- `simulink/User-Defined Functions/Fcn` 模块



- `simulink/User-Defined Functions/Fcn` 模块








```
function [y1, y2] = fcn(u1, u2)

y1 = u1 + u2;
y2 = u1 - u2;
```

使用命令行方式进行动态系统仿真

使用命令行方式，可以编写并运行系统仿真的脚本文件来完成动态系统的仿真，在脚本文件中**重复地对同一系统在不同的仿真参数或不同的系统模块参数下进行仿真**，而无需一次又一次地启动 Simulink 仿真平台中的 Run 进行仿真。

如果需要需要分析某一参数对系统仿真结果的影响，读者可以很容易地通过for循环自动修改任意指定的参数，即可达到目的。这样可以非常容易地分析不同参数对系统性能的影响，并且也可以从整体上加快系统仿真的速度。

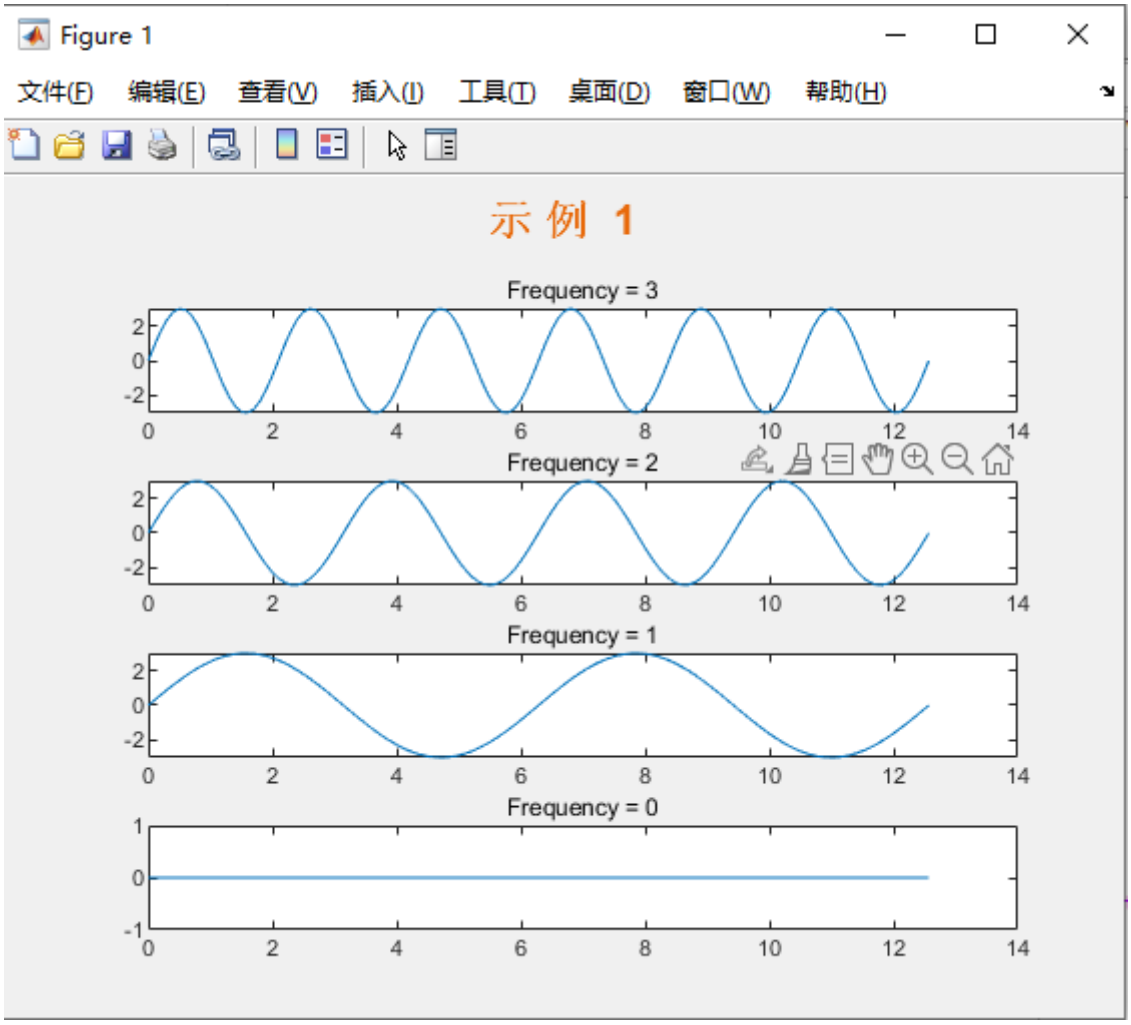
使用 sim 命令进行动态系统仿真

```
%% 模型仿真
sim(modelName);

% 模型和参数设置不同，simOut 有不同的格式，这里不详细讨论
simOut = sim(modelName);

%% 以设定的模型参数进行仿真
% 对系统模型model进行系统仿真，其仿真参数ParameterName1、ParameterName 2等的取值分别为
Value1、Value2 等。
simOut = sim(modelName, 'ParameterName1',Value1, 'ParameterName2', Value2...);

% 功能同上，只不过所有仿真参数被包含于一个结构体 ParameterStruct。
simOut = sim(modelName, ParameterStruct)

%% 设置仿真时间
sim('model', StopTime)
sim('model', [StartTime StopTime])
% StartTime:Interval:StopTime表示输出时间向量，每间隔时间 Interval 系统就输出一次。
sim('model', StartTime:Interval:StopTime)

%% 详细设置
% 'StopTime', '15.0' 仿真结束时间：15.0s
% 'MaxStep', '0.1' 最大步长：0.1
% 'SaveFormat', 'Array' 以数组形式保存输出数据
% 'SaveState', 'on' 是否保存状态变量：是
% 'StateSaveName', 'xout' 保存状态变量的名称：xout
% 'SaveOutput', 'on' 是否保存输出变量：是
% 'OutputSaveName', 'yout' 保存保存变量的名称：yout
simOut = sim(systemName, 'StopTime', '4*pi', 'MaxStep', '0.1', ...
        'SaveFormat', 'Array', ...
        'SaveState', 'on', 'StateSaveName', 'xout', ...
        'SaveOutput', 'on', 'OutputSaveName', 'yout');
tout=simOut.get('tout');
yout=simOut.get('yout');
plot(tout, yout);
```

示例1：修改模型参数，进行迭代仿真

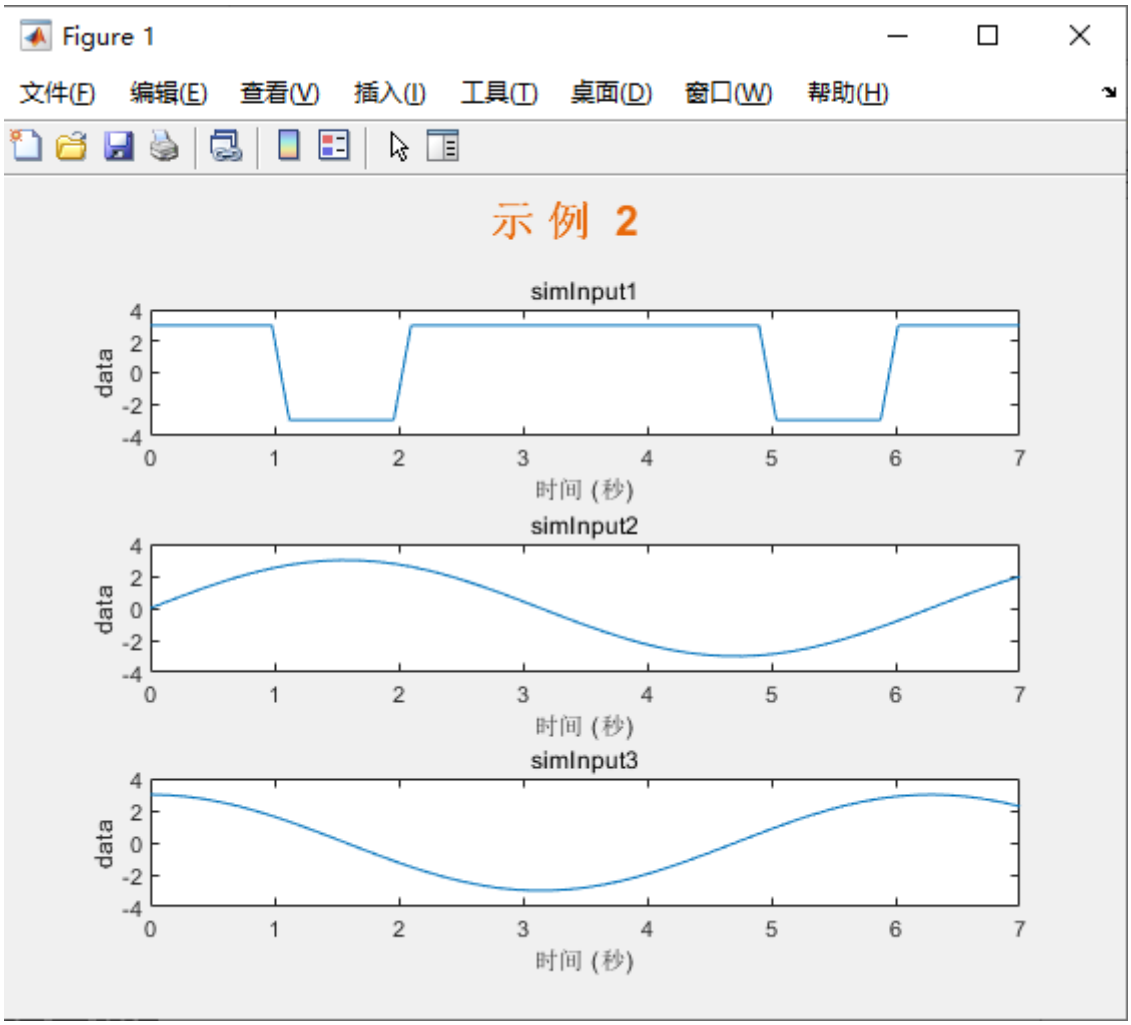
```
clc;clear;

systemName = 'UsingTheCommandLineForDynamicSystemSimulation001';
load_system(systemName)

for index = 1:4
```

```
    tmp = 4 - index;
    % 每次循环修改 Sine Wave 模块的频率
    set_param([systemName, '/Sine Wave'], 'Frequency', num2str(tmp));
    % 进行仿真
    simOut = sim(systemName, 'StopTime', '4*pi', 'MaxStep', '0.1', ...
        'SaveFormat', 'Array', ...
        'SaveState', 'on', 'StateSaveName', 'xout', ...
        'SaveOutput', 'on', 'OutputSaveName', 'yout');

    % 输出结果
    subplot(4,1,index);
    plot(simOut.tout, simOut.yout);
    title(['Frequency = ', num2str(tmp)]);
end
sgtitle('示 例  1', 'Color', [0.9102, 0.4124, 0.0379], 'FontWeight', 'bold',
'FontSize', 16);
close_system(systemName, 0)
```



示例2: 使用不同工作空间参数作为输入

```
%% 示例2: 使用不同工作空间参数作为输入
clc;clear;
systemName = 'UsingTheCommandLineForDynamicSystemSimulation006';

load_system(systemName)

simInput1=[ 0, 1;
            1,1;
            1,-1;
            2,-1;
            2, 1;
            3, 1;
            3, 1;
            4, 1;
            4, 1;
            5, 1;
            5,-1;
            6,-1;
            6, 1;
            7, 1 ];
t=0:0.1:7;
simInput2=[t', sin(0:0.1:7)'];
simInput3=[t', cos(0:0.1:7)'];

sim(systemName,7, [], simInput1);
subplot(3,1,1);
plot(simout);
ylim([-4, 4]);
title('simInput1');

sim(systemName,7, [], simInput2);
subplot(3,1,2);
plot(simout);
ylim([-4, 4]);
title('simInput2');

sim(systemName,7, [], simInput3);
subplot(3,1,3);
plot(simout);
ylim([-4, 4]);
title('simInput3');

sgtitle('示 例  2', 'Color', [0.9102, 0.4124, 0.0379], 'FontWeight', 'bold',
'FontSize', 16);
close_system(systemName, 0)
```



> Sinks 接收器模块
>
> Sources 信号源