第四章 动态仿真集成环境——Simulink

CONTENT

01

PART ONE

第一节 SIMULINK简介

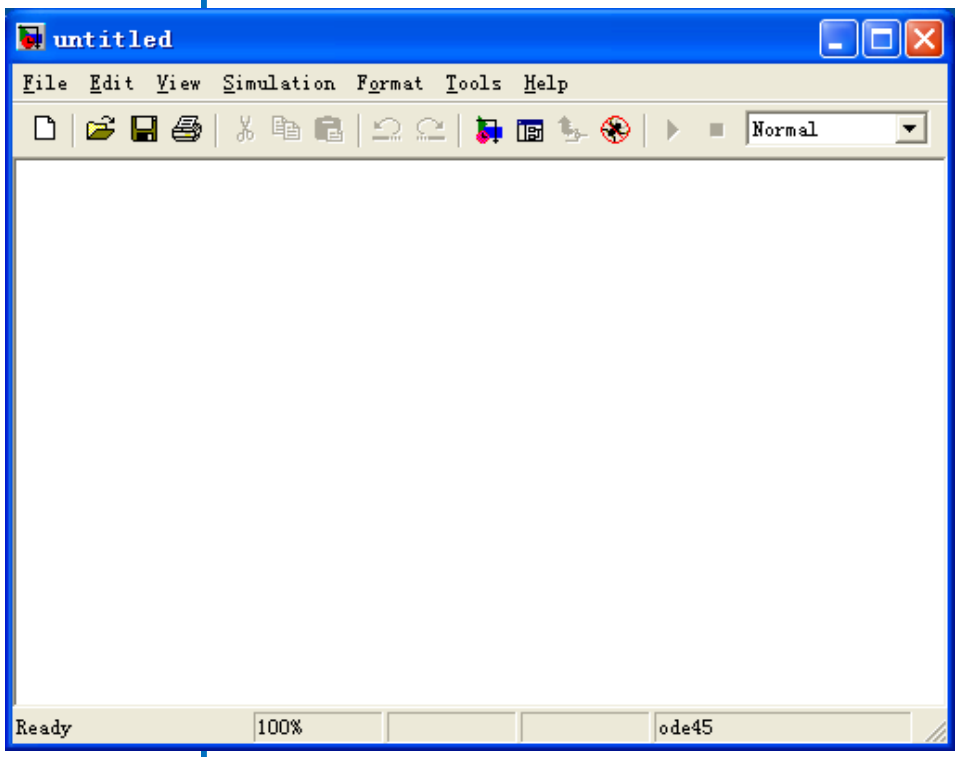
一.SIMULINK的启动

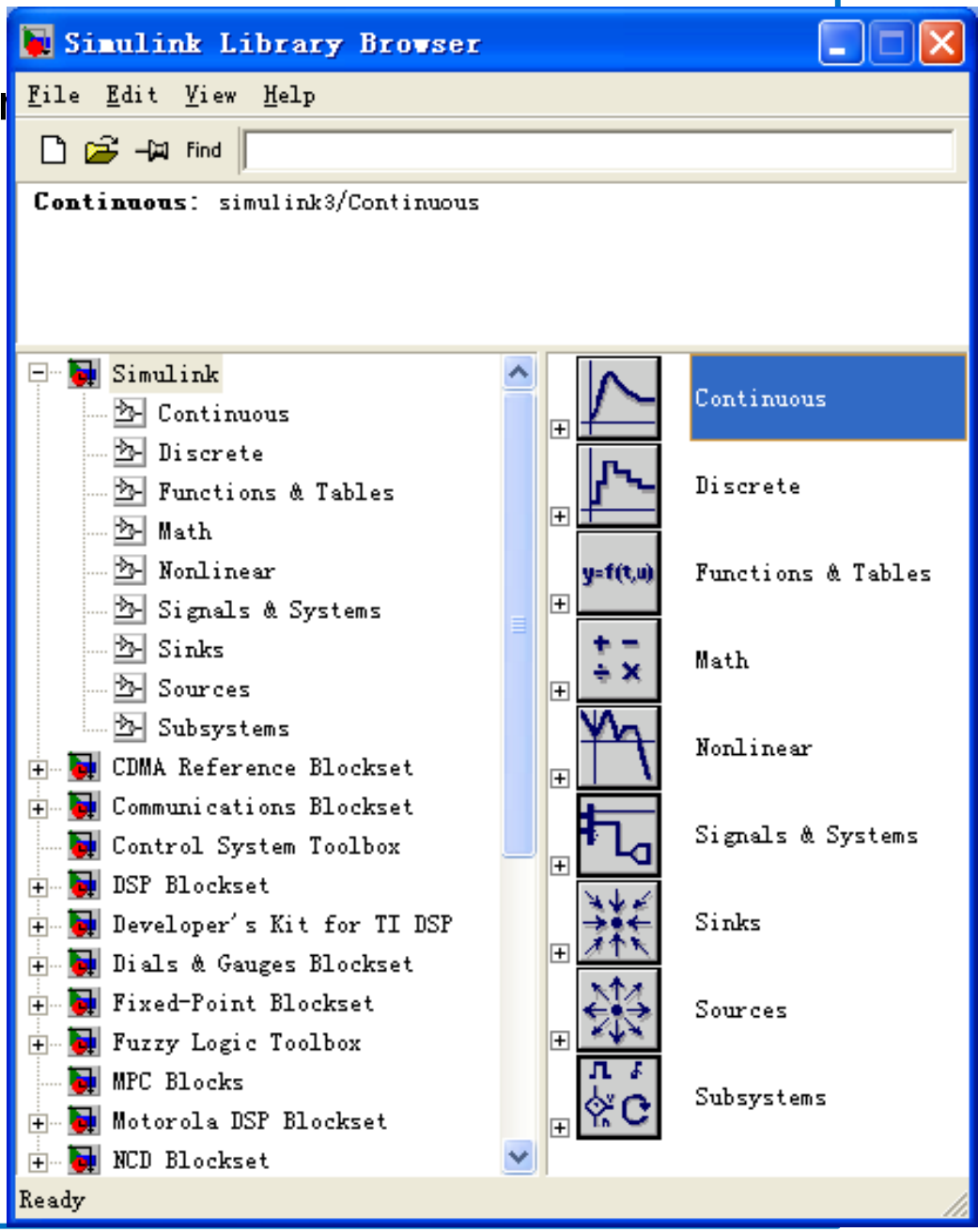
二.SIMULINK库浏览窗口的功能菜单

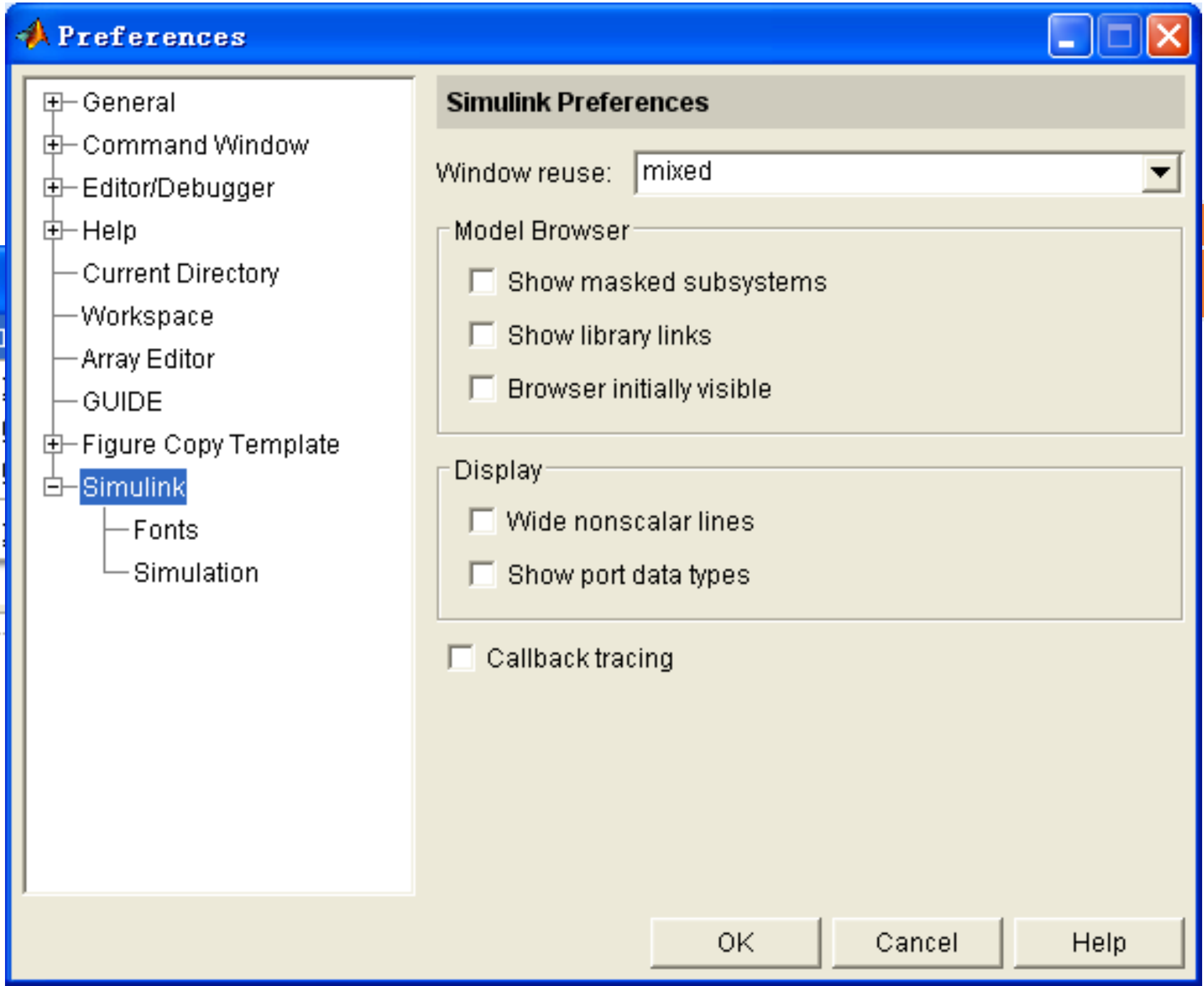
三.SIMULINK的仿真模块库

一.SIMULINK的启动

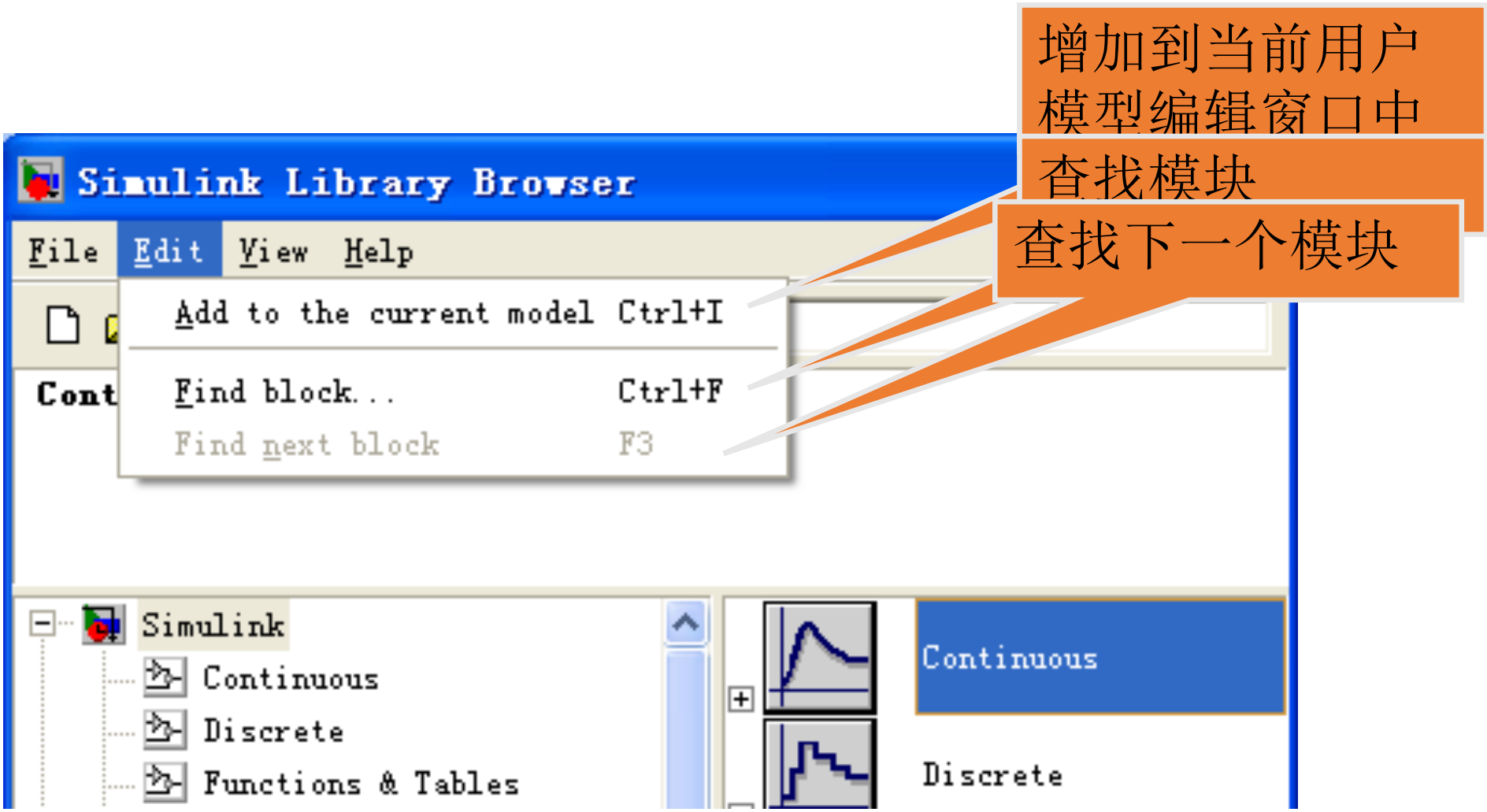
- 点击工具条上的图标
- 在命令窗口直接输入simulin



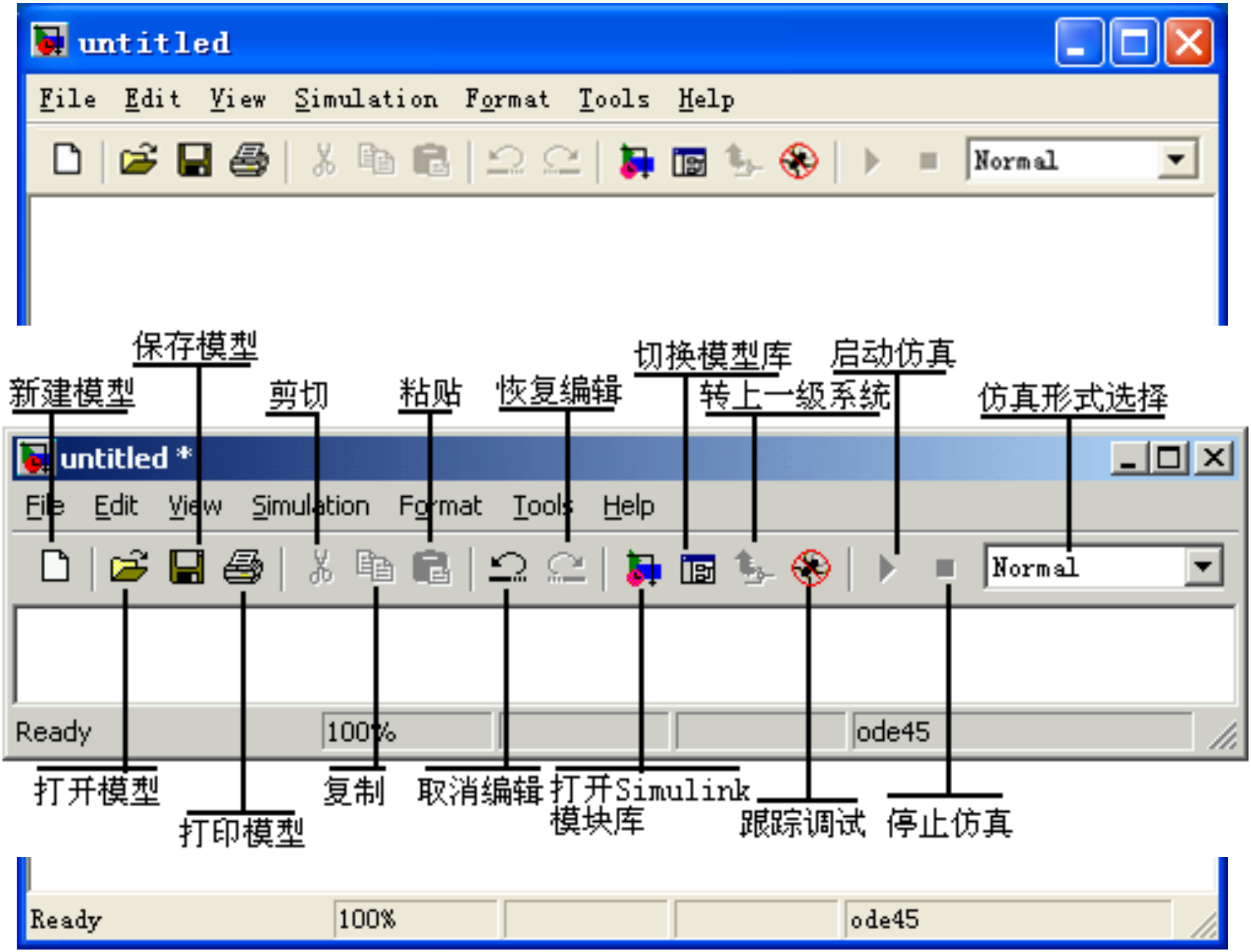



Preferences
General
Command Window
Editor/Debugger
Help
Current Directory
Workspace
Array Editor
GUIDE
Figure Copy Template
Simulink
Fonts
Simulation
Simulink Preferences
Window reuse:
mixed
Model Browser
Show masked subsystems
Show library links
Browser initially visible
Display
Wide nonscalar lines
Show port data types
Callback tracing
OK
Cancel
Help

二.SIMULINK库浏览窗口的功能菜单



二.SIMULINK库浏览窗口的功能菜单


untitled
File Edit View Simulation Format Tools Help
Normal
Ready
100%
ode45
新建模型
保存模型
剪切
粘贴
恢复编辑
切换模型库
转上一级系统
启动仿真
仿真形式选择
打开模型
打印模型
复制
取消编辑
打开Simulink
模块库
跟踪调试
停止仿真

三.SIMULINK的仿真模块库

- ◆1．Continuous：提供线性系统元件(指连续性)。
- ◆2．Discrete：提供离散型元件。
- ◆3．Function & Tables：函数以及表。
- ◆4．Math：数学函数功能。
- ◆5．Nonlinear：提供非线性元件。
- ◆6．Signal & Systems：信号以及系统变换函数
- ◆7．Sink：提供输出设备元件。
- ◆8．Source：提供信号源元件。
- ◆9．Subsystems：子系统函数

02

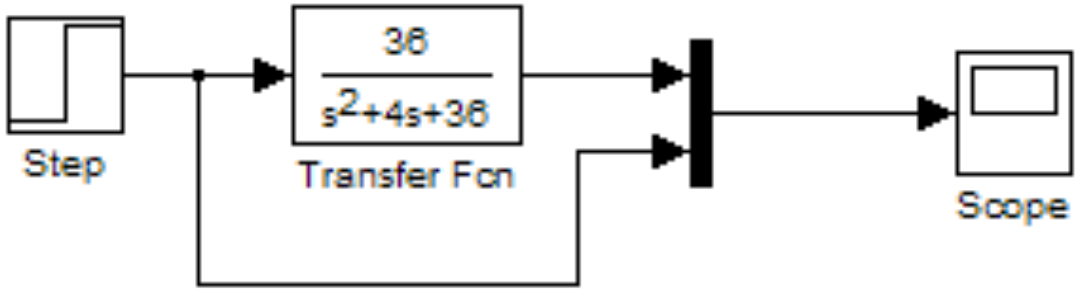
PART TWO

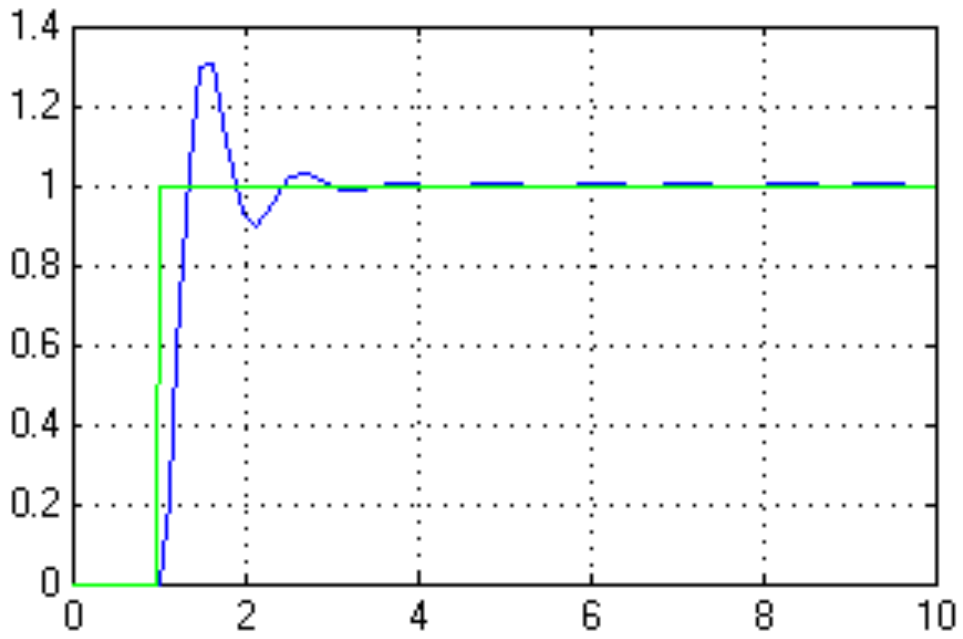
第二节SIMULINK应用实例

一.简单的模型实例
二.倒车系统的模型实例
三.倒立摆模型实例

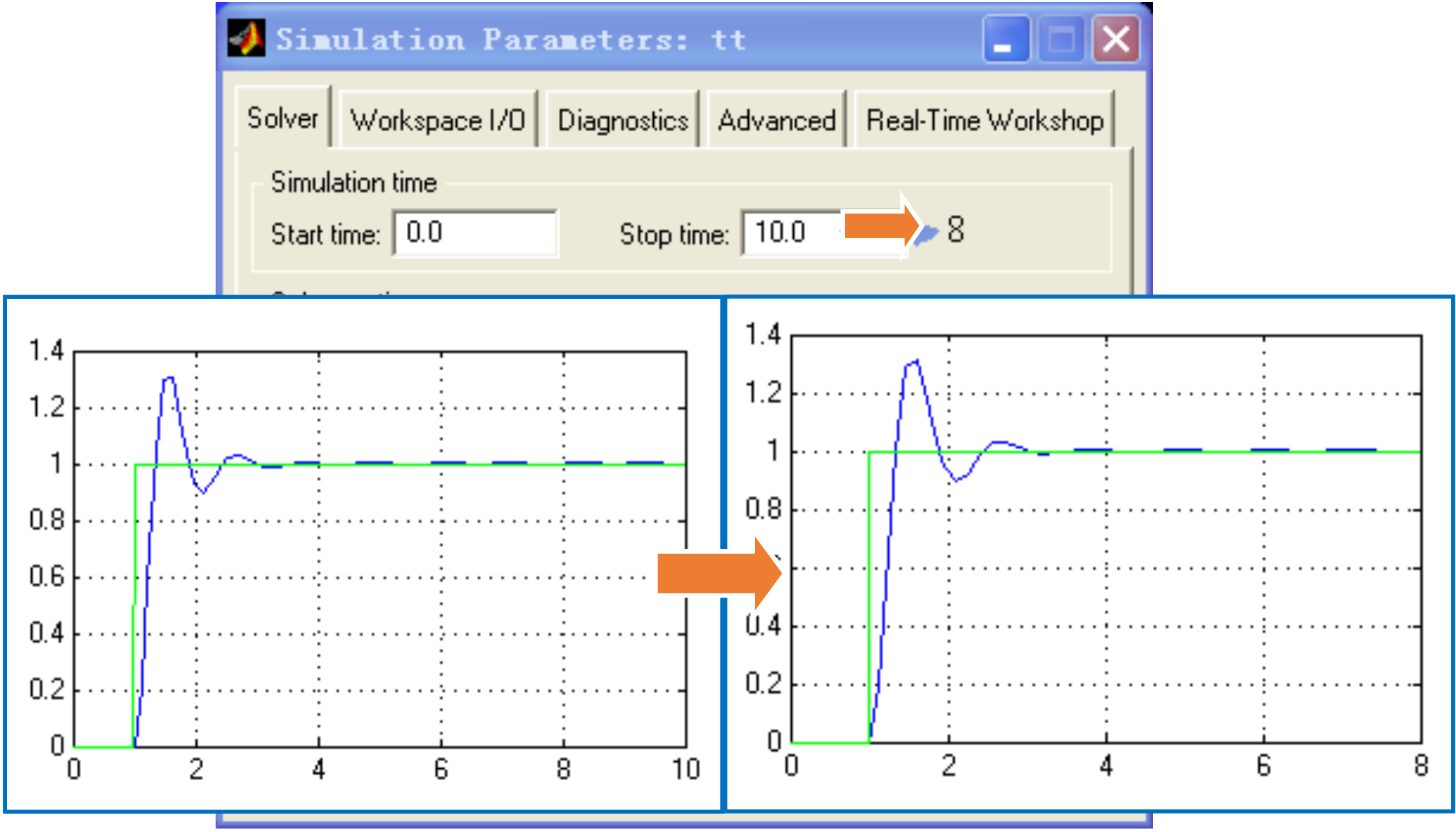
一.简单的模型实例

例题1.针对某二阶系统，绘制在单位阶跃信号作用下的响应曲线。





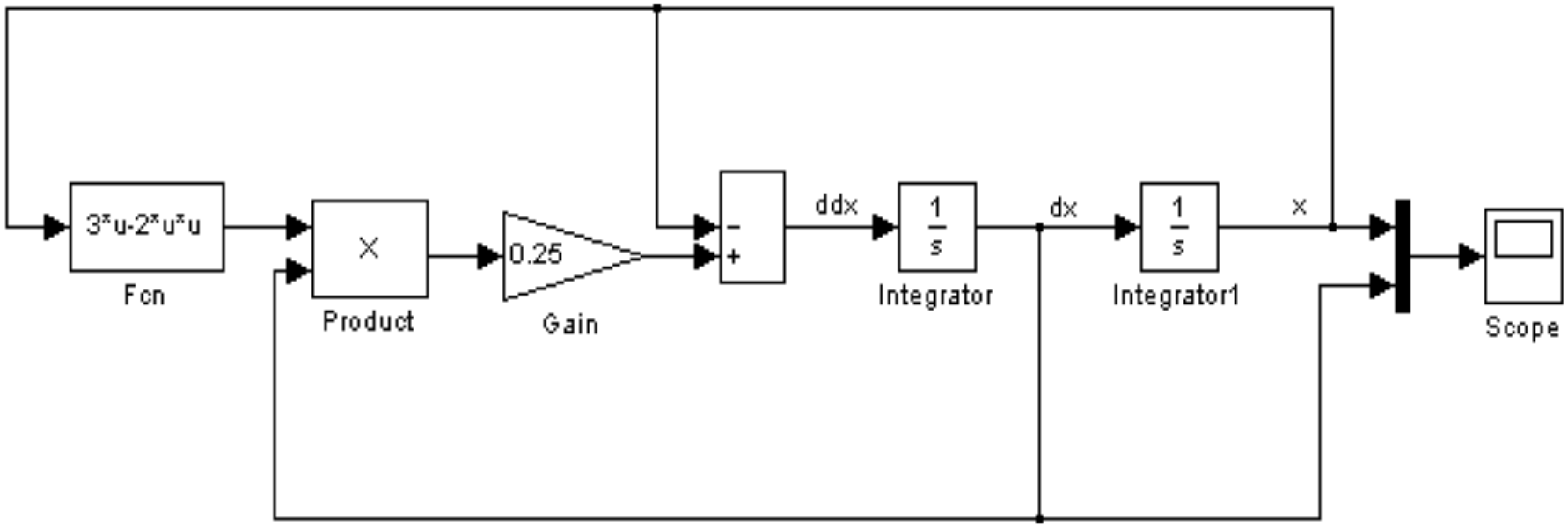
一.简单的模型实例


Simulation Parameters: tt
Solver
Workspace I/O
Diagnostics
Advanced
Real-Time Workshop
Simulation time
Start time: 0.0
Stop time: 10.0
8
1.4
1.2
1
0.8
0.6
0.4
0.2
0
0
2
4
6
8
10
1.4
1.2
1
0.8
0.4
0.2
0
0
2
4
6
8

一.简单的模型实例

例题2. 使用Simulink创建系统，求解非线性微分方程$(3x-2x^2)\dot{x}-4x=4\ddot{x}$，其初始值为$\dot{x}(0)=0, x(0)=2$绘制函数的波形。

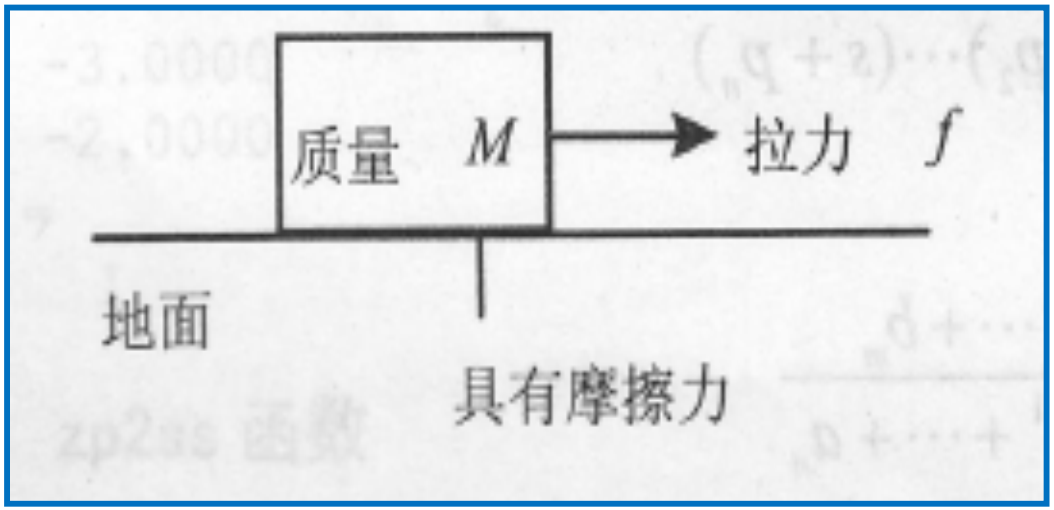
创建仿真系统为

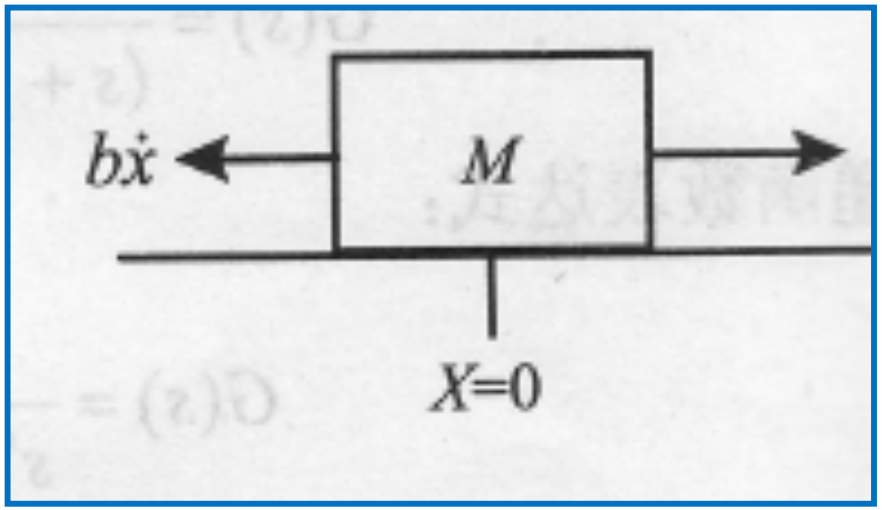


一.简单的模型实例

一.简单的模型实例

例题3.力-质量系统，要拉动一个箱子（拉力f=1N)，箱子质量为M(1kg)，箱子与地面存在摩擦力,其大小与车子的速度成正比，比例系数为b=0.4N/m/s 。





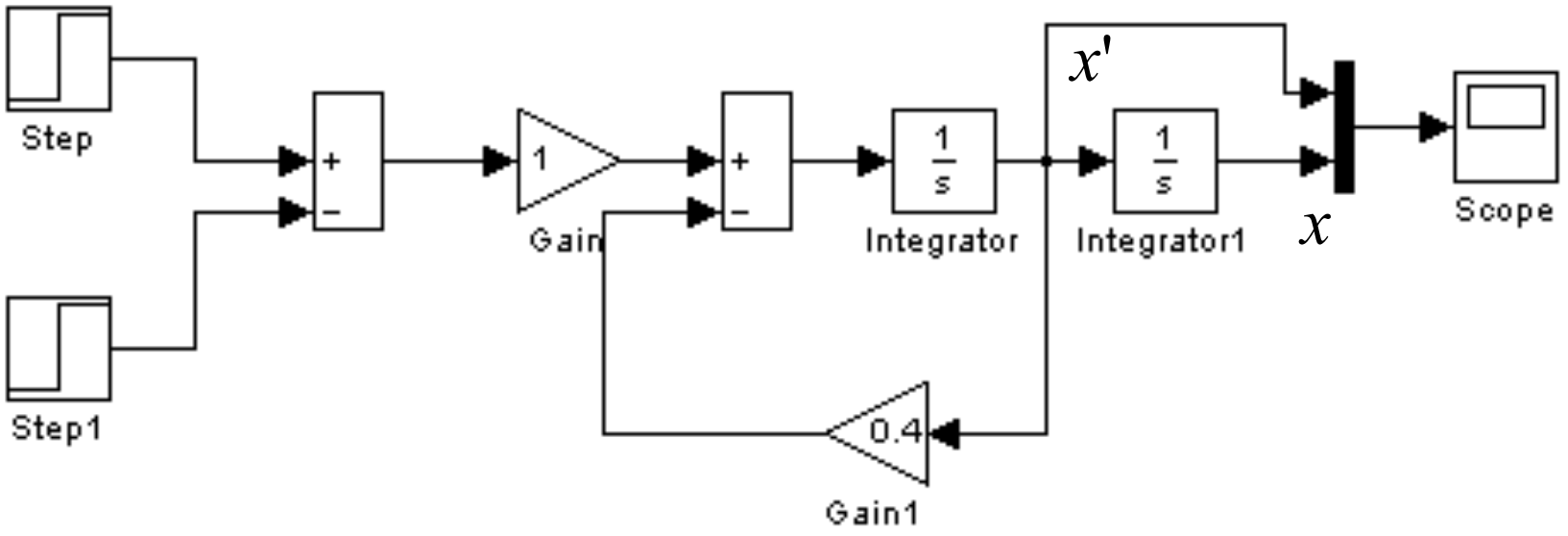
其运动方程式为 $$f - b\dot{x} = M\ddot{x}$$

一.简单的模型实例

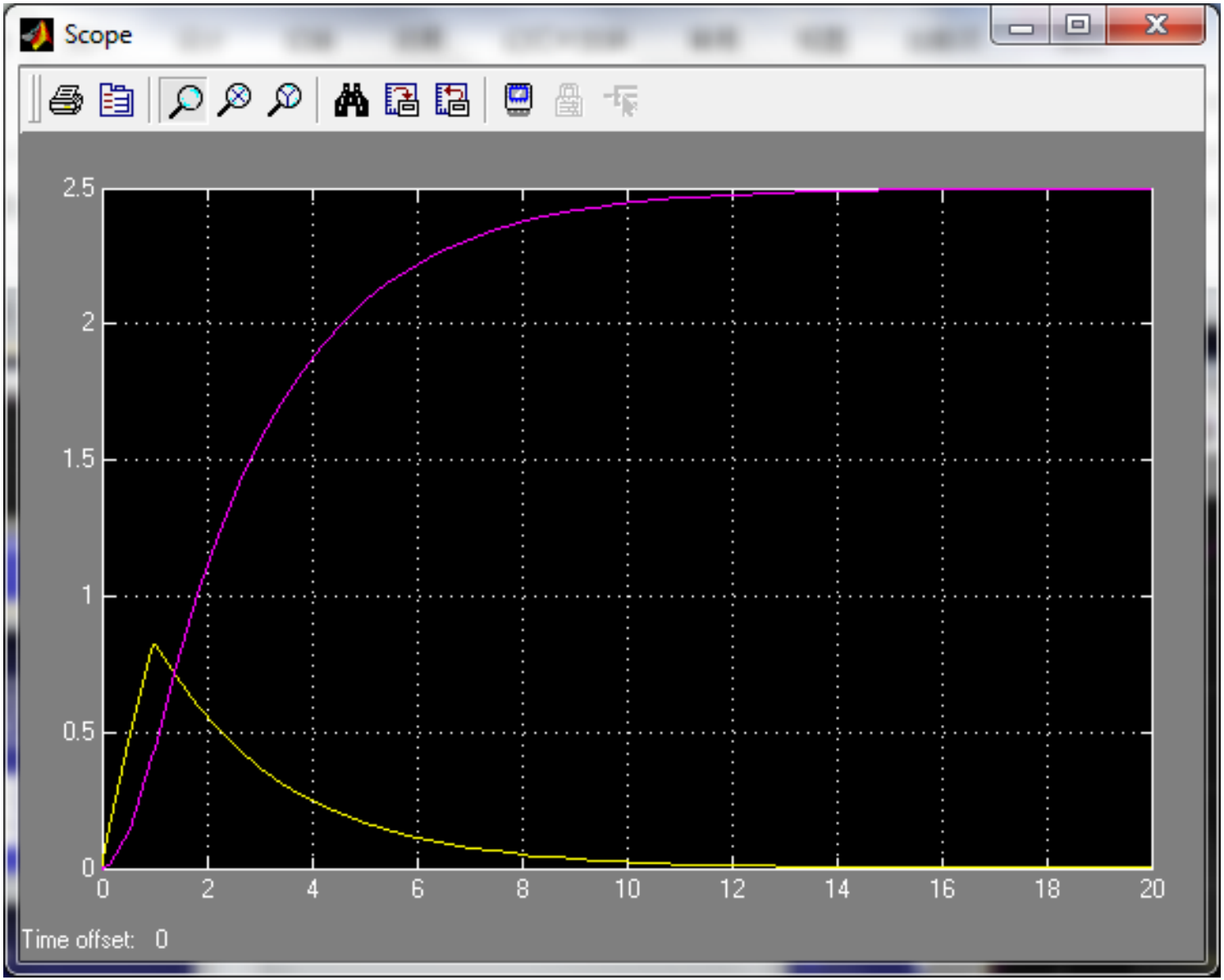
例题3.力-质量系统，要拉动一个箱子（拉力f=1N)，箱子质量为M(1kg)，箱子与地面存在摩擦力,其大小与车子的速度成正比，比例系数为b=0.4N/m/s 。

其运动方程式为 $f - b\dot{x} = M\ddot{x}$

拉力作用时间为2s，建构的模型为



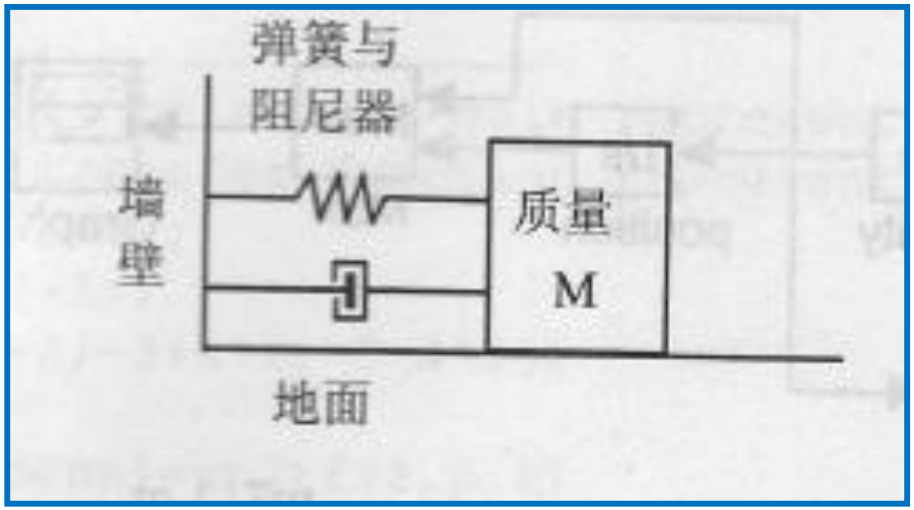
一.简单的模型实例

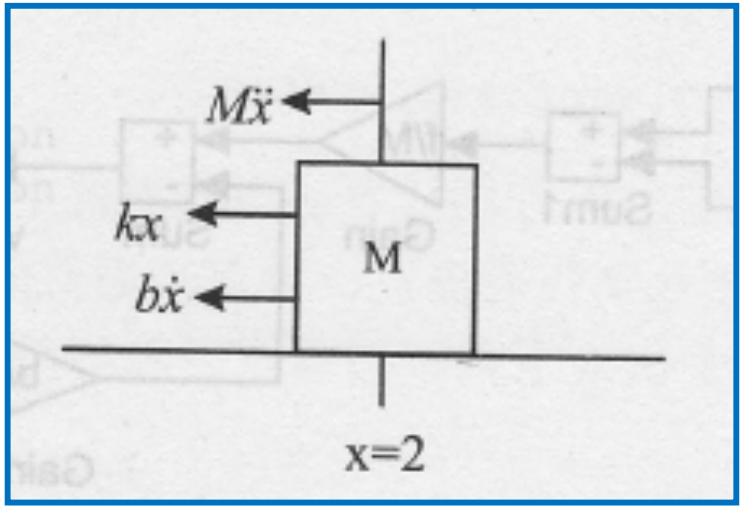


因有摩擦力存在，箱子最终将会停止前进。

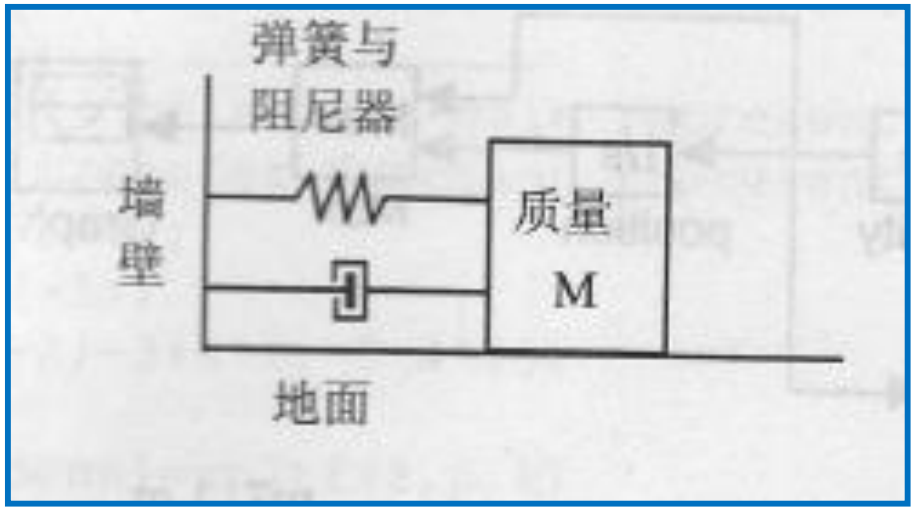
一.简单的模型实例

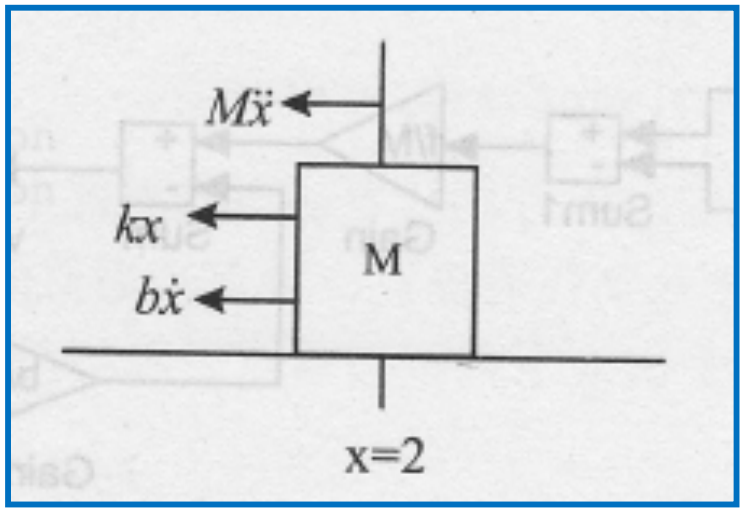
例题4.力-弹簧-阻尼系统，假设箱子与地面无摩擦存在，箱子质量为M(1kg)，箱子与墙壁间有线性弹簧(k=1N/m)与阻尼器(b=0.3N/m/s)。阻尼器主要用来吸收系统的能量，吸收系统的能量转变成热能而消耗掉。现将箱子拉离静止状态2cm后放开，试求箱子的运动轨迹。





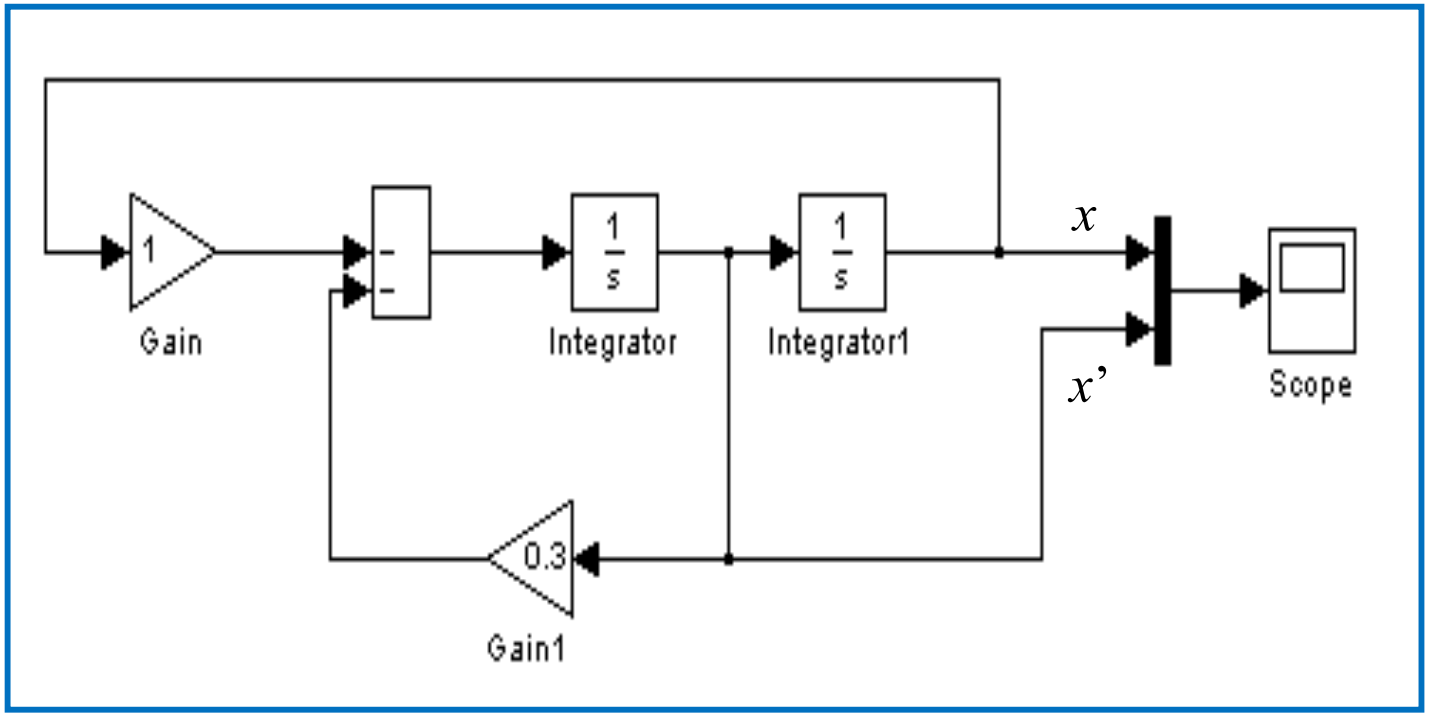
一.简单的模型实例



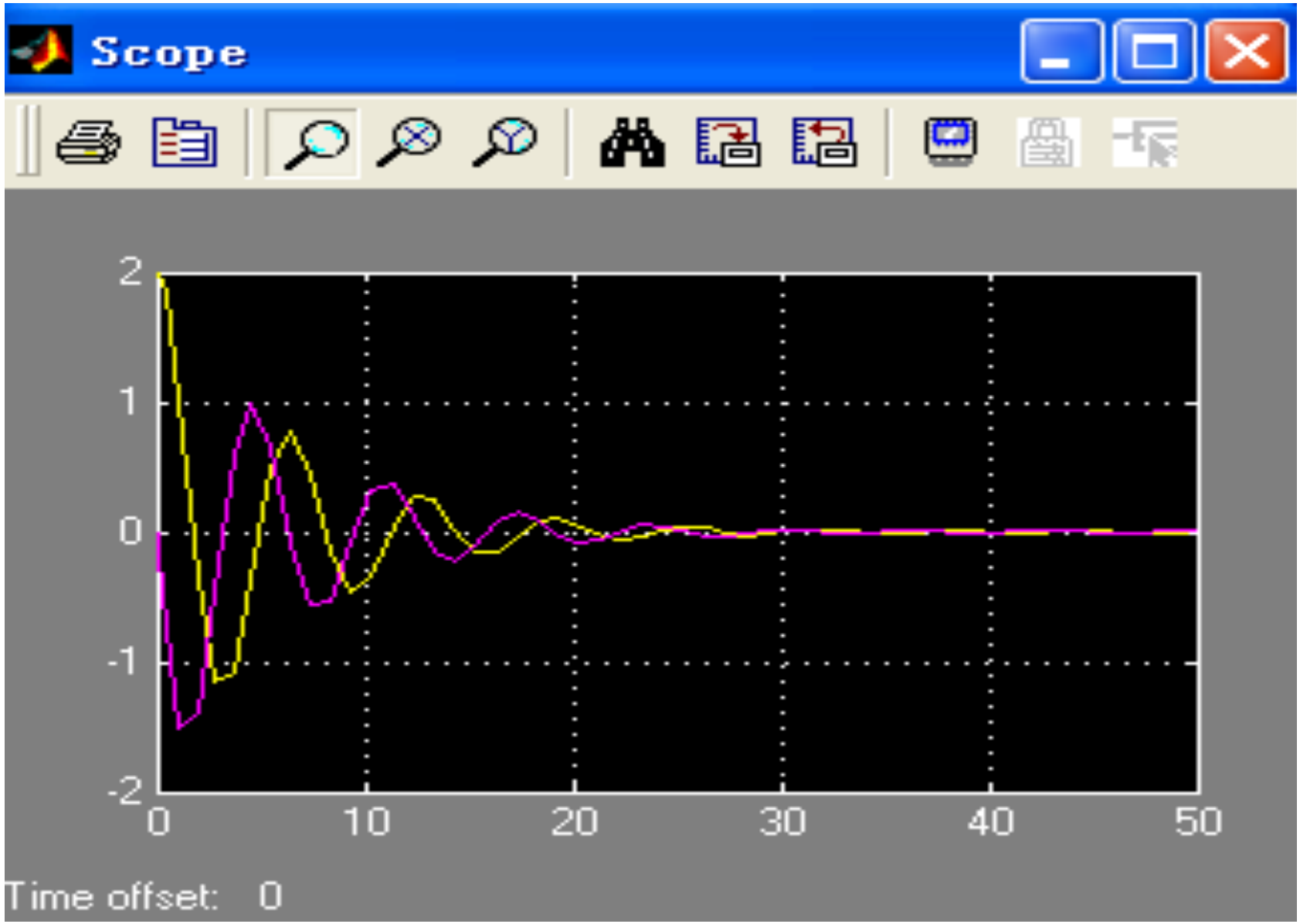


运动方程式为 $M\ddot{x} + kx + b\dot{x} = 0$

构建的模型为



一.简单的模型实例

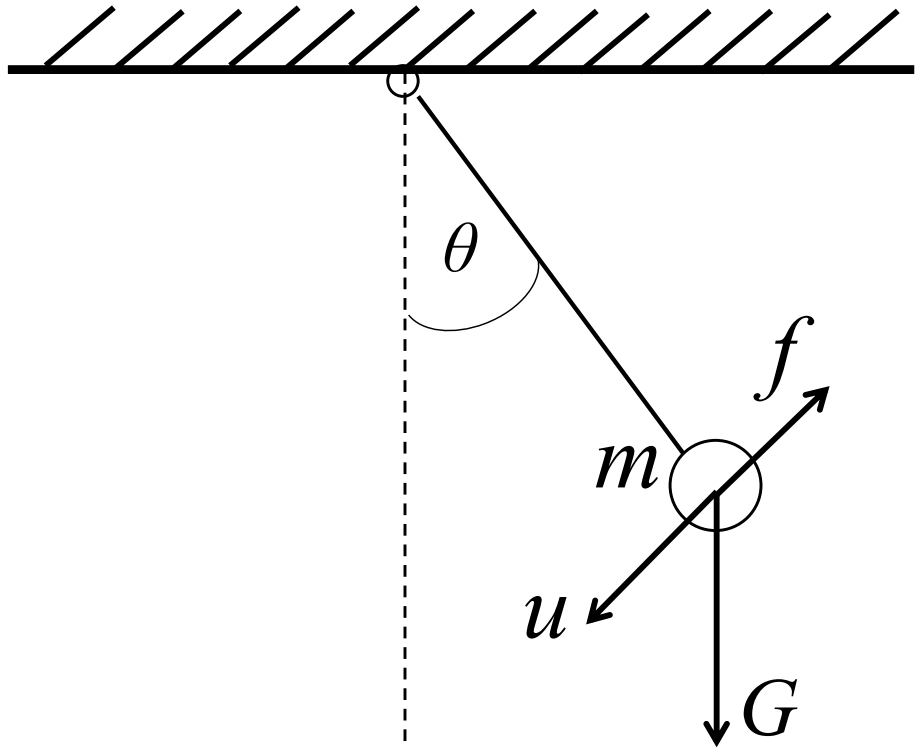


因有阻尼器存在，故箱子最终会停止运动。

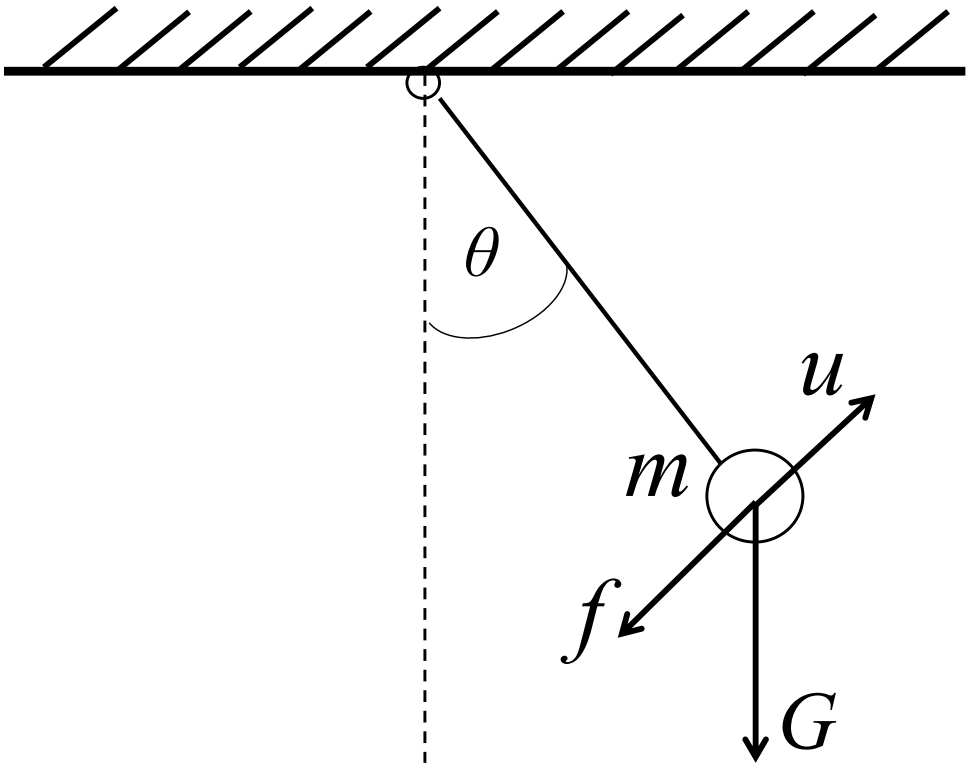
一.简单的模型实例

例题5.下图所示简单的单摆系统,假设杆的长度为L（米m）,且杆的质量不计,钢球的质量为m（千克kg）。单摆受重力和摩擦阻尼力以及外力共同作用，其中u（牛N）为作用在单摆上的外力；单摆的运动可以以线性的微分方程式来近似,但事实上系统的行为是非线性的,而且存在粘滞阻尼,假设粘滞阻尼系数为ξ（kg/ms^{-1}）。

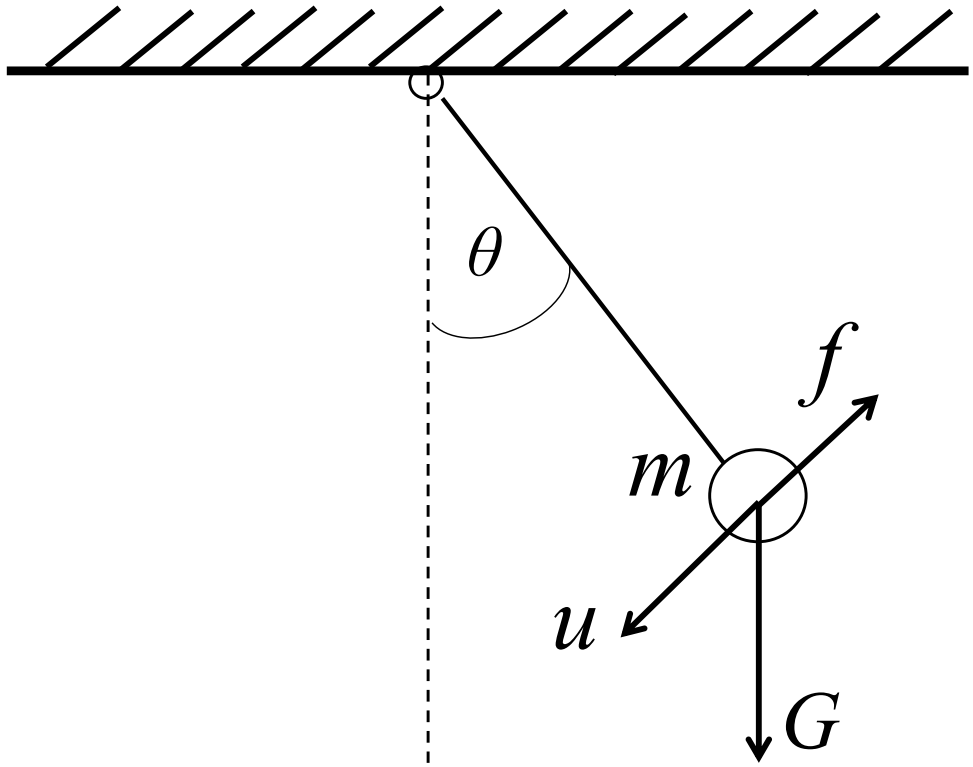
$\xi = 0.03$
$g = 9.8$
$L = 0.8$
$m = 0.3$
$u = 0$



系统数学建模



单摆系统向上受力时受力分析图



单摆系统向下受力时受力分析图

根据牛顿第二定律： $\sum F = ma$

问题中： $\ddot{\theta} = \dot{\omega}$，$a = L \times \dot{\omega} = L \times \ddot{\theta}$，$f = \xi \times v = \xi \times L \times \omega = \xi L\dot{\theta}$，$G = mg$

推导出单摆系统的动力学方程为：

第一种情况： $mL\ddot{\theta} + \xi L\dot{\theta} = u - Gsim(\theta) = u - mgsin(\theta)$

第二种情况： $mL\ddot{\theta} + \xi L\dot{\theta} = u + Gsim(\theta) = u + mgsin(\theta)$

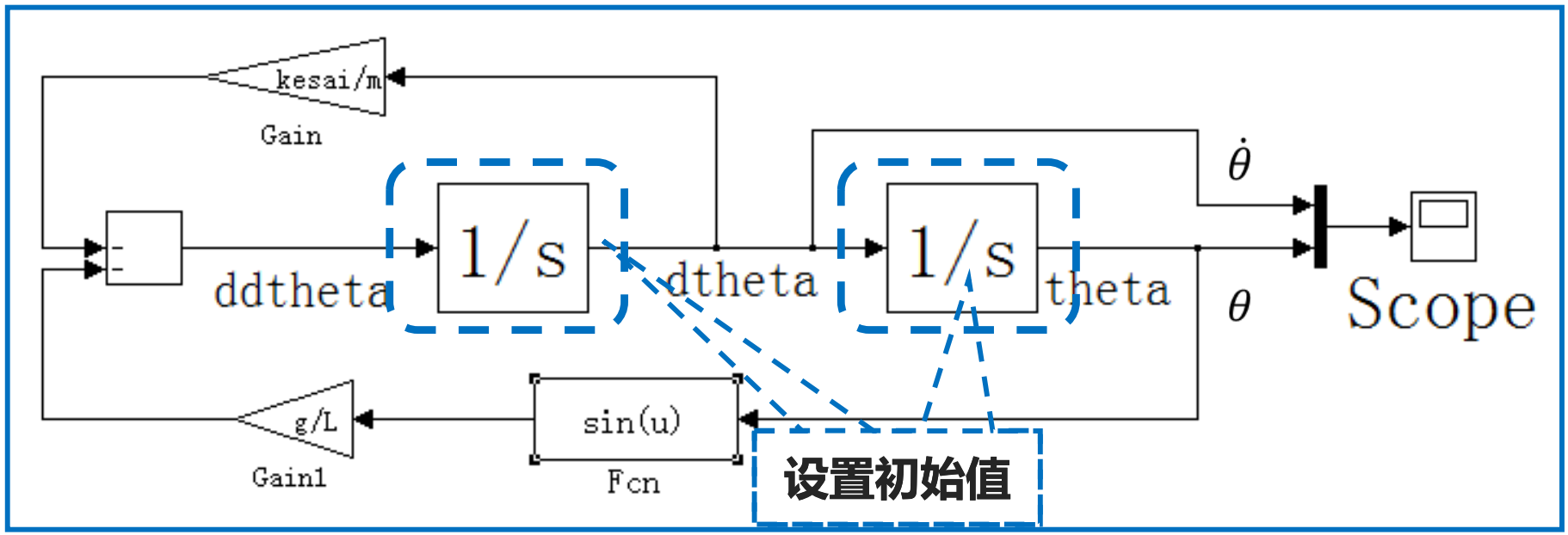
系统数学建模

第一种情况

根据单摆系统的动力学方程：

$$mL\ddot{\theta} + \xi L\dot{\theta} = u - Gsim(\theta) = u - mgsin(\theta)$$

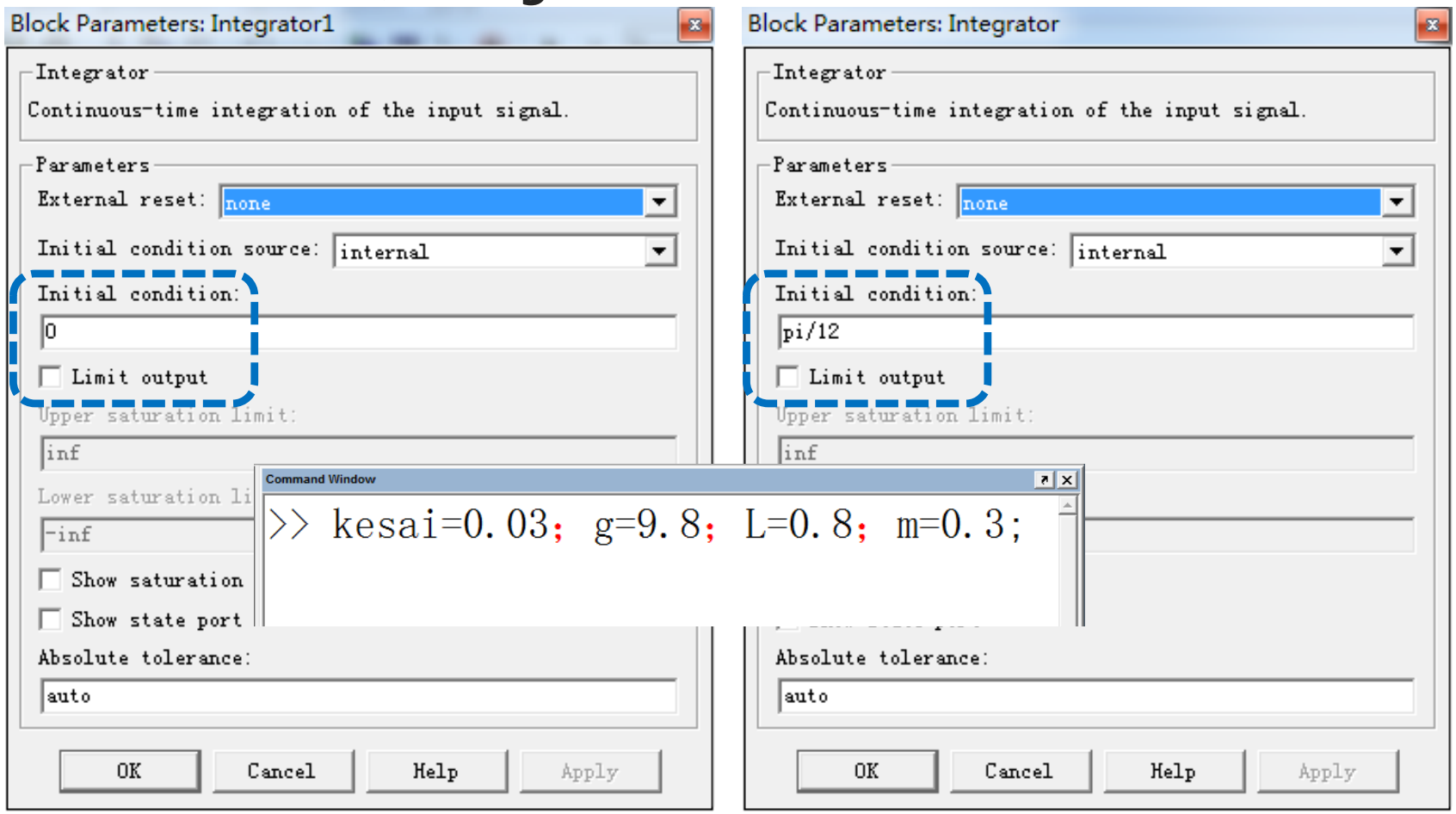
设ξ=0.03，g=9.8，L=0.8，m=0.3，u=0 ，所构建的模型



系统数学建模

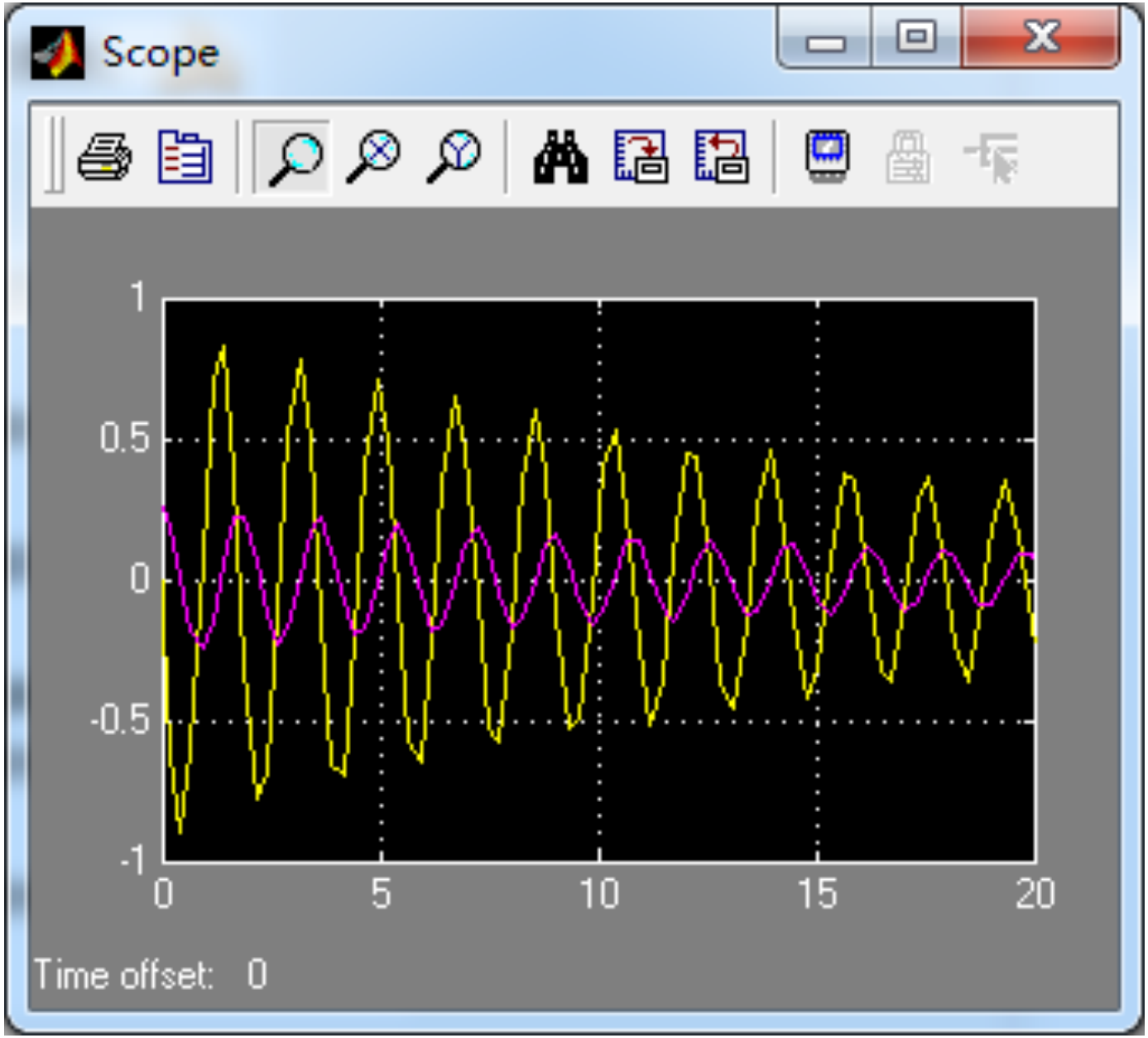
第一种情况

仿真过程中，设起始角度为pi/12；
弹性系数kesai为0.03，g为9.8m/s²，杆长L为0.8，钢球质量为0.3kg



系统数学建模

第一种情况



一.简单的模型实例

例题6.蹦极跳作为一个连续系统的例子。当一个90kg的人系着弹力绳从桥上跳下来时，是否安全？

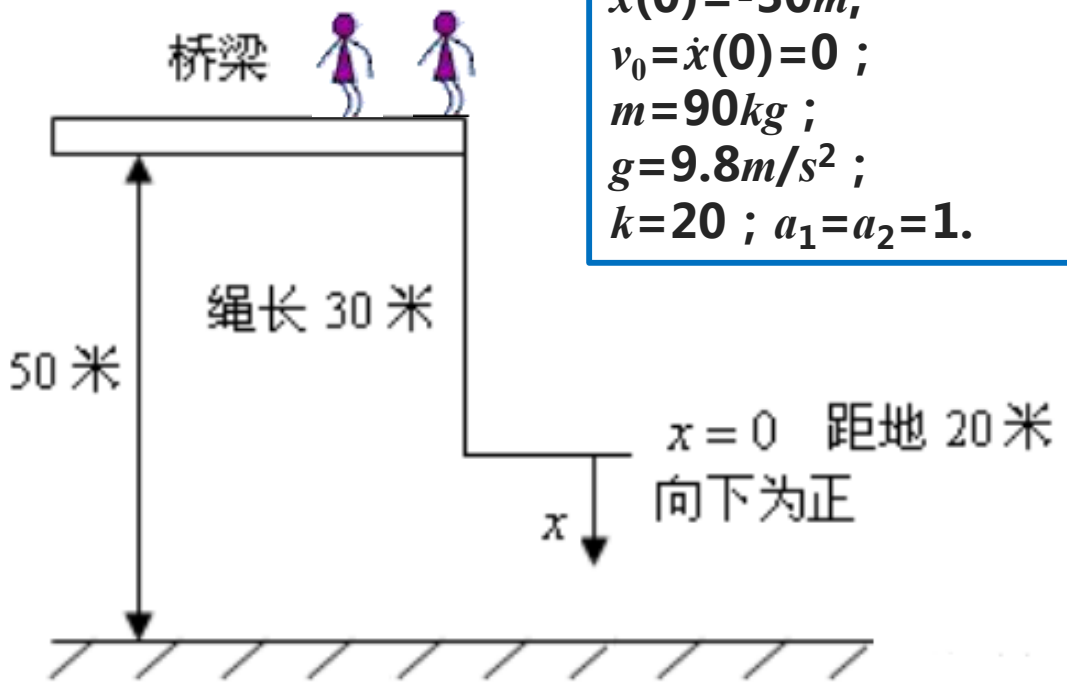
已知：m为人体的质量，g是重力加速度，x为拉伸位置，蹦极者受到的弹性力是$b(x)$，弹性系数为 k ,满足：

$$b(x)=\begin{cases}-kx, x>0\\ 0, x\le 0\end{cases}$$

a_1,a_2是空气阻尼系数，空气的阻力为：

$$f=-a_1\dot{x}-a_2|\dot{x}|\dot{x}$$

试:仿真分析其安全性。



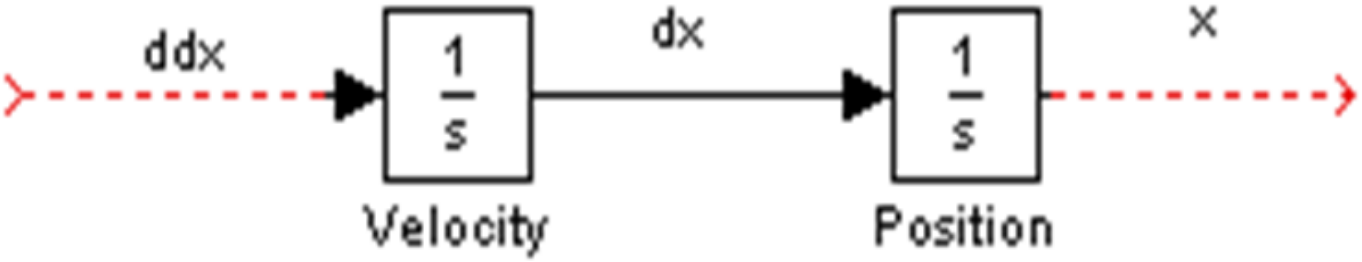
设绳索长度30m,即
$x(0)=-30m$;
$v_0=\dot{x}(0)=0$；
$m=90kg$；
$g=9.8m/s^2$；
$k=20$；$a_1=a_2=1$.

一.简单的模型实例

分析：系统方程可以表示为

$$m\ddot{x} = mg + b(x) - a_1\dot{x} - a_2|\dot{x}|\dot{x}$$

在MATLAB中建立这个方程的Simulink模型，这里需要使用两个积分器。



一旦x和x' 搭好，就可以：

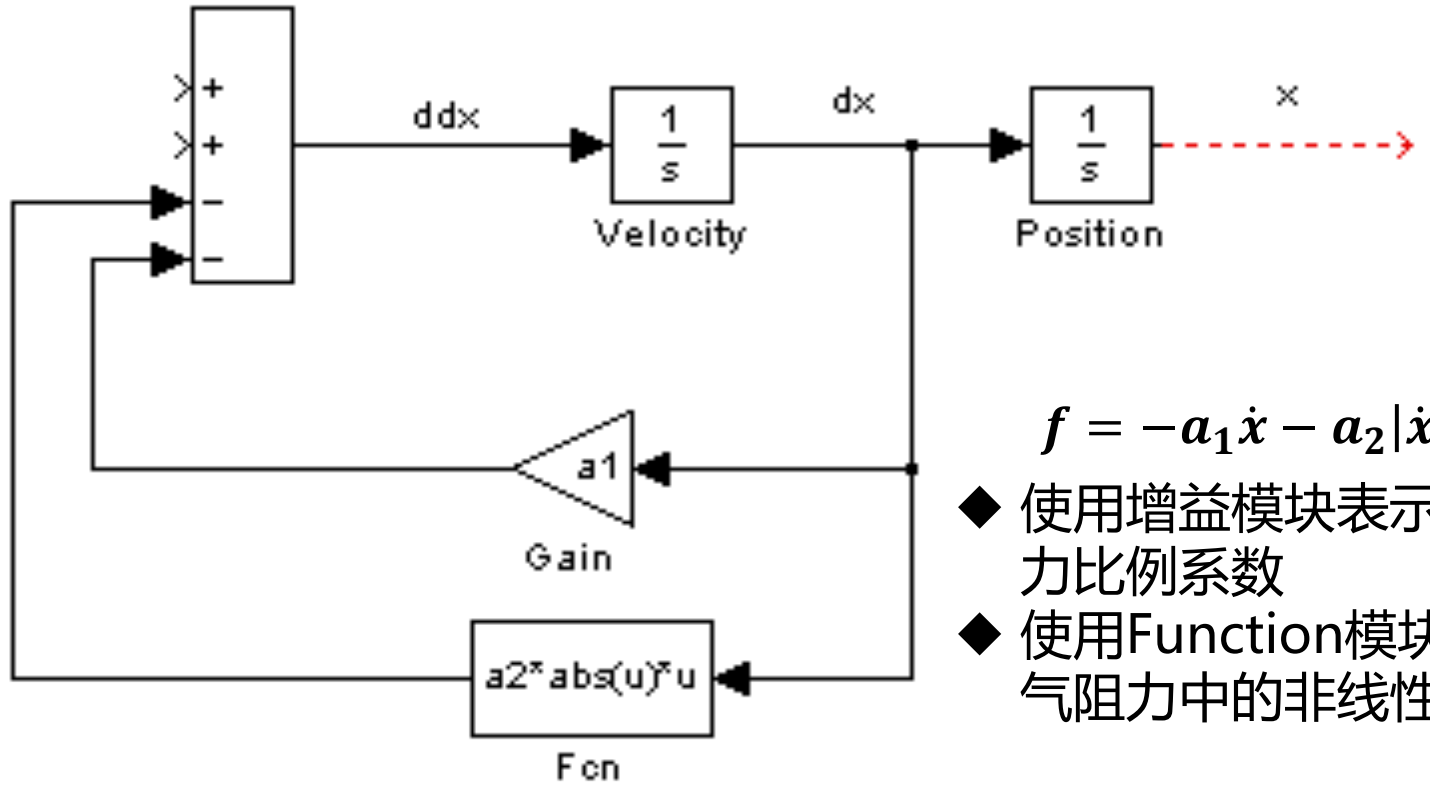
◆使用增益模块表示空气阻力比例系数

◆使用Function模块表示空气阻力中的非线性部分。

$$f = -a_1\dot{x} - a_2|\dot{x}|\dot{x}$$

一.简单的模型实例

$$m\ddot{x} = mg + b(x) - a_1\dot{x} - a_2|\dot{x}|\dot{x}$$

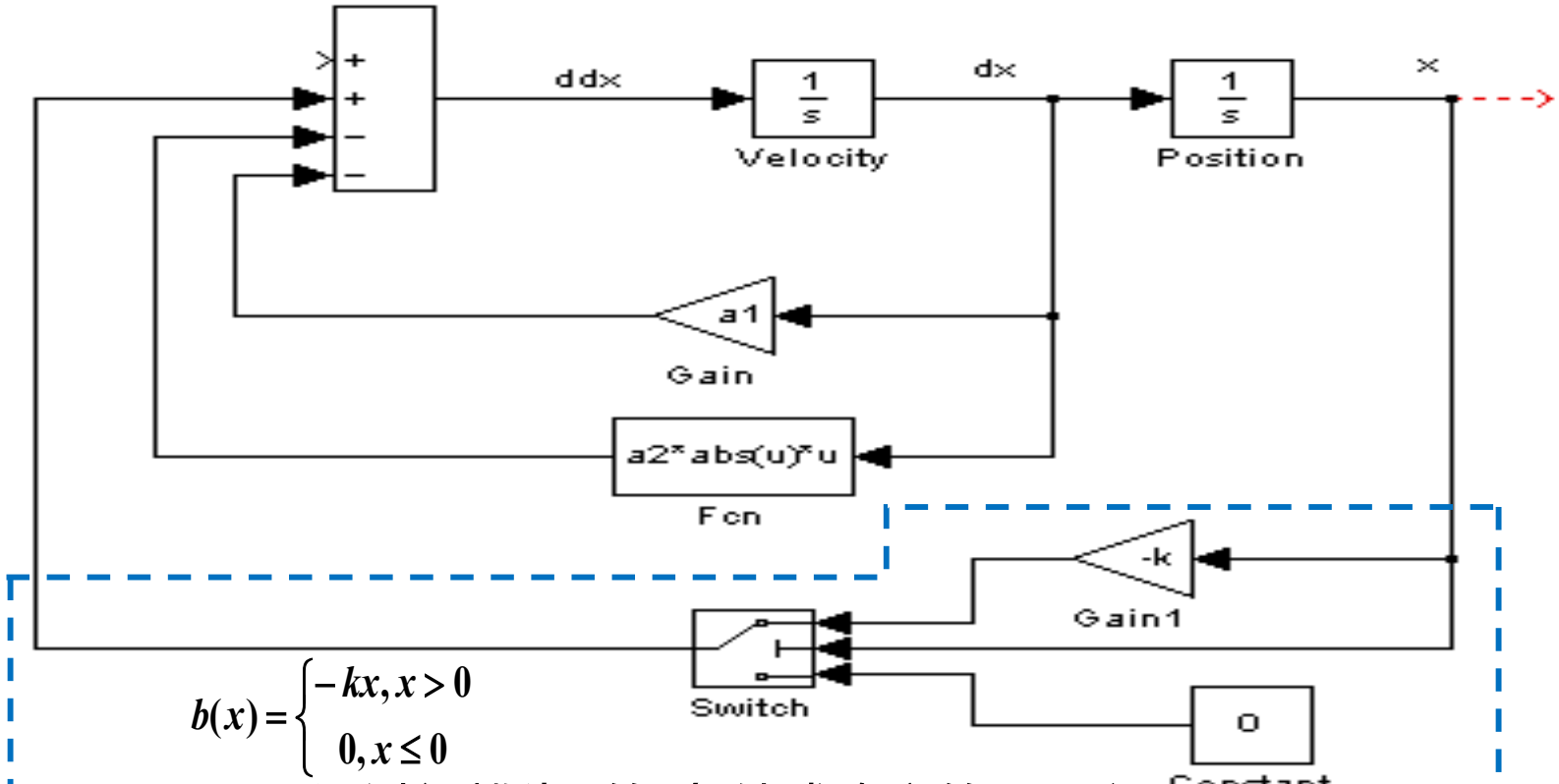


$$f = -a_1\dot{x} - a_2|\dot{x}|\dot{x}$$

- 使用增益模块表示空气阻力比例系数
- 使用Function模块表示空气阻力中的非线性部分。

一.简单的模型实例

$$m\ddot{x} = mg + b(x) - a_1\dot{x} - a_2|\dot{x}|\dot{x}$$

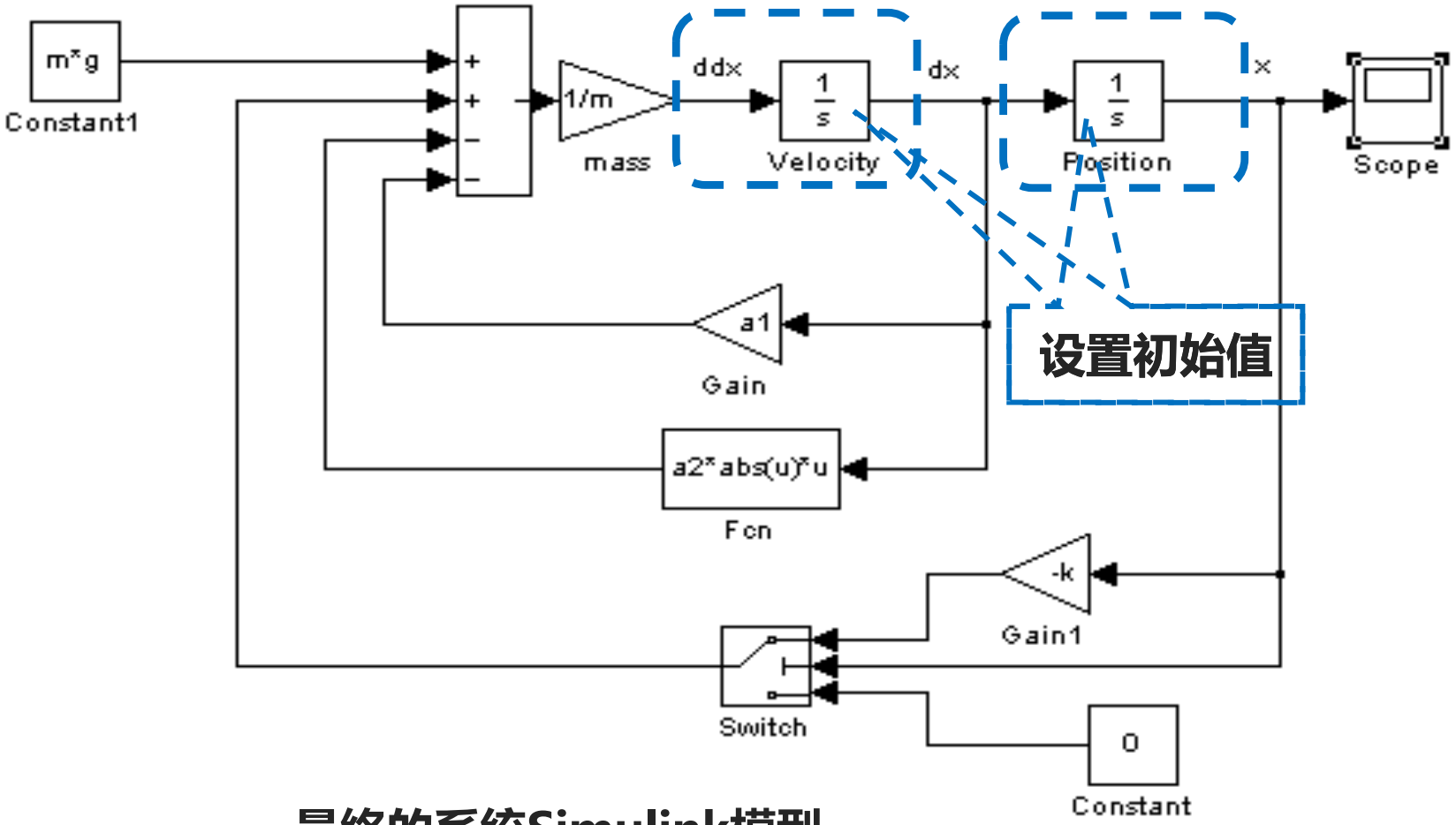


$$b(x) = \begin{cases} -kx, x > 0 \\ 0, x \le 0 \end{cases}$$

$b(x)$是通过门槛为0的x条件式确定的，可以使用一个Switch模块来实现判断条件。

一.简单的模型实例

$$m\ddot{x} = mg + b(x) - a_1\dot{x} - a_2|\dot{x}|\dot{x}$$

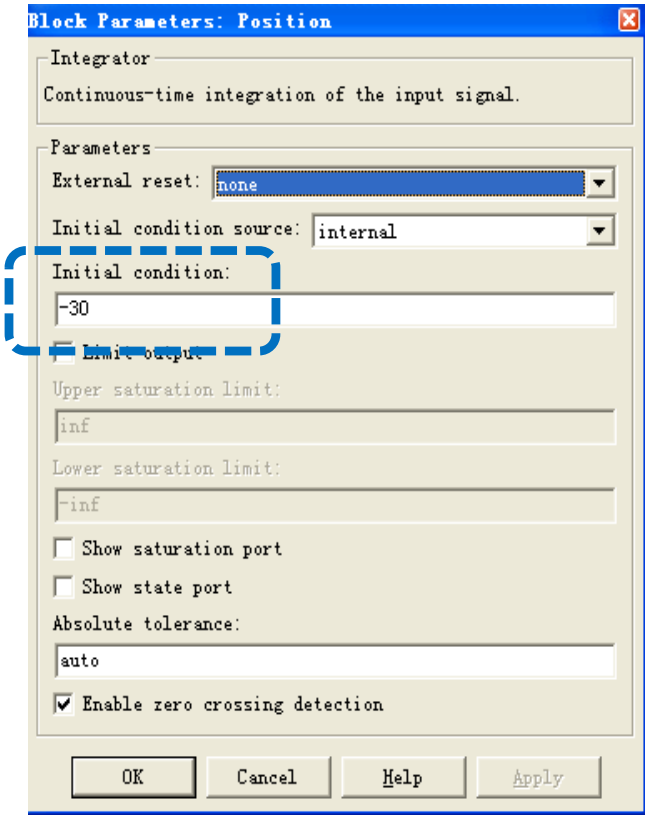


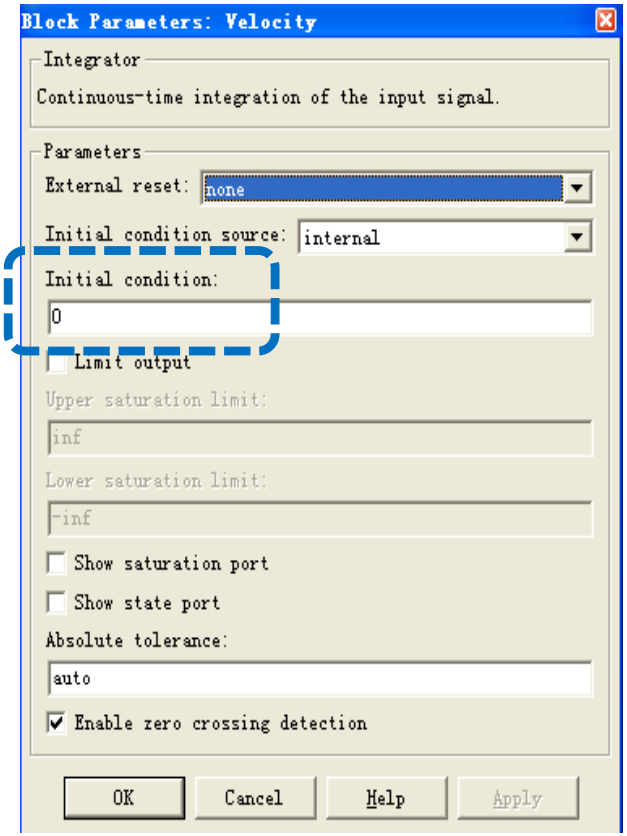
最终的系统Simulink模型

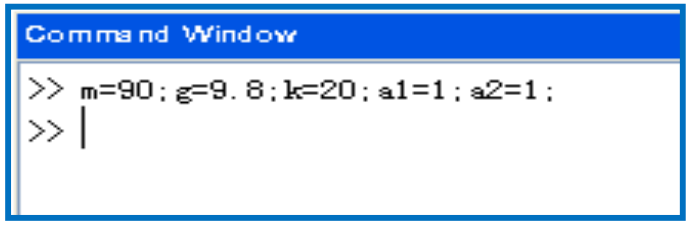
一.简单的模型实例

仿真过程中，设绳索长度-30m，起始速度为0；

物体质量为90kg，g为9.8m/s^2,弹性系数k为20，a_1和a_2均为1.

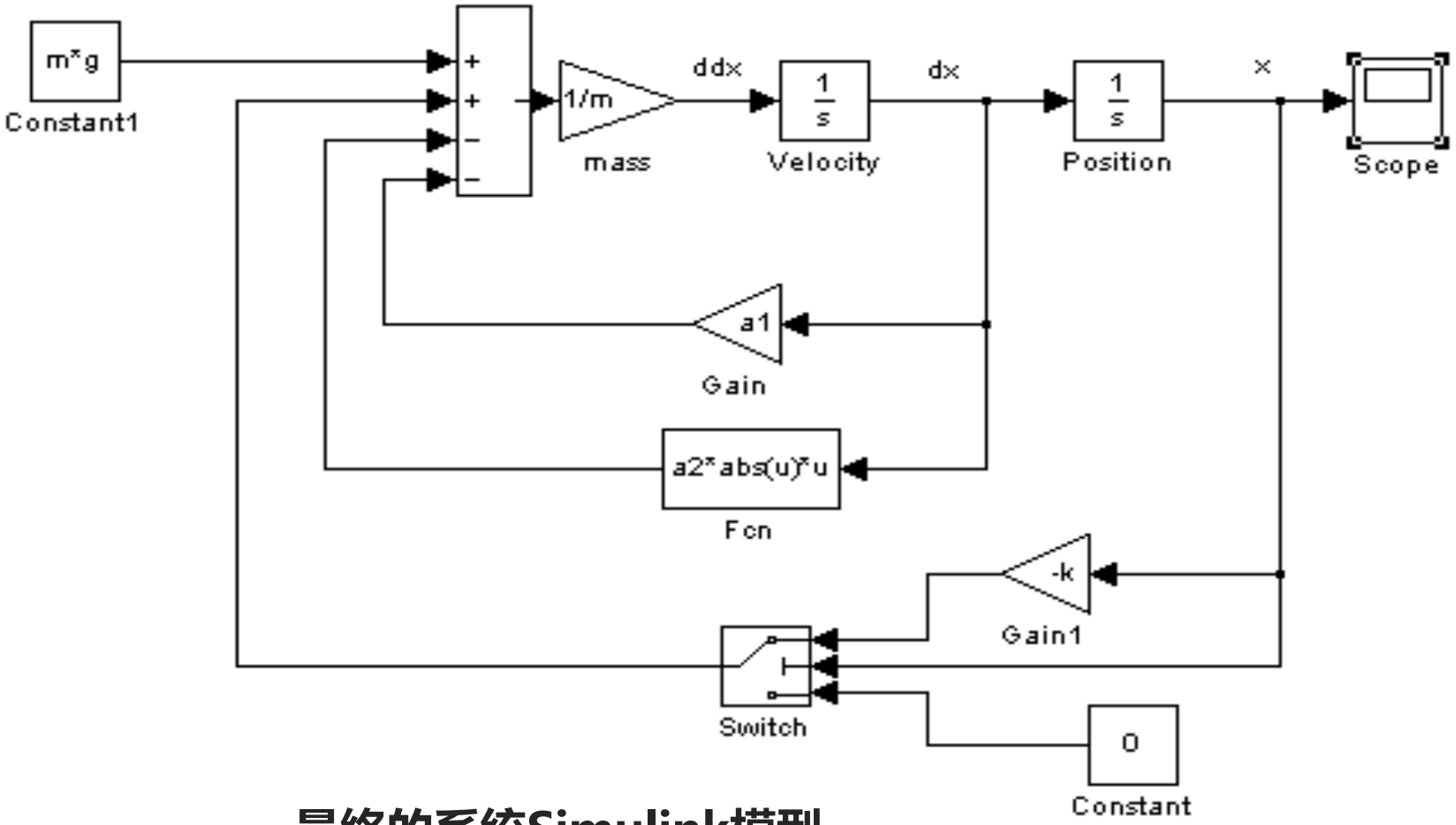






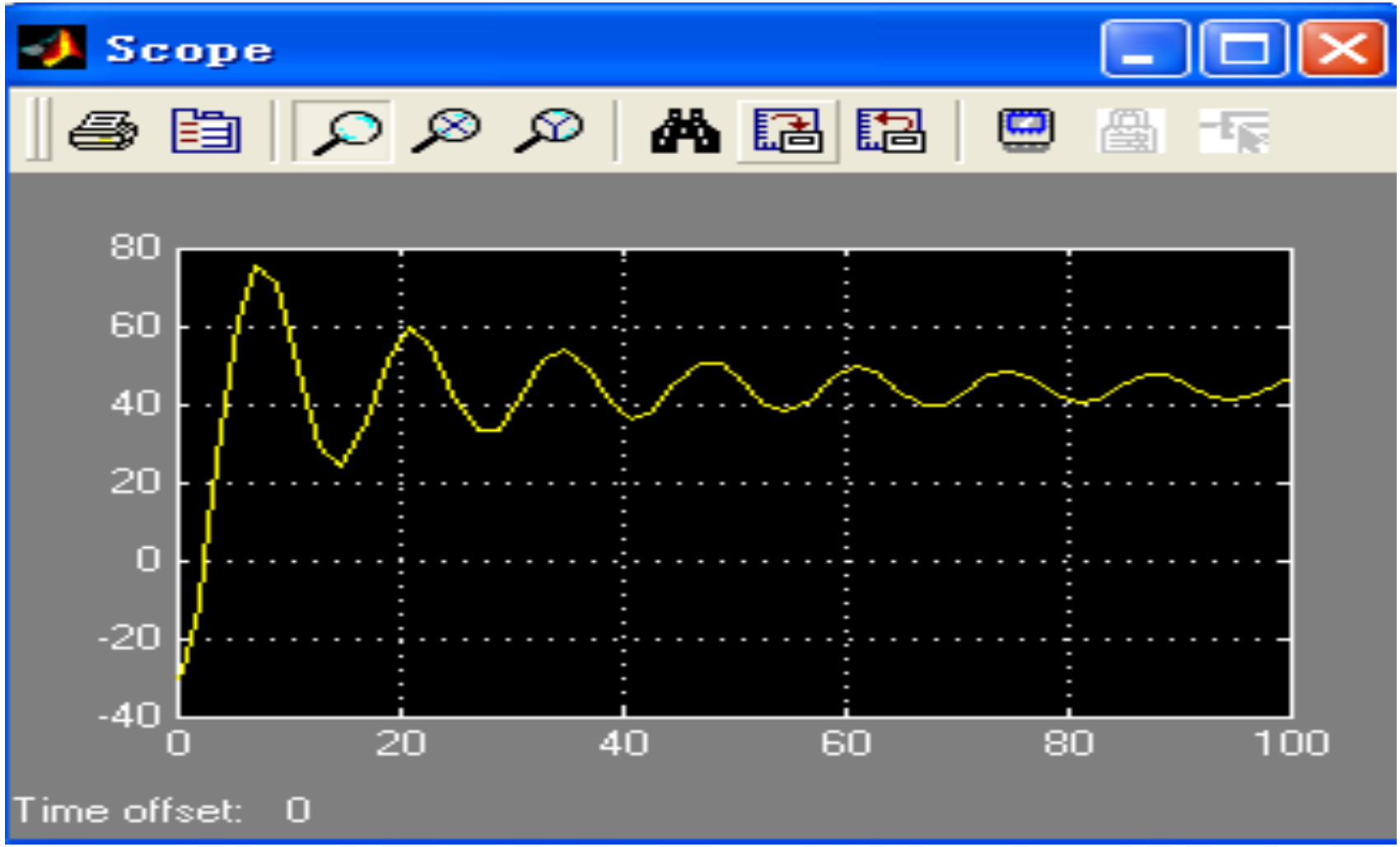
一.简单的模型实例

$$m\ddot{x} = mg + b(x) - a_1\dot{x} - a_2|\dot{x}|\dot{x}$$



最终的系统Simulink模型

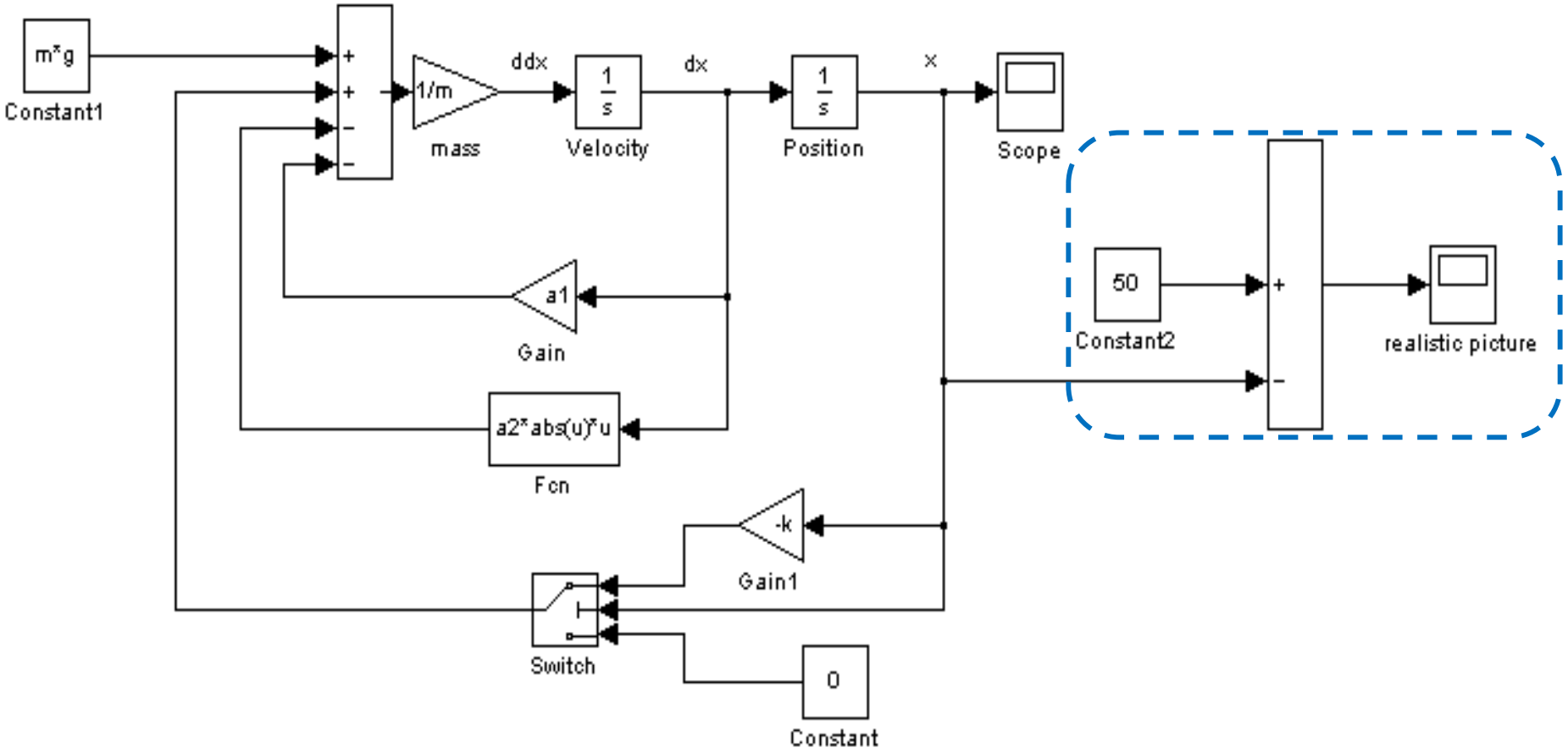
一.简单的模型实例



Simulink模型的仿真曲线

一.简单的模型实例

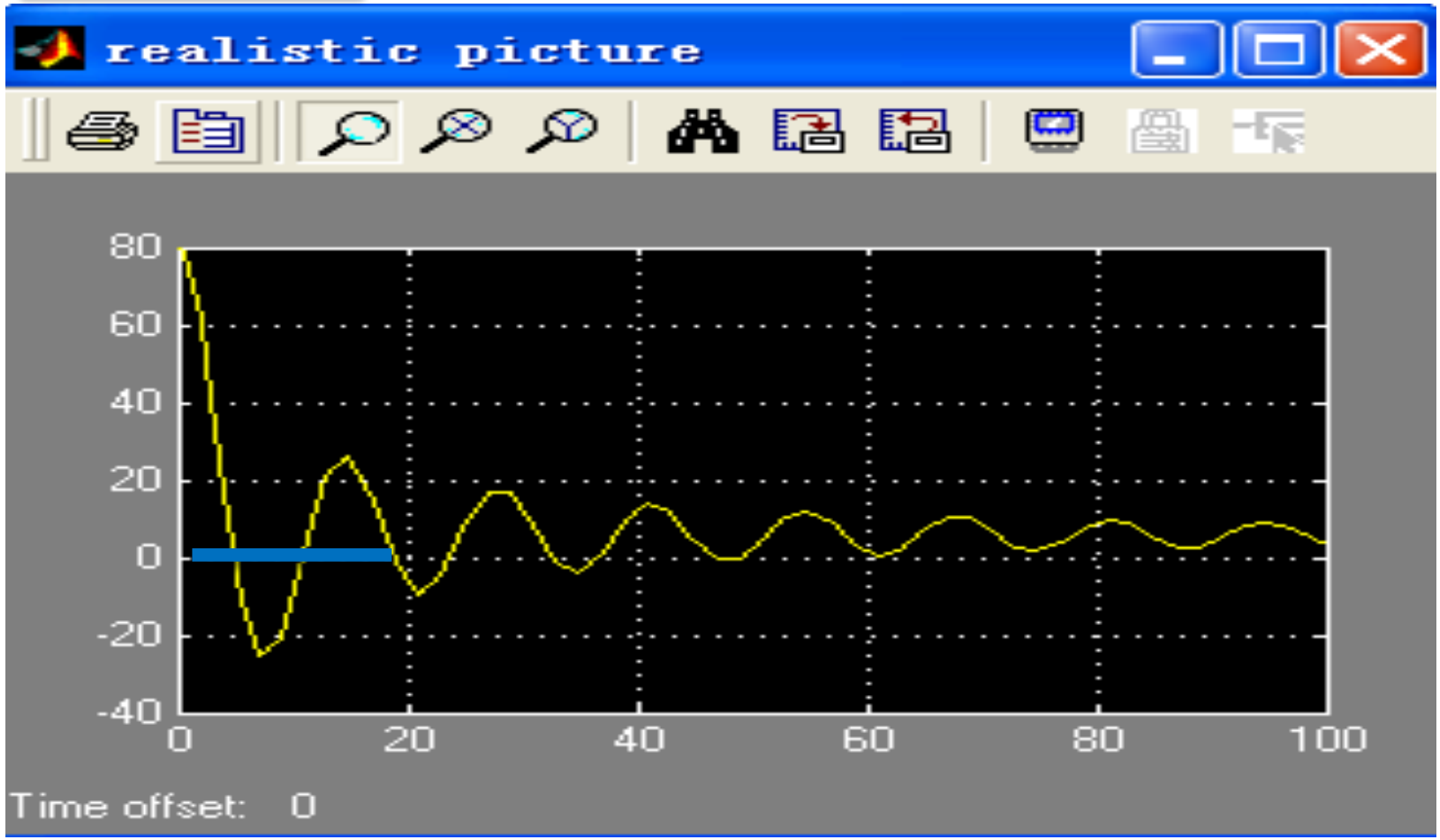
人站在桥的端部，距地面为50m,为了得到更真实的曲线，我们求人距离地面的高度，用50减去输出位置x。



考虑人距离地面高度时系统Simulink模型

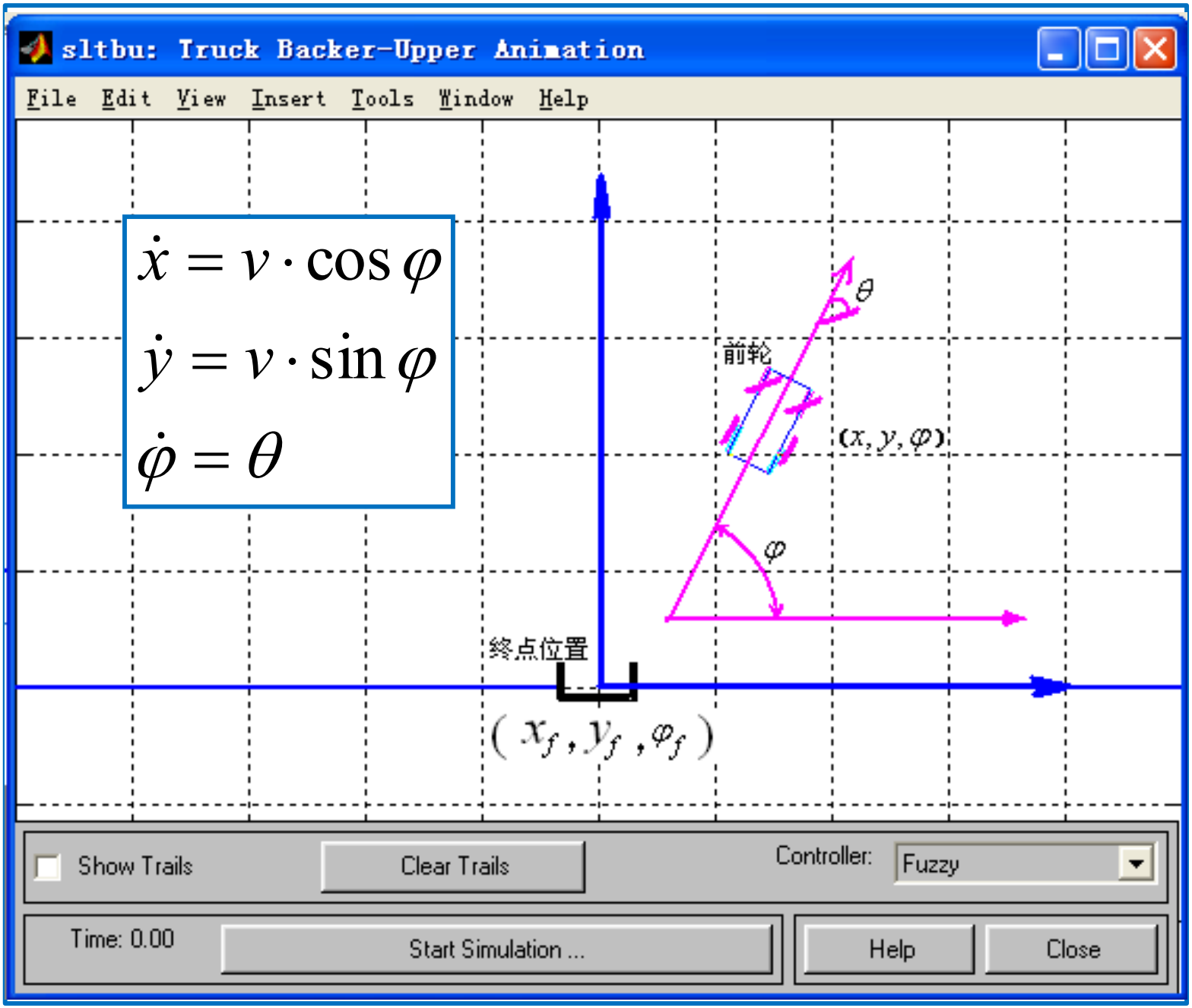
一.简单的模型实例

人站在桥的端部，距地面为50m,为了得到更真实的曲线，我们求人距离地面的高度，用50减去输出位置x，可以看到，眺跃者已经撞到了地上。



考虑人距离地面高度时系统Simulink模型

二.倒车系统的模型实例

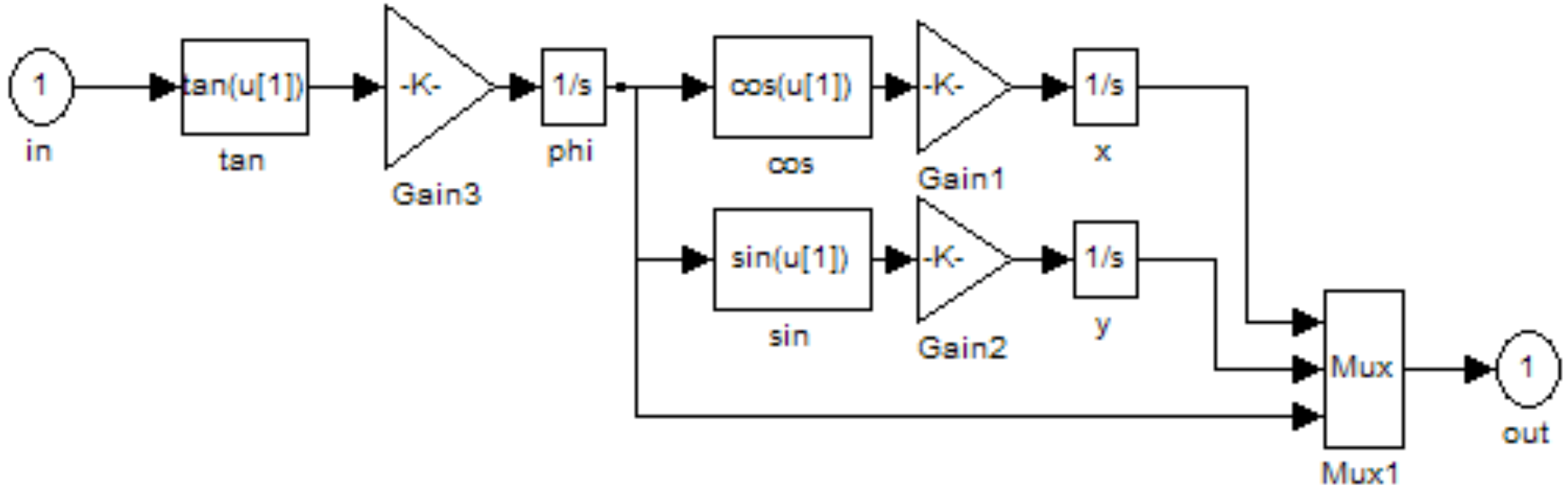

sltbu: Truck Backer-Upper Animation
File Edit View Insert Tools Window Help
$\dot{x} = v \cdot \cos\varphi$
$\dot{y} = v \cdot \sin\varphi$
$\dot{\varphi} = \theta$
θ
前轮
(x, y, φ)
φ
终点位置
(x_f, y_f, φ_f)
Show Trails
Clear Trails
Controller:
Fuzzy
Time: 0.00
Start Simulation ...
Help
Close

二.倒车系统的模型实例

$$\dot{x} = v \cdot \cos\varphi$$

$$\dot{y} = v \cdot \sin\varphi$$

$$\dot{\varphi} = \theta$$



三.倒立摆模型实例

精确倒立摆模型

$$\begin{cases} \ddot{x} = \dfrac{(J+ml^2)F + lm(J+ml^2)\sin\theta\cdot\dot{\theta}^2 - m^2l^2g\sin\theta\cos\theta}{(J+ml^2)(m_0+m) - m^2l^2\cos^2\theta} \\ \ddot{\theta} = \dfrac{ml\cos\theta\cdot F + m^2l^2\sin\theta\cos\theta\cdot\dot{\theta}^2 - (m_0+m)m\lg\sin\theta}{m^2l^2\cos^2\theta - (J+ml^2)(m_0+m)} \end{cases}$$

简化的倒立摆模型

$$\begin{cases} \ddot{x} = \dfrac{(J+ml^2)F - m^2l^2g\theta}{J(m_0+m)+m_0ml^2} \\ \ddot{\theta} = \dfrac{(m_0+m)m\lg\theta - mlF}{J(m_0+m)+m_0ml^2} \end{cases}$$

三.倒立摆模型实例

当小车的质量m_0=1kg；倒摆振子的质量m=1.5kg；倒摆长度 2l=0.8m；

重力加速度取$g=10m/s^2$时得

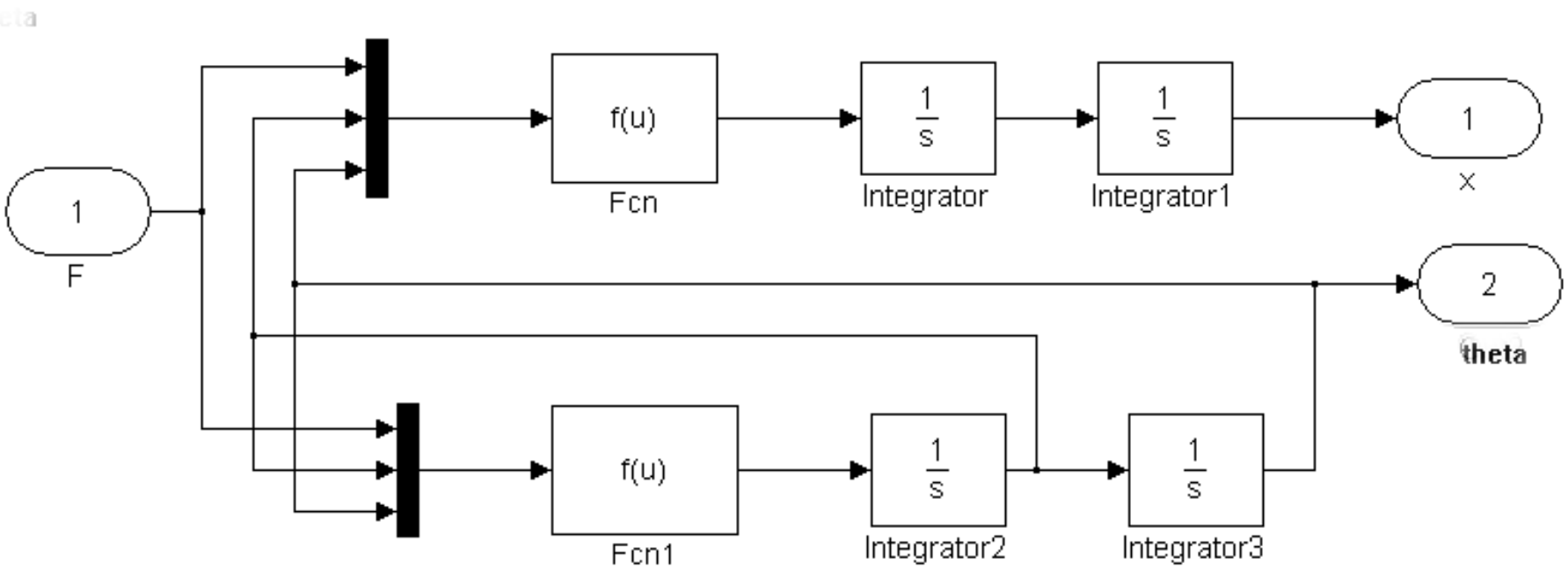
精确倒立摆模型

$$\begin{cases} \ddot{x}=\dfrac{0.32F+0.0192\sin\theta\cdot\dot{\theta}^2-3.6\sin\theta\cos\theta}{0.8-0.36\cos^2\theta} \\ \ddot{\theta}=\dfrac{0.06\cos\theta F+0.36\sin\theta\cos\theta\cdot\dot{\theta}^2-15\sin\theta}{0.36\cos^2\theta-0.8} \end{cases}$$

简化的倒立摆模型

$$\begin{cases} \ddot{x}=\dfrac{-3.6\theta+0.32F}{0.44} \\ \ddot{\theta}=\dfrac{15\theta-0.6F}{0.44} \end{cases}$$

三.倒立摆模型实例


theta
1
F
f(u)
Fcn
1
s
Integrator
1
s
Integrator1
1
x
2
theta
f(u)
Fcn1
1
s
Integrator2
1
s
Integrator3

三.倒立摆模型实例

Fcn:

(0.32*u[1]+0.0192*sin(u[3])*power(u[2],2)-3.6*sin(u[3])*cos(u[3]))/(0.8-0.36*powe

r(cos(u[3]),2))

Fcn1:

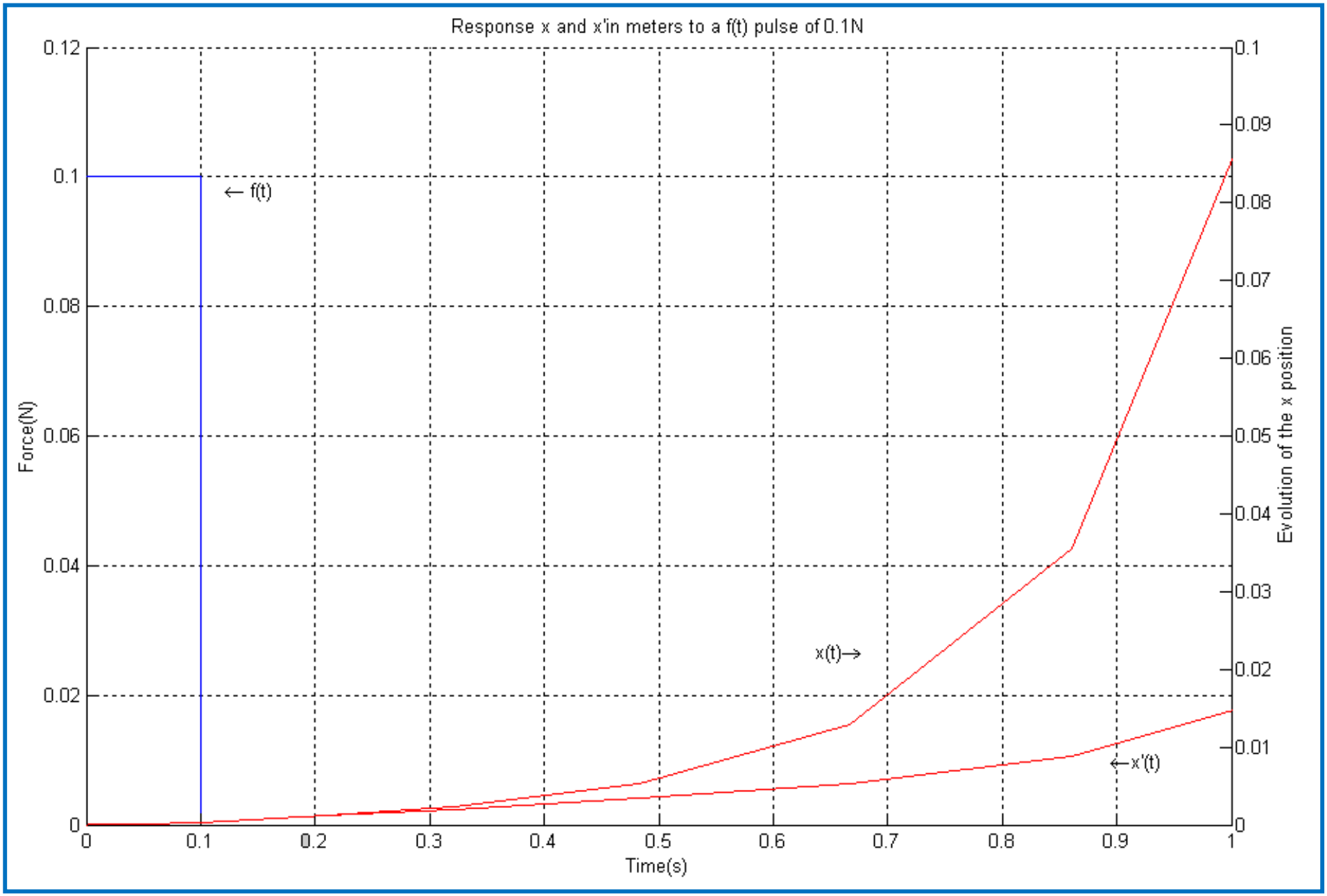
(0.06*cos(u[3])*u[1]+0.36*sin(u[3])*cos(u[3])*power(u[2],2)-15*sin(u[3]))/(0.36*po

wer(cos(u[3]),2)-0.8)

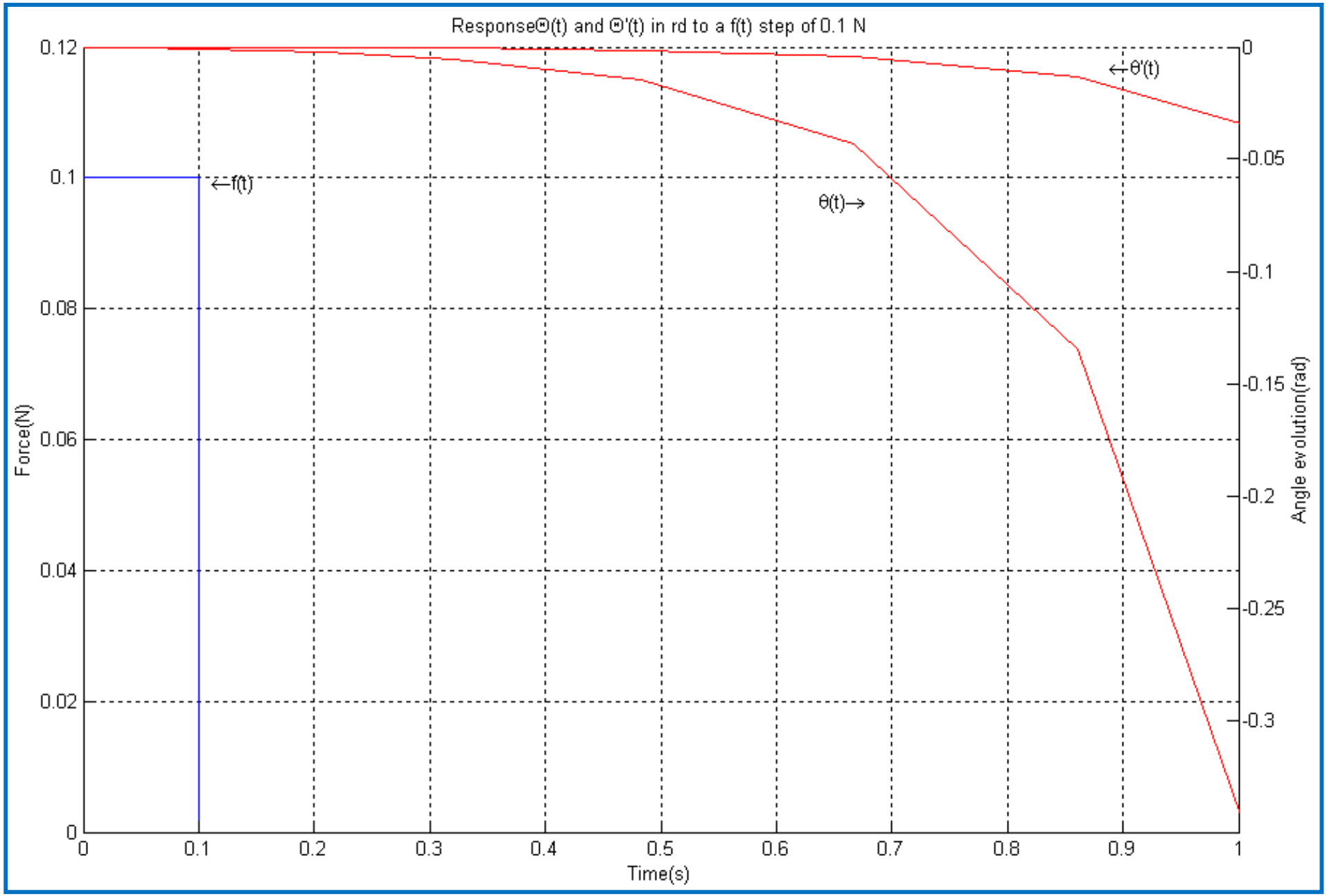
Fcn2: 0.32*u[1]/0.44-3.6*u[3]/0.44

Fcn3: 15*u[3]/0.44-0.6*u[1]/0.44

三.倒立摆模型实例

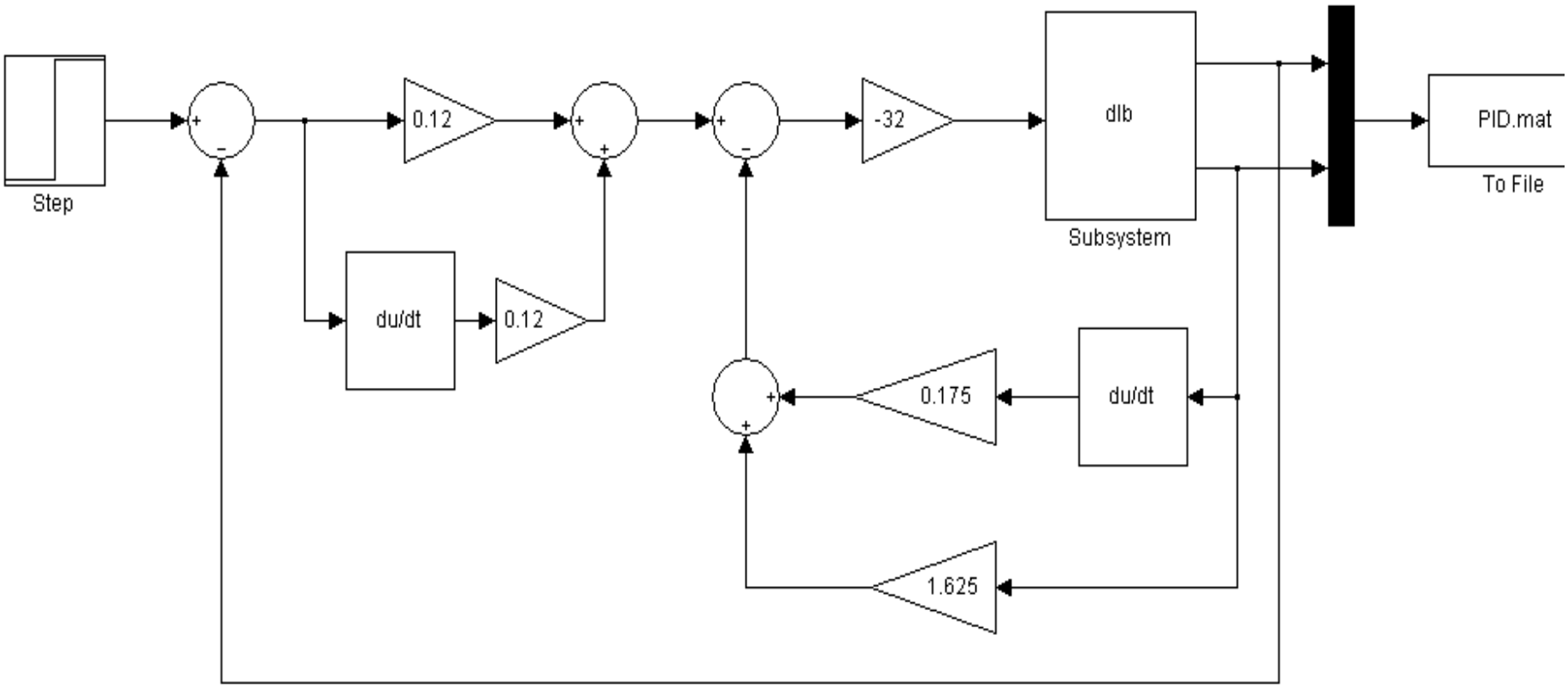

Response x and x'in meters to a f(t) pulse of 0.1N
← f(t)
x(t)→
←x'(t)
Force(N)
Evolution of the x position
Time(s)
0.12
0.1
0.08
0.06
0.04
0.02
0
0.09
0.07
0.05
0.03
0.01
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

三.倒立摆模型实例


Response Θ(t) and Θ'(t) in rd to a f(t) step of 0.1 N
←f(t)
θ(t)→
←θ'(t)
Force(N)
Angle evolution(rad)
Time(s)
0.12
0.1
0.08
0.06
0.04
0.02
0
-0.05
-0.1
-0.15
-0.2
-0.25
-0.3
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

三.倒立摆模型实例

具有双闭环PID控制器的倒立摆控制模型



三.倒立摆模型实例

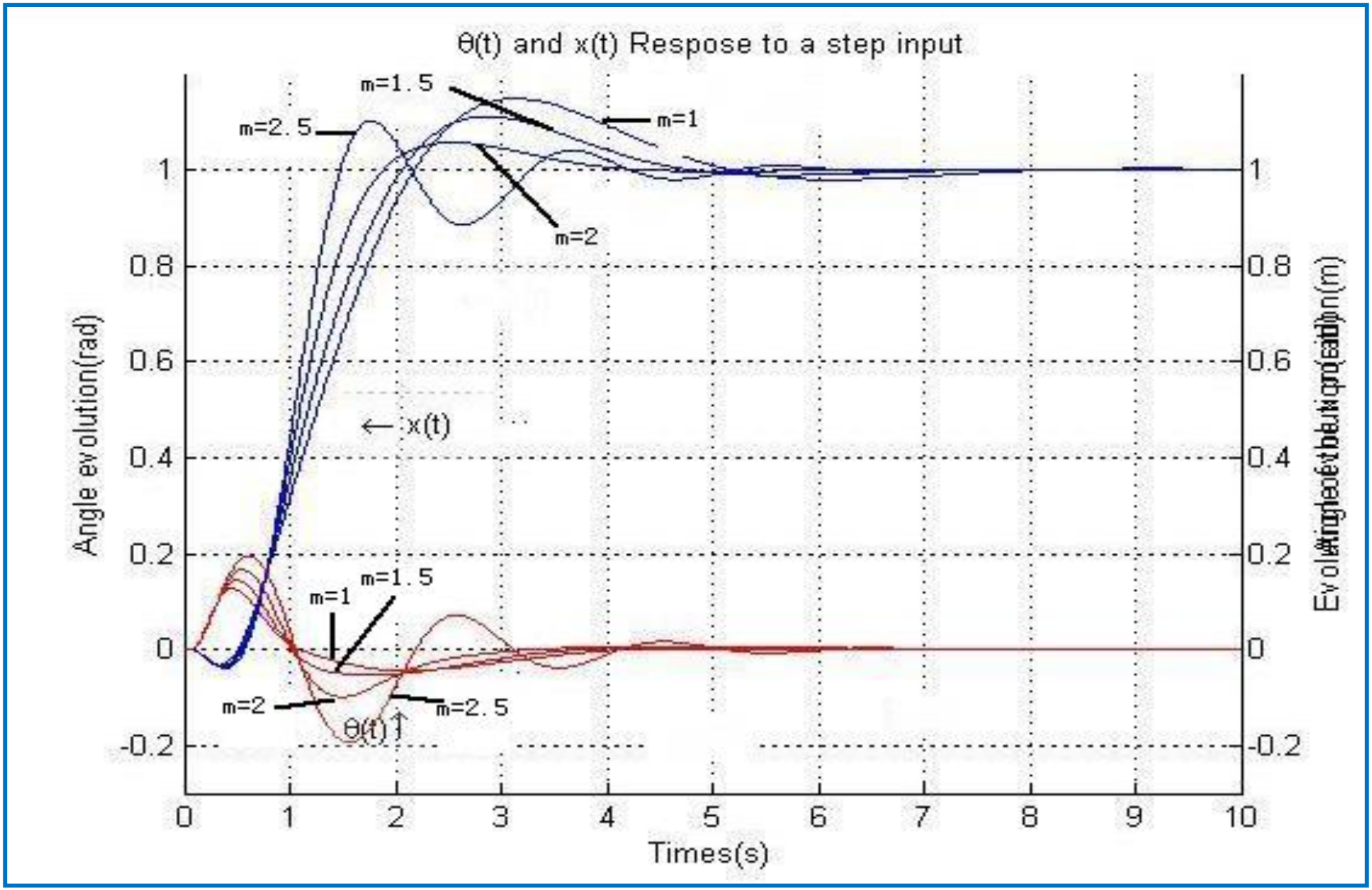


图3-12摆长不变而摆杆质量变化时系统仿真结果

三.倒立摆模型实例

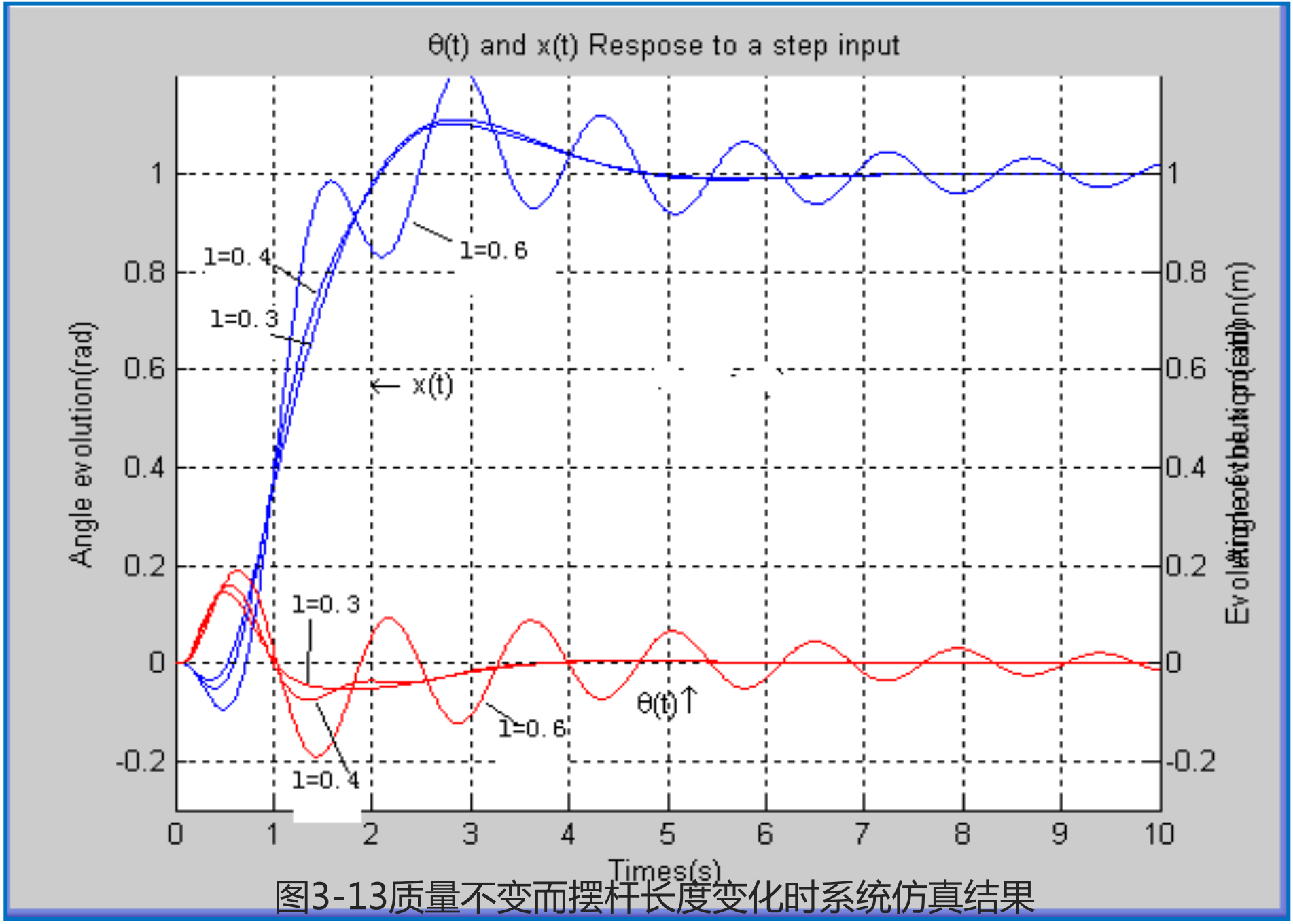


图3-13质量不变而摆杆长度变化时系统仿真结果

PART THREE

数字仿真中的代数环问题

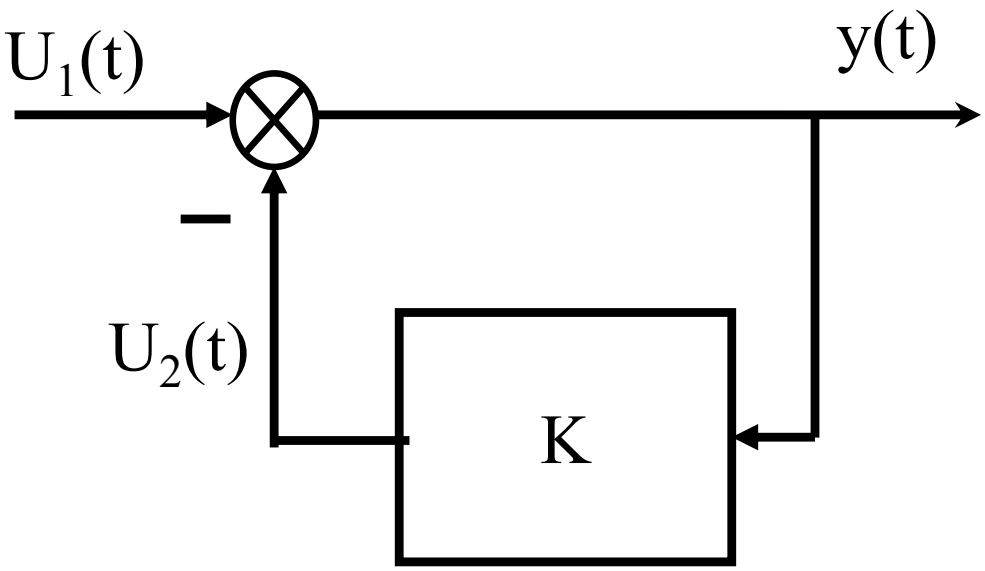
- 一、问题的提出
- 二、代数环产生的条件
- 三、消除代数环的方法

一.问题的提出

反馈是控制系统中普遍存在的环节，在进行数字仿真时，计算机按照一定的时序执行相应的计算步骤，对于反馈回路就有一个输入和输出计算顺序的问题。

在相当普遍的条件下，当一个系统的输入直接取决于输出，同时输出也直接取决于输入时，仿真模型中便出现“代数环”问题。

一.问题的提出



a）仿真模型

$y(t)=u_1(t)-u_2(t)$

$u_2(t)= Ky(t)$

b）死锁环路

仿真模型的输出反馈信号作为输入信号的一部分，在进行仿真时，按正常的计算顺序应该先计算模块的输入，然后再计算由输入驱动的输出。

然而由于输入与输出相互制约，这就形成了一个死锁环路，也就是所谓的“代数环”问题。

一.问题的提出

代数环的定义：

当一个仿真模型中存在一个闭合回路，并且回路中每个模块/环节都是直通的，即模块/环节中的一部分直接到达输出，这样一个闭合回路就是“代数环”。

二.“代数环”产生的条件

代数环存在的充分必要条件：

在系统仿真模型中，存在一个闭合回路，该闭合回路中每个模块/环节都是直通模块/环节。

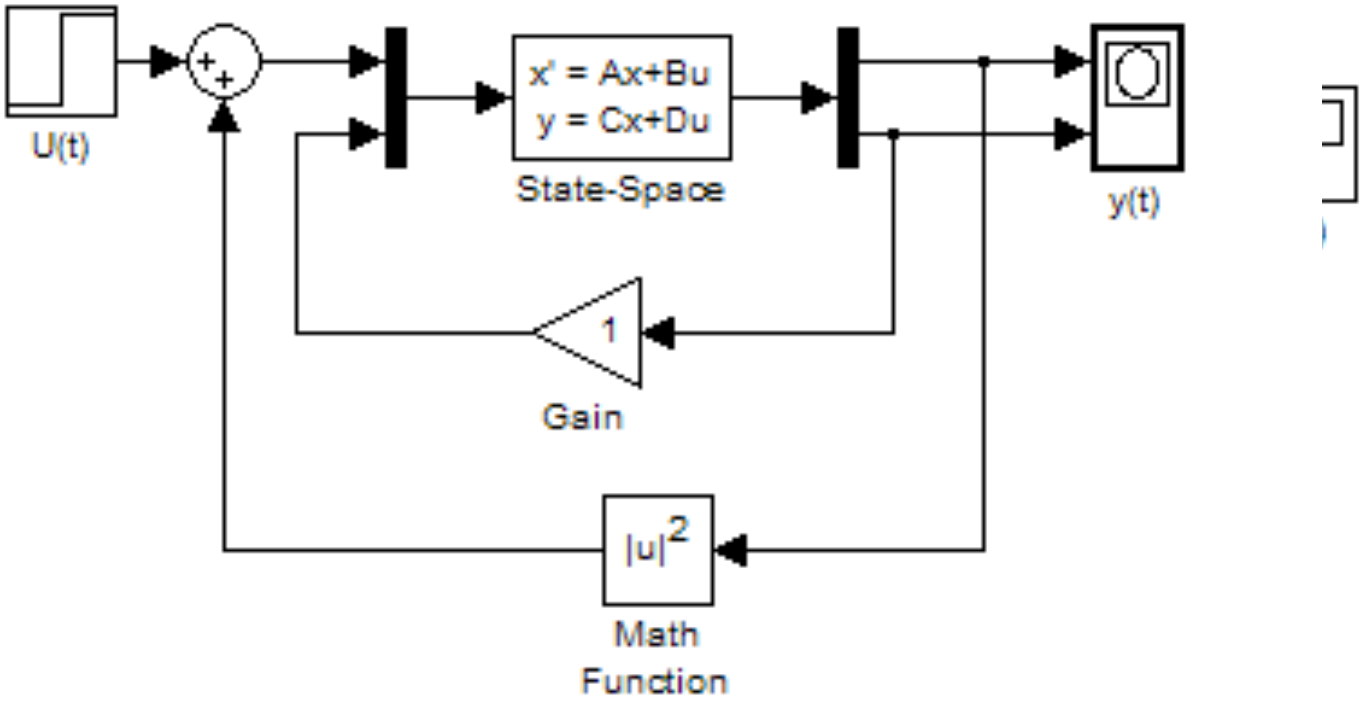
所谓直通，指的是模块/环节输入中的一部分直接到达输出。

如果一个反馈回路的正向通道和反向通道都由直通模块组成，则次反馈回路一定构成“代数环”。

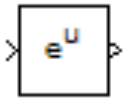
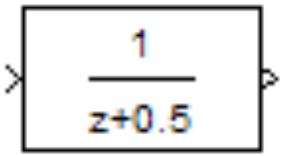
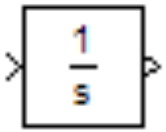
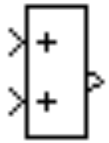
对于复杂反馈回路来说，只要能够找到由直通模型构成的闭合路径，则也一定构成“代数环”。

二.“代数环”产生的条件

常见的几种代数环：

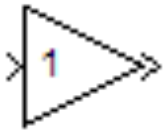
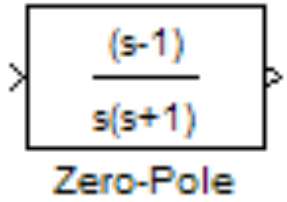
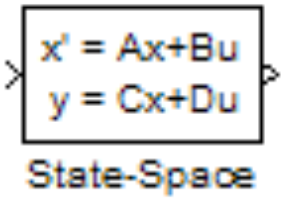
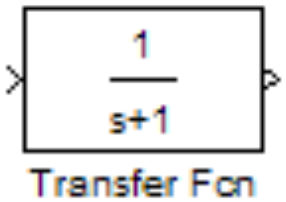


二.“代数环”产生的条件——直通模块

模块	所属模块库	说明
	Simulink/Math	任意情况
	Simulink/Discrete	当离散传递函数的分子与分母阶次相同时
	Simulink/Continuous	从初始条件输入端到输出端的直通
	Simulink/Math	任意情况

Simulink模块库中一些典型的直通模块：

二.“代数环”产生的条件——直通模块

模块	所属模块库	说明
	Simulink/Math	任意情况
	Simulink/Continuous	当极点数目零点数目相同时
	Simulink/Continuous	当D矩非零时
	Simulink/Continuous	当传递函数的分子与分母阶次相同时

Simulink模块库中一些典型的直通模块：

二.“代数环”产生的条件——直通模块

产生“代数环”的一般条件：

1.前馈通道中含有信号的“直通”模块，如比例环节或含有初值输出的积分器。

2.系统中的大部分模型表现为非线性。

3.前馈通道传递函数的分子与分母同阶。

4.用状态空间描述系统时，输出方程中D矩阵非零。

三.消除代数环的方法

- ✓变换法
- ✓拆解法

三.消除代数环的方法

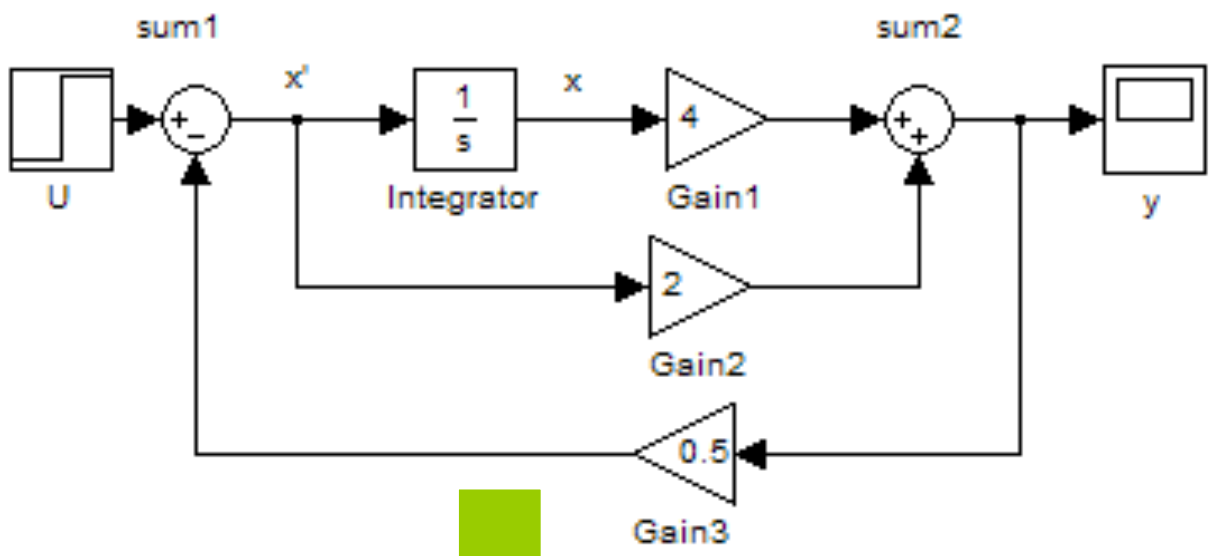
✓变换法

“代数环”在形式上是一种数字仿真模型，对应的是数学模型，通常表现为一个非常或方程组。当方程的右边项中包含有方程的左边项时，如果用simulink去直接实现该方程，将产生“代数环”。

如果先将原始数学模型进行变换，使得方程的右边项中不包含有方程的左边项，然后再用simulink去实现，则可以消除“代数环”，从本质上是将“代数环”隐函数变换为显函数的方法。

三.消除代数环的方法

$$\begin{cases} \dot{x} = u - 0.5y \\ y = 4x + 2\dot{x} \end{cases}$$




```
Warning: Block diagram 'daishuhuan4' contains 1 algebraic loop(s).
Found algebraic loop containing block(s):
  'daishuhuan4/Gain3'
  'daishuhuan4/Sum'
  'daishuhuan4/Gain2'
  'daishuhuan4/Sum1' (algebraic variable)
>>
```

三.消除代数环的方法

✓变换法局限性：

- 并不是所有的隐函数都可以求解得到显函数；
- 原始的数学模型往往反映了仿真对象的物理结构，按照物理结构构造仿真模型可以实现所谓的同构仿真，按照变换后得到的数学模型构造的仿真模型则只能实现同态仿真，同构仿真比同态仿真具有更好的可信度，也具有更大的灵活性。

三.消除代数环的方法

- ✓拆解法

“代数环”在形式上是一个闭合回路，而且回路中的每一个模块都必须是直通模块，在保持功能不变的同时，如果能够在回路中产生一个非直通模块，则该“代数环”就被拆解了。

三.消除代数环的方法

✓拆解法

“代数环”的拆解有多种方法，这里主要介绍三种方法：

1.插入存储器模块拆解“代数环”；

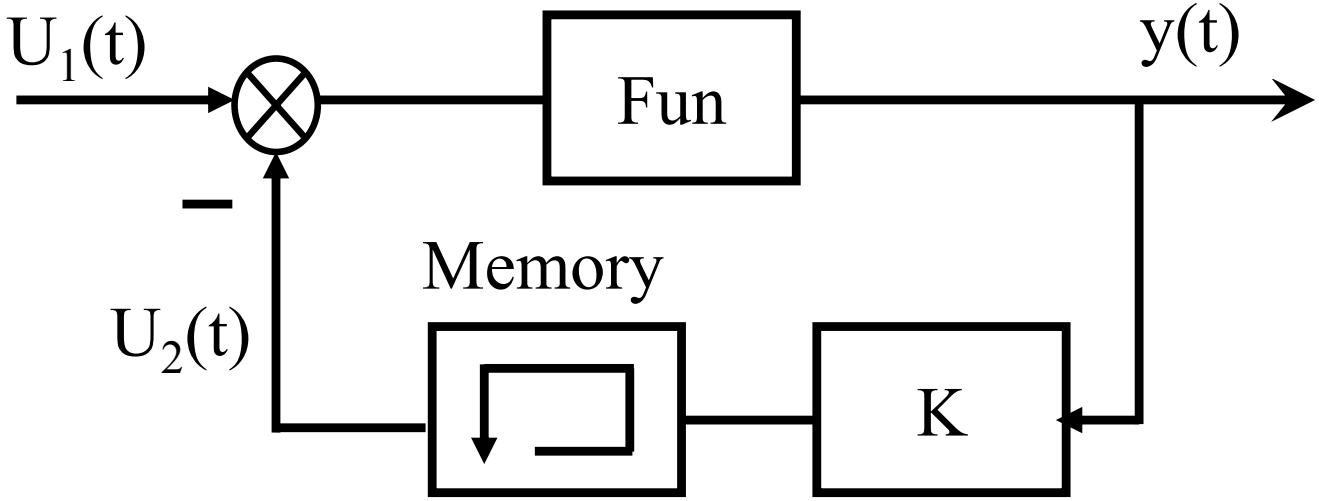
2.通过模型替代或重构的方法消除直通模块；

3.用simulnk提供的专门手段拆解代数环；

三.消除代数环的方法

✓拆解法

1.插入存储器模块拆解“代数环”；



三.消除代数环的方法

✓拆解法

2.通过模型替代或重构的方法消除直通模块；

在一定条件下，“代数环”一些直通模块可以用具有相同功能的非直通模块替代

三.消除代数环的方法

✓拆解法

3.用simulnk提供的专门手段拆解代数环；

- 代数约束模块
- 积分模块的状态输出端

04

PART FOUR

SIMULINK的扩展工具——S-函数

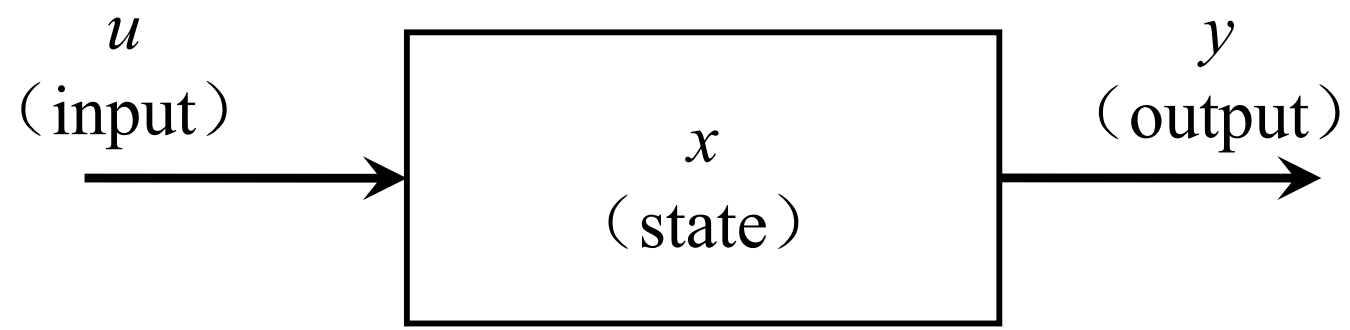
- 一、什么是S-函数
- 二、 S-函数的仿真流程
- 三、 S-函数的编写以及应用

一. 什么是S-函数

- S-函数为Simulink的“系统”函数
- 能够响应Simulink求解器命令的函数
- 采用非图形化的方法实现一个动态系统
- 可以开发新的simulink模块
- 可以与已有的代码相结合进行仿真
- 采用文本方式输入复杂的系统方程
- 扩展Simulink功能——M文件S-函数扩展图形功能
- S-函数的语法结构是为实现一个动态系统而设计的（默认用法），其他S-函数的用法是默认用法的特例（如用于显示目的）

一. 什么是S-函数

- **S-函数工作原理**



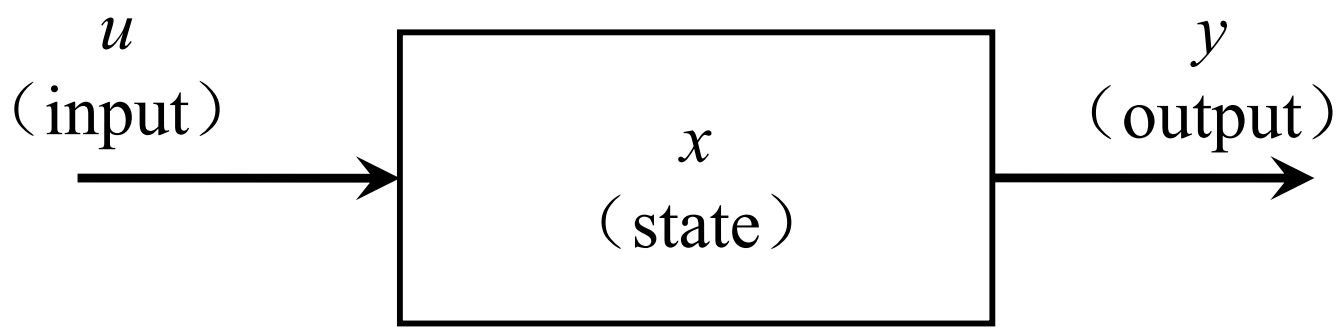
输出方程：$y = f_0(t, x, u)$

连续状态方程：$dx = f_d(t, x, u)$

离散状态方程：$x_{k+1} = f_u(t, x, u)$　其中$x = [dx, x_{k+1}]$

一. 什么是S-函数

- **S-函数工作原理**

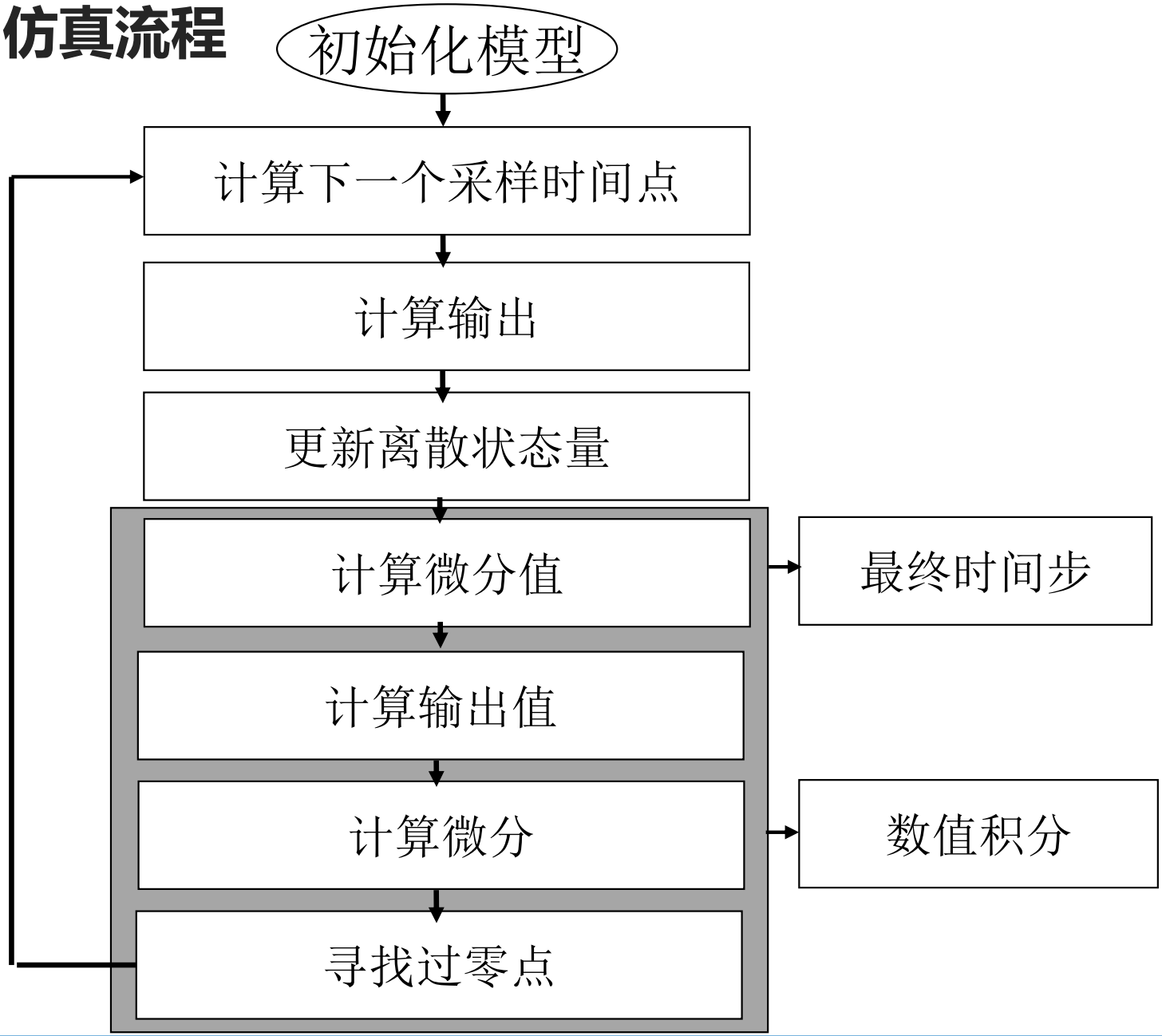


状态方程：$\dot{x} = f_c(x,u,t)$

输出方程：$y = g(x,u,t)$

二. S-函数的仿真流程

- **S-函数的仿真流程**



二. S-函数的仿真流程

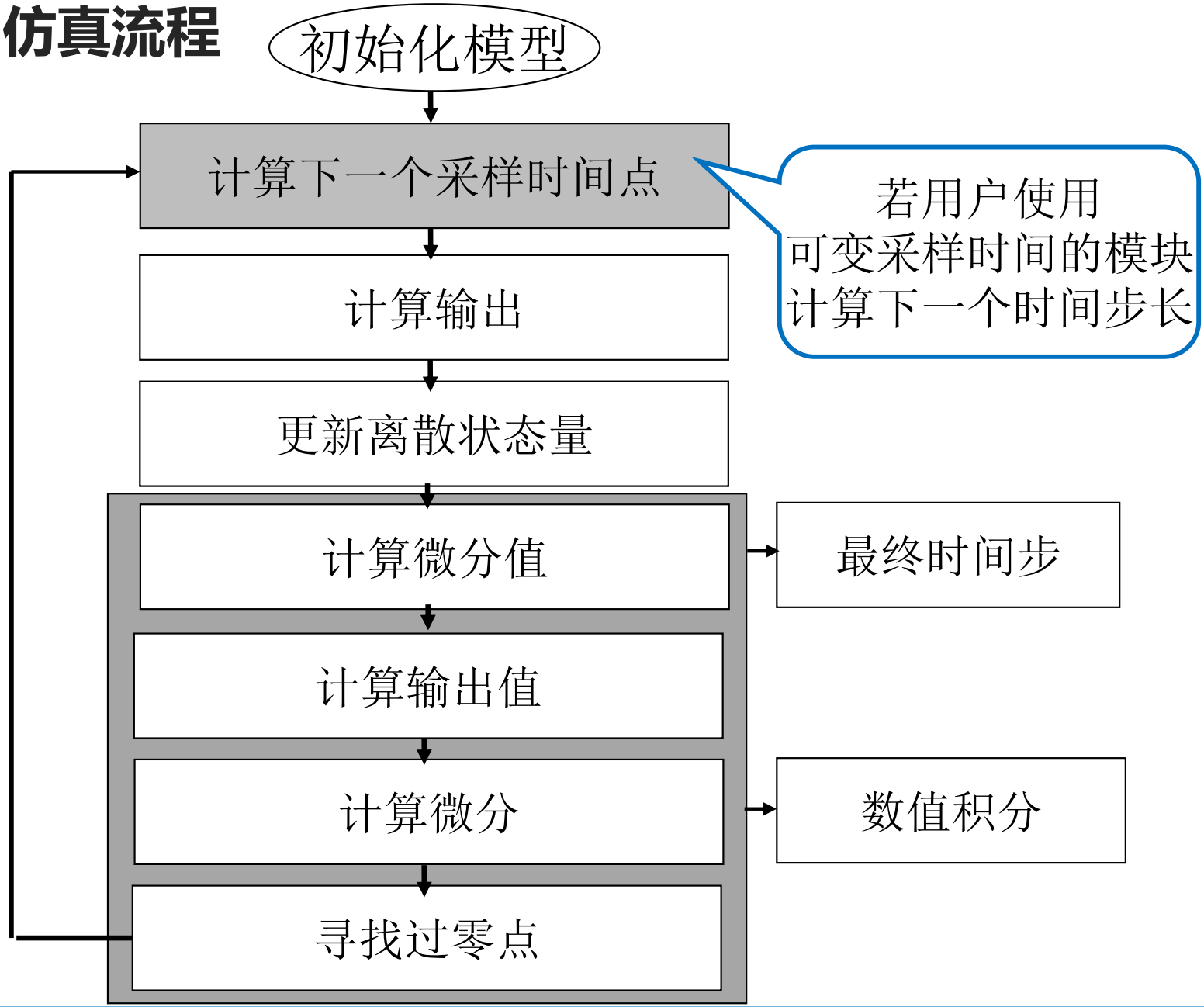
- S-函数的仿真流程

初始化

- 初始化包含S函数信息的仿真结构SimStruct
- 设置输入输出端口的数目和维数
- 设置模块的采样时间
- 分派内存区和sizes数组

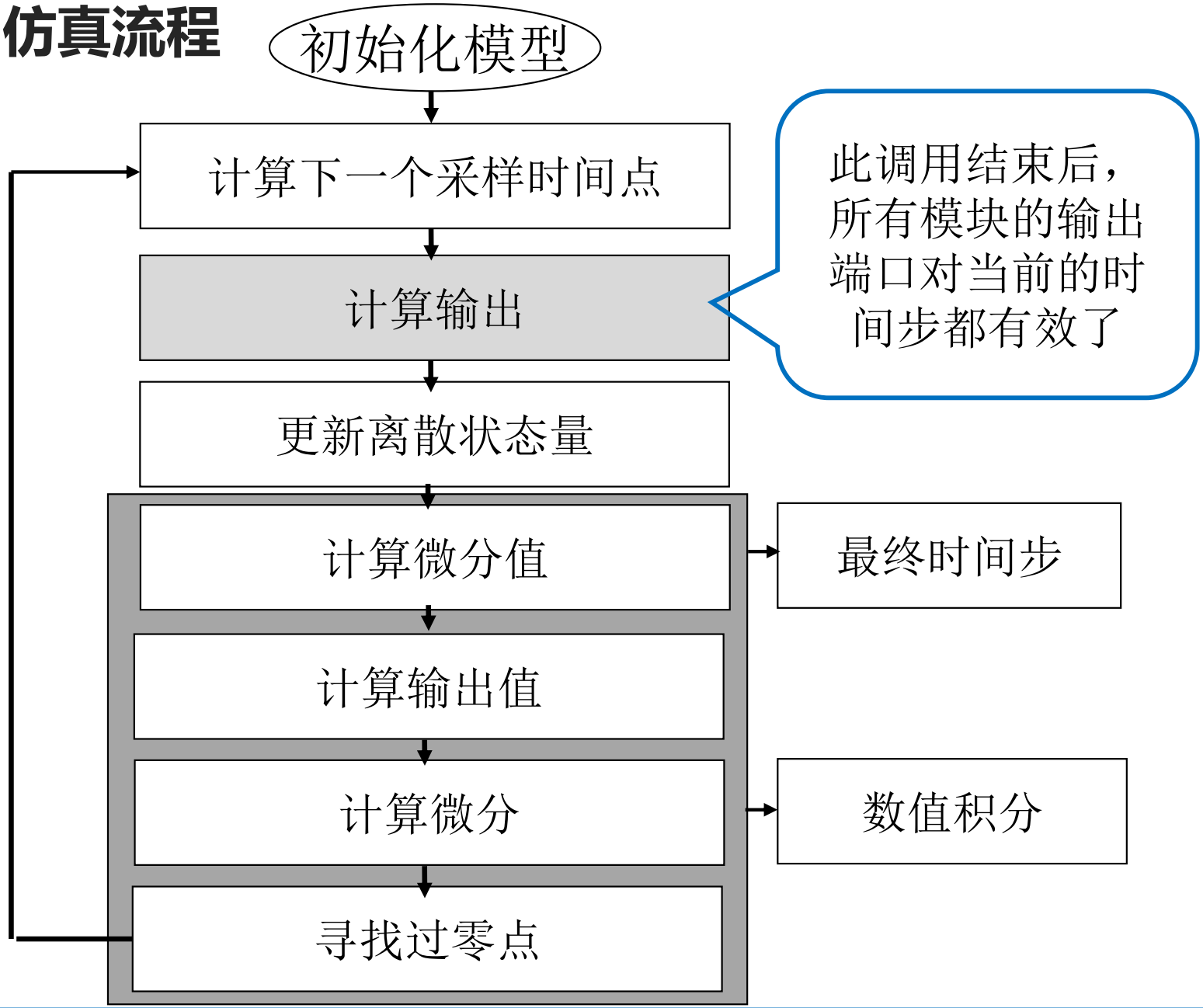
二. S-函数的仿真流程

- **S-函数的仿真流程**



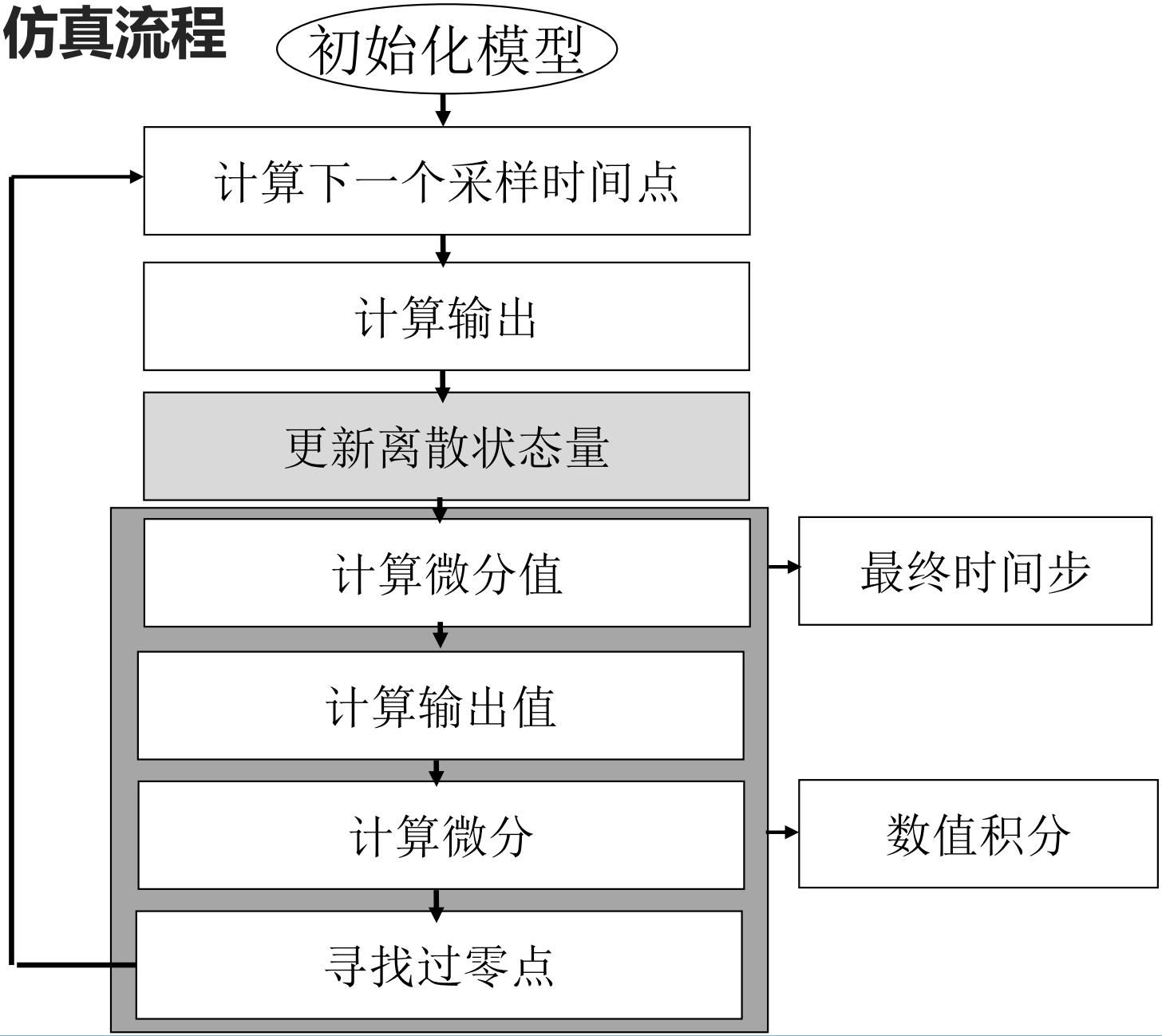
二. S-函数的仿真流程

- **S-函数的仿真流程**



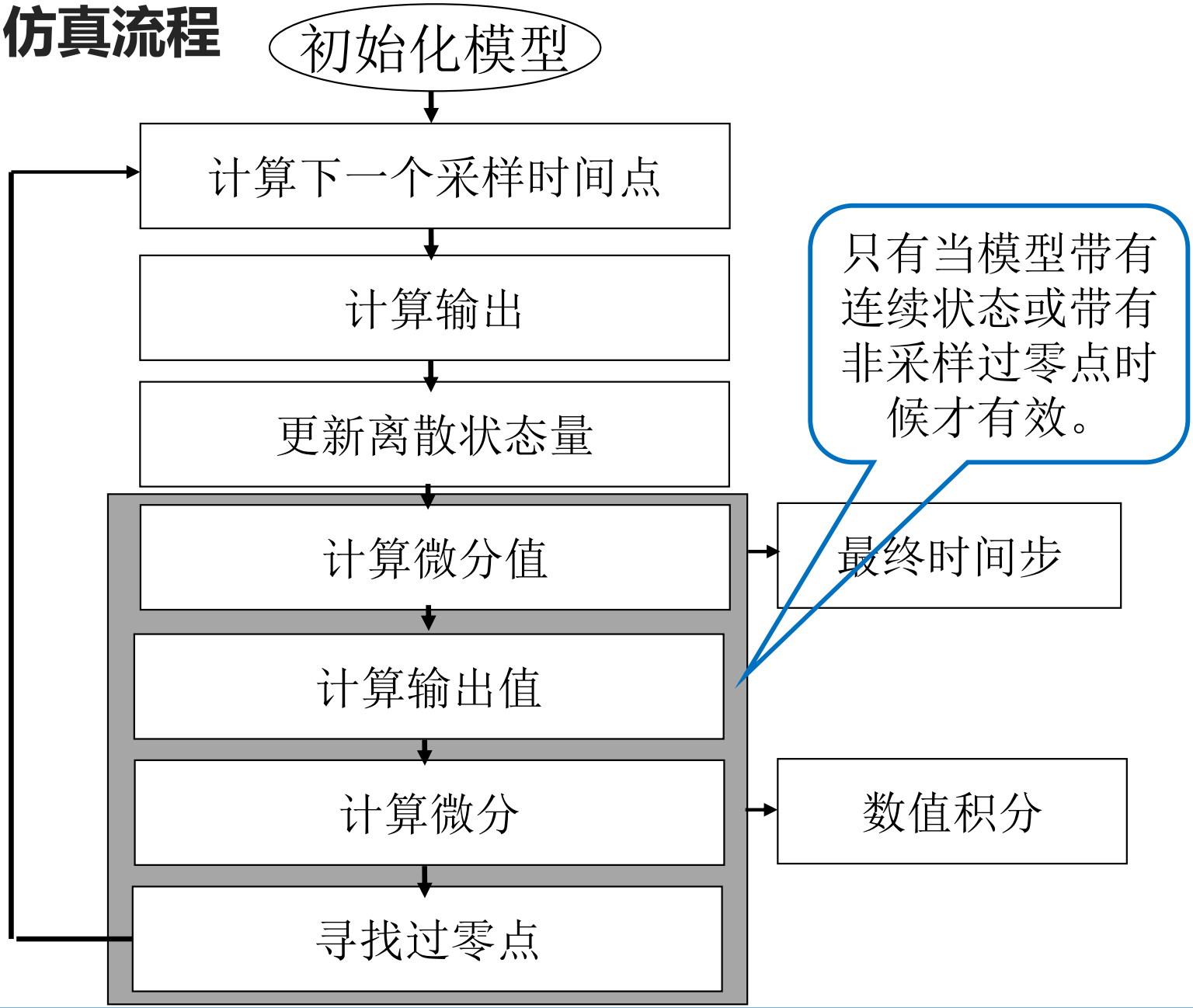
二. S-函数的仿真流程

- **S-函数的仿真流程**



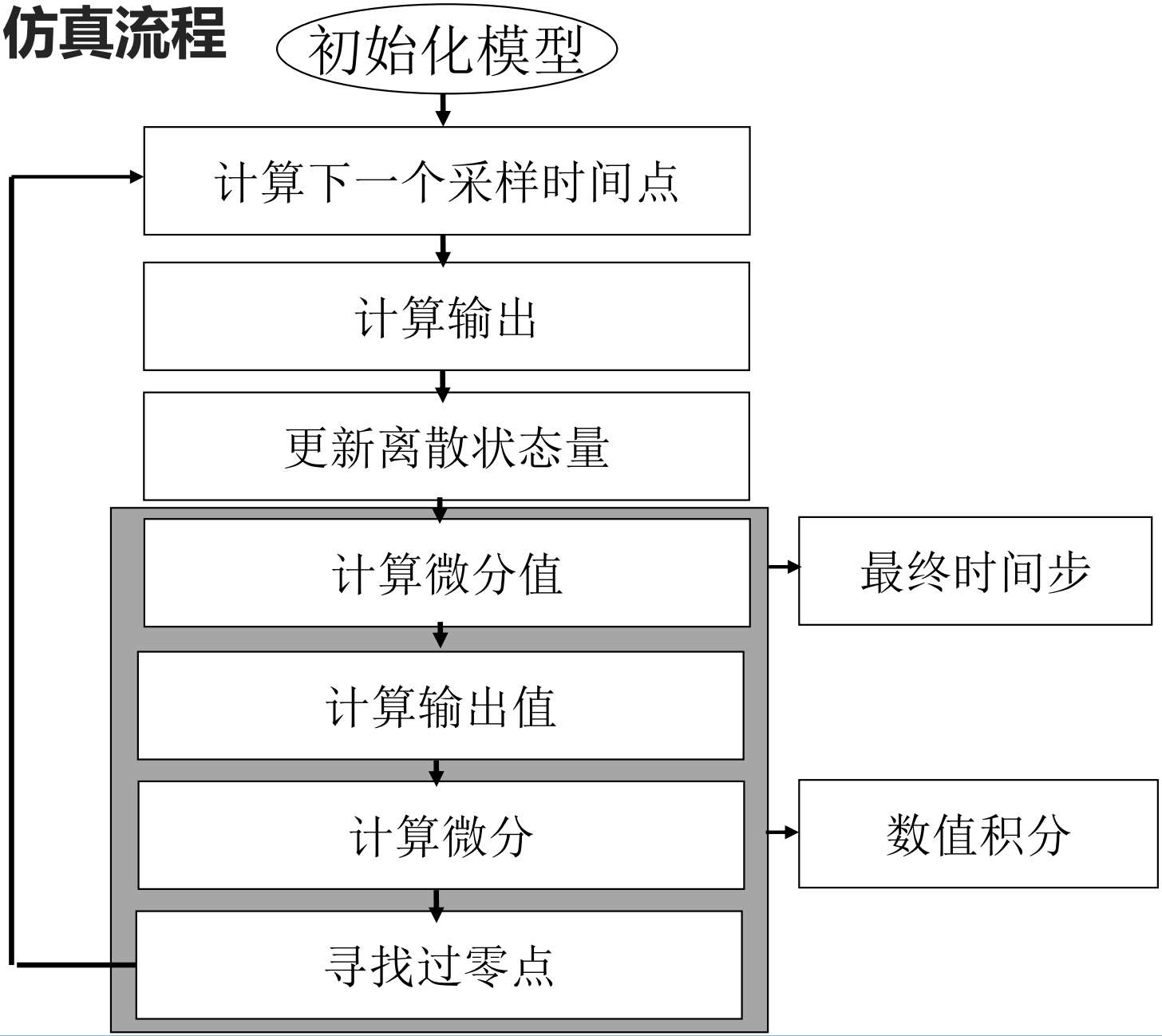
二. S-函数的仿真流程

- **S-函数的仿真流程**



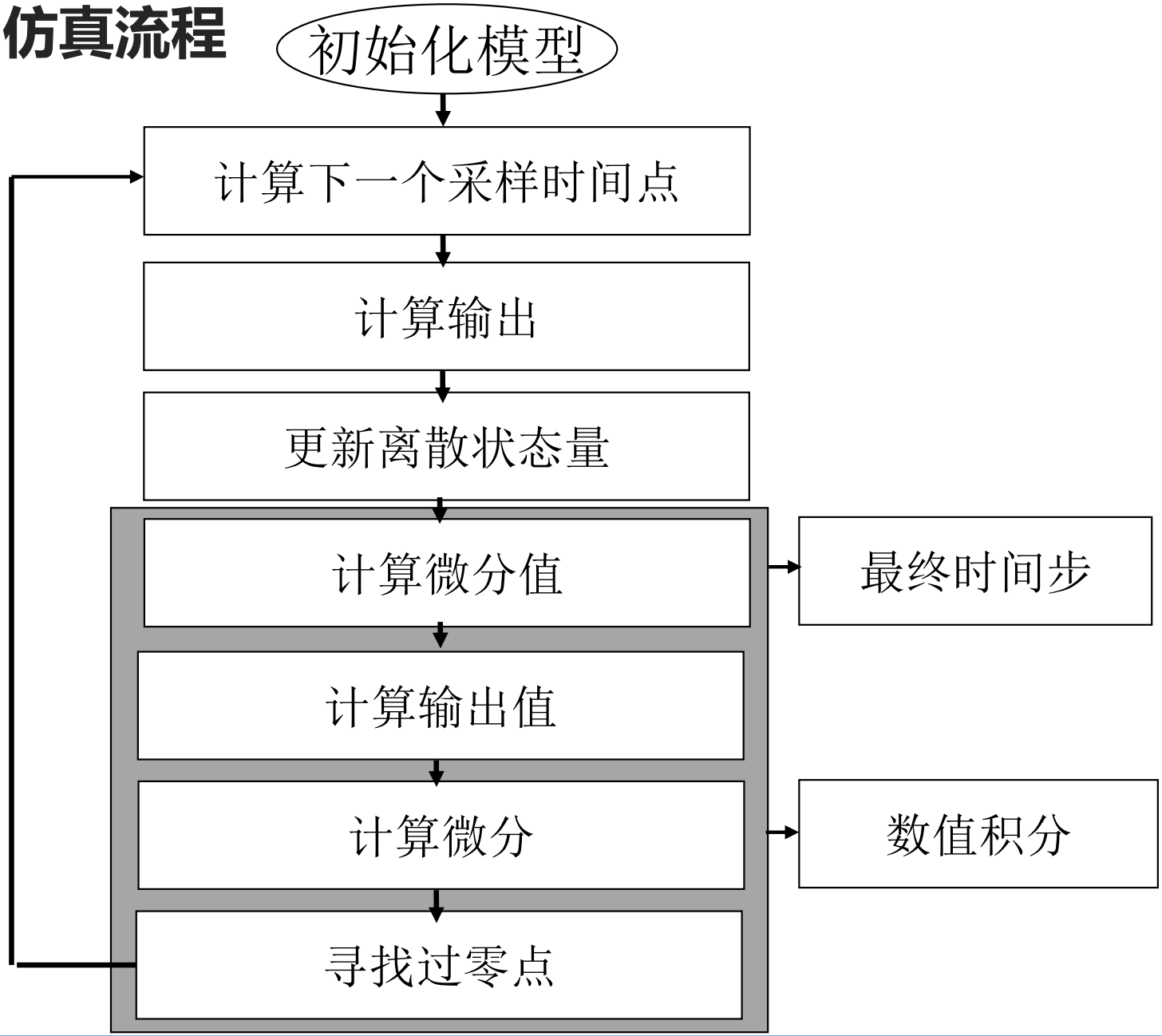
二. S-函数的仿真流程

- **S-函数的仿真流程**



二. S-函数的仿真流程

- **S-函数的仿真流程**



三. S-函数的编写以及应用

- **M文件S-函数的编写**

fucntion [sys,x0,str,ts]=sfunc_name(t,x,u,flag,p1,p2,...,pn)

S函数输入变量名	定　　义
t	仿真时间的当前值
x	S函数状态向量的当前值
u	输入向量的当前值
flag	S函数行为的标志
p1,p2,…,pn	选择参数列表

三. S-函数的编写以及应用

- **M文件S-函数的编写**

fucntion [sys,x0,str,ts]=sfunc_name(t,x,u,flag,p1,p2,...,pn)

S函数输出变量名	定　　义
sys	多目标输出变量，sys的定义取决于flg的值
x0	S函数状态向量的初始值包括连续和离散两种状态
str	设置输出变量为一个空矩阵
ts	设置采样时间、采样延迟矩阵，此矩阵必须为一个双列矩阵。

三. S-函数的编写以及应用

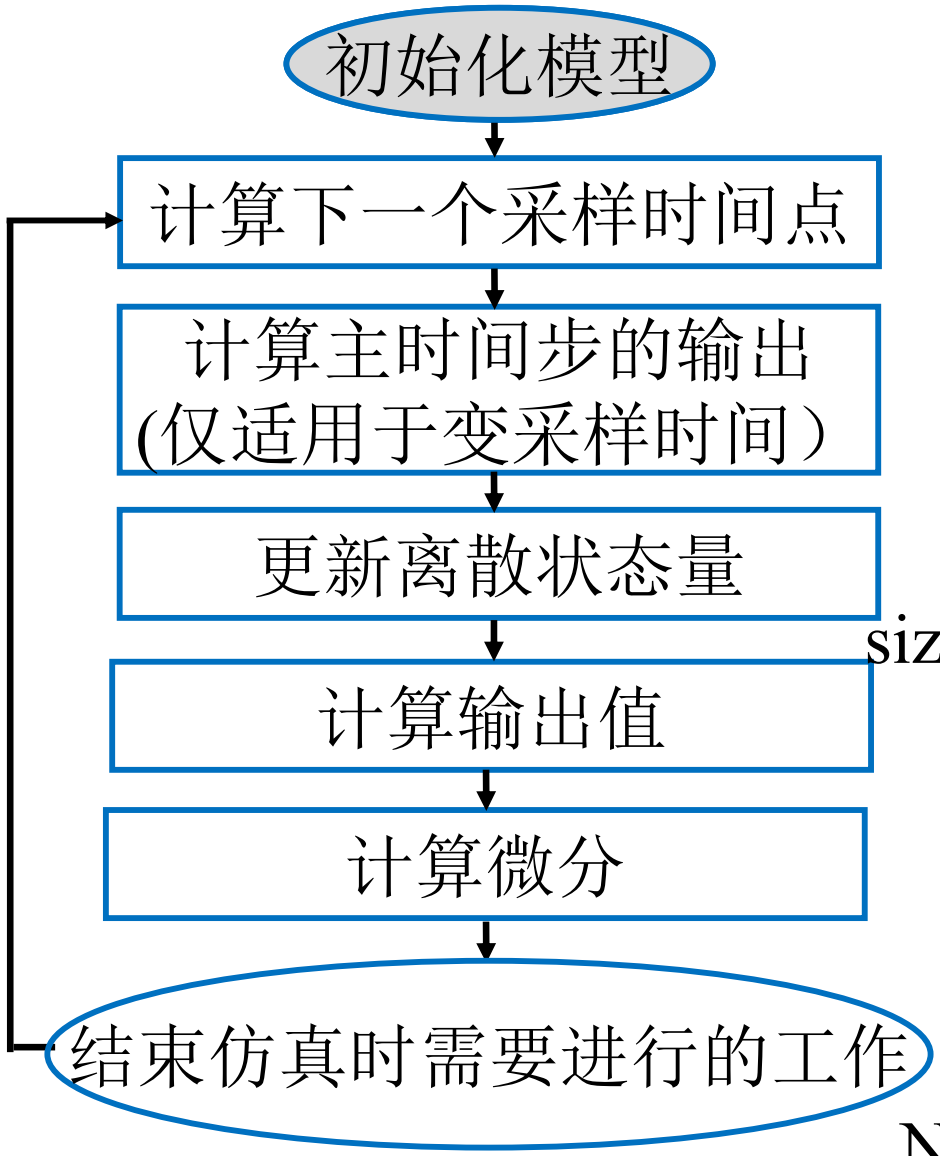
- **M文件S-函数的编写**

fucntion [sys,x0,str,ts]=sfunc_name(t,x,u,flag,p1,p2,...,pn)

S函数flag的值	S函数的行为
0	初始化size结构，并将size的值付给sys，设置x0，ts
1	计算连续状态的微分值
2	更新离散状态
3	计算输出。设置sys为输出向量的值。
4	计算下一次采样时间
9	执行必要的结束仿真任务

三. S-函数的编写以及应用

- **M文件S-函数仿真流程图**



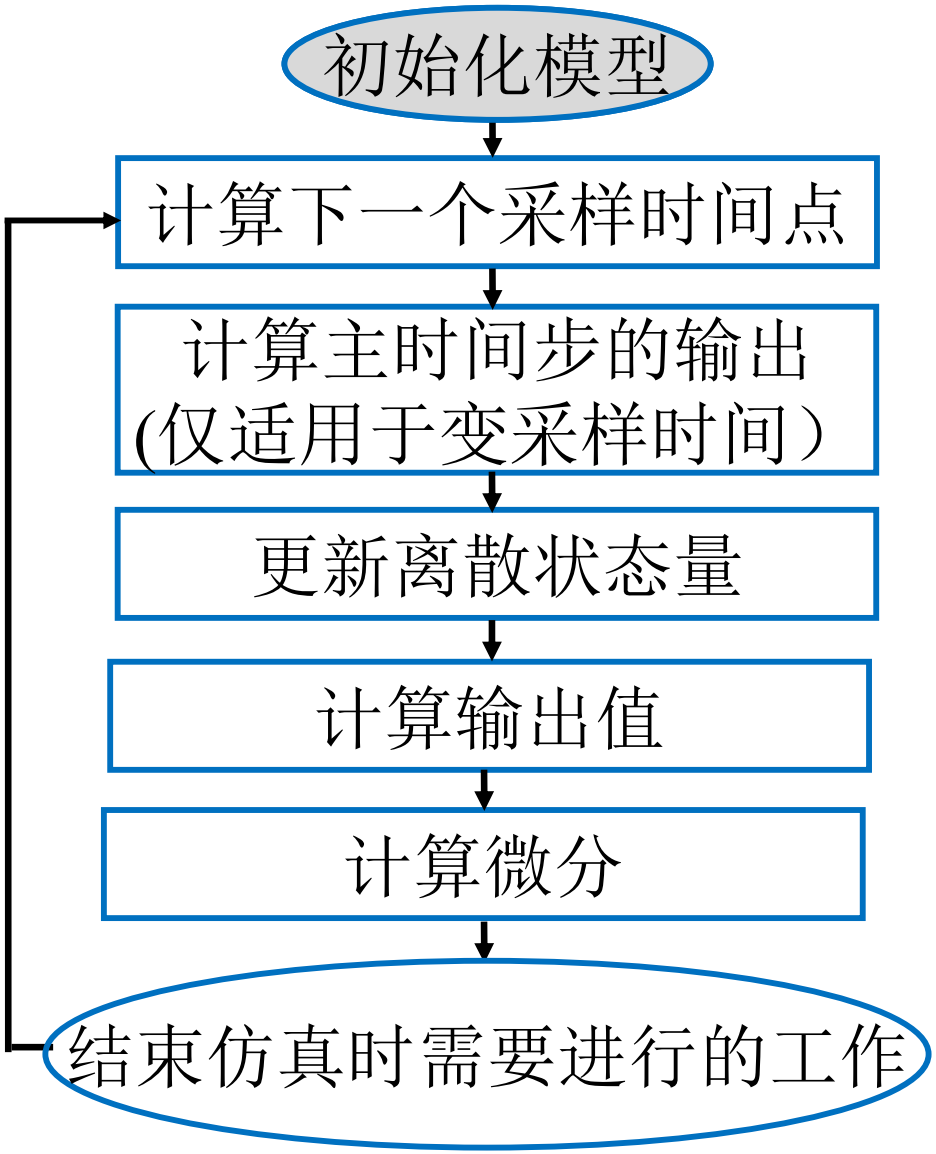
Flag=0
mdlInitializeConditions
mdlInitializeSizes
mdlInitializeSampleTimes

所完成的操作：
1.设置size结构，并将此结构分配到多目标返回变量sys中。
sizes=simsizes

sizes =
NumContStates: 0 连续状态个数
NumDiscStates: 0 离散状态个数
NumOutputs: 0 输出的个数
NumInputs: 0 输入的个数
DirFeedthrough: 0 有无直接馈入
NumSampleTimes: 0 采样时间个数

三. S-函数的编写以及应用

- **M文件S-函数仿真流程图**



Flag=0
mdlInitializeConditions
mdlInitializeSizes
mdlInitializeSampleTimes

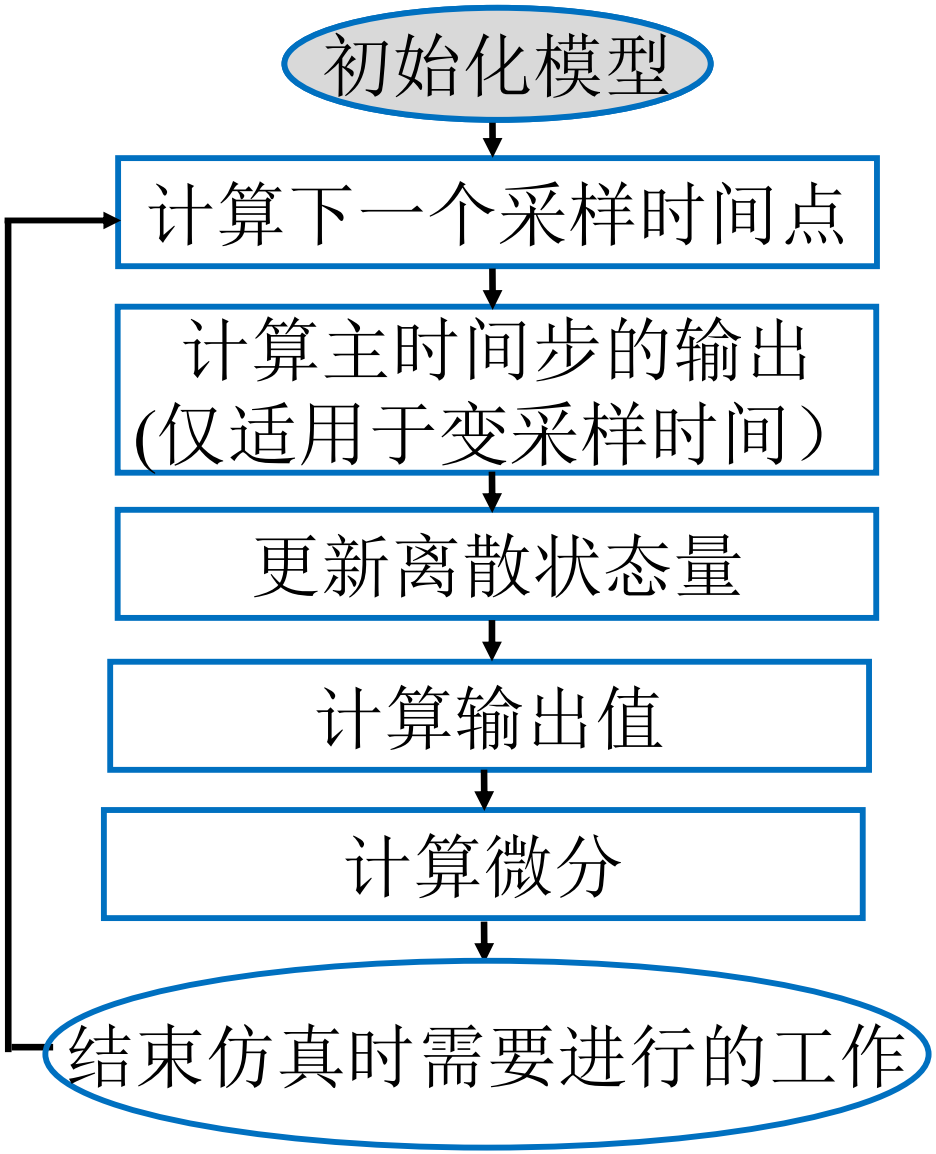
所完成的操作：
1.设置size结构，并将此结构分配到多目标返回变量sys中。
sizes=simsizes

sys=simsizes(sizes)

↳将size付给输出变量sys

三. S-函数的编写以及应用

- **M文件S-函数仿真流程图**



Flag=0
mdlInitializeConditions
mdlInitializeSizes
mdlInitializeSampleTimes

所完成的操作：

2.设置初始化条件向量x0。

例如：有两个连续状态，初始变量为1.0，有两个离散变量，初始变量为0.0，则设置x0为：

x0=[1.0,1.0,0.0,0.0];

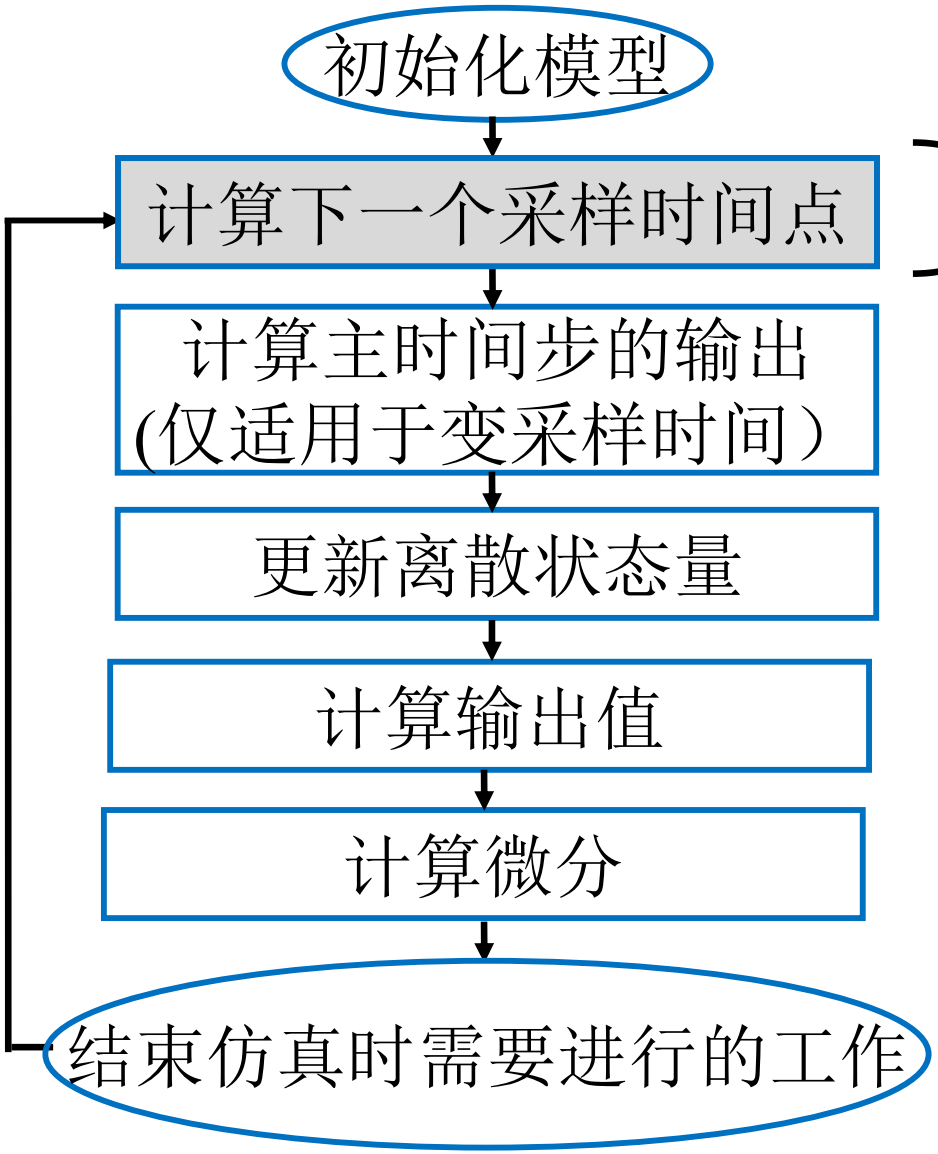
3.设置str变量

str=[];

4.创建采样时间及延迟矩阵ts。

三. S-函数的编写以及应用

- **M文件S-函数仿真流程图**



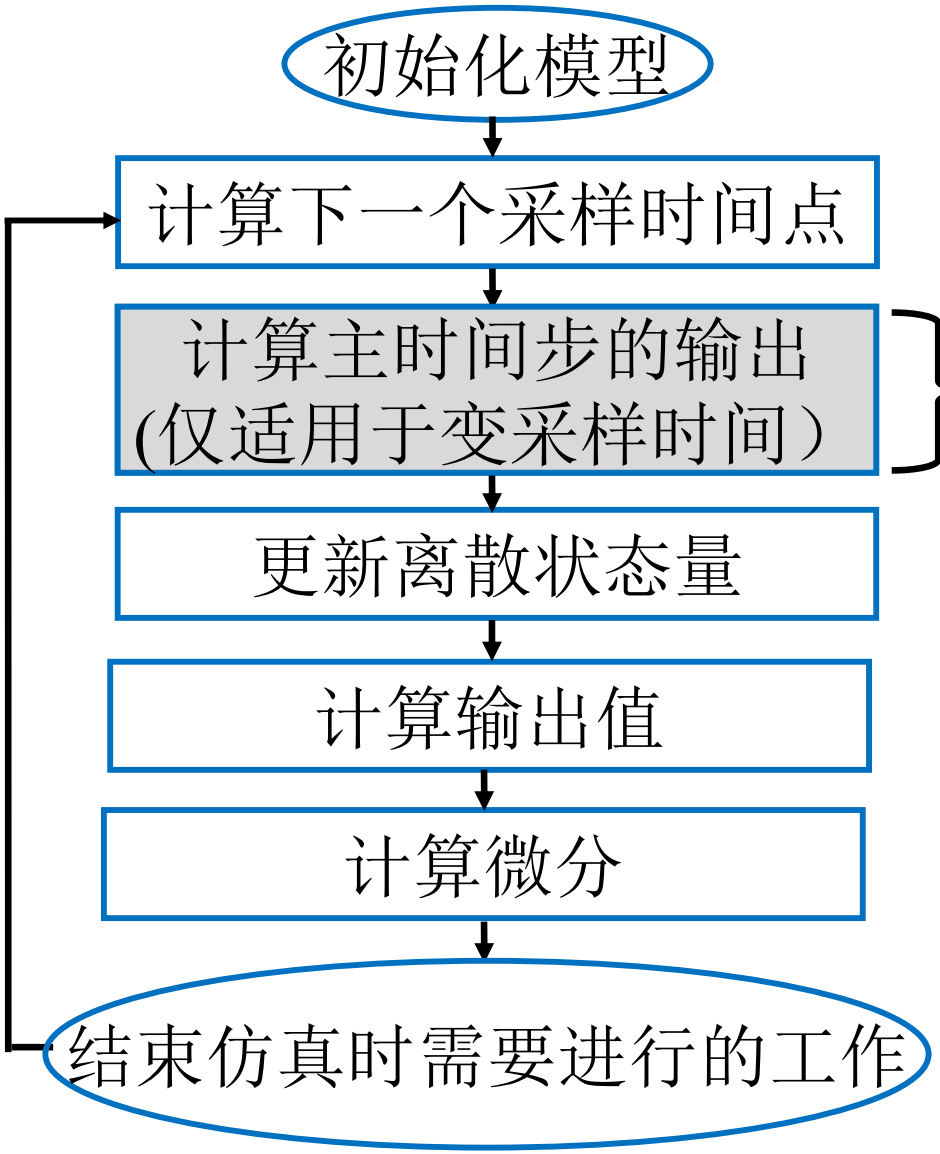
Flag=4
mdlGetTimeNextVarHit

所完成的操作：

赋sys的值为下一次采样时刻的值

三. S-函数的编写以及应用

- **M文件S-函数仿真流程图**



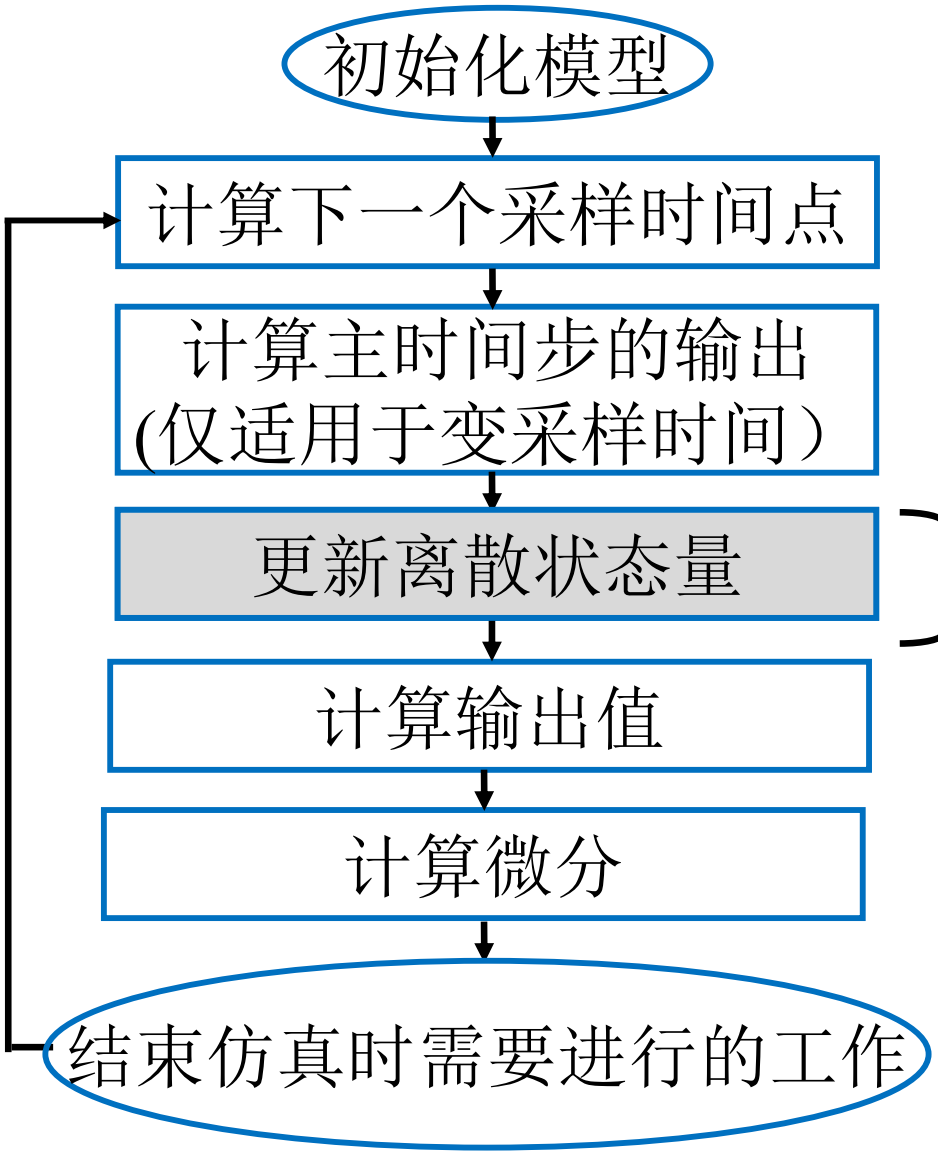
Flag=3
mdlOutputs

所完成的操作：

赋S函数的输出给sys。

三. S-函数的编写以及应用

- **M文件S-函数仿真流程图**



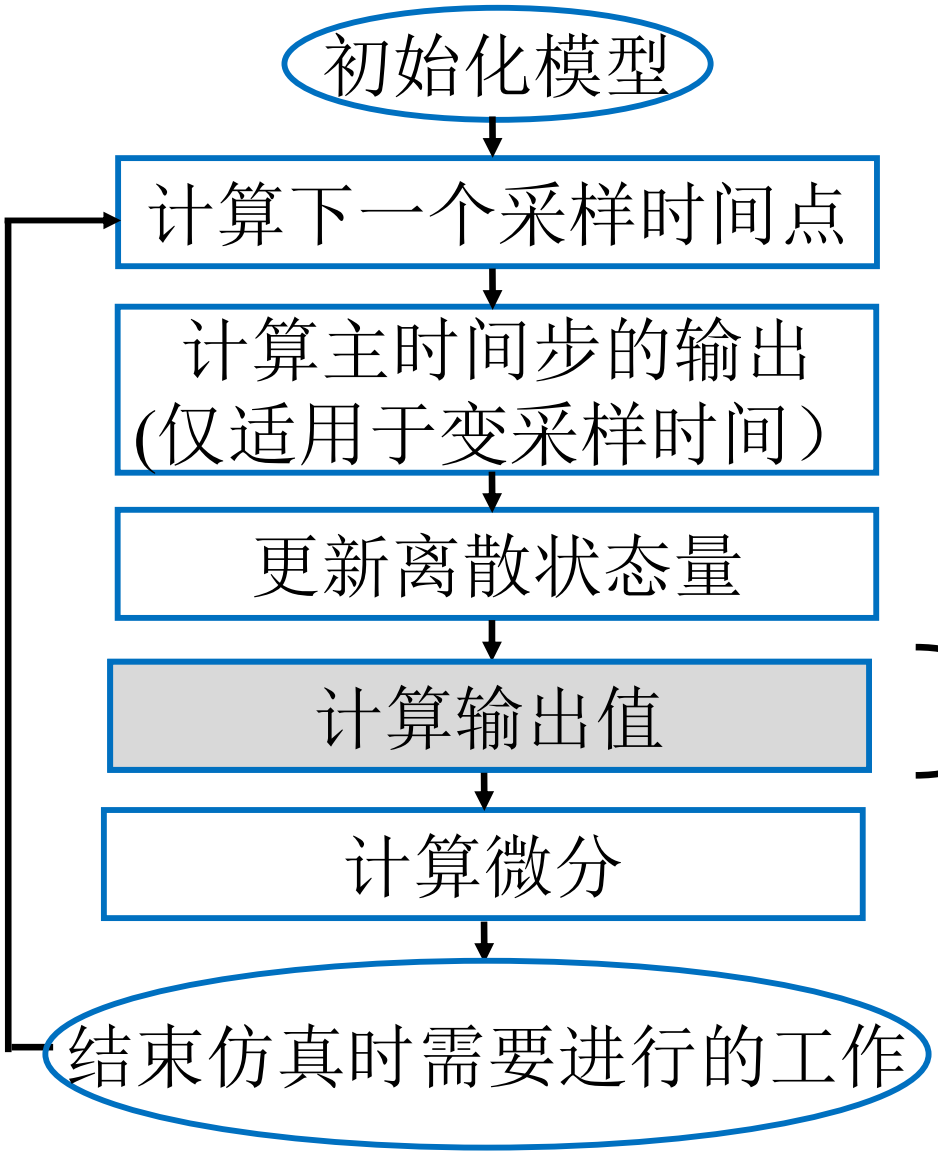
Flag=2
mdlUpdate

所完成的操作：

将状态向量离散部分的更新赋给sys

三. S-函数的编写以及应用

- **M文件S-函数仿真流程图**



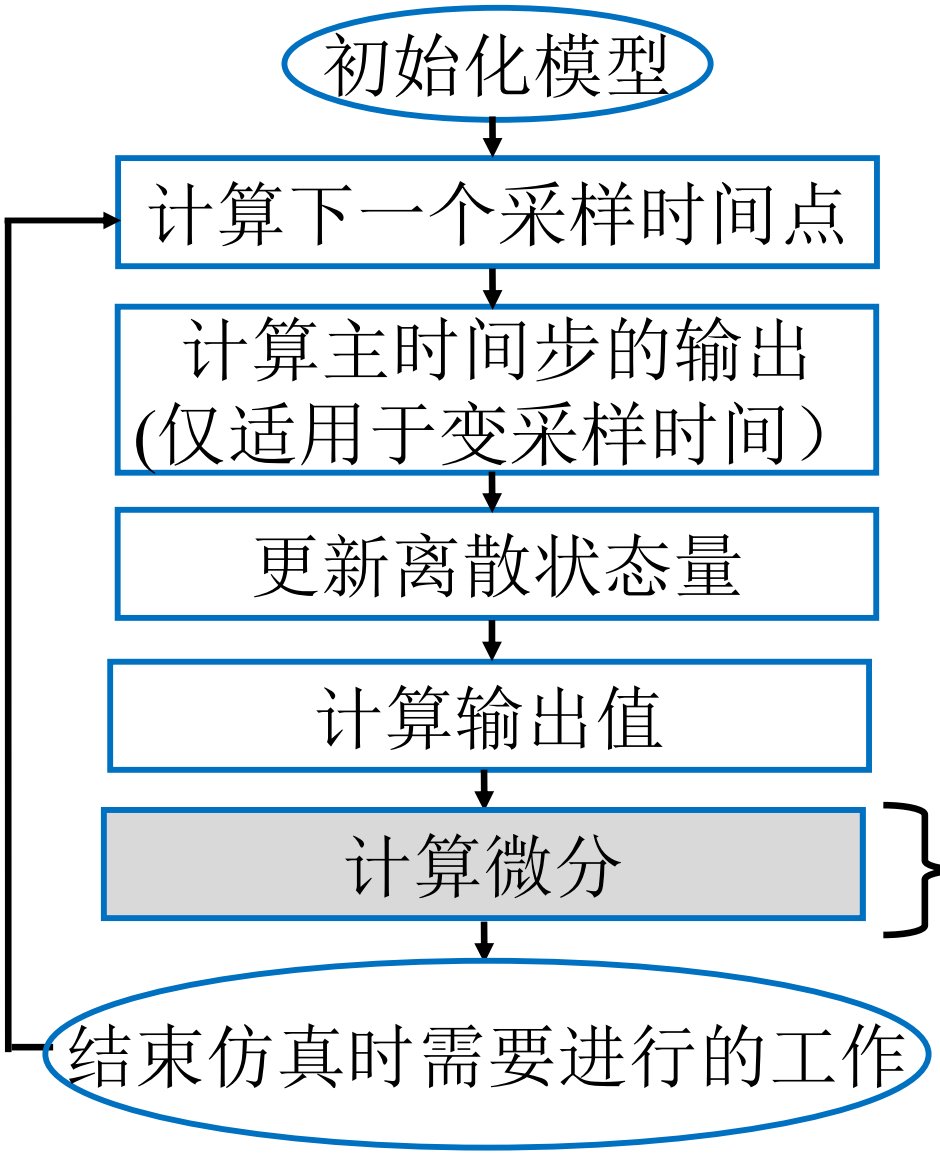
Flag=3
mdlOutputs

所完成的操作：

赋S函数的输出给sys。

三. S-函数的编写以及应用

- **M文件S-函数仿真流程图**



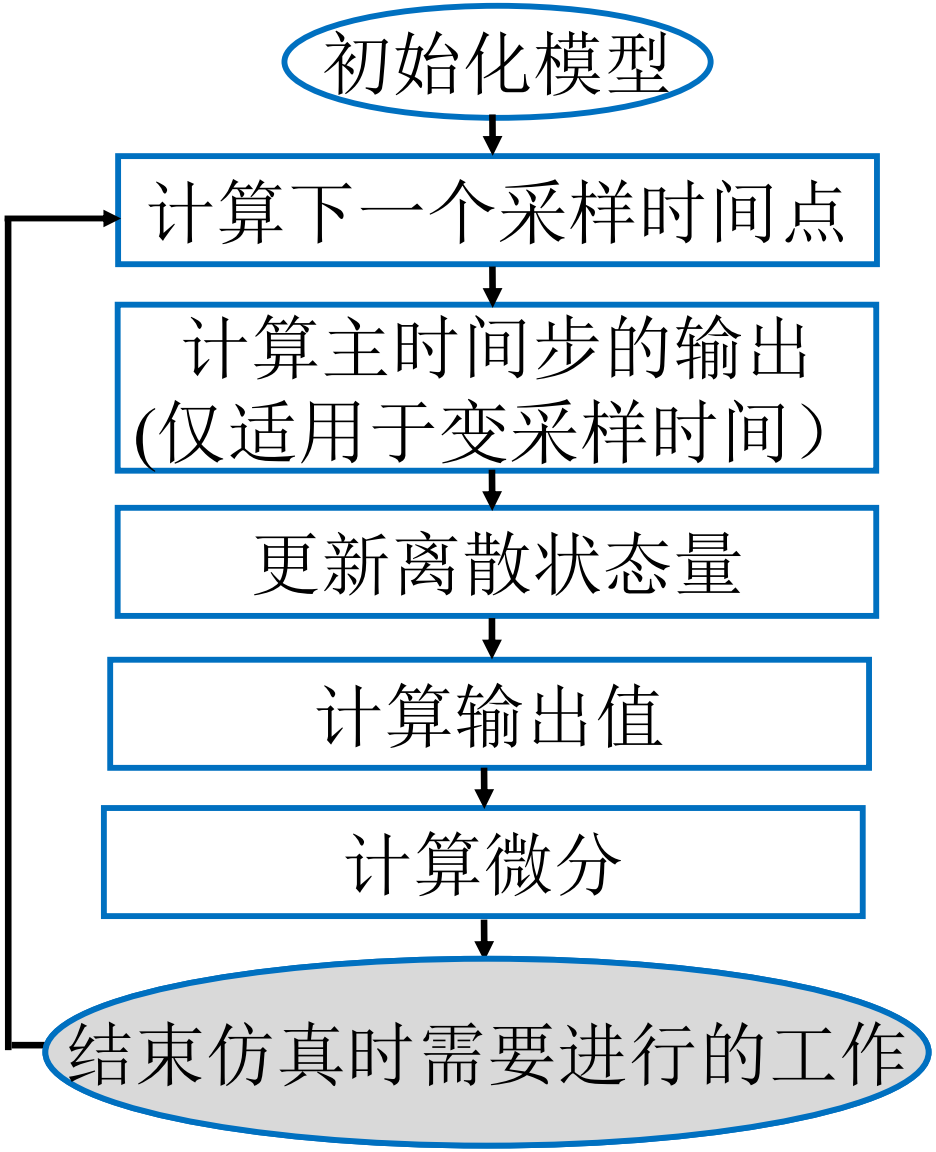
所完成的操作：

将状态向量连续部分的微分值赋给sys

Flag=1
mdlDerivatives

三. S-函数的编写以及应用

- **M文件S-函数仿真流程图**



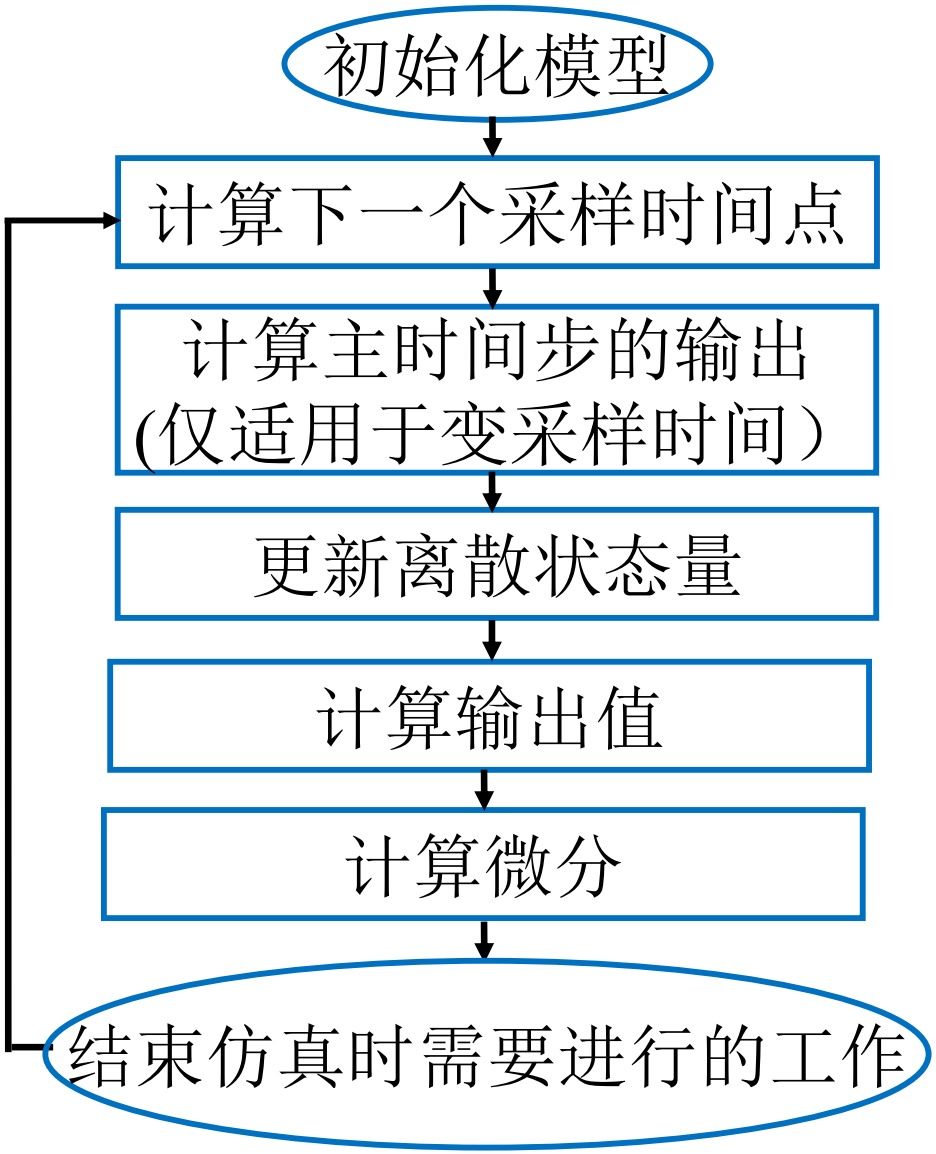
所完成的操作：

终止仿真

Flag=9
mdlTerminate

三. S-函数的编写以及应用

- **M文件S-函数仿真流程图**



M文件S-函数模版

edit sfuntmpl

三. S-函数的编写以及应用

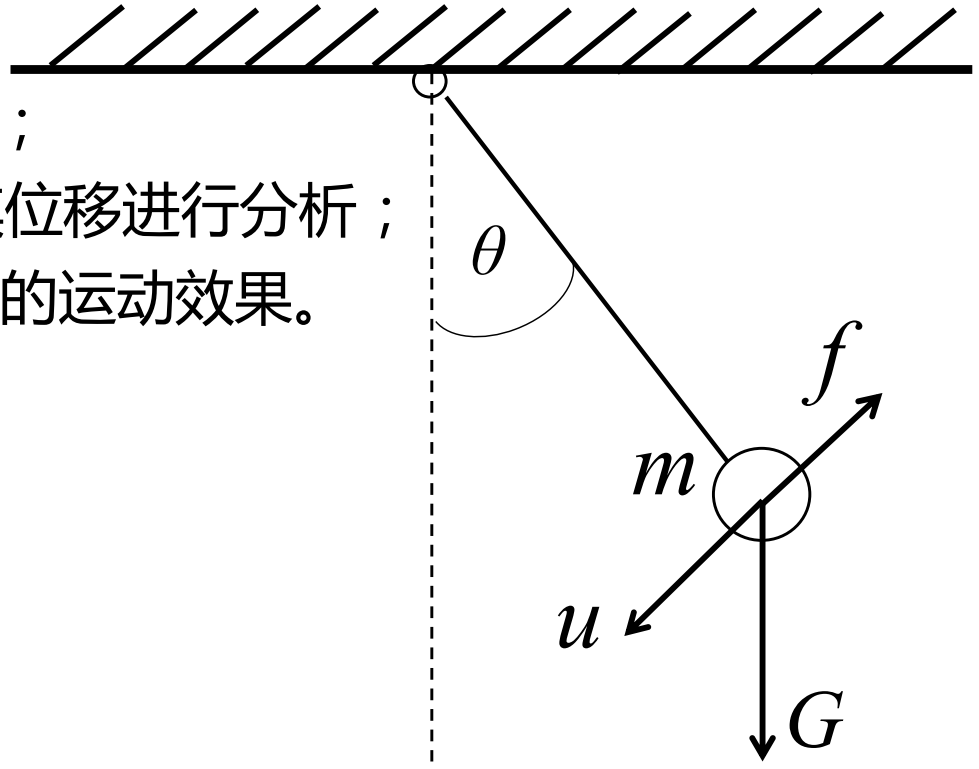
•M文件S-函数的应用 下图所示简单的单摆系统,假设杆的长度为L（米m）,且杆的质量不计,钢球的质量为m（千克kg）。单摆受重力和摩擦阻尼力以及外力共同作用，其中u（牛N）为作用在单摆上的外力；单摆的运动可以以线性的微分方程式来近似,但事实上系统的行为是非线性的,而且存在粘滞阻尼,假设粘滞阻尼系数为ξ（kg/ms^{-1}）。

1）利用s-函数对单摆系统进行建模；

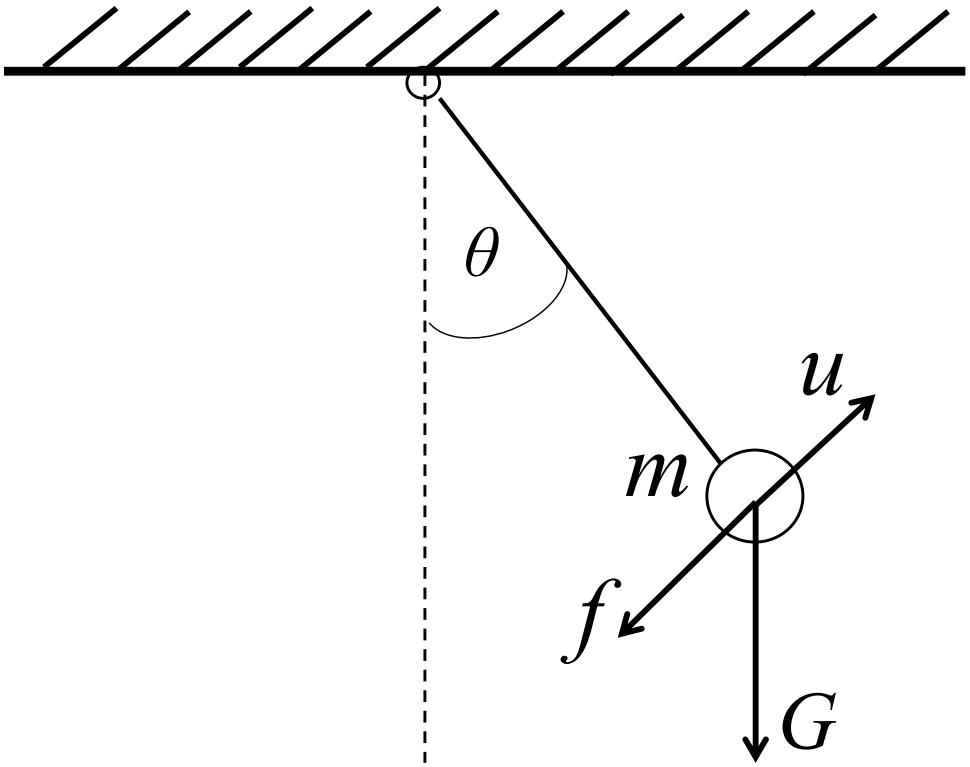
2）利用Simulink进行仿真，并对其位移进行分析；

3）利用S-函数动画模块来演示单摆的运动效果。

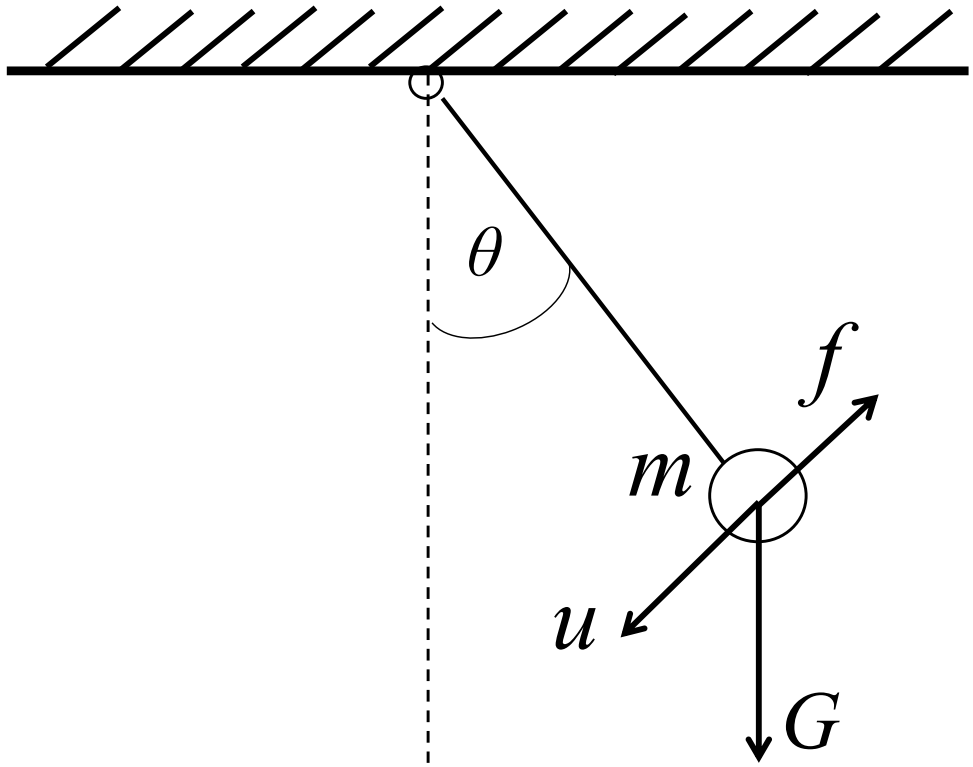
$$\begin{aligned} \xi &= 0.03 \\ g &= 9.8 \\ L &= 0.8 \\ m &= 0.3 \\ u &\neq 0 \end{aligned}$$



系统数学建模



单摆系统向上受力时受力分析图



单摆系统向下受力时受力分析图

根据牛顿第二定律： $\sum F = ma$

问题中： $\ddot{\theta} = \dot{\omega}$， $a = L \times \dot{\omega} = L \times \ddot{\theta}$， $f = \xi \times v = \xi \times L \times \omega = \xi L\dot{\theta}$， $G = mg$

推导出单摆系统的动力学方程为：

第一种情况： $mL\ddot{\theta} + \xi L\dot{\theta} = u - Gsim(\theta) = u - mgsin(\theta)$

第二种情况： $mL\ddot{\theta} + \xi L\dot{\theta} = u + Gsim(\theta) = u + mgsin(\theta)$

第一种情况

根据单摆系统的动力学方程：

$$mL\ddot{\theta} + \xi L\dot{\theta} = u - Gsim(\theta) = u - mgsin(\theta)$$

将动力学方程转化为状态方程如下。
令$x_1 = \theta, x_2 = \dot{\theta}$，则动力学方程变为：

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -\frac{\xi}{m}x_2 + \frac{1}{ml}u - \frac{g}{l}sin(x_1) \end{bmatrix}$$

则根据初始值为0的条件，可知$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

S-函数动态仿真建模

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix}=\begin{bmatrix}x_2\\ -\frac{\xi}{m}x_2+\frac{1}{ml}u-\frac{g}{l}sin(x_1)\end{bmatrix}$$

- 根据状态方程，在S-函数模板的基础上编辑修改针对这个问题的S-函数，并保存为dbyd_exm.m。
- 建立Simulink模型，设置S-函数模块以及其他模块参数

```
function [sys,x0,str,ts] = dbyd_exm(t,x,u,flag,kesai,g,l,a,m)
%  根据本例子参数要求，加入了参数如下：
%  阻尼系数kesai，重力加速度g，初始状态值a，质量m，摆杆长度l。
%  flag标志来控制仿真的流程
switch flag,
  %%%%%%%%%%%%%%%%%%%%%%
  % Initialization 调用初始化函数模块     %
  %%%%%%%%%%%%%%%%%%%%%%
  case 0,
    [sys,x0,str,ts]=mdlInitializeSizes(a);
  %%%%%%%%%%%%%%%%%%%%
  % Derivatives 计算导数模块函数%
  %%%%%%%%%%%%%%%%%%%%
  case 1,
     sys=mdlDerivatives(t,x,u,kesai,g,l,m);
  %%%%%%%%%%%%%%%%%%%%
  % Update 调用更新状态输出模块%
  %%%%%%%%%%%%%%%%%%%%
  case 2,
    sys=mdlUpdate(t,x,u);
  %%%%%%%%%%%%%%%%%%%%%%
  % Outputs 调用结束仿真子函数模块%
  %%%%%%%%%%%%%%%%%%%%%%
```

```
  case 3,
    sys=mdlOutputs(t,x,u);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  % GetTimeOfNextVarHit 调用离散状态更新子函数模块%
  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  case 4,
    sys=mdlGetTimeOfNextVarHit(t,x,u);
  %%%%%%%%%%%%%%%%%%%%%%%
  % Terminate 调用结束仿真子函数模块 %
  %%%%%%%%%%%%%%%%%%%%%%%
  case 9,
    sys=mdlTerminate(t,x,u);
  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  % Unexpected flags 如果flag标志位状态不正确，返回错误%
  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  otherwise
    error(['Unhandled flag = ',num2str(flag)]);
end
% end sfuntmpl
```

```
%=============================================================
% mdlInitializeSizes
% Return the sizes, initial conditions, and sample times for the S-function.
%=============================================================
%
function [sys,x0,str,ts]=mdlInitializeSizes(a)
%
% call simsizes for a sizes structure, fill it in and convert it to a
% sizes array.
%
% Note that in this example, the values are hard coded.  This is not a
% recommended practice as the characteristics of the block are typically
% defined by the S-function parameters.
%
sizes = simsizes;

sizes.NumContStates  = 2;   %  包含连续状态变量的个数，这里设为2个。
sizes.NumDiscStates  = 0;   %  包含离散状态变量的个数，这里设为0个。
sizes.NumOutputs     = 1;       %  包含一路输入信号，设为1。
sizes.NumInputs      = 1;       %  包含一路输出信号，设为1。
sizes.DirFeedthrough = 0;   %  因为没有反馈值，所以设为0。
sizes.NumSampleTimes = 1;       %  采样时间个数，至少有一个采样时间，因此设置为1。

sys = simsizes(sizes);
%
% initialize the initial conditions
%
x0  = a;                %  状态变量初始值
%
% str is always an empty matrix
%
str = [];
%
% initialize the array of sample times
%
ts  = [0 0];
% end mdlInitializeSizes
```

```
%==========================================================================
% mdlDerivatives  计算导数子函数
% Return the derivatives for the continuous states.
%==========================================================================
%
%function sys=mdlDerivatives(t,x,u,kesai,g,m)
function sys=mdlDerivatives(t,x,u,kesai,g,l,m)
dx(1)=x(2);
dx(2)=-(kesai/m)*x(2) - (g/l)*sin(x(1))+(1/(m*l))*u;
sys = dx;

% end mdlDerivatives
```

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix}=\begin{bmatrix}x_2\\ -\frac{\xi}{m}x_2+\frac{1}{ml}u-\frac{g}{l}sin(x_1)\end{bmatrix}$$

```
%==========================================================================
% mdlUpdate  计算状态更新子函数
% Handle discrete state updates, sample time hits, and major time step
% requirements.
%==========================================================================
%
function sys=mdlUpdate(t,x,u)

sys = [];

% end mdlUpdate
```

```
%=============================================================================
% mdlOutputs  计算输出子函数
% Return the block outputs.
%=============================================================================
%
function sys=mdlOutputs(t,x,u)

sys = x(1);
% end mdlOutputs
%
%=============================================================================
% mdlGetTimeOfNextVarHit
% Return the time of the next hit for this block.  Note that the result is
% absolute time.  Note that this function is only used when you specify a
% variable discrete-time sample time [-2 0] in the sample time array in
% mdlInitializeSizes.
%=============================================================================
%
function sys=mdlGetTimeOfNextVarHit(t,x,u)

sampleTime = 1;    %  Example, set the next hit to be one second later.
sys = t + sampleTime;
% end mdlGetTimeOfNextVarHit
%
%=============================================================================
% mdlTerminate
% Perform any end of simulation tasks.
%=============================================================================
%
function sys=mdlTerminate(t,x,u)

sys = [];
% end mdlTerminate
```

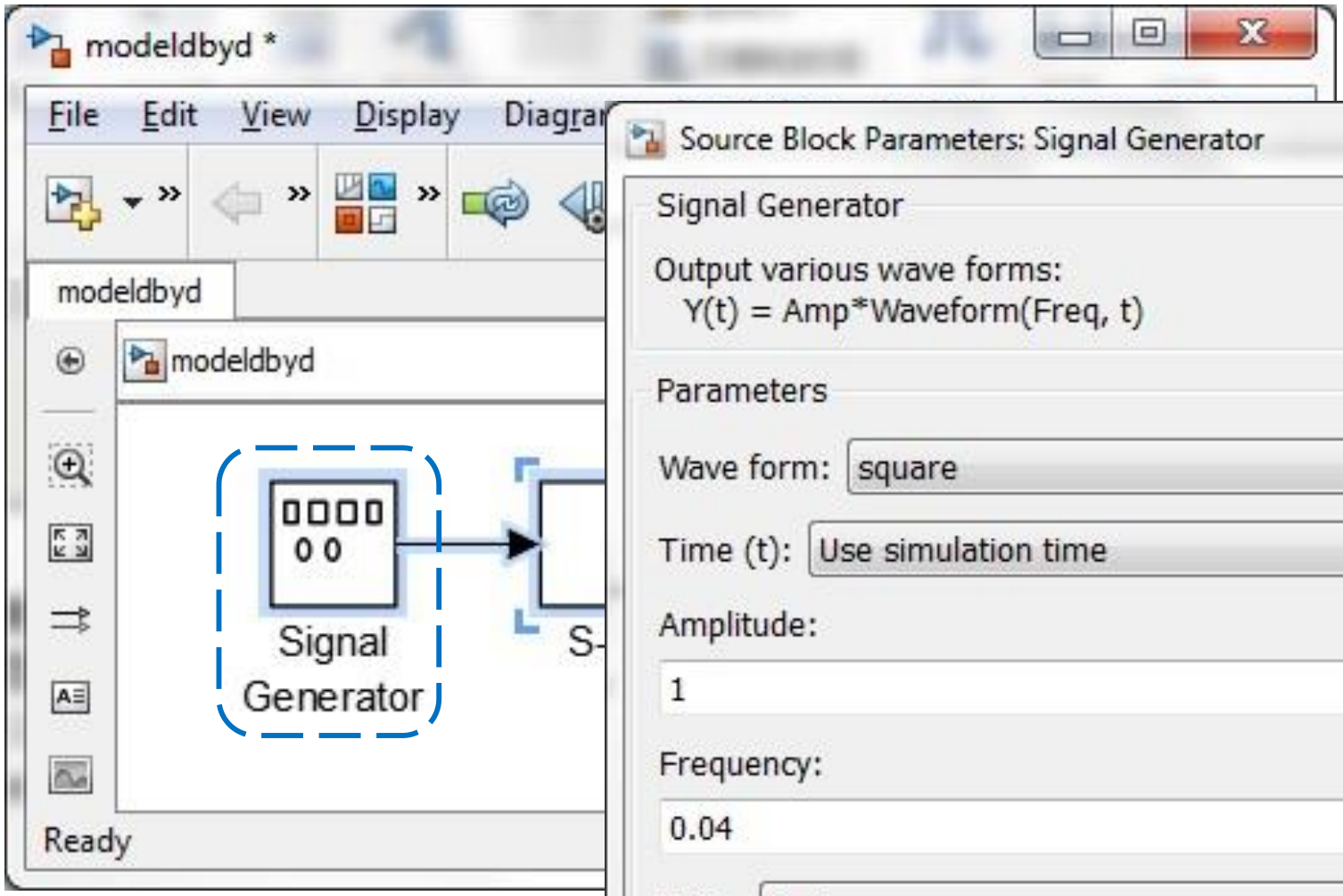
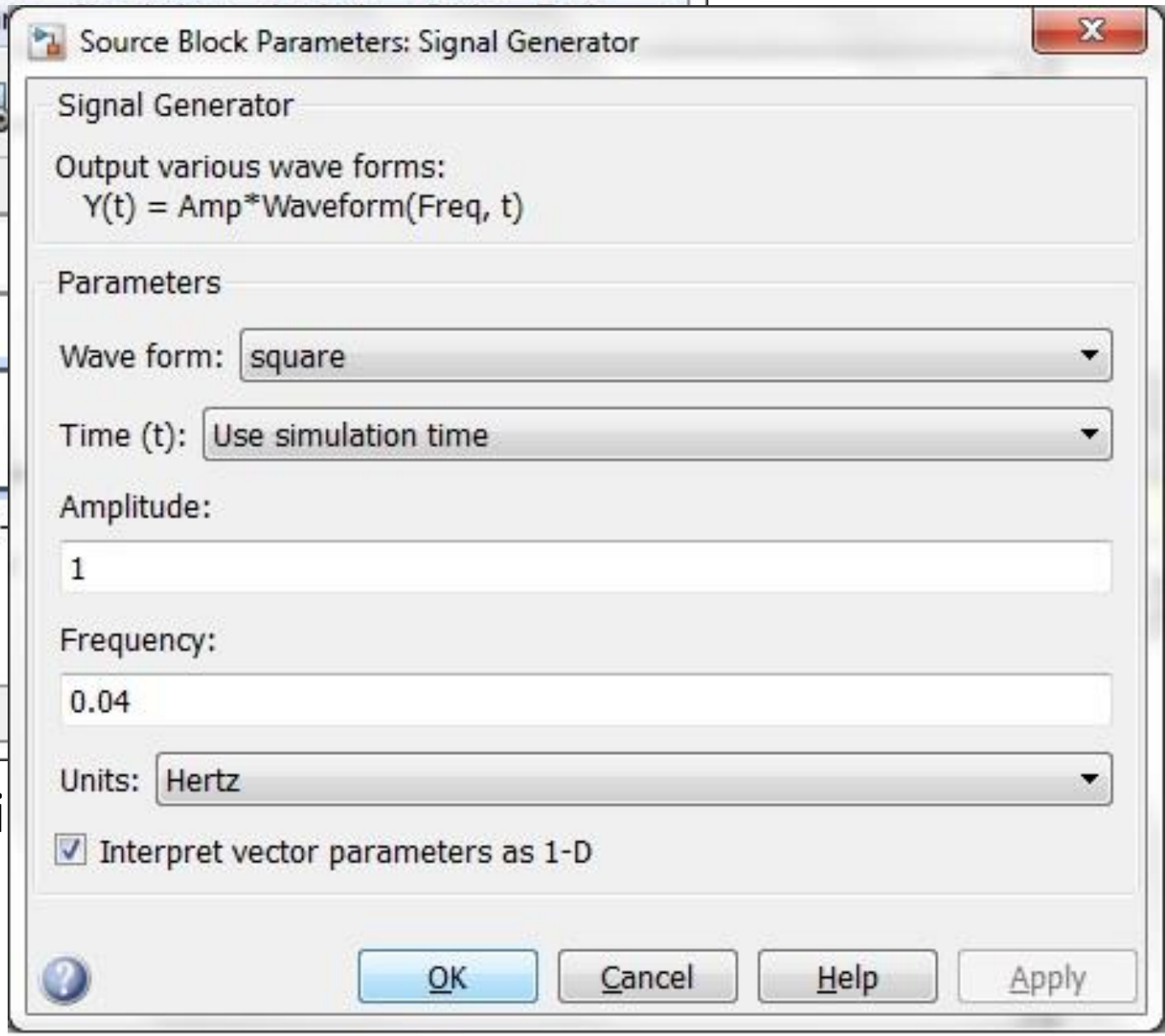


图4.21 单摆Si



Signal Generator模块参设置图

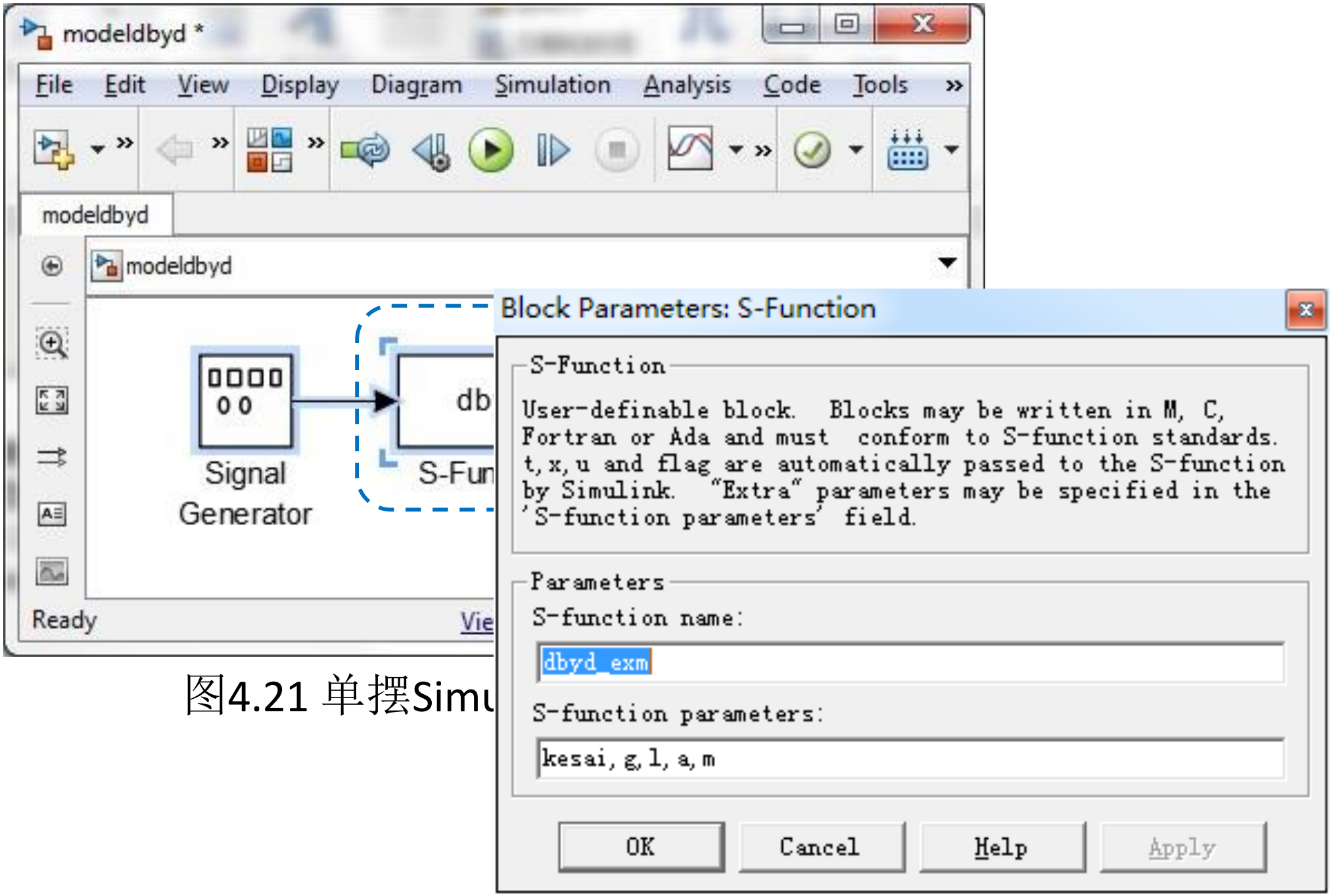
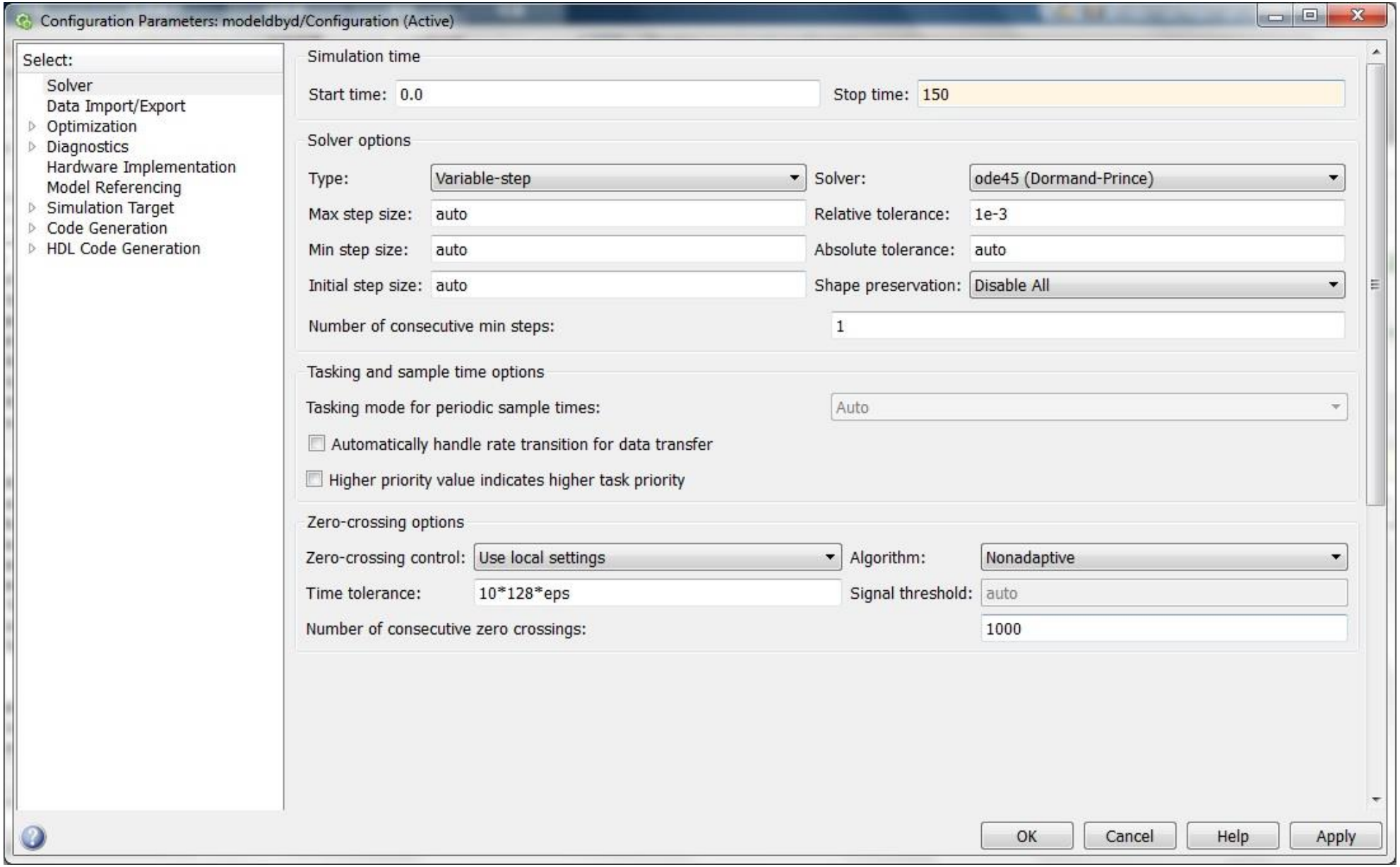


图4.21 单摆Simu

S-Function块参数设置

图4.21 单摆Sim

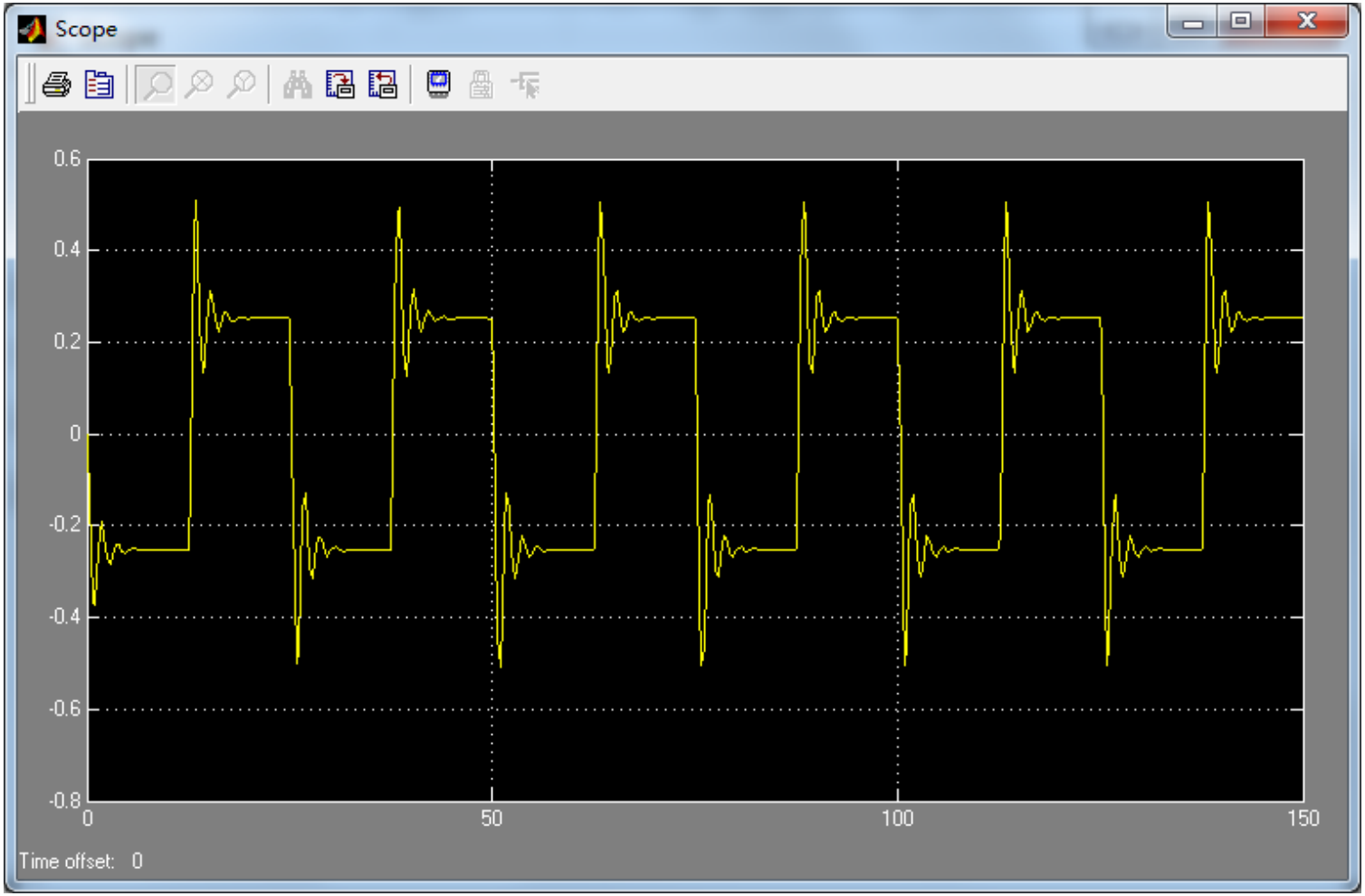
调用Simulink仿真模型参数配置对话框


Configuration Parameters: modeldbyd/Configuration (Active)
Select:
Solver
Data Import/Export
Optimization
Diagnostics
Hardware Implementation
Model Referencing
Simulation Target
Code Generation
HDL Code Generation
Simulation time
Start time: 0.0
Stop time: 150
Solver options
Type: Variable-step
Solver: ode45 (Dormand-Prince)
Max step size: auto
Relative tolerance: 1e-3
Min step size: auto
Absolute tolerance: auto
Initial step size: auto
Shape preservation: Disable All
Number of consecutive min steps: 1
Tasking and sample time options
Tasking mode for periodic sample times: Auto
Automatically handle rate transition for data transfer
Higher priority value indicates higher task priority
Zero-crossing options
Zero-crossing control: Use local settings
Algorithm: Nonadaptive
Time tolerance: 10*128*eps
Signal threshold: auto
Number of consecutive zero crossings: 1000
OK
Cancel
Help
Apply

仿真模型参数设置图

设置完成后，在Matlab命令窗口输入参数变量的初始值。

```
>> kesai=0.6;g=10;a=[0;0];m=0.4;l=0.8
```



单摆仿真效果

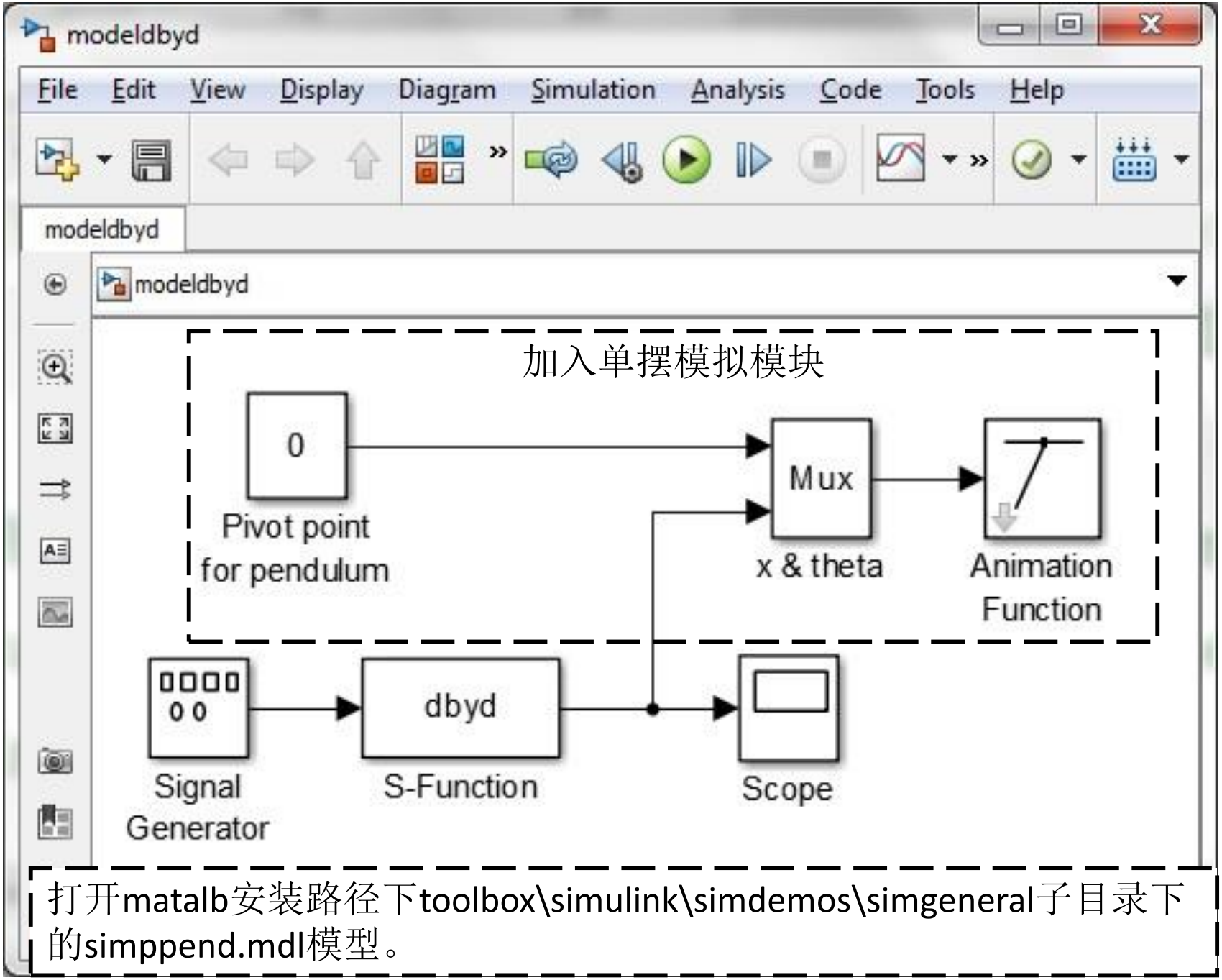


图4.26修改后Simulink模型

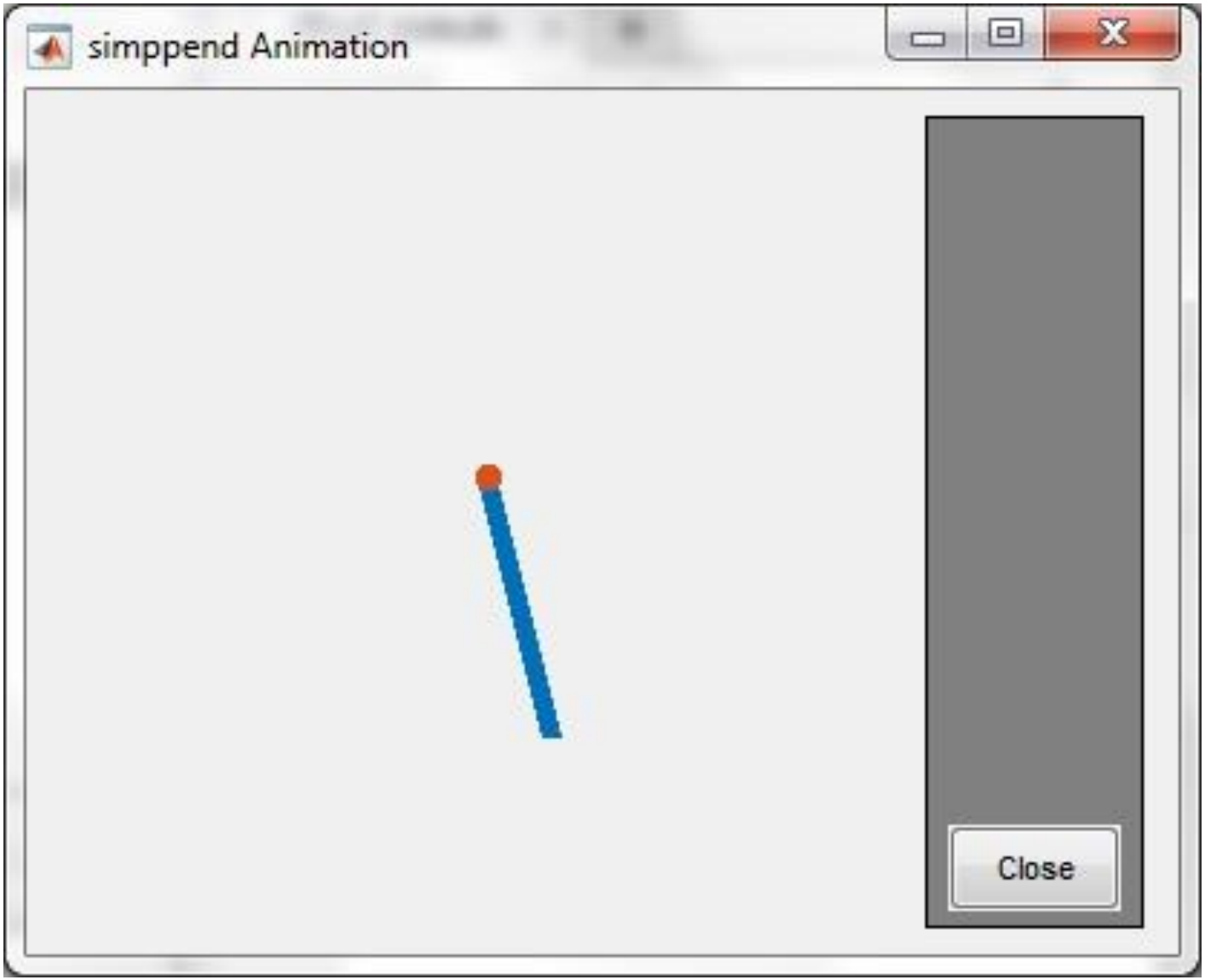
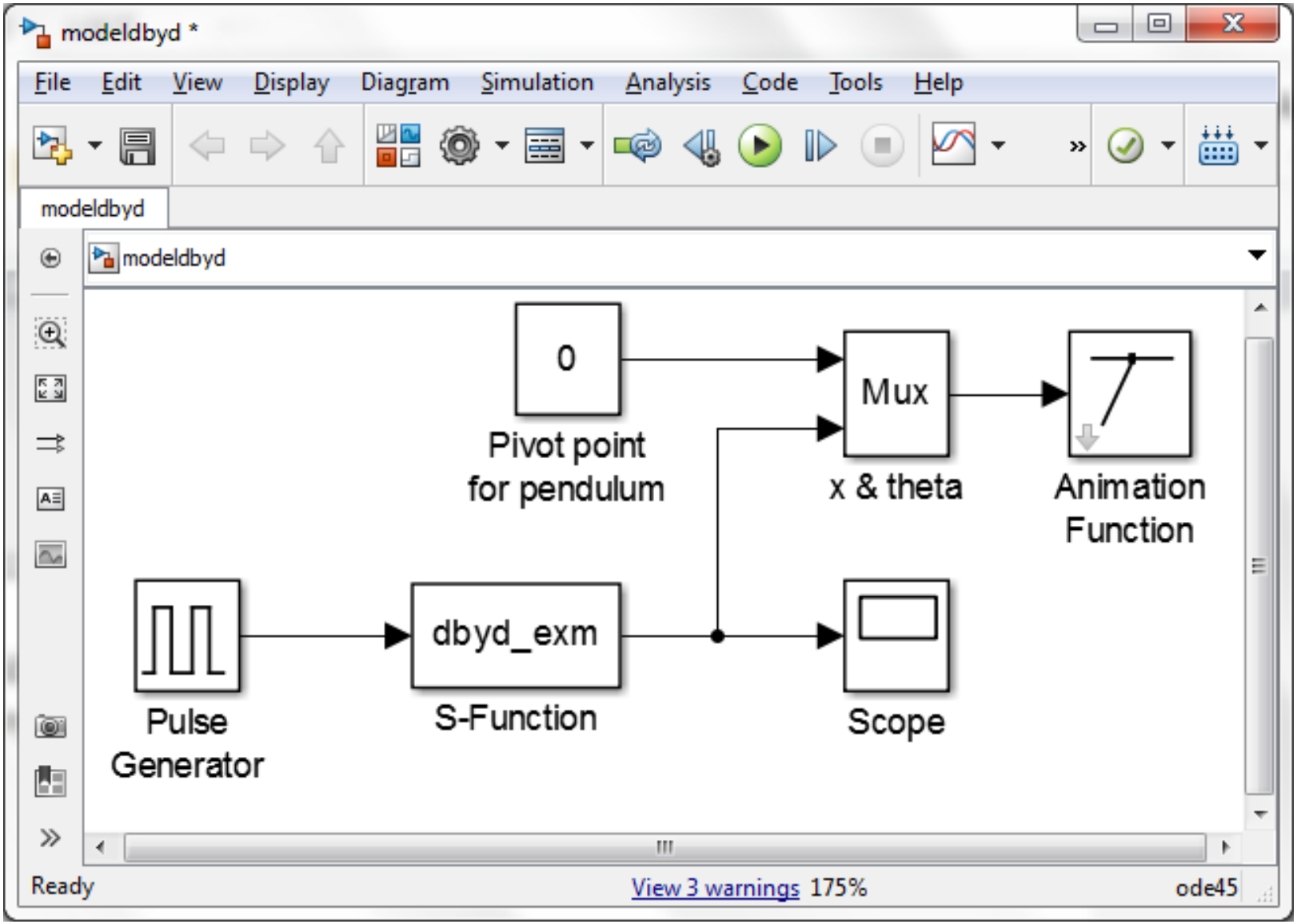


图4.27 摆效果图

把信号发生器替换成脉冲发生器



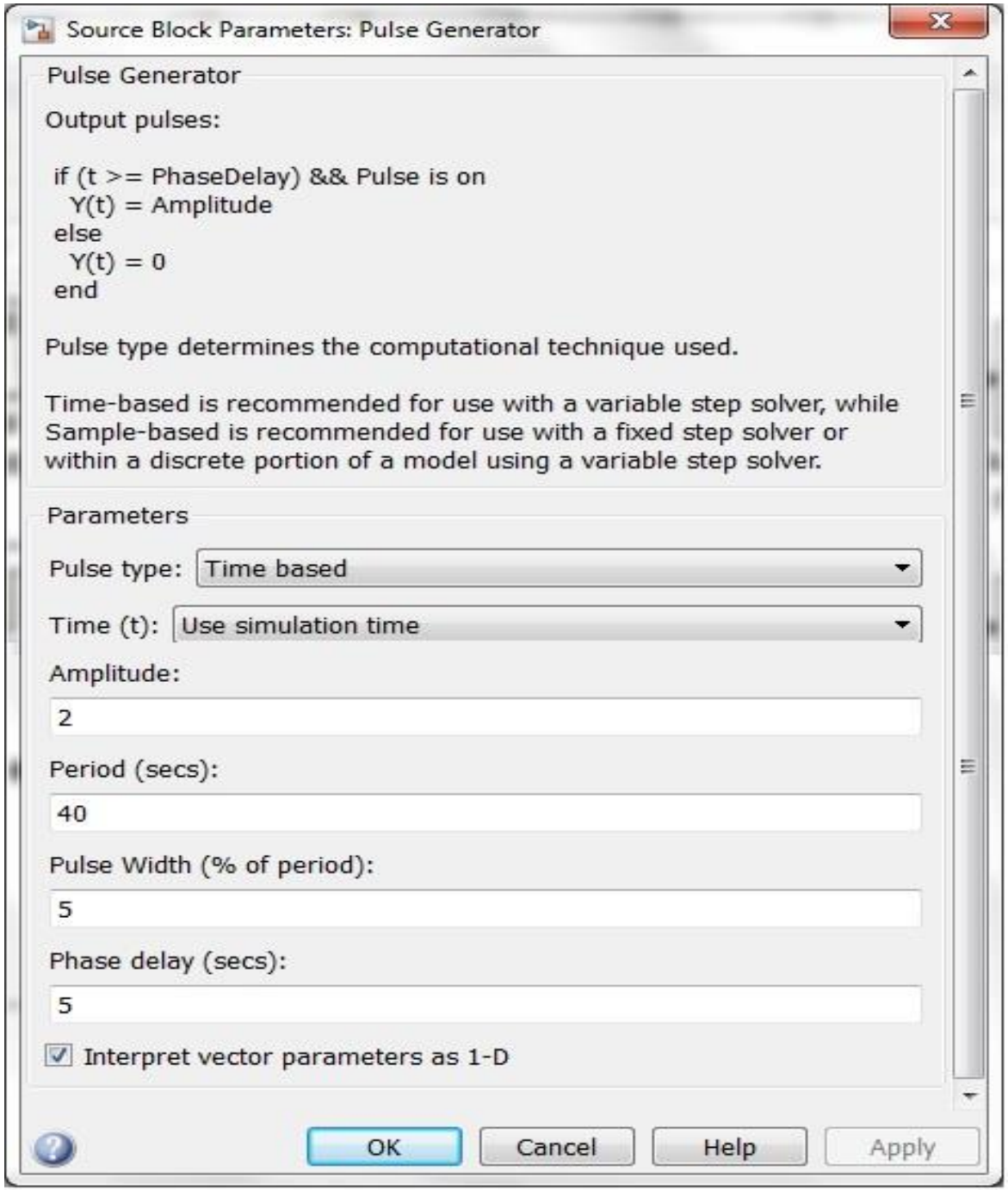
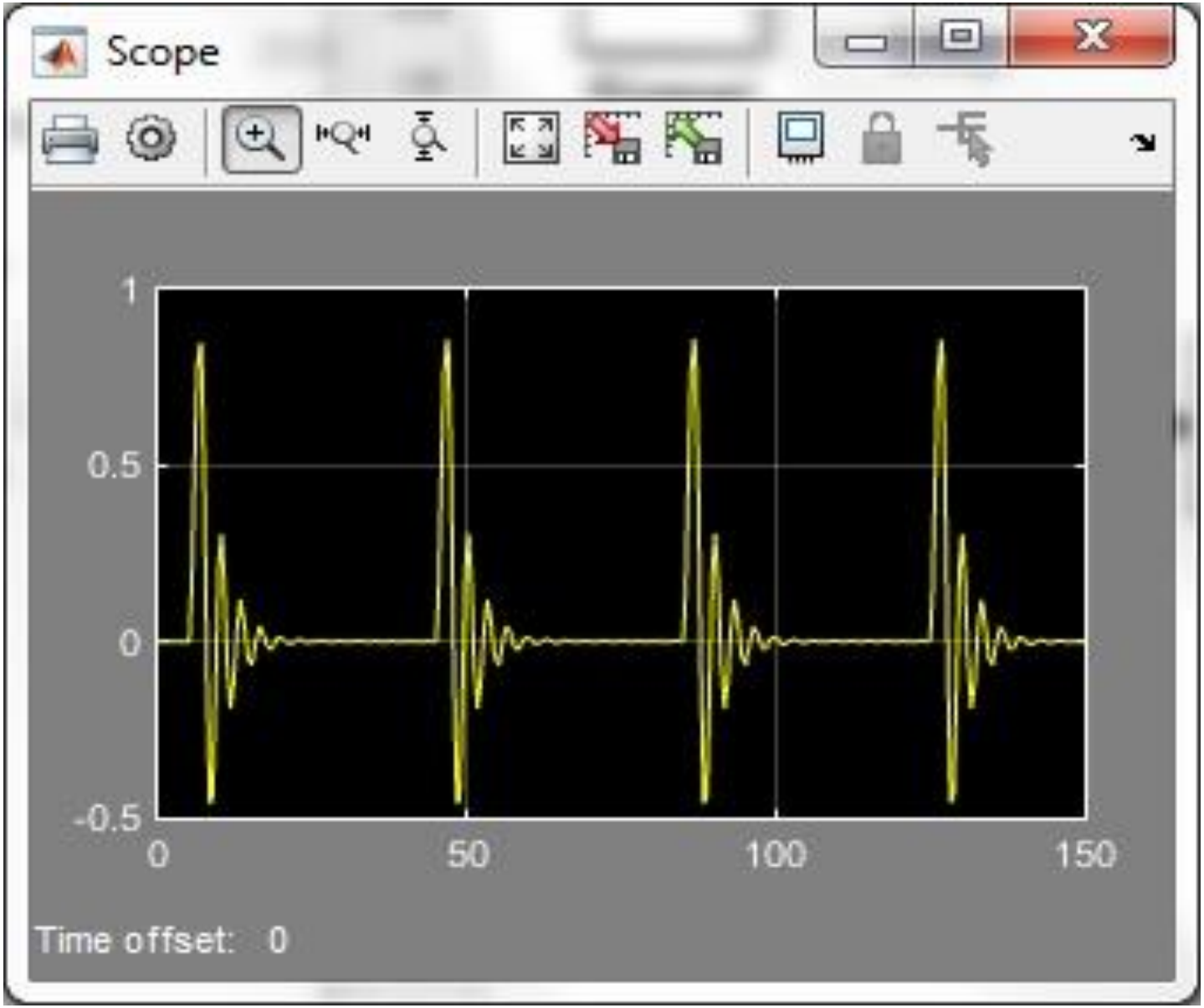


图4.29 脉冲发生器参数设置对话框



仿真结果图

总结

本章给大家介绍了如何将一个系统进行建模仿真。

1. SIMULINK简介。

2. SIMULINK应用实例

3. SIMULINK的扩展工具：S-函数

   利用S-函数模板进行S-函数的编制

4. Simulink仿真模型搭建，S-函数设置，进行系统仿真分析。

Thanks For Listening