МИНИСТЕРСТВО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Сивохин А.В. Мещеряков Б.К.

РЕШЕНИЕ ЗАДАЧ ОПТИМАЛЬНОГО УПРАВЛЕНИЯ С ИСПОЛЬЗОВАНИЕМ МАТЕМАТИЧЕСКОЙ СИСТЕМЫ MATLAB И ПАКЕТА ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ SIMULINK

Лабораторный практикум по основам теории управления

ПЕНЗА 2006

УДК 681.3
С

Рецензенты

Научно – технический совет
Федерального государственного унитарного предприятия
“НПП Рубин”

Кандидат технических наук, доцент кафедры “Прикладная математика и информатика” Пензенского государственного педагогического университета им. В. Г. Белинского В. В. Дрождин

Сивохин А. В., Мещеряков Б. К. Решение задач оптимального управления с использованием математической системы MATLAB и пакета имитационного моделирования SIMULINK. Лабораторный практикум по основам теории управления. – Пенза: Изд-во Пенз. гос. ун-та, 2006 – 120с.

Рассматриваются вопросы применения математического моделирования для решения задач оптимального управления динамическими процессами с использованием системы MATLAB 7.0 и пакета Simulink 5.0.

Лабораторный практикум подготовлен на кафедре “Математическое обеспечение и применение ЭВМ” и предназначен для студентов специальности 23.01.05 при изучении дисциплины “Основы теории управления”.

УДК681.3

Издательство Пензенского государственного университета, 2006

Введение

Дальнейшее развитие науки, техники и производства требует решения всё более и более сложных проблем управления различными динамическими процессами в области механики, электротехники, радиоэлектроники, в экологических и социальных системах. Увеличение производительности, быстроты движений, размеров и мощности машин, увеличение точности и усложнение характера производственного процесса или научного эксперимента затрудняют или даже делают невозможным для человека достаточно быстрое и точное управление движением машин или ходом производственного процесса. Для автоматизации управления машинами и производственными процессами необходимо, прежде всего, глубокое теоретическое и экспериментальное изучение самих процессов управления, а затем создание на основе полученных знаний широкого набора надежных и эффективных приборов, устройств и систем автоматики.

В настоящее время для решения многих из указанных проблем с успехом применяются персональные компьютеры, оснащенные специальным программным обеспечением. Их использование делает более эффективным реализацию аналитических расчетов и многих численных методов, а также позволяет производить визуальное графическое моделирование, имитируя работу исследуемой системы или воспроизводя изменяющуюся во времени структуру сложного прибора или устройства. Такие методы исследования получили название соответственно имитационного и ситуационного моделирования.

Среди ряда современных специальных программных систем компьютерной математики особо выделяется матричная математическая система MATLAB корпорации MathWorks Inc. Эта система является идеальным средством для реализации всех видов моделирования: аналитического, численного, имитационного и ситуационного. Система имеет мощные средства диалога, графики и комплексной визуализации, а также многочисленные программные пакеты для расширения функций системы: символического дифференцирования и интегрирования, идентификации систем, построения и исследования искусственных нейронных систем, обработки сигналов и изображений, решения обыкновенных дифференциальных уравнений и т.д. Одним из таких пакетов системы MATLAB является пакет визуального имитационного и ситуационного моделирования Simulink, позволяющий исследовать многие линейные и нелинейные блочные динамические системы и устройства произвольного назначения. Модель создается из стандартных функциональных графических блоков, набор которых в пакете очень велик и постоянно расширяется. Параметры блоков задаются с помощью удобных диалоговых панелей. Результаты работы регистрируются либо в виде графиков, либо в цифровой форме для последующего применения. По желанию пользователя формируется отчет в формате HTML, который содержит структурную схему модели, перечень

её блоков, таблицы параметров блоков и записи регистрирующих устройств в виде соответствующих графиков и диаграмм.

В данном методическом пособии показано, каким образом надо умело сочетать аналитические подходы и визуальное компьютерное моделирование для решения сложных математических задач, возникающих при исследовании систем оптимального управления, - начиная от решения обыкновенных дифференциальных уравнений и кончая нахождением экстремумов в вариационных задачах и созданием программных нейрорегуляторов для оптимального управления различными динамическими системами.

Лабораторная работа № 1

РАБОТА И ОСНОВЫ ПРОГРАММИРОВАНИЯ В ИНТЕГРИРОВАННОЙ СРЕДЕ МАТЕМАТИЧЕСКОЙ СИСТЕМЫ MATLAB

Цель работы: приобретение навыков и приёмов работы в интегрированной среде математической системы MATLAB и изучение инструментальных средств командного входного языка и языка программирования этой среды.

1.1 Работа в интегрированной среде MATLAB

1.1.1 Запуск системы и доступ к ресурсам

Запуск системы MATLAB 7.01 производится либо с помощью пиктограммы, либо с помощью команды из главного меню рабочего стола Windows. После загрузки программной оболочки системы на экране появляется уменьшенное окно однодокументного приложения MATLAB с главным меню, панелью инструментов, рабочим окном и строкой состояния.

В рабочем окне приложения открываются, как правило, три интерфейсных окна интегрированной среды MATLAB: два слева и одно справа. Это – стандартное и наиболее удобное расположение окон среды. Если такое расположение окон нарушено по каким-либо причинам, то для его восстановления надо исполнить команду View/Desktop Layout/Default. При работе желательно увеличить во весь экран окно приложения и ширину правого окна среды.

После появления в правом окне приглашения в виде знаков “>>” и мигающего курсора, система MATLAB готова к работе. С этого момента начинается сеанс работы среды (сессия), который закончится вместе с завершением работы системы.

Для регистрации событий сессии надо создать дневник, указав имя файла на диске, и включить запись данных из правого окна в этот файл:

```
diary Date_29_01_05;  % - имя содержит текущую дату;
diary on;             % - начало записи в дневник.
```

Перед выходом из системы следует подать команду diary off для прекращения записи в дневник. Дневник позволит сделать анализ работы, обнаружить ошибки и создать обновленный сценарий для более быстрого выполнения запланированной работы. В дальнейшем он станет так же основой электронного отчета по лабораторной работе.

Запустив систему MATLAB и создав для себя дневник, можно приступить к работе в интегрированной среде этой системы, пользуясь ее огромными ресурсами (см. табл. 1.1), дружеским интерфейсом и умными помощниками – специализированными процессорами, которые появляются в нужный момент и в нужном месте, как кудесники или эксперты.

Таблица 1.1

Ресурсы интегрированной среды MATLAB 7.01

Тип	Назначение ресурса среды	Количество	Объём в Кб
.	Все ресурсы в 5877 папках	81449	1843357
*.exe	Исполняемые программы	118	145692
*.dll	Динамически подключаемые библиотеки	2100	235175
*.lib	Библиотеки объектных модулей	624	109463
*.def	Файлы определения модулей	86	309
*.ocx	Управляющие элементы OLE	24	6857
*.tlb	Библиотеки типов OLE	4	27
*.idl	Файлы описания интерфейсов OLE	12	55
*.bat	Командные файлы ОС	97	675
*.p	Псевдокоды библиотечных функций	16241	47816
*.m	Библиотека исходных модулей ядра	37428	98209
*.mat	Библиотека данных ядра	699	111741
*.mdl	Библиотека имитационных моделей	1617	91708
*.asm	Модули на ассемблере	38	148
*.h	Заголовочные файлы C и C++	1672	10133
*.c	Исходные модули C	2843	58647
*.cpp	Исходные модули C++	154	1963
*.rc	Файлы описания ресурсов C и C++	19	59
*.jar	Архивные файлы JAVA	204	340778
*.txt	Текстовые файлы	140	1321
*.doc	Документы WORD	4	471
*.rtf	Обогащенные текстовые файлы	43	2170
*.hlp	Справочные файлы	26	9340
*.htm	Гипертекстовые файлы HTML	104	480
*.html	Гипертекстовые файлы HTML	880	8557
*.xml	Гипертекстовые файлы XML	546	8270
*.bmp	Файлы растровой графики BMP	602	2997
*.gif	Файлы растровой графики GIF	862	9026
*.fig	Графические файлы системы	100	12171
*.db	Базы данных системы	78	22848
.???	Остальные ресурсы	14084	1337106

Основной формой работы в интегрированной среде является диалог на почти естественном входном языке системы, который ориентирован на матричную математику и матричные методы решения задач. Единицами действия языка являются команды, которые выполняются, или интерпретируются немедленно, как только нажимается клавиша ENTER. Команда набирается в правом, командном окне (Command Window) после знака приглашения. Выполнение команд обеспечивает самый главный процессор системы – вычислитель, или интерпретатор команд. Ответ выдается немедленно в это же окно. Более того, все сообщения среды, в том числе и сообщения об ошибках, поступают в это окно. Оно же используется для взаимодействия с другими инструментальными средствами такими, как пакет имитационного моделирования Simulink, пакет символьных вычислений, пакет для решения дифференциальных уравнений в частных производных и т.д.

Настоящим помощником исследователя является еще один процессор – собственный браузер справочной системы среды, который вызывается из меню командой Help/MATLAB Help и обеспечивает все виды работ со справками: поиск информации по оглавлению (вкладка Contents), поиск информации по индексному каталогу (вкладка Index), поиск тем по любой фразе или слову (вкладка Search) и доступ к специальным возможностям системы (Favorites), в том числе к электронной документации в формате PDF при наличии программ Acrobat Reader или Adobe Acrobat (MATLAB Printable Documentation/MATLAB Manuals). Для поиска новых функций, включенных в систему MATLAB 7.01, следует на вкладке Search задать режим Full Text и Functions. Нужную функцию можно найти, установив тип поиска – Function Name и указав имя функции, например, sin. Текст функции можно получить на экране, исполнив команду type. Команды help, doc, lookfor, computer version, help script, help function и другие позволяют получать справки в командном режиме.

За окном команд внимательно следит левое нижнее окно (Command History), которое переписывает к себе все исполняемые команды. С помощью мыши их легко перенести в правое окно для повторного выполнения или записать в файл, предварительно выделив нужный фрагмент. Сразу после запуска системы в этом окне появляется запись с текущей датой и текущим временем запуска системы, например, “%--18.01.05 10.32--%”.

Файлы и папки системы отображаются в левом верхнем окне на вкладке текущего каталога (Current Directory). Требуемый каталог выбирается с помощью инструментов Current Directory, находящихся на панели инструментов вверху справа. Встроенный браузер файлов и папок позволяет быстро переместиться в начало или конец списка просматриваемых файлов, подняться на один уровень вверх по файловой структуре, открыть новую папку, найти файл и т.д. В списке файлов отображается их тип и время последней модификации. Данный браузер позволяет, таким образом, ориентироваться в файловой системе среды и

установить собственный рабочий каталог для обеспечения интерпретации новых программных модулей (m-файлов).

На время сеанса работы пользователю выделяется значительная область оперативной памяти для хранения определяемых для решения задачи новых функций и глобальных данных. Она называется рабочей областью и управляется специальным браузером, активизируемым при щелчке левой клавиши мыши по вкладке Workspace в левом верхнем окне. Окно содержит список используемых объектов с указанием их имен, размеров (как массивов данных), количества занимаемых байтов памяти и классов. Двойной щелчок мыши по выбранной строке данных активизирует редактор массивов (Array Editor), который позволяет модифицировать или проанализировать эти данные.

С помощью браузера рабочей области можно выполнять следующие действия:

а) очистить рабочую область от ранее использованных данных и сделать ее более компактной (команда pack);

б) удалить ненужные данные (команды clear x и clear x,y,z,…);

в) очистить всю рабочую область (команда clear);

г) сохранить данные в файле для использования в последующих сеансах работы (команды save имя файла x и save имя файла x,y,z…);

д) сохранить всю рабочую область в файле для продолжения работы в будущем (команда save имя файла);

е) восстановить данные или всю рабочую область (команда load имя файла).

Во всех этих случаях данные сохраняются в двоичных файлах типа *.mat. При необходимости можно использовать и текстовые форматы.

Команды who и whose позволяют отобразить в командном окне либо список имен объектов рабочей области, либо всю структуру используемых данных и функций. Это позволяет записывать рабочую область в дневник для последующего анализа.

1.1.2 Определение и обработка скалярных данных

Команда clear:

```
x=1
y=2
z=3
whos
clear x y
whos
pause
```

```
x=1
y=2
z=3
clear x,y,z
whos
pause
```

Действительные и комплексные числа:

```
i
j
z=2+3i
abs(z)
real(z)
imag(z)
angle(z)
pause
```

Форматы чисел:

```
format short
x=[4/3 1.234e-6]
format short e
x=[4/3 1.234e-6]
format long
x=[4/3 1.234e-6]
format long e
x=[4/3 1.234e-6]
format bank
x=[4/3 1.234e-6]
pause
```

Свойства вещественных чисел:

```
x=1/0
y=-1/0
w=exp(710)
z=0/0
x=z+1
pause
```

Комплексные числа:

```
x=1.5-0.5i
y=1.5-0.5j
pause
a=1
x=i/a
y=(i)*a
y
z=a*i
```

```
pause
z=(1.5-0.5i)*(2.5+0.8j)
real(z)
imag(z)
pause
```

Конструирование комплексных чисел:

```
complex(1, -1)
conj(z)
x=4.15+0.05i'
y=x'
pause
```

Преобразование числа в строку:

```
num2str(pi)
whos
pause
```

Преобразование числа в строку с заданной точностью:

```
num2str(pi, 10)
num2str(pi, 20)
pause
```

Преобразование комплексного числа в строку:

```
x=sqrt(-2)
num2str(x, 10)
whos
pause
```

Форматное преобразование числа в строку:

```
num2str(pi, 'pi%6.2f')
whos
pause
```

Решение системы линейных уравнений

```
b=[1;2;3]
a=[1 -1 1; 2 0 2; 0 1 1]
x=inv(a)*b
whos
```

```
pause
```

MATLAB в роли суперкалькулятора:

```
i = 2 + 3
sin(1)
type sin
help sin
V=[1 2 3 4]
sin(V)
3*V
V^2
V.^2
V+2
pause
```

1.1.3 Определение и обработка числовых векторов

Особенности задания векторов и матриц:

```
V=[1 2 3]
M=[1 2 3; 4 5 6; 7 8 9]
M
V=[2+2/(3+4) exp(5) sqrt(10)]
V
M(2, 2)
M(2, 2)=10
M=[1 2 3; 4 5 6; 7 8 9]
M(2)
M(8)
M(9)
M(5)=100
M
i=sqrt(-1);
CM = [1 2; 3 4] + i*[5 6; 7 8]
CM = [1+5*i 2+6*i; 3+7*i 4+8*i]
M=magic(4)
sum(M)
sum(M')
sum(diag(M))
M(1,2)+M(2,2)+M(3,2)+M(4,2)
```

```
pause
```

Символьный вектор:

```
sv='abcdefgh'
whos
pause
```

Символьная матрица:

```
sm=['abcdefgh';'12345678']
whos
pause
```

1.1.4 Определение и обработка числовых матриц

Файл-функция для заполнения матрицы:

```
function A=exam(n)
A=zeros(n);
for i=1:n
   for j=1:n
      A(i,j)=1/(i^2+j^2);
   end
end
```

Процедура перемножения матриц:

```
subroutine multmtr(n1, n2, n3, X, Y, Z)
C Вычисление Z=X*Y, где X - n1xn2,  Y - n2xn3,  Z - n1Xn3
integer n1, n2, n3, i, j, k
real*8 X(n1,n2), Y(n2,n3), Z(n1,n3)
     do 30 i=1,n1
       do 20 j=1,n3
          Z(i,j)=0.0d0
          do 10 k=1,n2
            Z(i,j)=Z(i,j)+X(i,k)*Y(k,j)
10        continue
20      continue
30   continue
end
```

Вычисление элементов матрицы с предварительным выделением памяти:

```
A=zeros(400);
for i=1:400
   for j=1:400
      A(i,j)=1/(i^2+j^2);
   end
end
```

Сравнение матричных операндов:

```
a=[1 2 3; 4 5 6]
b=[2 2 2; 5 5 5]
c=a >= b
pause
```

Объединение малых матриц в большую:

```
A=magic(3)
B=[A A+16; A+32 A+16]
sum(B)
sum(B.')
D=magic(6)
sum(D)
sum(D.')
pause
```

Удаление столбцов и строк матриц:

```
M=[1 2 3; 4 5 6; 7 8 9]
M(:,2)=[]
M(2,:)=[]
pause
```

1.1.5 Определение и обработка символьных данных

Константы и символьные переменные:

```
2*pi
eps
realmin
realmax
1/0
0/0
'Hello my friend'
'Привет'
'2+3'
pause
```

Конкатенация строк с пробелами в конце:

```
strcat('C ', 'Новым ', 'Годом!');
strcat('C', 'Новым', 'Годом!')
['C ', 'Новым ', 'Годом!']
pause
```

Сравнение символьных данных:

```
s1='ABCDEFGH'
s2='ABCDefgh'
s3='ABCabc'
strcmp(s1,s2)
strcmpi(s1,s2)
strncmp(s1,s2,3)
whos
sm1=['1234567'; 'ABCDEFG']
sm2=['1234567'; 'ABCDefg']
strcmp(sm1, sm2)
strncmp(sm1, sm2, 4)
strcmpi(sm1, sm2)
sm3=['1234567'; 'ABCDEF']
sm4=['1234567'; 'ABCdef']
strcmpi(sm3, sm4)
sc1=[{'1234'}; 'ABCDEFGH']
sc2=[{'1235'}; 'ABCDefgh']
strcmp(sc1, sc2)
strcmpi(sc1, sc2)
whos
strncmp(sc1, sc2, 3)
pause
```

Преобразование регистра:

```
lower('Happy New Year - С Новым Годом!')
upper('Happy New Year - С Новым Годом!')
pause
```

Выделение лексем:

```
s='С Новым Годом!'
[t1, r1] = strtok(s)
[t2, r2] = strtok(r1)
[t3, r3] = strtok(r2)
pause
```

Поиск символьных данных:

```
s1='00'
s2='2003'
s3='100002'
findstr(s1,s2)
findstr(s2,s1)
strfind(s1,s2)
strfind(s2,s1)
s3='100002'
findstr(s3,s1)
whos
pause
```

Выделение лексем с нестандартными разделителями:

```
s='a+b*c'
[t1, r1] = strtok(s, '+*')
[t2, r2] = strtok(s, '+*')
[t3, r3] = strtok(s, '+*')
pause
```

Поиск символьных данных с использованием strmatch:

```
sm=strvcat('com', 'compare', 'computer')
strmatch('com', sm)
strmatch('com', sm, 'exact')
s='123com'
```

```
strmatch('com',s)

sc={'com';'compare';'computer'}
strmatch('com', sc)
whos
pause
```

Поиск и замена символьных данных:

```
s='12341234'
s1=strrep(s, '123', 'ABCD')
s2=strrep(s, '124', 'ABCD')

s3=strep(s, '123', '')
pause
```

Присваивание символьным переменным:

```
str='ABCDE'
whos
str_mas=['1234';'ABCD']
whos
str_cell=[{'123', 'ABCD'};{'4567', 'abc'}]
whos
pause
```

Печать таблицы ASCII:

```
char(32:64)
char(65:96)
char(97:126)
char(168:184)
char(192:223)
char(224:255)
pause
```

Заполнение строки:

```
repmat('=', 1,4)
repmat('*', 3,4)
pause
```

Выравнивание строки:

```
s='    123456     '
sl=strjust(s, 'left')
sc=strjust(s, 'center')
sr=strjust(s, 'rught')
whos
pause
```

Выполнение фрагментов строки:

```
s='С Новым Годом!';
s(1:2)
s(3:8)
s(9:end)
```

1.1.6 Определение и обработка многомерных массивов

Формирование трехмерного массива:

M=[1 2 3; 4 5 6; 7 8 9]

```
M(: , : , 2)=[10,11,12; 13,14,15; 16,17,18]
M
```

Объединение многомерных массивов:

```
M1=[1 2;3 4]
M2=[5 6;7 8]
cat(1,M1,M2)
cat(2,M1,M2)
M=cat(3,M1,M2)
```

Перестановка размерностей массива:

```
A=[1 2;3 4];
B=[5 6;7 8];
C=[9 10;11 12];
D=cat(3,A,B,C)
size(D)
size(permute(D, [3 2 1]))
size(permute(D, [2 1 3]))
```

Сдвиг и удаление размерностей многомерных массивов:

```
A=randn(1,2,3,4);
[B,N]=shiftdim(A)
```

```
A=randn(1,2,1,3,1);
B=squeeze(A)
```

1.1.7 Определение и обработка структур

Формирование структуры:

```
authors = struct('name', 'Сидоров С.С.','age',33,'phone','33-33-33');
whos
pause
```

Файл-программа для заполнения массива структур при помощи операторов присваивания:

```
% Заполнение первой структуры массива
GR201(1).Family = 'Алексеев';
GR201(1).Name = 'Иван';
GR201(1).Year = 1980;
GR201(1).Marks = [5 4 4 5 5 4];
% Заполнение второй структуры массива
GR201(2).Family = 'Васильев';
GR201(2).Name = 'Сергей';
GR201(2).Year = 1981;
GR201(2).Marks = [3 4 4 3 5 4];
% Заполнение третьей структуры массива
GR201(3).Family = 'Кашин';
GR201(3).Name = 'Павел';
GR201(3).Year = 1979;
GR201(3).Marks = [4 3 4 4 5 4];
% Заполнение четвертой структуры массива
GR201(4).Family = 'Серова';
GR201(4).Name = 'Наталья';
GR201(4).Year = 1981;
GR201(4).Marks = [4 3 3 5 4 5];
% Заполнение пятой структуры массива
GR201(5).Family = 'Терехова';
GR201(5).Name = 'Ольга';
GR201(5).Year = 1980;
GR201(5).Marks = [5 5 5 5 4 5];
```

Вывод значений полей структуры в командное окно:

```
Len = max(size(GR201));
for k = 1:Len
```

```
  disp(GR201(k))
end
```

Создание файл-функций для работы с массивами структур:

```
function meanmarks = groupprog(GROUP);
% Функция вычисляет средний балл студентов по каждому предмету
% и выводит результат в виде столбцевой диаграммы.
% Возвращает массив, каждый элемент которого равен
% среднему баллу по предмету с соответствующим номером
% Использование meanmark = groupprog(GROUP)
%    GROUP - массив структур с полями
%       Family (строка), Name (строка), Year (число),
%       Marks (вектор-строка с отметками)

% Нахождение числа студентов в группе
N = max(size(GROUP));
% Определение количества курсов по информации для
% первого студента
Courses = length(GROUP(1).Marks);
% Инициализация массива meanmarks
meanmarks = zeros(1, Courses);
% Перебор курсов и вычисление средней успеваемости
for course = 1:Courses
  % Суммирование баллов, полученных каждым из студентов по
  % курсу с номером course
  for student = 1:N
    meanmarks(course) = meanmarks(course) +...
      GROUP(student).Marks(course);
  end
  % Нахождение среднего арифметического
  meanmarks(course) = meanmarks(course) / N;
end
% Построение столбцевой диаграммы
bar(meanmarks);
```

Запись данных в текстовый файл:

```
function writegroup(filename, GROUP)
% Файл-функция для записи таблицы с успеваемостью группы
% студентов в текстовый файл.
% Использование  writegroup(filename, GROUP)
%    filename - имя файла
%    group - массив структур с полями
%       Family (строка), Name (строка), Year (число),
```

```
%       Marks (вектор-строка с шестью отметками)

% Нахождение числа студентов в группе
N = max(size(GROUP));
% Открытие файла с именем filename для записи
F = fopen(filename, 'w');
% Запись шапки таблицы
fprintf(F, '%-14s %-10s %-4s  %-6s\n',...
     'Фамилия', 'Имя', 'Год' , 'Оценки');
% Запись в файл содержимого полей каждой структуры в строку
for s = 1:N
  fprintf(F, '%-14s %-10s %4.0f...
    %2.0f %2.0f %2.0f %2.0f %2.0f %2.0f\n',...
    GROUP(s).Family, GROUP(s).Name, GROUP(s).Year, GROUP(s).Marks);
end
% Закрытие файла
fclose(F);
```

1.1.8 Определение и обработка ячеек

Список ячеек:

```
authors=struct('name', {'Сидоров С.С.', 'Иванов И.И.'},'age',{31, 33},'phone',{'11-11-11','33-33-33'});
whos
pause
```

Поиск и замена в массиве ячеек:

```
s1='12341234'
s2=strrep(s1,'123', {''})
whos
pause
```

Преобразование массива ячеек в символьный массив:

```
s=char({'ABC'}, '1234')
whospause
```

Извлечение строки из массива ячеек:

```
s_c=[{'123','ABCD'};{'45678','abc'}]
s_c(1,2)
s_c(1, : )
```

```
pause
```

Конкатенация массивов ячеек:

```
s=strcat({'123','ABC'},{'5678','xz'})
whos
pause
```

Вертикальная конкатенация массива ячеек:

```
s=strvcat({'123', 'ABCD'})
whos
pause
```

Файл-функция students, выделяющая фамилии студентов из массива ячеек:

```
 function strmas = students(CELLMAS)
% Файл-функция формирует массив строк с фамилиями
% студентов, участвующих в эксперименте.
% Использование strmas = students(CELLMAS)

% Определение размеров массива ячеек
SizeMas = size(CELLMAS);
% Нахождение числа студентов(информация о деятельности
% каждого сдудента хранится в столбце)
NStudents = SizeMas(2);
if NStudents >= 1
% Если число студентов больше или равно единице, то
% считываем из поля Family третьей ячейки первого столбца
% фамилию студента и заносим в массив строк strmas
   strmas = CELLMAS{3, 1}.Family;
end
% Продолжаем считывание по всем оставшимся столбцам,
% начиная со второго
for k = 2:NStudents
   % Считываем фамилию из поля Family третьей ячейки k-го столбца
   fam = CELLMAS{3, k}.Family;
   % Добавляем фамилию в массив строк
   strmas = char(strmas, fam);
end
```

1.1.9 Определение и обработка объектов классов

Справка по конкретному объекту:

```
help hsin
help sinh
help pi
pause
```

Справка по группе объектов:

```
help timefun
pause
```

Файл-программа для вывода текстового объекта в точку с заданными координатами:

```
% Создание графического окна и осей
HF = figure('Menu', 'none', 'Color', 'w');
HAx = axes;
% Генерация векторов значений аргумента и функции
x = [-2:0.01:3];
y = exp(-x.^2);
plot(x,y)
% Задание координат положения текстового объекта
Xtext = 1.17;
Ytext = exp(-Xtext.^2);
% Создание и отображение текстового объекта
HTxt = text(Xtext, Ytext, '\leftarrow Функция {\ity} = {\ite}^{-x^2}');
```

Операторы для получения ординаты точки минимума в zeroandmin.m:

```
% Поиск локального минимума функции на [a,b]
% Получение значения функции и аргумента локального минимума
[minvalue, fmin] = fminbnd(funname, a, b, options);
```

Заголовки и пояснения:

```
axes(HAx(1)) % левые верхние оси текущие
title('Лок. минимум') % добавление заголовка
```

```
axes(HAx(2)) % правые верхние оси текущие
title('Ноль функции') % добавление заголовка
axes(HAx(3)) % нижние оси текущие
% Создание текстового объекта, расположенного в нуле функции
HTxtZer = text('String', '\leftarrow ноль', 'Position' , [zero 0]);
% Создание текстового объекта, расположенного в лок. минимуме функции
HTxtMin = text('String', '\leftarrow лок.мин.',...
   'Position' , [minvalue fmin], 'Rotation', 90);
```

Операторы, демонстрирующие изменение свойств линий и осей при помощи указателей:

```
% Установка свойства XGrid осей с указателем HAx в 'on'
set(HAx,'XGrid','on')
% Задание цвета первой линии (графика синуса) с указателем HLines(1)
set(HLines(1), 'Color', 'k')
% Задание толщины первой линии (графика синуса) с указателем HLines(1)
set(HLines(1), 'LineWidth', 3)
% Задание цвета второй линии (графика косинуса) с указателем HLines(1)
set(HLines(2), 'Color', 'k')
% Задание типа маркера второй линии (графика косинуса)
% с указателем HLines(2)
set(HLines(2), 'Marker', 'o')
% Задание цвета маркеров второй линии (графика косинуса)
% с указателем HLines(2)
set(HLines(2), 'MarkerFaceColor', 'w')
% Задание цвета границ маркеров второй линии (графика косинуса)
% с указателем HLines(2)
set(HLines(2), 'MarkerEdgeColor', 'k')
```

Файл-функция, использующая указатели на объекты:

```
function maxfun(a,b)
% Файл-функция с интерфейсом из командной строки для
% исследования функций на максимум на отрезке [a,b]
% Использование: maxfun(a,b)

% Создание графического окна (оно становится текущим)
HF = figure;
% Создание осей в текущем графическом окне
HAx = axes;
% Задание вектора значений аргумента
x = [a:(b-a)/30:b];
str = ''; % инициализация строки запроса ввода пользователя
```

```
funcount = 0; % инициализация счетчика введенных функций
maximums = [];% инициализация массива с максимумами функций
hFuns = []; % инициализация массива указателей на линии графиков
% Обработка ввода пользователя в бесконечном цикле
while 1
  str = input('Введите функцию, или new, или end: ', 's');
  switch str
  case 'new'  % Пользователь задал очистку осей
    axes(HAx); % оси с указателем HAx стали текущими
    cla % очистка текущих осей
  case('end') % Пользователь завершает работу с программой
    break % выход из цикла
  otherwise  % Пользователь ввел новую функцию
    funcount = funcount + 1; % увеличение счетчика функций
    % Формирование команды для вычисления массива значений
    eval(strcat('y =', str, ';'));
    % Оси HAx должны быть текущими для вывода графика
    axes(HAx)
    hold on % график следует добавить на оси
    % Построение графика функции, введенной пользователем,
    % и добавление указателя на него в массив указателей
    HFuns(funcount) = plot(x,y)
    % Установка требуемых свойств линии графика новой функции
    set(HFuns(funcount), 'Marker', 'o')
    set(HFuns(funcount), 'MarkerEdgeColor', 'k')
    set(HFuns(funcount), 'MarkerFaceColor', 'w')
    set(HFuns(funcount), 'LineWidth', 1)
    set(HFuns(funcount), 'Color', 'k')
    % Вычисление максимума новой функции и добавление
    % его значения в массив, содержащий максимумы
    maximums(funcount) = max(y);
    if funcount > 1 % пользователь ввел две или более функций
      % Удаление маркеров с графика предыдущей функции
      set(HFuns(funcount-1), 'Marker', 'none')
      % Поиск функции с максимальным значением среди ранее
      % введенных, в oldmax записывается значение,
      % а в N - номер требуемой функции
      [oldmax, N] = max(maximums(1:funcount-1));
      % График найденной функции должен отображаться жирной линией
      set(HFuns(N), 'LineWidth', 3)
      % Сравнение максимального значения новой функции с
      % максимальным из значений предыдущих функций
      if maximums(funcount) > oldmax
        % Новая функция принимает самое большое значение среди
        % всех функций, введенных пользователем, поэтому
```

```
          % график последней введенной функции должен рисоваться
          % жирной линией
          set(HFuns(funcount), 'LineWidth', 3)
          % Линия графика функции, которая ранее имела максимальное
          % значение, теперь не должна быть жирной
          set(HFuns(N), 'LineWidth', 1)
        end
      end
    end
  end
```

1.1.10 Определение и использование скриптов и функций

Демонстрация работы функции isfinite(), isinf() и isnan():

```
x=[1 0 inf nan]
y=1./x
zf=isfinite(y)
zi=isinf(y)
zn=isnan(y)
pause
```

Демонстрация работы функции issorted():

```
a=[1 2 3 4 5]
issorted(a)
b=[5 4 3 2 1]
issorted(b)
c='abc'
issorted(c)
d='bac'
issorted(d)
e=[1 2 3; 2 3 4]
issorted(e, 'rows')
pause
```

Демонстрация работы функции isreal():

```
q{1,1}='Text';
q{1,2}=3;
q{1,3}=magic(2);
q{1,4}=1+i;
q{1,5}=isreal(pi);
```

```
q
a
pause
```

Демонстрация работы функции isnumeric():

```
q{1,1}='Text';
q{1,2}=3;
q{1,3}=magic(2);
q{1,4}=1+i;
q{1,5}=isreal(pi);
for j=1:5, a(j)=isnumeric(q{1,j}) end
pause
```

Демонстрация работы функции ischar():

```
q{1,1}='Text';
q{1,2}=3;
q{1,3}=magic(2);
q{1,4}=1+i;
q{1,5}=isreal(pi);
for j=1:5, a(j)=ischar(q{1,j}) end
pause
```

Демонстрация работы функции islogical():

```
q{1,1}='Text';
q{1,2}=3;
q{1,3}=magic(2);
q{1,4}=1+i;
q{1,5}=isreal(pi);
for j=1:5, a(j)=islogical(q{1,j}) end
pause
```

Демонстрация работы функции isletter():

```
x='a1b2c3'
isletter(x)
y=['a1';'b2';'c3']
isletter(y)
pause
```

Передача аргументов функции:

```
function [varargout] = array2vec(a)
```

```
for k=1:nargout
   varargout{k} = a{k, : }
end

a1={1 2; 3 4; 5 6; 7 8; 9 10; 11 12}
[p1 p2]=array2vec(a1)
[q1 q2 q3 q4 q5]=arra2vec(a1)
whos
pause
```

Файл-функция с переменным числом входных аргументов

```
function where = point(varargin)
% Файл-функция определяет попадание точки с заданными
% координатами (px, py) в круги с центрами
% в (x1,y1), (x2, y2) и т. д. и радиусами R1, R2 и т. д.
% Возвращает
%        1 в случае попадания
%        0 в случае непопадания
% Использование where = point(px,py,[x1,y1,R1],[x2,y2,R2],...)

% Проверка числа входных аргументов (числа ячеек varargin)
if length(varargin) < 3
  error('Недостаточно входных аргументов')
end
% Выделение координат точки из первых двух ячеек
Xpoint = varargin{1};
Ypoint = varargin{2};
% Нахождение числа заданных кругов
% (число ячеек varargin без первых двух)
Ncircle = length(varargin) - 2;
% Извлечение координат центров и радиусов кругов
for i = 1:Ncircle
  Xcircle(i) = varargin{i+2}(1);
  Ycircle(i) = varargin{i+2}(2);
  Rcircle(i) = varargin{i+2}(3);
end
% Полагаем where=0, т. е. пока нет ни одного нужного круга
where = 0;
% Перебор кругов в цикле
for i = 1:Ncircle
  % Вычисление расстояния от точки до центра текущего круга
  dist = sqrt((Xpoint-Xcircle(i))^2+(Ypoint-Ycircle(i))^2);
  % Сравнение расстояния с радиусом круга
```

```
  if  dist <= Rcircle(i)
    where = 1;  % Требуемый круг найден
    break       % Дальше проверять нет смысла
  end
end


%
% Операторы для проверки входных аргументов файл-функции point
%

if ~isnumeric(varargin{1}) | ~isreal(varargin{1}) |...
    max(size(varargin{1}) ~= 1)
  error('Аргумент N1 должен быть вещественным числом')
end
if ~isnumeric(varargin{2}) | ~isreal(varargin{2}) |...
    max(size(varargin{2}) ~= 1)
  error('Аргумент N2 должен быть вещественным числом')
end
for i = 3:length(varargin)
  if ~isnumeric(varargin{i}) | ~isreal(varargin{i}) |...
      min(size(varargin{i})) ~= 1 | length(varargin{i}) ~=3 |...
      varargin{i}(3) < 0
    str1 = 'Аргумент N';
    str2 = num2str(i);
    str3 = ' долж. быть вещ. вектором длиной 3 с третьим эл-том >= 0';
    strerror = strcat(str1, str2, str3);
    error(strerror)
  end
end


%
% Операторы вывода в командное окно для файл-функции point
%

% Отображение данных в графическое окно
figure; % Создание окна
% Построение окружностей
t = [0:pi/20:2*pi]; % задание вектора параметра
for i = 1:Ncircle
  % Вычисление векторов, соответствующих параметрически
  % заданным функциям, которые определяют окружности
  x = Rcircle(i)*cos(t) + Xcircle(i);
  y = Rcircle(i)*sin(t) + Ycircle(i);
```

```
  plot(x,y) % построение окружности
  hold on
end
% Вывод точки красным маркером
plot(Xpoint,Ypoint, 'or')
hold off
axis square % сохранение одинакового масштаба осей


%
% Файл-функция  переменным числом входных и выходных аргументов
%

function [where, varargout] = point(varargin)
% Файл-функция определяет попадание точки с заданными
% координатами (px, py) в круги с центрами
% в (x1,y1), (x2, y2) и т. д. и радиусами R1, R2 и т. д.
% Использование
%   where = point(px,py,[x1,y1,R1],[x2,y2,R2],...)
%       where равно 1, если точка попала в какой-либо круг,
%                  0, в противном случае
%   [where, NC] = point(px,py,[x1,y1,R1],[x2,y2,R2],...)
%       NC равно числу кругов, содержащих точку
%   [where, NC, Nums] = point(px,py,[x1,y1,R1],[x2,y2,R2],...)
%       В вектор Nums записываются номера кругов, содержащих точку

% Выделение координат точки из первых двух ячеек
Xpoint = varargin{1};
Ypoint = varargin{2};
% Нахождение числа заданных кругов
% (число ячеек varargin без первых двух)
Ncircle = length(varargin) - 2;
% Извлечение координат центров и радиусов кругов
for i = 1:Ncircle
  Xcircle(i) = varargin{i+2}(1);
  Ycircle(i) = varargin{i+2}(2);
  Rcircle(i) = varargin{i+2}(3);
end
% Сначала where=0, т. е. пока нет ни одного нужного круга
where = 0;
% Сначала число кругов, содержащих точку, равно нулю
NC = 0;
% Перебор кругов в цикле
for i = 1:Ncircle
  % Вычисление расстояния от точки до центра текущего круга
```

```
  dist = sqrt((Xpoint-Xcircle(i))^2+(Ypoint-Ycircle(i))^2);
  % Сравнение расстояния с радиусом круга
  if  dist <= Rcircle(i)
    where = 1;  % Требуемый круг найден
    % Увеличение числа найденных кругов на единицу
    NC = NC + 1;
    % Сохранение номера круга в массиве Nums
    Nums(NC) = i;
  end
end
% Запись полученных результатов в выходной массив ячеек в зависимости
% от числа выходных аргументов, с которыми была вызвана point
switch nargout
case(2)
  % Было указано два выходных аргумента, следовательно, надо записать
  % только число кругов в первую ячейку varargout
  varargout{1} = NC
case(3)
  % Было указано три выходных аргумента, следовательно, надо записать
  % число кругов и массив с их номерами в первые две ячейки varargout
  varargout{1} = NC;
  varargout{2} = Nums;
end



%
% Файл-функция half для решения уравнений методом половинного деления
%

function root = half(fname, left, right, epsilon)
% Файл-функция находит корень уравнения f(x)=0
% методом половинного деления
% Использование
% root = half(fname, left, right, epsilon)
%      fname - имя файл-функции, вычисляющей f(x)
%      left, right - левая и правая границы отрезка
%      epsilon - точность вычислений

% Проверка значений функции на границах отрезка
if feval(fname, left)*feval(fname, right) > 0
  error('Одинаковые знаки функции на границах отрезка')
end
% Деление отрезка пополам
while (right - left) > epsilon
  center = (right + left)/2; % вычисление середины
```

```
  % Проверка на равенство f(x) нулю в center
  if feval(fname, center) == 0
    break    % найден точный корень, дальше делить нет смысла
  end
  % Выбор нужной половины отрезка, на границах которой
  % f(x) принимает значения разных знаков
  if feval(fname, left)*feval(fname, center) < 0
    right = center;
  else
    left = center;
  end
end
% Приближенное значение корня равно координате любой границы
% последнего полученного отрезка
root = center;


%
% Файл-функция half с переменным числом аргументов
%

function [root, varargout] = half(fname, left, right, varargin)
% Файл-функция находит корень уравнения f(x)=0
% методом половинного деления
% Использование
% root = half(fname, left, right, epsilon)
%      fname - имя файл-функции, вычисляющей f(x)
%      left, right - левая и правая границы отрезка, на
%                  котором находится корень
%      epsilon - точность вычислений, если не задана, то
%                по умолчанию 1.0e-03
% [root, Fun] = half(fname, left, right, epsilon)
%      Fun = f(root)

% Если число входных аргументов равно четырем, то последний
% аргумент содержит точность вычислений, а если трем, то точность
% устанавливается по умолчанию 1.0e-03
switch nargin
case(4)
  epsilon = varargin{1};
case(3)
  epsilon = 1.0e-03;
otherwise
  error('Может быть три или четыре входных аргумента')
end
```

```
% Проверка значений функции на границах отрезка
if feval(fname, left)*feval(fname, right) > 0
  error('Одинаковые знаки функции на границах отрезка')
end
% Деление отрезка пополам
while (right - left) > epsilon
  center = (right + left)/2; % вычисление середины отрезка
  % проверка на равенство f(x) нулю в середине отрезка
  if feval(fname, center) == 0
    break % найден точный корень, дальше делить нет смысла
  end
  % Выбор нужной половины отрезка, на границах которой
  % f(x) принимает значения разных знаков
  if feval(fname, left)*feval(fname, center) < 0
    right = center;
  else
    left = center;
  end
end
% Приближенное значение корня равно координате любой границы
% последнего полученного отрезка
root = center;
if nargout == 2
  varargout{1} = feval(fname, root);
end



%
% Файл-функция simple с подфункцией f в одном файле
%

function simple;
% Основная функция
f1 = f(1.1,2.1)
f2 = f(3.1,4.2)
f3 = f(-2.8,0.7)

function z = f(x,y)
% Подфункция
z = x^3 - 2*y^3 + 3*(x^2+y^2)- x*y + 9;
```

1.1.11 Работа с файлами

Работа с папками:

```
cd
cd E:\matlabr12\tool
cd E:\matlabr12\toolbox\
cd
pwd
cd E:\matlabr12\toolbox\matlab
dir
pause
```

Некоторые другие команды:

```
getenv('temp')
tempdir
computer
[C S] = computer
pause
```

Файл-функция simple с подфункцией f в одном файле:

```
function simple;
% Основная функция
f1 = f(1.1,2.1)
f2 = f(3.1,4.2)
f3 = f(-2.8,0.7)

function z = f(x,y)
% Подфункция
z = x^3 - 2*y^3 + 3*(x^2+y^2)- x*y + 9;
```

Чтение данных из двоичного файла:

```
f_id=fopen(‘test.m’,’r’);
[a k]=fread(f_id)
a=char(a)
a’
```

Запись и чтение из текстового файла:

```
 k=fopen(‘a.txt’,’wt’);
 a=magik(5)
 n=fprintf(k,’%d’,a)
 fclose(k);
 k=fopen(‘a.txt’,’rt’);
[b n]=fscan(k,’%d’,[5 5])
frewind(k);
[c n]= (k,’%d’,5)
whos
```

1.1.12 Графическая визуализация

Построение трехмерных графиков:

```
[X,Y]=meshgrid(-5:0.1:5);
Z=X.*sin(X+Y);
meshc(X, Y, Z)
pause
```

Формирование линий и маркеров для графика нескольких функций:

```
x=-6:.1:6;
plot(x, sin(x), x, sin(x).^3, x, sin(x).^5);
pause
```

Работа с камерой 3D-графики:

```
Z=peaks(40);
mesh(Z);
pause
```

Построение в одном окне графиков нескольких функций:

```
y1=sin(x); y2=cos(x); y3=sin(x)/x;
plot(x, y1, x, y2, x, y3)
pause
```

Маркировка линий уровня на контурных графиках:

```
[X, Y] = meshgrid([-3:0.1:3]);
Z = sin(X)./(X.^2+Y.^2+0.3);
C = contour(X, Y, Z, 10);
colormap(gray)
clabel(C)
pause
```

Управление свойствами осей графиков:

```
x = -5:0.1:5;
plot(x, sin(x));
axis([-10 10 -1.5 1.5])
pause
```

Включение и выключение сетки:

```
x = -5:0.1:5;
plot(x, sin(x));
axis([-10 10 -1.5 1.5])
grid on
```

Наложение графиков друг на друга:

```
x = -5:0.1:5;
plot(x, sin(x))
hold on
plot(sin(x), cos(x))
plot(2*sin(x), cos(x))
plot(4*sin(x), cos(x))
hold off
pause
```

Разбиение графического окна:

```
x = -5:0.1:5;
subplot(2, 2, 1), plot(x, sin(x))
subplot(2, 2, 2), plot(sin(5*x), cos(2*x+0.2))
subplot(2, 2, 3), contour(peaks)
subplot(2, 2, 4), surf(peaks)
pause
```

Изменение масштаба графика:

```
x = -5:0.01:5;
plot(x, sin(x.^5)./(x.^5+eps))
zoom on
pause
```

Установка палитры псевдоцветов:

```
z = peaks(40)
colormap(hsv)
pcolor(z)
pause
```

Создание закрашенного многоугольника:

```
X = [1 2 3 2 1];
Y = [1 2 0 5 1];
patch(X, Y, [1 0 0])
pause
```

Создание плоских многоугольников:

```
X = [1 2 3 2 1];
Y = [5 0.5 0 4 5];
fill(X, Y, [0 0 1])
pause
```

Вывод шкалы цветов:

```
fill3(rand(5, 4), rand(5, 4), rand(5, 4), rand(5, 4))
colorbar('vert')
pause
```

Цветные плоские круговые диаграммы:

```
X = [1 2 3 4 5];
pie(X, [0 0 0 0 2])
pause
```

Окрашенные многоугольники в пространстве:

```
fill3(rand(5, 4), rand(5, 4), rand(5, 4), rand(5, 4))
pause
```

Цветные объемные круговые диаграммы:

```
X = [1 2 3 4 5];
pie3(X, [0 0 1 0 1])
pause
```

Построение цилиндра:

```
[X, Y, Z] = cylinder(10, 30)
surf(X, Y, Z, X)
pause
```

Построение сферы:

```
[X, Y, Z] = sphere(30);
surfl(X, Y, Z)
pause
```

Трехмерная графика с треугольными плоскостями:

```
x = rand(1, 40);
y = rand(1, 40);
z = sin(x.*y);
tri = delaunay(x, y);
trimesh(tri, x, y, z)
pause
```

1.1.13 Анимационная и дескрипторная графика

Файл-программа, демонстрирующая задание свойств в аргументах графических функций:

```
t = [0:0.1:7];
x = sin(t);
y = cos(t);
axes('XGrid','on')
hold on
plot(t,x,'Color','k','LineWidth',3)
plot(t,y,'Color','k','Marker','o','MarkerFaceColor','w',...
'MarkerFaceColor','w','MarkerEdgeColor','k');
```

Файл-программа, располагающая графическое окно на весь экран:

```
% Нахождение размеров экрана
SCRsize = get(0, 'ScreenSize')
% Область окна начинается от левого и нижнего края экрана,
% рамка не нужна
left = SCRsize(1);
bottom = SCRsize(2);
% Ширина области окна равна ширине экрана
width = SCRsize(3);
% Высота окна вычисляется с учетом ширины заголовка окна
height = SCRsize(4)-19;
% Создание окна без меню, рабочая область и заголовок растянуты
% на весь экран, границы не отображаются
figure('Position', [left bottom width height], 'Menu', 'none')
```

Файл-программа, демонстрирующая вывод текста в графическое окно:

```
% Формирование массива ячеек строк
strmas{1} = ['Fmin = ', num2str(fmin)];
strmas{2} = ['Xmin = ', num2str(minvalue)];
strmas{3} = ['Xzero = ', num2str(zero)];
% Создание вспомогательных невидимых осей,
% таких же размеров, как графическое окно
HHelpAx = axes;
set(HHelpAx, 'Position', [0 0 1 1], 'Visible', 'off')
% Вывод текста во вспомогательные оси
HInfo = text(0.6, 0.3, strmas);
% Установка свойств шрифта
set(HInfo, 'FontSize', 12, 'FontWeight', 'bold')
```

Файл-функция для создания двух графических окон:

```
function HFigs = fig2
% Создание двух графических окон равной высоты, делящих экран
% на две части по вертикали. Возвращает вектор указателей на окна
% Использование HFigs = fig2

% Нахождение размеров экрана
SCRsize = get(0, 'ScreenSize');
% Выделение ширины и высоты экрана
SCRwidth = SCRsize(3);
SCRheight = SCRsize(4);
```

```
% Ширина, высота и отступ слева одинаков для обоих окон
width = SCRwidth-5-3;
height = (SCRheight-19-5-3-19-5-3)/2;
left = 5;
% Отступ снизу для нижнего графического окна равен 5 пикселям
bottom2 = 5;
% Вычисление отступа снизу для верхнего окна с учетом толщины
% рамок и заголовка нижнего окна и толщины верхней рамки нижнего
% окна
bottom1 = 5+height+19+5+3;
% Создание окон без меню, рамки отображаются
HFigs(1) = figure('Position', [left bottom1 width height],...
  'Menu', 'none', 'Color', 'w');
HFigs(2) = figure('Position', [left bottom2 width height],...
  'Menu', 'none', 'Color', 'w');
```

Файл-функция, предназначенная для исследования функций:

```
function zeroandmin(funname, a, b);
% Исследование поведения функций вблизи корня и
% локального минимума на отрезке [a,b]
% Использование zeroandmin(funname, a, b);

% Формирование параметров функций fzero и fminbnd
% Подавление вывода в командное окно информации о ходе вычислений
options = optimset('Display', 'off');
% Поиск нуля функции на [a,b]
zero = fzero(funname, [a b], options);
% Поиск локального минимума функции на [a,b]
minvalue = fminbnd(funname, a, b, options);
% Создание окна
HF = figure('Menu', 'none', 'Color', 'w');
% Использование axes3 для построения трех осей
HAx = axes3(HF);
% Задание delta для вывода графиков вблизи корня и
% локального минимума
delta = (b-a)/30;
axes(HAx(1)) % оси с указателем HAx(1) стали текущими
% Вывод графика функции вблизи ее минимума на верхние левые оси
fplot(funname, [minvalue-delta  minvalue+delta])
axes(HAx(2)) % оси с указателем HAx(2) стали текущими
% Вывод графика функции вблизи ее нуля на верхние правые оси
fplot(funname, [zero-delta zero+delta])
axes(HAx(3)) % оси с указателем HAx(3) стали текущими
```

```
% Вывод графика функции на [a,b] на нижние оси
fplot(funname, [a b])
```

1.2 Основы программирования в системе MATLAB

1.2.1 Алфавит и лексемы

Язык программирования сверхвысокого уровня MATLAB позволяет создавать полномасштабные модульные программы и хранить их во внешней памяти в текстовом формате. Для записи операторов программы используются те же лексические средства, что и для записи команд входного языка интегрированной среды системы. Дополнительные средства языка позволяют оформлять отдельные модули и объединять их в проблемную программу или пакет.

Основной текст любого модуля состоит из букв латинского алфавита, десятичных цифр и ограниченного набора специальных знаков или их несложных комбинаций. Русский язык можно использовать лишь для записи комментариев и символьных констант.

Логическими единицами текста модуля являются лексемы. Их можно разделить на следующие виды:

1)целые числа: 0,1,2,3,4,5,6,7,8,9,10,11,125,999;

2)дробные числа: 1/2, 1/3, 1/4, 5/7, 1/125, 1/300;

3)вещественные числа: 12.34, 5e-5, 6.1e+10, 3.1e2;

4)символьные строки: 'A', '1', 'ABC37', 'Весна!', '?';

5)комментарии – строки из любых знаков, начинающиеся с символа % и заканчивающиеся в конце строки;

6) поименованные системные константы:

i или j – мнимая единица,

pi – число 3.1415926,

eps – погрешность вычисления процессора (2^{-52}),

realmin – наименьшее число (2^{-1022}),

realmax – наибольшее число (2^{1023}),

inf – машинная бесконечность,

NaN(Not a Number) – нечисловые данные

7)системные переменные:

ans – имя результата последнего использованного оператора;

8)идентификаторы, начинающиеся с латинской буквы и содержащие произвольное количество латинских букв, десятичных цифр и знака подчёркивания; они используются

для обозначения операций: kron;

в качестве имён функций-операций: Plus, Times;

в качестве имён данных (переменных): x, y, WL;

в качестве имён встроенных функций: plot, sin, abs;

в качестве имён файлов данных: fdat, fext;

в качестве имён файлов модулей: Mysin, Kiw, Abc;

в качестве имён классов: network, NewClass;

в качестве имён функций пользователя: Mysin, Kiw;

в качестве имён для указателей функций: hdKiw;

9)ключевые слова для обозначения операторов и их составных частей: for, if, end, function, global;

10)специальные знаки и их комбинации для обозначения математических операций и разделения или продолжения текста: +, .*, /, = =, <=, …, [,], (,);

11)имена объектов встроенных и пользовательских классах: NetObj, MyLin, Obj.

1.2.2 Типы данных и определение переменных

1.2.2.1 Программные и проблемные типы данных

Тип данных в языках программирования определяет множество значений, обладающих некоторыми фиксированными свойствами, и набор допустимых операций, функций и процедур по их созданию, преобразованию и использованию. В современной интерпретации тип данных – это класс(class). Ориентация системы MATLAB на математическое моделирование с представлением данных в универсальной матричной форме определила систему построения типов в ее языке программирования и способ определения переменных. На уровне языкового представления базовым типом является массив(array), а все остальные типы являются производными от этого типа, т.е. массивами каких либо объектов: чисел, символов, структур, ячеек и т.п. Даже число, например, рассматривается как одноэлементный вектор или матрица.

На самом деле, язык скрывает от программиста детали представления данных и дает ему возможность работать на проблемном уровне, оперируя числами, символами, векторами, матрицами, разреженными матрицами и другими математическими объектами. Это достигается тем, что типы переменных заранее не декларируются отдельно от данных, переменные вводятся в программу динамически, по мере необходимости, а их тип устанавливается автоматически по используемым данным. Таким образом, переменные никогда не определяются отдельно от данных, хотя данные могут присутствовать в программе отдельно, например, 1+1, тогда их приписывают системной переменной ans.

Проблемный подход в языке подкрепляется и чрезвычайно простым способом представления всех числовых данных во внутреннем формате с плавающей точкой двойной точности(double). Это облегчает понимание семантики используемых операций, функций и процедур, что повышает надежность создаваемых программ. Другие форматы представления

числовых данных используются лишь для их более компактного хранения в оперативной памяти или для удобного отображения на экране. Для изменения способа представления числовых данных в оперативной памяти (рабочей области) используются следующие функции: int8, uint8, int16, uint16, int32, uint32, single, double. Однако, перед выполнением арифметических операций и вызовом функций с вещественными параметрами их необходимо преобразовать к формату double. Форма представления числовых данных на экране может быть задана оператором format или командой меню File/Preference/Command Window/Numeric format, а так же спецификацией формата в операторе вывода строки на экран sprintf. Для этих целей имеются следующие форматы: rational (дробный), short, long, short e, long e**,** short g, long g, hex, bank u +. При вводе-выводе данных могут потребоваться функции преобразования строк и чисел такие как: num2str, int2str, str2double, str2num, dec2bin, dec2hex, dec2base, bin2dec, hex2dec, base2dec, hex2num, mat2str, str2mat. Наиболее гибким способом размещения на экране чисел и строк является применение шаблона преобразования, задаваемого спецификаторами : %c, %d, %e, %E, %f, %g, %G, %o, %s, %u, %x и %X. Кратко рассмотрим все типы проблемного уровня, которые представляет пользователю система программирования MATLAB.

1.2.2.2 Числовые типы

Скалярные числовые типы включают все известные в математике классы чисел: целые числа в форматах int8, uint8, int16, uint16, int32, uint32, single**,** double; дробные числа в формате single, double; вещественные и комплексные числа в форматах single, double. Из этих чисел в любой комбинации можно формировать числовые векторы, матрицы и многомерные массивы, в том числе и разреженные матрицы для повышения эффективности вычислений. Для всех числовых типов определены арифметические и алгебраические операции, а так же операции поэлементного сравнения, которые приведены в таблице 1.2. Для реализации функционального программирования каждая операция имеет в языке свой двойник в виде функции. Например, операцию сложения можно записать как X+Y или как Plus(X,Y) . Во всех операциях числа должны быть представлены в формате double. В таком же формате необходимо задавать и числовые аргументы при вызове библиотечных функций, встроенных в систему (стандартных функций). Ядро системы MATLAB включает полный набор арифметических, алгебраических, тригонометрических, обратных тригонометрических, гиперболических, обратных гиперболических функций, функций округления и знака, а так же функций комплексного аргумента. Справочная информация по операциям и функциям для числовых типов находится в разделах: ops, elmat**,** elfun, specfunc, matfunc, soarfun. При программировании в системе MATLAB следует помнить, что скалярный тип – это массив размером 1x1.

Следует отметить, что операции **<, <=, >, >=** при комплексных операндах используют для сравнения только действительные части операндов – мнимые отбрасываются. В то же время операции == и ~= ведут сравнение с учетом как действительной, так и мнимой части операндов.

1.2.2.3 Логические типы

Логические значения в языке обычно представляются числами 0 (ложь) и 1 (истина), хотя возможен и другой способ: ноль – ложь, любое ненулевое число – истина. Вектор, матрица или многомерный массив, сформированные непосредственно из таких чисел не рассматриваются системой как логические типы и в рабочей области они отображаются как double array. Поэтому для скалярных типов, векторов, матриц и многомерных массивов такого вида разрешается применять все арифметические и алгебраические операции, указанные в табл. 1.2. Применение к ним операций сравнения формирует логические типы соответствующей структуры. Указанные в той же таблице логические операции, а также логические функции Xor (исключающее или), Any (проверка наличия хотя бы одного ненулевого элемента в массиве) и All (проверка наличия в массиве только ненулевых элементов) – определены для логических данных и возвращают логические результаты.

В библиотеке системы MATLAB имеется большое число функций, которые также возвращают логические массивы (logical array). Например, функция isa определяет принадлежность объекта классу. Она имеет формат:

isa(obj, ‘class name’),

где obj – имя анализируемого объекта;

class name – имя предполагаемого класса.

В качестве имени класса можно задавать: char, numeric, logical, int8, uint8, int16, uint16, int32, uint32, int64, uint64, single, double, cell, struct, function_handle и имена классов пользователя.

Следующие функции также проверяют класс данных и возвращают логический массив: iscell, iscellstr, ischar, isempty, isequal (проверка идентичности структуры массивов), isfield, isfinite, isglobal, ishandle, ishold, isinf, isletter, islogical, isnan, isnumeric, isobject, isprime, isreal, isspace, isparse (разреженная матрица) и isstruct. При необходимости преобразование числового массива в логический можно выполнить с помощью функции logical.

Заметим, что функции strcmp, strcmpi и strncmp, с помощью которых сравниваются символьные данные, возвращают логические массивы, которые также могут использоваться в качестве операндов логических операций или аргументов логических функций.

1.2.2.4 Символьные типы

Символьное представление данных в системе MATLAB лежит в основе символьной математики, обеспечивающей нахождение аналитических решений некоторых математических задач (решение систем линейных и нелинейных уравнений, разыскание пределов и производных, нахождение определенных и неопределенных интегралов, поиск аналитических решений дифференциальных уравнений и систем), арифметики произвольной точности и многочисленных программных конструкций (векторов, массивов любой размерности, ячеек, структур, классов, java-объектов), а также в базах данных и других пакетах расширения. Чаще всего при этом используются строки символов, представляемые вектор-строками размерностями 1xM, массивы строк равной длины (матрицы размерности NxM) и массивы ячеек, каждая из которых представлена строками разной длины.

Таблица 1.2

Операции для числовых и логических типов

Знак	Выполняемые действия
-	Покомпонентное вычитание массивов одинаковой структуры или вычитание числа из каждого элемента массива
+	Покомпонентное сложение массивов одинаковой структуры или добавление числа к каждому элементу массива
*	Алгебраическое умножение матриц MxK и KxN или умножение на число каждого элемента массива
.*	Покомпонентное умножение массивов одинаковой структуры
/	Деление на число каждого элемента массива или матричное деление квадратных матриц одного порядка
./	Покомпонентное деление массивов одинаковой структуры
\	Левое деление квадратных матриц одного порядка
.\	Левое покомпонентное деление массивов одинаковой структуры
^	Возведение числа в любую степень или вычисление целой степени квадратной матрицы NxN
.^	Покомпонентное возведение массива в степень
'	Вычисление сопряженной матрицы и транспонирование
.'	Транспонирование матрицы. Для вещественных матриц операции ' и . ' приводят к одному результату – транспонированию исходной матрицы
kron	Тензорное произведение матриц
==	Сравнение числа на равенство с каждым элементом массива или покомпонентное сравнение на равенство массивов одинаковой структуры
~=	Проверка на неравенство (см. операцию ==)
>	Проверка на больше (см. операцию ==)
>=	Проверка на больше или равно (см. операцию ==)
<	Проверка на меньше (см. операцию ==)
<=	Проверка на меньше или равно (см. операцию ==)
~	Логическое отрицание скаляра или элементов массива
&	Логическое умножение (см. операцию ==)
|	Логическое сложение (см. операцию ==)

Каждый символ строки имеет внутреннее представление в виде числового кода из диапазона от 0 до 255, определяемого заданной при установке таблицей кодов. Первые 127 чисел – это коды американского стандарта ASCII, включающего латинские буквы, десятичные цифры, специальные знаки и управляющие символы. Они образуют основную таблицу кодов. Вторая таблица (коды от 128 до 255) является дополнительной и используется для кодирования национальных алфавитов и других знаков.

Создание символьных переменных того или иного вида (строки, матрицы, ячейки) осуществляется операторами присваивания, в правой части которых записываются символьные строки или функциональные выражения, возвращающие символьные строки. Для символьных данных в языке не предусмотрено никаких операций – вся их обработка реализуется библиотечными функциями: deblank, findstr, lower, strcat, strmatch, strrep, strtok, strcat, upper, char, int2str, mat2str, num2str, sprintf, sscanf, str2double, str2num, bin2dec, dec2bin, dec2hex, hex2dec, hex2num. Функции strcmp, strcmpi, и strncmp осуществляют сравнение строк и подстрок, возвращая логические значения 0 или 1 для каждой пары сравниваемых строк, при этом функция strcmpi не различает прописные и строчные буквы.

1.2.2.5. Битовые типы

Битовые типы обеспечивают более удобные и эффективные способы обработки логических массивов. Значения битового типа представляют собой последовательности двоичных нулей и единиц, которые хранятся в одном или нескольких байтах оперативной памяти в зависимости от длины этих значений. Если рассматривать двоичный ноль как ложь, а двоичную единицу как истина, то для хранения такого логического значения потребуется лишь один бит памяти, в то время как для хранения обычного логического значения – один байт. При представлении логических значений числами (ноль и не ноль) может потребоваться и еще больше памяти. К тому же в процессоре имеются очень быстрые операции над битовыми значениями, так что использование битовых типов в языке может обеспечить также повышение скорости обработки логических данных.

Выделение памяти в рабочей области и ее инициализация производятся с помощью определения переменных целого или иного типа, при этом количество резервируемых бит не должно превышать возможности компьютера: число с плавающей точкой, записанное в этих битах, не должно превышать максимально возможного, которое определяется функцией bitmax. Для большинства компьютеров оно равно 2^{53}-1, или 9.0072e+015.

В библиотеке MATLAB имеются следующие функции для работы с битовыми типами:

bitset (A, bit, [v]) – устанавливает бит в позиции bit равным значению v, которое может быть 0 или 1; если v не задано, то устанавливается 1;

bitget (A, bit) – возвращает значение бита в позиции bit операнда A;

bitcmp (A, n) – возвращает поразрядное дополнение аргумента A как n – битного неотрицательного целого числа;

bitand (A, B) – возвращает поразрядное И двух неотрицательных целых аргументов A и B;

bitor (A, B) – возвращает поразрядное ИЛИ двух неотрицательных целых аргумента A и B;

bitxor (A, B) – возвращает поразрядное исключающее ИЛИ двух неотрицательных целых аргумента A и B;

bitshift (A, n) – возвращает значение аргумента A, сдвинутое на n битов, т.е. умноженное на 2^n.

При работе с битовыми типами необходимо использовать функции перевода чисел из одной системы счисления в другую такие, как dec2bin, bin2dec, dec2hex, hex2dec, dec2base, base2dec, num2str, int2str, str2num, str2double.

1.2.2.6. Множества

Векторы и матрицы с одинаковым числом столбцов могут рассматриваться как множества и обрабатываться специальными функциями системы MATLAB, при этом элементами множества являются либо компоненты вектора, либо строки матрицы. Для матрицы в соответствующих функциях указывается параметр ‘rows’. В библиотеке имеются следующий набор функций:

unique(a), unique(a,’rows’),[b,I,j]=unique(a),[b,i,j]=unique(a,’rows’) – для нахождения неповторяющихся элементов и их индексов;

intersect (a,b), intersect (a,b,’rows’), [c,ia,ib]= intersect (a,b) – для определения пересечения двух множеств и индексов общих элементов;

setdiff (a,b), setdiff (a,b,’rows’),[c,i]= setdiff (a,b),[c,i]= setdiff (a,b,’rows’) – для вычисления разности множеств, т.е. таких элементов из a, которые не содержатся в b;

union(a,b), union(a,b, ‘rows’), [c,ia,ib]= union(a,b), [c,ia,ib]= (a,b, ‘rows’) – для объединения элементов из a и b без их повторения и вычисления индексов таких элементов;

setxor(a,b), setxor(a,b, ‘rows’), [c,ia,ib]= setxor(a,b), [c,ia,ib]=setxor (a,b, ‘rows’) – для реализации исключающего ИЛИ множеств a и b, а так же определения индексов элементов результирующего множества;

ismember (a,S), ismember (a,S,’rows’) – для определения принадлежности элементов множества a множеству S в виде вектора логических значений.

Определение переменной типа множества и задание элементов множества производится соответственно присваиванием переменной вектора или матрицы требуемых элементов.

1.2.2.7. Календарь, время и дата

В системе MATLAB имеется несколько специальных типов для обслуживания текущего времени и его измерения. Функции, связанные с этими типами, обеспечивают выдачу календаря на тот или иной месяц (calendar), даты в требуемом формате (date, datenum, datevec, datestr, eomday) и текущего времени (clock,now).Для измерения времени можно использовать следующие функции:

cputime – для получения времени работы процессора (в секундах), использованное системой Matlab с момента её запуска;

etime (t2,t1) – для определения длительности промежутка времени (в секундах) между t1 и t2, заданными в формате даты и времени;

tic – для запуска таймера при профилировании программы;

toc – для определения времени после запуска таймера.

Функция [N,S]=weekday (D) возвращает день недели в виде числа N и в виде строки S для каждой даты массива D.

1.2.2.8 Разреженные матрицы

Числовые матрицы без нулевых элементов называются полными. Матрицы, содержащие некоторое число элементов с нулевыми значениями, в системе MATLAB называют разреженными матрицами. К таким матрицам приводит решение многих задач математической физики, теории прочности, пластичности, электротехники и электроники. Кроме нескольких диагоналей или небольших блоков, заполненных коэффициентами, в этих матрицах находится огромное число нулей. Хранить их в полном объеме невыгодно, поэтому в ряде случаев хранят лишь значения ненулевых элементов и их индексы. На самом деле, хранятся не индексы отдельных элементов, а диапазоны индексов, соответствующих подряд идущим ненулевым элементам. В библиотеке системы имеется довольно много функций по обработке разреженных матриц, которые используют факт наличия нулевых элементов для исключения пустых операций (умножение на нуль, вычитание нуля, и т.п.) и повышения эффективности вычислений. Функции преобразования осуществляют переход от полных матриц к разреженным и от разреженных к полным.

Определение переменных для разреженных матриц осуществляется так же, как и для полных, но при этом используется функция приведения sparse, например, b=sparse(eye(100)), где eye(100) генерирует единичную матрицу размером 100x100. Ряд функций обеспечивает непосредственное создание разреженных матриц простой структуры (элементарных), позволяет осуществлять разложение матриц по методу Холецкого или LV-разложение, производить упорядочение элементов, вычислять нормы, числа обусловленности, ранги, собственные значения и сингулярные числа, а также визуализацию разреженных матриц.

Естественно, что все операции над матрицами, описанные в таблице 1.2, применимы и для разреженных матриц, при этом они выполняются быстрее, чем для соответствующих полных матриц.

1.2.2.9 Многомерные массивы

Векторы являются одномерными массивами, задаваемыми либо в виде строки (вектор-строки), либо в виде столбца (вектор-столбец) [1 2 3] или [1; 2; 3]. Матрицы являются двумерными массивами: число измерений у них равно 2, а их размер, т.е. число элементов, определяется как произведение NxM, где N – число строк, а M – число столбцов. Если число измерений массива больше 3, то он называется многомерным. Размер такого массива определяется произведением количества компонент по каждому измерению: строк, столбцов, страниц, блокнотов и т.д.

Элементами многомерных массивов могут быть однородные объекты скалярных типов, а также строки символов или последовательности бит одинаковой длины. Для выполнения операций над этими объектами используется поэлементный принцип (см. табл.1.2). К массивам применены многие функции, определенные для векторов и матриц. Доступ к отдельным элементам массива и его компонентам осуществляется с помощью индексов и знаков двоеточия, которые задаются в круглых скобках после имени массива и отделяются друг от друга запятой.

Определение многомерных массивов производится с помощью операторов присваивания, в которых за именем определяемого массива записываются индексы, а за оператором присваивания – отдельные значения, строки, векторы и матрицы, а также другие многомерные массивы – компоненты определяемого. Многомерный массив можно определить также путем присваивания другого массива.

Для удаления любой размерности массива используется оператор присваивания, в правой части которого указывается пустой массив []. Если в правой части задана константа того или иного типа, а в левой части индексы определяют некоторую компоненту массива, то всем элементам этой компоненты будет присвоено значение константы.

Определение и инициализацию многомерных массивов можно произвести также с помощью встроенных функций: ones - единицами, zeros – нулями, rand – случайными числами с нормальным распределением. Параметрами этих функций являются размеры по соответствующим измерениям: ones(3,3,2) – массив с тремя строками, тремя столбцами и двумя страницами.

Функция конкатенации cat позволяет объединять векторы, матрицы и многомерные массивы в более сложные многомерные массивы:

cat (1,A,B) - вертикальная конкатенация A и B;

cat (2,A,B) - горизонтальная конкатенация A и B;

cat (DIM,A,B) - объединение массивов вдоль размерности DIM;

cat (DIM,A,B, C…..) - объединение нескольких массивов вдоль размерности DIM;

Размерность массива определяется функцией ndims (A), а размеры по измерениям DIM – с помощью функции size (A, DIM), так что size (A,1)- число строк и size (A,2) - число столбцов массива A. Функция [m,n,k,…..]= size (A) возвращает размер первых DIM размерностей A.

Структуру массива можно изменить, переставляя его размерности в заданном порядке ORDER с помощью функции permute (A, ORDER) или ipermute(A, ORDER).

Здесь ORDER- перестановка чисел от 1 до N, где N - размерность массива.

Порядок следования размерностей можно изменить с помощью функций shiftdim (X,N), где N - величина сдвига. В итоге строки становятся столбцами, страницы - строками, а столбцы - страницами. Если в массиве при преобразованиях оказались единичные размерности, то их можно удалить функцией squeere.

1.2.2.10 Структуры

Структуры относятся к сложным типам данных. Они состоят из поименованных компонент – полей любого типа. Чаще всего – это скалярные типы, однако компонентой может быть многомерный массив, ячейка и даже структура.

Определить структуру – это значит задать её имя, задать имена полей и их начальные значения. Делается это двумя способами: с помощью операторов присваивания или с помощью функции struct. Изменять значения полей также можно двумя способами: оператором присваивания и функцией setfield. Доступ к полю обеспечивается составным именем, включающим имя структуры и имя поля, которое отделяется от имени структуры точкой. Функция getfield также позволяет получить значение любого поля структуры.

Строение структуры можно динамически менять, добавляя в неё новые поля описанными выше способами или удаляя ненужные поля функций – методом rmfield. Функция fieldnames позволяет получить имена полей неизвестной структуры, а функции isfield и isstruct производят проверку строения структуры.

Структура может быть элементом многомерного массива ячейки. В этих случаях используется индексация. Структура является также основой такого мощного типа, как класс.

1.2.2.11 Ячейки

Ячейки, как и структуры, состоят из многих компонент разного типа, которые упорядочены в виде многомерного массива. Доступ к компонентам осуществляется с помощью индексов. Отличительным атрибутом ячеек на уровне языка является задание их содержимого в фигурных скобках {}. Используя фигурные скобки, ячейки можно создавать двумя способами:

а) с помощью индекса ячеек: A(1,2) = {[1 2; 3 4]};

б) с помощью индексации содержимого: A{1,2}=[1 2; 3 4].

Доступ к компонентам ячейки осуществляется с помощью индексов, которые можно записывать как в круглых ( ), так и в фигурных скобках, так что A(1,2) и A {1,2} будут обозначать одну и ту же компоненту ячейки A.

Специальная функция cell позволяет создавать ячейки из пустых матриц: cell(N) – из NxN матриц, cell(M,N) - из MxN матриц и т.д. Для отображения компонент ячейки используется функция celldisp, для визуализации – набор функций cellplot с различными аргументами.

Проверка типа ячеек производится функциями iscellstr и iscell.

Для создания из массива S, содержащего символьные строки, ячейки с теми же строками можно использовать функции cellstr. Для подобного преобразования массива чисел имеется функция num2cell. Преобразование структур и ячеек выполняются соответственно функциями struct2cell и cell2struct, а матриц и ячеек - функциями mat2cell и cell2mat.

1.2.2.12 Классы

Классы и объекты позволяют добавлять новые типы данных и новые операции. Класс описывает тип переменной и определяет, какие операции и функции могут быть применены к этому типу переменной. Объект – это структура или образец некоторого класса.

Добавление классов осуществляется в рамках операционной среды системы MATLAB, которая обеспечивает возможность хранения созданных объектов и организации каталога m-файлов, определяющих допустимые методы обработки для данного класса объектов. Каталог класса имеет имя @<имя класса> и включает m-функции, определяющие способы, с помощью которых операции системы MATLAB, включая арифметические, обработки индексов и конкатенации, обрабатывают объекты данного класса. Переопределение встроенных операций для нового класса объектов в рамках объектно-ориентированного подхода называется переопределением методов.

Каталог класса должен обязательно содержать m-файл, называемый конструктором класса. Название конструктора должно совпадать с названиями класса и каталога без префикса @. Объекты классов создаются динамически посредством вызова конструктора класса. Конструктор создает

объекты, используя данные в виде массива записей (структуры) и приписывая им метку класса.

Функции isa и class используются конструктором, но могут применяться и вне каталога класса.

Функция isa(a, 'class_name') проверяет, принадлежит ли объект a данному классу.

Функция class(a) при использовании вне контекста методов допускает только один аргумент, и возвращает строку, содержащую имя класса для объекта a.

Вызов функции преобразования класса имеет вид

b = class_name(a),

где a – объект некоторого класса, отличного от class_name.

В этом случае система MATLAB ищет метод с именем class_name в каталоге классов для объекта a. Такой метод преобразовывает объект одного класса в объект другого класса. Если данный объект уже является объектом класса class_name, то система MATLAB вызывает функцию конструктора, который просто возвращает этот объект.

Наиболее важными функциями преобразования классов являются double и char. Преобразование к классу double создает традиционный массив системы MATLAB, хотя это может и не отражать требуемого соответствия для некоторых классов. Преобразование к классу char полезно для вывода на печать.

Каталоги классов могут иметь связанные с ними частные каталоги. Такие каталоги могут содержать как частные методы, которые работают с объектами данного класса, так и частные функции, которые не работают с объектами, но выполняют общие вычисления. Можно устанавливать частный каталог в каталоге класса точно так же, как создается любой частный каталог, т. е. просто создать каталог, именуемый private, внутри каталога @class_name.

Во многих случаях можно изменить поведение операторов и функций системы MATLAB, когда в качестве аргумента выступает объект. Это осуществляется путем переопределения соответствующих функций. Переопределение класса открывает возможность обработки с помощью этой функции различных типов данных при произвольном количестве входных аргументов.

Каждая встроенная операция в системе MATLAB имеет имя. Поэтому любая операция может быть переопределена путем создания m-файла с соответствующим названием в каталоге классов.

Можно переопределить любую m-функцию, создавая функцию с тем же именем в каталоге класса. Когда функция применяется к объекту, MATLAB прежде всего просматривает каталог соответствующего класса, а уже потом другие пути доступа.

1.2.3 Имена и выражения

Имена – это названия скалярных переменных различного типа, векторов, матриц, многомерных массивов, структур, ячеек и объектов классов. Для указания элементов и компонент векторов, матриц, многомерных массивов и ячеек используются индексы. Поля структуры и атрибуты объектов обозначаются составными именами.

Имена используются для записи выражений. Операции над именами задаются либо с помощью знаков (см. табл. 1.2), либо с помощью имен функций-операций. В выражении могут использоваться любые функции, как встроенные, так и написанные программистом.

В математических выражениях языка системы MATLAB операции имеют определенный приоритет исполнения: приоритет логических операторов выше, чем арифметических, приоритет возведения в степень выше приоритетов умножения и деления, приоритет умножения и деления выше приоритета сложения и вычитания. Для изменения приоритета операций в математических выражениях используются круглые скобки. Степень вложения скобок не ограничивается.

1.2.4 Операторы

На уровне программирования команды и функции системы MATLAB называются операторами, а их последовательности – скриптами.

Помимо скриптов с линейной структурой, инструкции которых исполняются строго по порядку, MATLAB позволяет создавать программы, структура которых имеет ветвление, циклы, ввод данных и приостановки.

Для организации диалогового вывода используются функции input и disp.

Функция input имеет следующий синтаксис:

```
переменная = input(строка)
```

При выполнении этого оператора вначале выводится строка, затем происходит остановка работы программы и ожидается ввод значения. Ввод подтверждается нажатием клавиши ENTER, после чего введенное значение присваивается переменной.

Функция disp предназначена для вывода ее параметра на экран:

```
disp (Выводимое значение)
```

Условный оператор if в общем виде записывается следующим образом:

```
if Условие
   Список_инструкций_If
elseif Условие
   Список_инструкций_Elsif
else
   Список_инструкций_Else
end
```

Циклы типа for...end обычно используются для организации вычислений с заданным числом повторяющихся циклов. Конструкция такого цикла имеет следующий вид:

```
for var=Выражение
    Список_инструкций
end
```

Выражение чаще всего записывается в виде s:d:e, где s – начальное значение переменной цикла var, d – приращение этой переменной и e – конечное значение управляющей переменной, при достижении которого цикл завершается. По умолчанию d равно 1.

Цикл типа while...end выполняется до тех пор, пока остается истинным условие:

```
while Условие
   Список_инструкций
end
```

Досрочное завершение циклов реализуется с помощью операторов break или continue.

Для осуществления множественного выбора (или ветвления) используется конструкция с переключателем типа switch:

```
switch switch_Выражение
case case_Выражение
     Список_инструкций
case {case_Выражение1,case_Выражение2,case_Выражение3,...}
     Список_инструкций
...
otherwise,
      Список_инструкций
```

```
end
```

Case_выражение может быть числом, константой, переменной, вектором ячеек или даже строчной переменной. В последнем случае оператор case истинен, если функция strcmp(значение, выражение) возвращает логическое значение “истина”.

Конструкция блока вывода ошибок try...catch...end имеет следующий синтаксис:

```
try
  Список_инструкций
catch
  Список_инструкций
end
```

Эта конструкция выполняет все списки инструкций. Если в каком-то списке до оператора catch появляется ошибка, то выводится сообщение об ошибке, но системная переменная последней ошибки lasterr не меняется. В выражениях после catch сообщение об ошибке не выводится.

Во всех управляющих структурах список инструкций, или тело, представляет собой последовательность выражений, команд или вложенных управляющих структур, разделяемых пробелом, запятой или точкой с запятой. Точка с запятой запрещает вывод данных на экран.

Для остановки выполнения инструкций используется оператор pause. Он используется в следующих формах:

pause – останавливает вычисления до нажатия любой клавиши;

pause(N) – останавливает вычисления на N секунд;

pause on – включает режим обработки пауз;

pause off – выключает режим обработки пауз.

1.2.5 Программные модули

Программные модули системы MATLAB называются m-файлами и делятся на два класса:

файлы-сценарии, не имеющие входных параметров;

файлы-функции, имеющие входные параметры.

Файл-сценарий или скрипт не имеет списка входных параметров. Он использует глобальные переменные, т. е. такие переменные, которые находятся в рабочей области и значения которых могут быть изменены в любой момент сеанса работы и в любом месте программы (если переменная объявлена как global). Для запуска файла-сценария из командной строки

MATLAB достаточно указать его имя в этой строке. Для запуска файла-функции надо указать список возвращаемых значений, имя функции и список аргументов, при этом после списка возвращаемых значений надо указать знак “=”.

Файл-сценарий имеет следующую структуру:

```
%Основной комментарий из одной строки
%Дополнительный комментарий из любого числа строк
Тело файла с любым списком инструкций
```

Основной комментарий выводится при выполнении команд lookfor и help имя_каталога. Полный комментарий выводится при выполнении команды help имя_файла, причём вывод производится до первой пустой строки.

Файл-функция отличается от файла-сценария прежде всего тем, что созданная им функция имеет входные параметры, список которых указывается в круглых скобках. Используемые в файлах-функциях переменные и имена параметров являются локальными переменными, изменение значений которых в теле функции не влияет на значения, которые те же самые переменные могут иметь за пределами функции.

Файл-функция имеет следующую структуру:

```
function var=f_name (список_входных параметров)
%Основной комментарий из одной строки
%Дополнительный комментарий из любого числа строк
Тело файла с любым списком инструкций
var=выражение
```

Последняя инструкция “var=выражение” вводится, если требуется, чтобы функция возвращала результат вычислений. Если необходимо большее количество выходных параметров, структура модуля будет иметь следующий вид:

```
function[var1,var2,...]=f_name(список_параметров)
%Основной комментарий  из одной строки
%Дополнительный комментарий из любого числа строк
Тело файла с любым списком инструкций
var1=выражение
var2=выражение
        ...
```

Имена var1, var2, ...должны быть глобальными.

1.2.6 Программа

Программа – это совокупность логически связанных программных модулей, находящихся во внешней памяти в виде m-файлов, написанных на языке программирования. Программные модули вызываются динамически. Один из модулей является начальным, хотя синтаксически он никак не выделяется. Для связи модулей используются переменные рабочей области, параметры, а также переменные, объявленные со словом global во всех модулях, где эти переменные используются.

1.3 Решение специальных задач

Файл-функция для решения модельной задачи:

```
function modelexam(gfile, bfile, err)
% Файл-функция находит решение модельной граничной
% задачи Дирихле для эллиптического дифференциального
% уравнения в прямоугольной области.
% Использование:
%    modelexam('файл с декомп. геом.', 'файл с гран. услов.', err)

% Инициализация сетки с максимальной стороной элемента 0.2
[p, e, t] = initmesh(gfile, 'Hmax', 0.2);
% Задание коэффициентов уравнения
a = 0;
c = 1;
% Определение строки с формулой правой части уравнения
f = '5*pi^2*sin(pi*x).*sin(2*pi*y)';

% Организация циклического измельчения сетки пока
% не достигнута требуемая точность
erhelp = 1;
while erhelp > err
  % Измельчение сетки
  [p, e, t] = refinemesh(gfile, p, e, t);
  % Решение уравнения
  u = assempde(bfile, p, e, t, c, a, f);
  % Вычисление точного решения в узлах сетки,
  % абсциссы и ординаты узлов храняться в строках матрицы p
  uex = exsol(p(1,:), p(2,:));
```

```
  % Вычисление нормы погрешности приближенного решения (u является
  % вектор-столбцом, а подфункция exsol вызывается от строк, поэтому
  % вектор с точным решением необходимо транспонировать)
  erhelp = norm(u-uex',Inf);
end
% Решение найдено с требуемой точностью
% Визуализация расчетной триангуляции и вывод контурного графика
% решения, залитого цветом
figure
subplot(2,1,1)
pdemesh(p,e,t)
subplot(2,1,2)
pdeplot(p, e, t, 'xydata', u, 'colormap', 'gray', 'colorbar', 'off')

function z = exsol(x,y)
% Подфункция для вычисления точного решения
z = sin(pi*x).*sin(2*pi*y);
```

Файл-программа для решения задачи о составлении рациона:

```
% Задание матрицы и вектора правой части системы неравенств
A = [4  6  15
    2  2  0
    5  3  4
    7  3  12];
A = -A;
b = [250; 60; 100; 220];
b = -b;
% Определение коэффициентов целевой функции
f = [44;    35; 100];
% Задание ограничений снизу на переменные
lb =[0; 0; 0];
% Решение и вывод результата в командное окно
x = linprog(f, A, b, [], [], lb, [])
```

Файл-функция, вычисляющая минимизируемую функцию:

```
function f = myfun(x)
f = 3*x(1)^2+2*x(2)^2;
```

Файл-функция с ограничениями:

```
function [c, ceq] = mycon(x)
```

```
% Задание ограничений
c(1) = x(1)^2+x(2)^2-1; % ограничения в виде неравенства
% Правая часть ограничений-равенств является пустым массивом,
% поскольку данных ограничений нет
ceq = [];
```

Файл-функция, вычисляющая левую часть системы уравнений:

```
function F = mysys(x)
F(1) = x(1)*(2-x(2))-cos(x(1))*exp(x(2));
F(2) = 2+x(1)-x(2)-cos(x(1))-exp(x(2));
```

Файл-функция, зависящая от вектора параметров и аргумента:

```
function y = fitfun(a, x)
y = a(1)*exp(x*a(2))+a(3)*sin(a(4)*x);
```

Файл-программа для решения задачи о подборе параметров:

```
% Ввод данных
xdat = (0:0.1:1);
ydat = [1.1 2.1 3.5 3.9 4.3 4.6 4.2 4.0 3.3 2.2 2.1];
% Отображение данных на графике
plot(xdat, ydat, 'o');
grid on
% Выбор начального приближения
a0 = [0.0 0.0 4.0 1.0];
% Построение графика функции от начального приближения
x = [0:0.05:1];
ya0 = fitfun(a0, x);
hold on;
plot(x, ya0, '--b')
% Задание границ области параметров
LB   = [-10 -10 -10 -10];
UB   = [10 10 10 10];
% Подбор параметров, точка с запятой в конце команды не ставится для
% вывода результата в командное окно
a = lsqcurvefit('fitfun',a0, xdat, ydat, LB, UB)
% Визуализация функции с найденными значениями параметров
ya = fitfun(a,x);
Hfit = plot(x, ya);
set(Hfit, 'LineWidth', 2)
legend('данные', 'начальное приближение', 'результат', 4)
```

Файл-функция, вычисляющая левую часть системы:

```
function F = largesys(x)
n = length(x);
F = rand(n,1);
F(1) = 2*x(1)^2-x(2)-1;
for i = 2:n-1
  F(i) = -x(i-1)+2*x(i)^2-x(i+1)-1;
end
F(n) = -x(n-1)+2*x(n)^2-1;
```

Файл-функция, вычисляющая левую часть системы и возвращающая якобиан:

```
function [F, J] = largesys(x)
n = length(x);
F = rand(n,1);
F(1) = 2*x(1)^2-x(2)-1;
for i = 2:n-1
  F(i) = -x(i-1)+2*x(i)^2-x(i+1)-1;
end
F(n) = -x(n-1)+2*x(n)^2-1;
% Если число выходных аргументов более единицы, то требуется найти
% разреженное представление якобиана
if nargout>1
  % якобиан является суммой трех матриц J = D+D1+D1'
  % Формирование диагонали матрицы D
  d = 4*x;
  % Инициализация разреженного представления для матрицы D
  Diag = sparse(1:n, 1:n, d, n, n);
  % Формирование вектора побочной диагонали
  d2 = -ones(1,n-1);
  % Инициализация разреженного представления для матрицы D1
  D1 = sparse(2:n, 1:n-1, d2, n, n);
  % Вычисление разреженного якобиана
  J = Diag+D1+D1';
end
```

Файл-программа для решения большой системы нелинейных уравнений:

```
n = 1000; % число переменных
x0 = ones(1,n); % начальное приближение
% Формирование структуры options
options = optimset('Display', 'iter', 'Jacobian', 'on',...
```

```
  'Diagnostics', 'on');
% Решение системы нелинейных уравнений и вывод результата
% в командное окно
x = fsolve('largesys', x0, options)
```

1.4 Порядок выполнения лабораторной работы

1. Запустить систему MATLAB.

2. Очистить интерфейсные окна Command Window, History Window и Workspace, используя команды меню Edit.

3. Создать дневник и включить запись в дневник содержимого окна Command Window, исполнив команды:

diary имя файла в виде – DAT_Число_Месяц_Год

diary on.

4. Выполнить команды из разделов 1.1.2 – 1.1.13.

5. На системном диске, где установлена математическая система MATLAB, найти папки инструментального пакета по нейронным сетям Neural Network Toolbox (NNT) и скопировать в рабочий каталог все методы класса network, обеспечивающие создание, инициализацию, обучение, моделирование и визуализацию нейронной сети. В рабочем каталоге найти m-файл конструкторов класса и проанализировать их структуру.

6. Запустить на пошаговое исполнение конструктор сети network, который не имеет параметров, и проследить последовательность действий по формированию объекта класса network и заданию значений по умолчанию для его атрибутов.

7. Создать объект класса нейронной сети network, исполнив команду

*net = network**,***

и вывести на экран все поля и все ячейки этого объекта, используя функцию **celldisp.**

8. Создать объект класса нейронной сети **network,** исполнив команду

net = network(2,3,[1;0;0], [1 1;0 0;0 0], ...

[0 0 0;1 0 0;0 1 0], [0 0 1], [0 0 1],

и вывести на экран все поля и все ячейки этого объекта, используя функцию celldisp.

9. Исполнив команду gensim(net)**,** получить на экране структурную схему созданной сети, раскрыть её блоки с помощью двойного щелчка мыши и выяснить смысл параметров конструктора network**.** Изменяя значения параметров, проследить их влияние на элементы структурной схемы.

10. Выключить запись в дневник, выполнив команду:

diary off.

11. Предъявить дневник преподавателю для оценки объема выполненной работы и анализа качества выполнения задания.

12. При выполнении задания следует пользоваться краткими пояснениями из раздела 1.2 и справочной системой MATLAB.

13. Используя конструктор графического интерфейса и язык программирования, разработать приложение для функциональной проверки возможностей системы MATLAB.

14. Разработать приложения для просмотра графических файлов.

15. Оценить производительность системы MATLAB при решении задач различных классов.

16. Используя конвертор m-файлов, получить P-код для заданной m-функции и измерить время выполнения m-файла и время выполнения P-кода. Сравнить результаты.

17. Оформить отчет в электронном формате.

1.5 Оформление отчета по лабораторной работе

На основании дневника сессии оформить m-файл для повторного выполнения задания, включив в него после каждого примера операторы остановки pause(10). В отчет включить также исходные модули приложения для функционального тестирования системы MATLAB и приложения для просмотра графических файлов.

Лабораторная работа № 2

ИНСТРУМЕНТАЛЬНЫЕ СРЕДСТВА ПАКЕТА SIMULINK ДЛЯ ВИЗУАЛЬНОГО ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ

Цель работы: изучение инструментальных средств для визуального проектирования имитационных моделей динамических систем и приобретение навыков конструирования, редактирования, отладки, исследования и документирования таких моделей.

2.1. Назначение и запуск пакета Simulink

Система MATLAB является мощным инструментом математического моделирования.

Математическое моделирование – это область науки и техники, которая обеспечивает выявление закономерностей протекания различных явлений окружающего нас мира или работы технических систем путем их математического описания и исследования полученных математических зависимостей без проведения, как правило, дорогостоящих натурных испытаний. Различают три вида математического моделирования: аналитическое, когда результаты представляются в виде аналитических выражений – формул; численное, когда с помощью численных методов производятся расчеты параметров исследуемых явлений и систем; имитационное, когда строится структура блоков, адэкватная структуре математического описания объектов предметной области с четко выделенными функциональными компонентами.

Ядро системы MATLAB и пакеты расширения поддерживают все три вида математического моделирования для самых различных областей применения. Для имитационного моделирования динамических систем, описываемых линейными и нелинейными дифференциальными уравнениями, используется пакет Simulink. Основные блоки этого пакета и их назначение приведены в табл. 2.1. Пакет обеспечивает визуальный режим проектирования модели и представляет большие возможности по визуализации и документированию процесса моделирования.

После инсталляции MATLAB пакет автоматически интегрируется с ее средой и на панели инструментов появляется справа пиктограмма с подсказкой Simulink. При щелчке по этой пиктограмме открывается окно пакета и дерево разделов его библиотеки. Исполнение команды simulink в командном окне MATLAB так же приводит к запуску пакета. И, наконец, его можно запустить вместе с открытием имитационной модели, если два раза щелкнуть по имени этой модели в файловом браузере.

Таблица 2.1

Основные блоки для моделирования динамических систем

Обозначение	Назначение блока	Раздел библиотеки
Constant	**Постоянное воздействие**	Sources
Step	Одиночный перепад(толчок)	Sources
Ramp	Наклонная линия k*t	Sources
Sine Wave	Синусоидальное воздействие a*sin (wt)	Sources
Random Number	Случайный сигнал	Sources
From Workspace	Сигнал из рабочей области	Sources
From File	Сигнал из файла	Sources
Signal Builder	Конструктор сигналов	Sources
Saturation	Блок ограничения	Nonlinear
Integrator	Интегрирующий блок	Continuous
Derivative	Дифференцирующий блок	Continuous
Sum	Суммирующий блок (+ и -)	Math
Product	Блок умножения и деления (* и /)	Math
Gain	Блок масштабирования	Math
Math Function	Блок математических функций	Math
Trigonometric	Блок тригонометрических функций	Math
Relational Operator	Операции отношения	Continuous
Logical Operator	Логические операции	Math
Fcn	Блок задания функции	Functions & Tables
MATLAB Fcn	Блок задания m-функции	Functions & Tables
Scope	Виртуальный осциллограф	Sinks
XY Graph	Виртуальный графопостроитель	Sinks
Display	Регистратор значений	Sinks
To Workspace	Запись в рабочую область	Sinks
To File	Запись в файл	Sinks
Stop	Блок остановки работы	Sinks
Goto	Передача данных без соединения	Signal & Systems
From	Получение данных без соединения	Signal & Systems
Mux	Мультиплексор данных	Signal & Systems

Управление моделированием осуществляется с помощью команд меню Simulink. Однако, возможно и прямое управление с помощью команд, используемых системой MATLAB (см. табл. 2.2)

Таблица 2.2

Команды управления моделированием

Команда	Назначение команды
Sim	Запустить модель Simulink
Sldebug	Отладить модель Simulink
Simset	Определить параметры структуры SIM
Simget	Выдать параметры структуры SIM
Linmod	Выделить линейную модель из системы
Dlinmod	Выделить линейную модель из дискретной системы
Trim	Найти рабочую точку устойчивого состояния
close_system	Закрыть модель или блок.
new_system	Создать новое окно с пустой моделью
open_system	Открыть существующую модель или блок
load_system	Загрузить существующую модель без визуализации
save_system	Сохранить открытую модель
add_block	Добавить новый блок
add_line	Добавить новую строку
delete_block	Удалить блок
delete_line	Удалить строку
find_system	Найти модель
hilite_system	Засветить объекты без модели
replace_block	Заменить существующие блоки новыми блоками
set_param	Установить значения параметров для модели или блока
get_param	Выдать значения параметров моделирования модели
Bdclose	Закрыть окно Simulink
Bdroot	Установить корневой уровень для имени модели
Gcb	Выдать имя текущего блока
Gcbh	Выдать дескриптор текущего блока
Gcs	Выдать имя текущей системы
Getfullname	Выдать полный путь для блока

2.2. Визуальное проектирование модели

2.2.1. Постановка задачи и начало создания модели

Решение любой задачи в системе Simulink должно начинаться с ее постановки. Чем глубже продумана постановка задачи, тем больше вероятность ее успешного решения в заданные сроки. В ходе постановки задачи нужно

оценить, насколько суть задачи отвечает возможностям пакета Simulink и какие компоненты последнего могут использоваться для построения модели.

Основные команды редактирования модели сосредоточены в меню Edit. В качестве примера применения этих команд рассмотрим построение простой модели, а точнее, сразу трех простых моделей в пределах одного окна. Это, кстати, будет весьма поучительный эксперимент, показывающий, что можно одновременно моделировать несколько систем.

Итак, сначала откроем пустое окно для новых моделей (кнопка Create new model на панели инструментов браузера библиотек Simulink).

2.2.2 Ввод текстовой надписи

Введем заголовок нашей будущей модели – «Simple model» или по-русски «Простая модель». Для этого достаточно установить курсор мыши в нужное место окна и дважды щелкнуть левой кнопкой мыши. Появится прямоугольная рамка, внутри которой находится мигающий маркер ввода в виде вертикальной палочки.

Теперь можно ввести нужную надпись по правилам, действующим для строчного редактора. Пока будем считать, что параметры надписи по умолчанию нас вполне устраивают.

Заметим, что русские надписи допускаются только в русифицированных версиях системы MATLAB.

2.2.3. Размещение блоков в окне модели

Из раздела библиотеки Sources перенесем с помощью мыши три источника сигнала: синусоидального, прямоугольного (дискретного) и пилообразного. Затем из раздела Sinks поместим в окно модели блок осциллографа.

2.2.4. Выделение блока модели

При выделении блока в меню Edit становится доступными команды редактирования свойств блока. Для выделения блока достаточно навести на него маркер мыши и нажать левую кнопку. В рамке блока по углам появится маленькие темные прямоугольники, которые являются признаком того, что блок выделен.

Если захватить курсором мыши уголок выделенного блока, то можно заметить, что курсор мыши превратился в перекрестие тонких диагональных двухсторонних стрелок. Это означает, что можно пропорционально увеличивать или уменьшать блок в диагональных направлениях.

2.2.5. Меню редактирования

Кратко рассмотрим основные команды меню Edit. Это меню содержит ряд типовых команд, которые разбиты на 6 групп. В первой группе есть 2 команды:

Undo (отмена последней операции) и Redo (восстановление последней отменен-ной операции). Эти команды являются контекстно-зависимыми.

Следующая группа команд связана с операциями в буфере обмена Windows:

Cut – перенос выделенных объектов в буфер;

Copy – копирование выделенных объектов в буфер;

Paste – вставка объектов из буфера в заданное курсором мыши место;

Clear – уничтожение выделенных объектов;

Select All – выделение всех объектов модели;

Copy model to clipboard – копирование всей модели в буфер;

Find – поиск в модели заданного объекта.

Остальные команды подменю Edit носят специальный характер и будут рассмотрены в дальнейшем.

2.2.6. Применение буфера обмена

Вернемся к нашей модели и покажем некоторые примеры работы с буфером обмена. Выделим блок осциллографа Scope. После выполнения команды Copy копия выделенного блока Scope поступает в буфер обмена и хранится в нем. При выполнении команды Cut помещенный в буфер обмена блок исчезает из окна модели.

Теперь для вставки копии блока Scope достаточно поместить в нужное место курсор мыши и выполнить команду Paste. Блок Scope1 появится в указанном месте. Аналогично добавляется еще один блок – Scope2 – к третьему источнику сигнала.

Теперь можно приступать к соединению выходов источников со входами осциллографов. Для этого достаточно указать курсором мыши на начало соединения (выход источника) и затем при нажатой левой кнопки мыши протянуть соединение в его конец (вход осциллографа). В итоге получим три законченные модели.

2.2.7. Выделение ряда блоков и их перенос

Блоки наших моделей размещаются в правой части окна модели. Допустим, мы задумали перенести их разом в левую часть окна. Для этого надо выделить все блоки. Это можно сделать двумя способами.

В первом способе для выделения надо использовать команду Select all. Во втором используется мышь. В стороне от выделяемых блоков надо установить курсор мыши и нажать левую клавишу. Теперь при перемещении мыши появится расширяемый прямоугольник из тонких пунктирных линий. Как только в нем окажется какой-то блок, он будет выделен. Охватив прямоугольником все блоки, можно выделить их.

Выделенный набор блоков можно перетаскивать мышью как единый объект. Отпустив левую клавишу мыши можно увидеть блоки на новом месте.

2.2.8. Запуск нескольких моделей одновременно

Теперь все готово для нашего первого серьезного эксперимента – одновременного запуска нескольких моделей. Чтобы получить приведенные далее результаты необходимо установить параметры: Start time=0 и Stop time=20 в окне установки параметров моделирования (напоминаем, что оно вызывается командой Simulation/Simulation parameters…). После этого, запустив моделирование нажатием кнопки Start Simulation или командой меню Simulation/Start, можно увидеть результат, показанный на осциллограммах экрана.

Чтобы получить осциллограммы от каждого из осциллографов, надо активизировать их, сделав на каждом из них двойной щелчок мышью. При этом появятся их осциллограммы в произвольных местах экрана. Полученные таким образом осциллограммы можно перетащить мышью в удобное для обзора положение. Их можно также растянуть или сжать в любом направлении с помощью мыши, и получить желаемый вид экрана.

Итак, мы видим, что все три модели работают и осциллограммы представляют временные зависимости сигналов, которые вырабатывают источники – синусоиду, прямоугольные импульсы и треугольные импульсы.

2.3. Визуальное редактирование модели

2.3.1. Постановка задачи

В качестве следующего примера рассмотрим тривиальную задачу моделирования работы идеального ограничителя сигналов, на вход которого подается синусоидальное напряжение с амплитудой 5В и частотой 1 рад/сек. Допустим, что пороги ограничения составляют +0.5 и –0.5В. Заметим, что такие параметры источник синусоидального сигнала имеет по умолчанию.

В данном случае очевидно, что основными блоками будут генератор синусоидальных сигналов и блок нелинейности, моделирующий передаточную характеристику ограничителя. Кроме того, к этим блокам надо добавить регистрирующий блок – осциллограф. Так как функциональная схема моделируемого устройства в данном случае вполне очевидна, то мы можем перейти к ее реализации.

2.3.2. Создание и запуск модели ограничителя

Для создания модели данного устройства проделаем следующие действия:

1. Откроем окно новой модели Simulink, нажав кнопку Create a new model.
2. Расположим это окно рядом с окном браузера библиотек.
3. Из раздела библиотеки Sources перенесем в окно модели источник синусоидального сигнала Sine Wave.
4. Из раздела библиотеки Nonlinear перенесем в окно модели нелинейный блок – ограничитель Saturation.

5.Из раздела библиотеки Sinks перенесем в окно модели блок осциллографа Scope.

6.Выполним соединение между блоками.

7.Проверим установку времени моделирования: Start time=0; Stop time=20.

8.Щелкнув дважды по блоку Sine Wave, в появившемся окне параметров источника синусоидального сигнала установим амплитуду, равную 5.

9.Запустим модель на исполнение, нажав кнопку Start Simulation в панели инструментов окна модели.

Результат представлен на осциллограмме экрана.

2.3.3. Настройка масштаба осциллограмм

Нетрудно заметить, что масштаб отображения осциллограммы у осциллографа на экране оказался не совсем удачным – изображение осциллограммы мало по высоте, поскольку при порогах 0.5 масштаб в 5 условных единиц уровня получается слишком крупным. Заметим, что мы не указываем размерность осциллограммы по вертикали. В зависимости от условий задачи это могут быть метры (задача на движение), вольты (электронный ограничитель) и т.д.

Для смены масштаба достаточно щелкнуть правой кнопкой мыши в окне осциллограммы. В появившемся контекстном меню нужно выбрать команду Axes Properties… которая служит для задания масштаба осциллограммы.

В открывшемся окне свойств осей надо заменить значения Y-min=-5 и Y-max=5, например на Y-min=-0.8 и Y-max=0.8. После этого, нажав кнопку Apply, можно увидеть осциллограмму с измененным масштабом.

Для получения максимального изображения следует исполнить команду Autoscale.

2.4 Визуальное редактирование модели

2.4.1 Добавление надписей и текстовых комментариев

Для изменения надписи нужно установить мышь в область надписи и щелкнуть левой кнопкой мыши – в надписи появится курсор ввода, и ее можно будет редактировать.

Чтобы убрать надпись, нужно выделить ее (кстати, как и любой другой объект) и выполнить команду Edit → Clear.

2.4.2 Вставка блоков и их соединение

Вставку блоков с помощью браузера библиотек Simulink мы уже обсудили достаточно подробно. Отметили также, что для переноса блоков, их копирования и размножения целесообразно использовать буфер обмена. Весьма плодотворным является подход, когда пользователь для создания своей модели использует ранее составленную модель – например, из

отлаженных демонстрационных примеров, которых в пакете Simulink великое множество.

Для подключения новых блоков нужны новые соединения. Они также легко выполняются с помощью мыши.

Тем не менее, отметить важнейшие приемы осуществления соединений полезно. Блоки моделей обычно имеют входы и выходы. Как правило, выход какого-либо блока подключается ко входу следующего блока и т.д. Для этого курсор мыши устанавливается на выходе блока, от которого должно исходить соединение. При этом курсор превращается в большой крестик из тонких линий. Держа нажатой левую кнопку мыши, надо плавно переместить курсор ко входу следующего блока, где курсор мыши приобретет вид крестика из тонких сдвоенных линий.

Добившись протяжки линий ко входу следующего блока, надо отпустить левую кнопку мыши. Соединение будет завершено, и в конце его появится жирная стрелка. Щелчком мыши можно выделить соединение, признаком чего будут черные прямоугольники, расположенные в узловых точках соединительной линии.

Иногда бывает нужно сделать петлю соединительной линии в ту или иную сторону. Для этого нужно захватить нужную часть линии и отвести ее в нужную сторону, перемещая мышь с нажатой левой кнопкой. Создание петли линии заканчивается отпусканием левой кнопки мыши. Существует возможность задания наклонных линий соединений при нажатой клавише Shift.

2.4.3 Создание отвода линии

Часто возникает необходимость сделать отвод от уже созданной линии. Заметим, что при нажатой клавише Shift отвод строится наклонными линиями.

В примере использована модель интегратора, подключенного к выходу источника прямоугольных импульсов. Чтобы было можно наблюдать осциллограммы как на выходе источника, так и на выходе интегратора, в схему включен блок мультиплексора сигналов Mux с двумя входами. Чтобы подключить нижний вход к уже задействованному выходу источника, нам и понадобилось создать отвод линии.

Нетрудно убедиться в том, что сигнал на выходе интегратора представляет собой ступенчато нарастающую линию. Когда на выходе генератора имеется высокий (условно) уровень напряжения, на выходе интегратора сигнал линейно нарастает. Когда уровень на генераторе равен 0, сигнал на выходе интегратора остается неизменным. Такой характер процесса, разумеется, хорошо знаком специалистам.

2.4.4 Удаление соединений

Для удаления соединительной линии достаточно выделить ее и выполнить команду Clear или Cut. Нужное соединение будет удалено. Напоминаем, что команда Undo позволяет восстановить удаленное соединение сразу после удаления.

2.4.5 Изменение размеров блоков

Simulink имеет расширенные возможности редактирования блок-схем. Так, блоки в окне редактирования можно не только перемещать с помощью мыши, но и изменять их размеры. Для этого блок выделяется, после чего курсор мыши надо установить на кружки по углам блока. Как только курсор мыши превратится в двунаправленную диагональную стрелку, можно будет при нажатой левой кнопке растягивать блоки по диагонали, увеличивая или уменьшая их размеры, при этом растягивается только графическое изображение (пиктограмма) блока, а размеры его названия в виде текстовой надписи не изменяются.

2.4.6 Перемещение блоков и вставка блоков в соединение

Блок, участвующий в соединении, можно перемещать в окне модели, выделив его и перетаскивая, как обычно, мышью. При этом соединение не прерывается, а просто сокращается или увеличивается в длине. В длинное соединение можно вставить новый блок, не разрушая его и не выполняя сложных манипуляций.

Однако подобная вставка возможна для блоков, имеющих один вход и один выход, которые включаются в соединение. Попытка вставить таким образом мультиплексор будет безуспешной, поскольку он имеет два входа и не стыкуется с разрываемым соединением.

Чтобы вставить мультиплексор, следует удалить соединение между дифференцирующим устройством и выходом осциллографа. Для этого соединение выделяется и выполняется команда Edit$\rightarrow$Clear. После этого мультиплексор помещают в нужное место и соединения создаются заново.

2.4.7 Моделирование дифференцирующего устройства

Итак, мы фактически создали модель дифференцирующего устройства и можем смотреть, что происходит при дифференцировании синусоидального сигнала. Внимательно присмотревшись к осциллограммам, мы видим, что при входном синусоидальном сигнале выходной сигнал является косинусоидой. Это вполне отвечает математическим соотношениям для данного случая.

Однако, в самом начале процесса дифференцирования хорошо виден изъян работы модели – при $t = 0$ производная равна не 1, а 0. Это связано с тем, что процесс начинается при нулевых начальных условиях. Но довольно быстро ситуация исправляется, и в дальнейшем выходной сигнал становится

косинусоидальным. Таким образом, дифференцирующее устройство можно использовать для точного сдвига на 90^0 гармонического сигнала любой частоты.

Обратите внимание, что в отличие от блока интегратора блок цифрового дифференциатора не имеет настраиваемых параметров. Его окно параметров является чисто информационным.

2.4.8 Команды Undo и Redo в окне модели

Большую помощь в редактировании оказывает команда Undo – отмена последней операции. Она поддерживает свыше ста различных операций, включая операции добавления и стирания линий. Эту команду можно вызвать с помощью кнопки в панели инструментов окна модели или из меню Edit. Для восстановления отмененной операции служит команда Redo.

2.4.9 Меню форматирования Format

Меню Format позволяет установить для надписей шрифт и его размер, способ выравнивания текста и его размещение по сторонам блока, а также поворачивать блоки на 90^0 и создавать их тени.

2.5 Отладка имитационной модели

Подготовка и запуск моделей в Simulink производится настолько просто и наглядно, что не возникает, как правило, необходимости в использовании специальных отладочных средств, которые находятся в меню Tools/Simulink Debugger. Обычно разработчики имитационных моделей, используя метод проб и ошибок, постепенно совершенствуют их и добиваются корректной работы. Для контроля состояния тех или иных блоков модели к ним подключаются регистраторы, например осциллографы или графопостроители, что позволяет оценивать уровни входных и выходных сигналов и их временные или иные графические зависимости.

Запуск отладчика можно произвести так же с помощью кнопки Debug панели инструментов Simulink или командной sldebug в командном окне MATLAB. Панель инструментов отладчика обеспечивает:

Step to next block – переход к следующему блоку;

Go to start of next time – переход к следующему такту времени;

Start/Continue – старт/продолжение отладки;

Break before selected block – установка точки прерывания перед выделенным блоком;

Display I/O of selected block with executed – отображение при исполнении входных и выходных данных выбранного блока;

Display current I/O of selected block – отображение текущих входных и выходных данных выбранного блока;

Help – вызов справочной системы по отладчику.

В окне отладочных данных выдается модельное время Tm, номер контролируемого блока и его входные U_i и выходные Y_i данные, а также порядок работы блоков. Их анализ позволяет оценить правильность работы модели.

Отладчик позволяет установить дополнительные условия остановки процесса моделирования:

Zero crossing – прохождение сигнала через нулевой уровень;

Step size limited by state – превышение допустимого значения шага;

Minor time step – недопустимо малые шаги времени;

NaN values – появление нечисловых значений;

Break at time – остановка в заданный момент времени.

В окне состояний отладчика (вкладка Status) отображается до 15 различных его состояний: количество точек прерываний, текущее модельное время и т.д.

2.6 Верификация математических моделей

Отлаженная имитационная модель должна быть тщательно проанализирована на предмет ее адекватности исследуемому динамическому процессу и достаточной точности полученных результатов. Для этих целей необходим какой-то минимум натурных испытаний, а так же использование других видов математического моделирования – аналитического и численного.

Например, для динамического процесса, описываемого следующим линейным обыкновенным дифференциальным уравнением с постоянными коэффициентами

$$Ax'' + Bx' + Cx = u(t) \tag{2.1}$$

реакцией на единичное толчкообразное возмущение $u(t) = 1(0)$ является

$$x(t) = \frac{1}{C} - A_2 e^{-Beta*t} + A_1 e^{Alfa*t} \tag{2.2}$$

где

$$Alfa = \frac{B + 2\sqrt{A*C*(R^2 - 1)}}{2A} \tag{2.3}$$

$$Beta = \frac{B - 2\sqrt{A*C*(R^2 - 1)}}{2A} \tag{2.4}$$

$$A_1 = \frac{\sqrt{A}*Beta}{2C\sqrt{C(R^2 - 1)}} \tag{2.5}$$

$$A_2 = \frac{\sqrt{A}*Alfa}{2C\sqrt{C(R^2 - 1)}} \tag{2.6}$$

$$R = \frac{B}{2\sqrt{A*C}} \geq 1 \tag{2.7}$$

Задавая значения коэффициентам A, B и C и соблюдая условие $R \geq 1$, можно сравнить расчетную функцию $x(t)$ с модельной и оценить абсолютную и относительную погрешность.

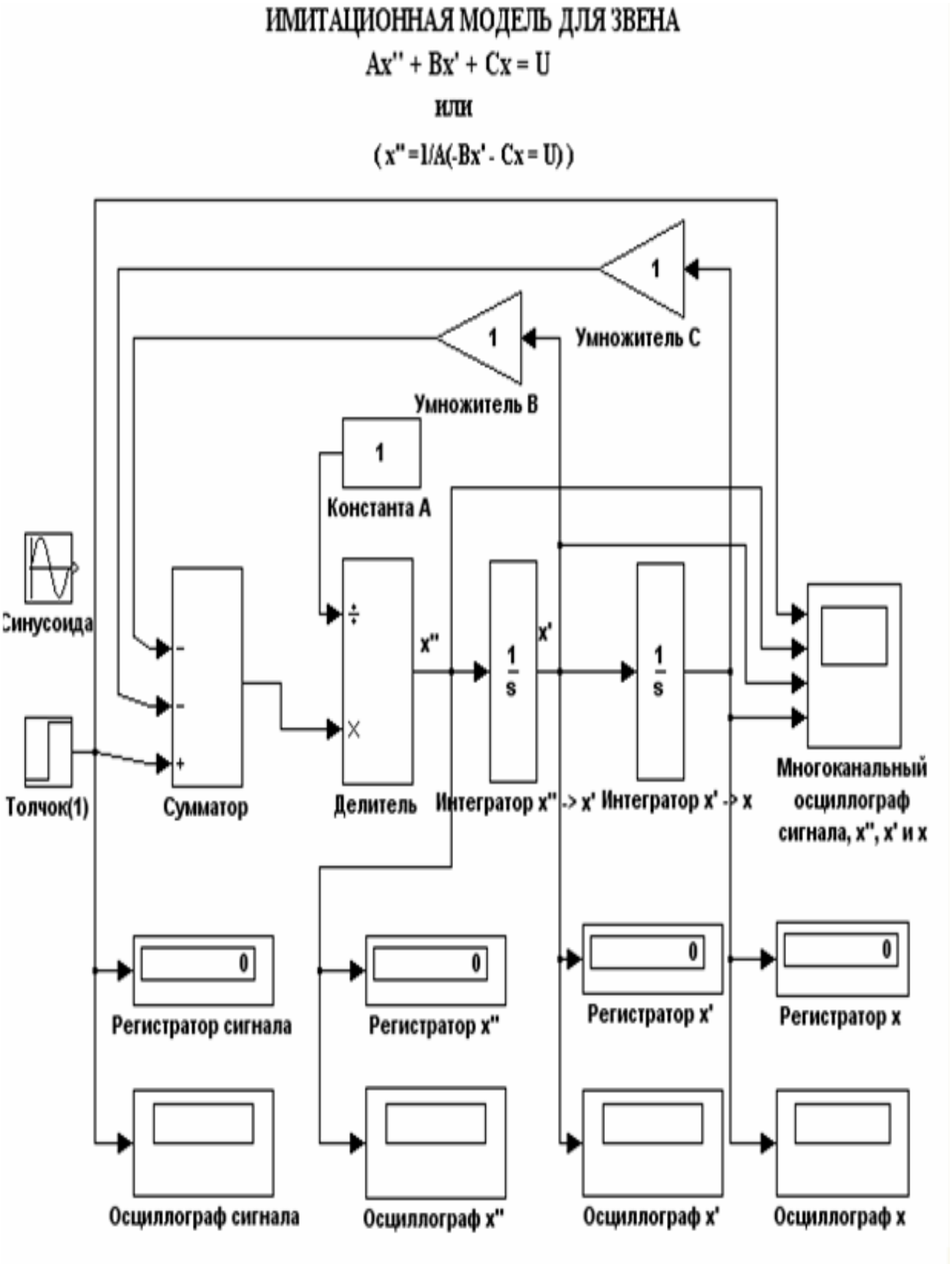

Рис. 2.1 Имитационная модель для динамического звена второго порядка

2.7 Генерация отчета по моделированию

Пакет Simulink обеспечивает автоматическое формирование отчета по моделированию в формате HTML. Для этих целей необходимо загрузить в рабочее окно файл с требуемыми моделями или моделью и выполнить следующие действия:

1. Выполнить команду Tools/Coverage settings и в диалоговом окне установить переключатель Generate HTML report.
2. Выполнить команду Tools/Report Generator.
3. В диалоговом окне выбрать требуемый шаблон редактирования, например, simulink-default.rpt и щелкнуть по кнопке Set.
4. Щелкнуть по кнопке Edit и в открывшемся окне еще раз щелкнуть по кнопке Report.
5. Подождать, пока не будет сформирован отчет и загружен его гипертекстовый файл в окно Интернет-браузера.

Отчет содержит названия моделей, таблицы параметров блоков, результаты моделирования в виде осциллограмм, структурные схемы моделей и оглавление с гипертекстовыми ссылками.

Таким образом, полученный отчет отличается детальностью. Его можно использовать как непосредственно в отчетных материалах по лабораторной работе, так и для подготовки справочных материалов для библиотек имитационных моделей.

2.8 Варианты заданий и порядок выполнения работы

Задания позволяют исследовать характеристики существующих источников сигналов и научиться конструировать новые, освоить средства регистрации и визуализации результата моделирования, правильно включать в модель и настраивать функциональные блоки, конструировать имитационные модели для динамических систем, рационально использовать средства их отладки и верификации, а также оформлять электронный отчет по проделанной работе. Предлагается выполнять задания в следующем порядке:

1. Открыть новое окно в среде Simulink и поместить в него источники Constant, Step, Ramp, Sine Wave и Random Number.
2. Подключить к каждому источнику осциллограф Scope и регистратор Display.
3. Промоделировать, используя параметры блоков и пакета Simulink, заданные по умолчанию, и команду Simulink/Start.
4. Проанализировать осциллограммы и конечные значения в регистраторах.
5. Сохранить модели под именем Lab2Zad1, сгенерировать отчет и предъявить преподавателю. Сохранить его под именем Lab2Rpt1.
6. К каждому осциллографу присоединить все источники, изменив количество окон в параметрах этих осциллографов.
7. Промоделировать и осциллограммы показать преподавателю, а затем сохранить модель под именем Lab2Zad2.

8.Используя мультиплексор данных Mux, соединить его выход со входами осциллографов, предварительно задав количество окон для них равным 1.

9.Промоделировать и осциллограммы показать преподавателю, а затем сохранить модель под именем Lab2Zad3.

10.Открыть новое окно и поместить в него источник Sine Wave, блок фиксированной задержки Transport Delay из раздела библиотеки Continuas и виртуальный графопостроитель XY Graph.

11.На один вход графопостроителя подать сигнал непосредственно, на второй – через блок задержки.

12.Изменяя величину задержки, проанализировать поведение фазовой траектории на экране графопостроителя. Модель сохранить под именем Lab2Zad4.

13.Построить модель, которая сохраняет выходные данные в рабочей области и файле, используя для этих целей блоки To Workspace и To File. Сохранить модель под именем Lab2Zad5 и проанализировать данные в рабочей области и в файле.

14.Построить модели, в которых источниками сигналов являются блоки From Workspace и From File, используя при этом данные, полученные на шаге 13. Модели сохранить под именем Lab2Zad6.

15.Используя источник Ramp, осциллограф Scope и различные функциональные блоки, построить осциллограммы (графики) этих функций, произвести сравнение значений с помощью блоков Rational Operator, а также выполнить логические операции с помощью блоков Logical Operator. Требуемые блоки находятся в разделах Math и Continuas. Модели сохранить в файле Lab2Zad7.

16.Сконструировать модель для построения графика функции $y = (5 + x)^2 - x^3$, используя блок Fcn. Модель сохранить под именем Lab2Zad8.

17.Повторить пункт 16, используя блок MATLAB Fcn, для функции $y = (5 + \sin(x))^2 - e^{-3x}$. Имя файла для модели Lab2Zad8.

18.Построить имитационную модель для динамического процесса, описываемого дифференциальным уравнением второго порядка

$$Ax'' + Bx' + Cx = u(t)$$

Модель сохранить под именем Lab2Zad9.

19.Провести исследование модели Lab2Zad9, изменяя коэффициенты A, B, C, задавая различные начальные условия и меняя источники воздействия $u(t)$. По результатам исследований сгенерировать отчет Lab2Rpt2. Сравнить их с аналитическими решениями.

20.Провести исследования модели Lab2Zad9, задавая A, B и C в виде функций времени A(t), B(t) и C(t). Отчет сохранить под именем Lab2Rpt3, а полученную модель – под именем Lab2Zad10.

21.Увеличивая количество интеграторов в модели Lab2Zad2, оценить характер получаемого решения x(t).

22.Для модели Lab2Zad9 включить отладочный режим и апробировать все опции отладчика Simulink.

23.Оформить электронный отчет по лабораторной работе, используя промежуточные отчеты Lab2Rpt1, Lab2Rpt2 и Lab3Rpt3, напечатать и предъявить преподавателю. Обосновать состоятельность полученных результатов.

2.9 Оформление отчета по результатам исследований

Заключительный отчет содержит самые существенные сведения по проектированию имитационных моделей, настройке параметров блоков, отладке и верификации. Он составляется на основании промежуточных отчетов, создаваемых автоматически, при этом используется русский язык. Прилагаются также результаты исследований в виде таблиц и осциллограмм. Сведения по моделям и блокам должны быть достаточными для воспроизведения моделей, а полученные результаты должны быть обоснованны.

Лабораторная работа № 3

ДИНАМИЧЕСКИЕ СИСТЕМЫ И МЕТОДЫ ИХ МАТЕМАТИЧЕСКОГО МОДЕЛИРОВАНИЯ

Цель работы: разработка аналитических моделей для определения поведения динамических систем, описываемых обыкновенными дифференциальными уравнениями n-го порядка с постоянными и переменными коэффициентами, реализация этих моделей в программной среде математической системы MATLAB, построение имитационных моделей с помощью пакета Simulink и верификация разработанных математических моделей путем сравнения результатов их работы.

3.1 Постановка задач исследования

Математическое описание обширного класса задач современной техники проводится с помощью нелинейных систем обыкновенных дифференциальных уравнений вида

$$\begin{aligned}
&\frac{dx_1}{dt} = f_1(x_1, x_2 ..., x_n, t);\\
&\frac{dx_2}{dt} = f_2(x_1, x_2 ..., x_n, t);\\
&\bullet\ \bullet\ \bullet\ \bullet\ \bullet\ \bullet\ \bullet\ \bullet\\
&\frac{dx_n}{dt} = f_n(x_1, x_2 ..., x_n, t),
\end{aligned} \tag{3.1}$$

называемых динамическими системами.

Динамические системы (3.1) описывают процессы разнообразной физической природы – механические, электрические, тепловые, химические и т.д. В табл. 3.1 приведены примеры дифференциальных уравнений для таких систем.

Движение динамической системы, т.е. решение

$$\begin{aligned}
&x_1 = \varphi_1(x_1, x_2 ..., x_n, t);\\
&x_2 = \varphi_2(x_1, x_2 ..., x_n, t);\\
&\bullet\ \bullet\ \bullet\ \bullet\ \bullet\ \bullet\ \bullet\ \bullet
\end{aligned} \tag{3.2}$$

$$x_n = \varphi_n(x_1, x_2 ..., x_n, t);$$

полностью определяется заданием начальных параметров $x_1^0, x_2^0, ..., x_n^0$ в момент $t = t_0$ (задача Коши) или заданием конечных значений $x_1^k, x_2^k, ..., x_n^k$ в момент $t = t_k$ (краевая задача), если функции $f_1, f_2, ..., f_n$ удовлетворяют некоторым общим условиям. Возможно также задание некоторой линейной комбинации значений $x_1^0, x_2^0, ..., x_n^0$ и $x_1^k, x_2^k, ..., x_n^k$ в точках t_0 и t_k соответственно, а также другие формулировки, например, нахождение функций $f_1, f_2, ..., f_n$, обеспечивающих оптимальное движение системы.

Независимо от происхождения исследуемой динамической системы оказывается полезной интерпретация системы (3.1) в пространстве координат $x_1, x_2 ..., x_n$, которые называют фазовым пространством. При такой интерпретации пространство представляют заполненным непрерывной средой, частицы которой перемещаются со скоростью

$$\vec{\upsilon} = \frac{dx_1}{dt}\vec{i}_1 + \frac{dx_2}{dt}\vec{i}_2 + ... + \frac{dx_n}{dt}\vec{i}_n = f_1\vec{i}_1 + f_2\vec{i}_2 + ... + f_n\vec{i}_n,$$

(3.3)

где $\vec{i}_1$, $\vec{i}_2$, ..., $\vec{i}_n$ являются единичными векторами в этом пространстве.

Кривые, соответствующие перемещающейся в фазовом пространстве точке $x_1, x_2 ..., x_n$, называются фазовыми траекториями. Их изучение оказывается весьма плодотворным для оценки поведения динамической системы.

Следующая теорема Коши дает точную формулировку условий, обеспечивающих существование и единственность решения в некоторой близости начальной точки:

Система обыкновенных дифференциальных уравнений (3.1) с начальными условиями $x_1(t_0) = x_{10}; x_2(t_0) = x_{20}; x_3(t_0) = x_{30}, ...; x_n(t_0) = x_{n0}$ имеет в неко-тором интервале $t_0 - h \le t \le t_0 + h$, где h – константа, зависящая от системы (3.1), единственное решение

$$x_1 = \varphi_1(t); x_2 = \varphi_2(t), ...; x_n = \varphi_n(t),$$

(3.4)

принимающее при $t = t_0$ заданные значения $\varphi_1(t_0) = x_{10}; \varphi_2(t_0) = x_{20}; ...; \varphi_n(t_0) = x_{n10}$, если:

1)функции $f_i(x_1, x_2 ..., x_n, t)$ непрерывны по всем аргументам в рассматриваемой замкнутой области D изменения аргументов $x_1, x_2 ..., x_n, t$;

2)функции f_i удовлетворяют условию Липшица относительно аргументов $x_1, x_2 ..., x_n$, т.е. при данном t для любых $x_1'', x_2'', ..., x_n''$ и $x_1', x_2', ..., x_n'$, являющихся двумя системами значений из области D, выполняются неравенства

$$\left| f_1(x_1', x_2', ...x_n', t) - f_1(x_1'', x_2'', ...x_n'', t) \right| \le k \left\{ \left| x_1' - x_1'' \right| + \left| x_2' - x_2'' \right| + ... + \left| x_n' - x_n'' \right| \right\},$$

(3.5)

где k - положительная постоянная.

Таблица 3.1

Варианты описаний динамических систем

Номер варианта	Варианты уравнений или их систем	Начальные условия
1	$x' = \frac{x-1}{t+1}$	$x(1) = 0$
2	$x' = \frac{1+x^2}{tx}$	$x(2) = 1$
3	$x' + \frac{3}{t}x = \frac{2}{t^3}$	$x(1) = 3$
4	$x' + 2tx = 2t^2 e^{-t^2}$	$x(0) = 1$
5	$x' = \frac{2tx}{t^2 + x^2}$	$x(1) = 2$
6	$t^3 x'' + t^2 x' = 1$	$x(1) = 1, \quad x(e) = 0$
7	$x'' - 2x' + x = t^3$	$x(0) = 24, \quad x(1) = 7$
8	$x'' - 3x' + 2x = e^t$	$x(0) = 1, \quad x(1) = 0$
9	$x'_1 = 2x_1 - x_2; \quad x'_2 = 3x_1 - 2x_2$	$x_1(0) = 1, \quad x_2(0) = 0$
10	$x'_1 = 4x_1 + 2x_2, \quad x'_2 = -x_1 + 2x_2$	$x_1(0) = 2, \quad x_2(0) = 1$
11	$x'_1 + 5x_1 - 2x_2 = 40e^t, \quad x'_2 - x_1 + 2x_2 = -9e^t$	$x_1(0) = 2, \quad x_2(0) = 1$

В качестве примера краевой задачи можно назвать задачу об изгибе балки, жестко заделанной в точке $x = 0$ и свободной в точке $x = 1$.

Следует заметить, что решение краевых задач значительно сложнее решения задач Коши за исключением тех случаев, когда в распоряжении исследователей имеется точное решение с произвольными константами, которые остается выбрать для удовлетворения краевых условий. В самом деле, при исследовании задачи Коши заданы все условия для приближенного построения решения; так, например, для системы

$$\frac{dx_1}{dt} = f_1(x_1, x_2, t);$$
$$\frac{dx_2}{dt} = f_2(x_1, x_2, t);$$

(3.6)

мы знаем x_{10}, x_{20} и, следовательно, $\frac{dx_1}{dt}\Big|_0$ и $\frac{dx_2}{dt}\Big|_0$. Поэтому у нас хватает данных для построения интегральной кривой хотя бы путем стыкования кусков касательных. Если же для той же системы заданы

$$x_1(t_0) = x_{10} \qquad t_0 < t_1 \ ,$$
$$x_2(t_1) = x_{21}$$

(3.7)

то данных для приближенного построения решения в точке $t = t_0$ не хватает, ибо мы непосредственно не может использовать значения $x_2(t_1)$. Вопросы существования и единственности решения краевых задач решаются много сложнее, чем аналогичные вопросы для задачи Коши даже в случае линейного уравнения.

Если при составлении уравнений принимаются во внимание все факторы, влияющие на динамическую систему, то уравнения (3.1) получаются сложными, по большей части нелинейными и трудно поддающиеся решению. Для качественного исследования таких систем уравнения отдельных элементов (звеньев) заменяют приближенными к ним линейными дифференциальными уравнениями с постоянными коэффициентами. Такое преобразование системы уравнений называется ее линеаризацией. На вопрос, в какой мере и при каких условиях допустима такая линеаризация уравнений, отвечают теоремы об устойчивости линеаризованных систем [4].

Линеаризованную систему можно легко привести к одной функции $x\ (t)$ и записать в операторной форме:

$$D(p)x(t) = U(p)u(t).$$

(3.8)

В этом уравнении $p = \frac{d}{dt}$ – оператор дифференцирования, а $D(p)$ и $U(p)$ - операторные полиномы с постоянными коэффициентами.

Как известно из теории дифференциальных уравнений, общее решение $x(t)$ уравнения (3.8) находится как сумма двух решений $x_c(t)$ и $x_b(t)$:

$$x(t) = x_c(t) + x_b(t)$$

(3.9)

где $x_c(t)$ – общее решение однородного уравнения

$$D(p)x_c(t) = 0,$$

(3.10)

а $x_b(t)$ – частное решение заданного уравнения

$$D(p)x_b(t) = U(p)u(t)$$

(3.11)

Движение системы, определяемое составляющей $x_c(t)$, называется свободным движением, а составляющей $x_b(t)$ – вынужденным движением.

Общее решение однородного уравнения (3.1) $x_c(t)$ состоит из суммы слагаемых, каждое из которых отвечает корню или группе корней характеристического уравнения

$$D(p) = 0.$$

(3.12)

Вид слагаемого зависит от типа корня или группы корней следующим образом:

а) каждому значению вещественного корня $p = p_k$ отвечает слагаемое вида

$$C_k e^{p_k t}$$

(3.13)

где C_k – постоянная интегрирования;

б) каждой группе из s вещественных кратных корней отвечает слагаемое вида

$$(C_1 + C_2 t + ... + C_s t^{s-1}) e^{p_k t}$$

(3.14)

где $C_1, C_2, C_3, ..., C_s$ – постоянные интегрирования;

в) каждой паре комплексных сопряженных корней $\alpha_k \pm i\beta_k$ отвечает слагаемое вида

$$e^{\alpha_k t}(C_k \cos \beta_k t + D_k \sin \beta_k t)$$

(3.15)

где C_k и D_k – постоянные интегрирования;

г) каждой группе из s комплексных сопряженных кратных корней отвечает слагаемое вида

$$e^{\alpha_k t}[(C_1 + C_2 t + ... + C_s t^{s-1}) c\beta_k t + (D_1 + D_2 t + ... + D_s t^{s-1}) \sin \beta_k t]$$

(3.16)

где C_i и D_i – постоянные интегрирования.

Число постоянных интегрирования во всех случаях должно быть равно порядку дифференциального уравнения (3.8). Они определяются из начальных условий движения системы регулирования, в качестве которых принимают значения x_c и всех её производных до (n-1)-го порядка включительно в начальный момент времени t = 0 согласно теореме Коши.

Свободное движение системы

$$x_{пер}(t) = x_c(t) = x(t) - x_b(t)$$

(3.17)

определяет переходной процесс регулирования, который имеет большое значение для работы системы.

В зависимости от значений корней характеристического уравнения могут иметь место следующие виды переходного процесса.

1.Один из вещественных корней положительный. Соответствующее этому корню слагаемое $C_k e^{p_k t}$ с течением времени неограниченно возрастает, в результате чего $x_{пер} \to \infty$ при $t \to \infty$. Такой переходной процесс называется неустойчивым.

2.Вещественная часть одной пары комплексных сопряженных корней положительна. Соответствующее этим корням слагаемое $e^{\alpha_k t}(C_k \cos\beta_k t + D_k \sin\beta_k t)$ определяет периодические колебания с неограниченно возрастающей амплитудой. Такой переходной процесс также называется неустойчивым.

3.Все вещественные корни, а также вещественные части комплексных сопряженных корней отрицательны. В этом случае все оставляющие, как нетрудно убедится, с течением времени стремятся к нулю, и $x_{пер} \to \infty$ при $t \to \infty$. Переходный процесс называется устойчивым, или затухающим. При этом различают затухание апериодическое, когда оно происходит без колебаний, и колебательное, когда затухающая составляющая совершает периодические колебания с амплитудой, стремящейся к нулю.

4.Один из вещественных корней равен нулю. В этом случае в составе $x_{пер}$ появляется постоянное слагаемое.

5.Пара комплексных сопряженных корней имеет нулевую вещественную часть. В этом случае $x_{пер}$ будет содержать слагаемое, представляющее колебательное движение с постоянной амплитудой. Такие колебания называются не затухающими.

6.Имеются кратные нулевые корни и кратные чисто мнимые корни. В этом случае переходной процесс неустойчив, так как одно из слагаемых $x_{пер}$ будет неограниченно возрастать апериодически или в виде колебаний с возрастающей амплитудой.

3.2 Разработка аналитических моделей

Рассмотрим пример динамической системы. Пусть дифференциальное уравнение, которое описывает ее поведение, имеет вид:

$$x''-2x'-3x = e^{4t} \quad , \quad x(0)=1 \quad x'(0)=0 . \tag{3.18}$$

Перепишем уравнение так:

$$x''-2x'-3x = e^{4t} . \tag{3.19}$$

Найдем отображения функции и ее производных по таблице или с помощью функции laplace системы MATLAB:

$$x \to X(p);$$
$$x' \to pX(p) - x(0);$$
$$x'' \to p^2 X(p) - px(0) - x'(0);$$
$$e^{4t} \to \frac{1}{p-4};$$

(3.20)

$$x' \to pX(p) - 1;$$
$$x'' \to p^2 X(p) - p.$$

Подставим в уравнение(3.19) и выполним преобразования:

$$p^2 X(p) - p - 2(pX(p) - 1) - 3X(p) = \frac{1}{p-4};$$
$$p^2 X(p) - p - 2pX(p) + 2 - 3X(p) = \frac{1}{p-4};$$

(3.21)

$$p^2 X(p)(p-4) - p(p-4) - 2pX(p)(p-4) + 2(p-4) - 3X(p)(p-4) = 1;$$
$$X(p)(p-4)(p^2 - 2p - 3) = 1 + p(p-4) - 2(p-4);$$
$$X(p)(p-4)(p^2 - 2p - 3) = p^2 - 6p + 9.$$

В результате получаем:

$$X(p) = \frac{p^2 - 6p + 9}{(p-4)(p^2 - 2p - 3)} = \frac{(p-3)^2}{(p-4)(p+1)(p-3)} = \frac{(p-3)}{(p-4)(p+1)}.$$

(3.22)

Разложим на простейшие дроби:

$$\frac{(p-3)}{(p-4)(p+1)} = \frac{A}{p-4} + \frac{B}{p+1} = \frac{A(p+1) + B(p-4)}{(p-4)(p+1)}.$$

(3.23)

Пусть $p = -1$, тогда $-5B = -4$; $B = \frac{4}{5}$.

Пусть $p = 4$, тогда $5A = 1$; $A = \frac{1}{5}$.

Находим отображение функции $x(t)$:

$$X(p) = \frac{(p-3)}{(p-4)(p+1)} = \frac{1}{5} \cdot \frac{1}{p-4} + \frac{4}{5} \cdot \frac{1}{p+1}.$$

(3.24)

Перейдем к оригиналу. Так как $e^{\alpha t} \to \frac{1}{p-\alpha}$, то

$$x(t) = \frac{1}{5} e^{4t} + \frac{4}{5} e^{-t}.$$

(3.25)

Рассмотрим еще один пример, когда динамика описывается системой дифференциальных уравнений следующего вида:

$$\begin{cases} \frac{dx_1}{dt} = 2x_1 + x_2 \\ \frac{dx_2}{dt} = x_1 + 2x_2 \end{cases},$$

(3.26)

$$x_1(0) = 1,$$
$$x_2(0) = 3.$$

Перепишем систему так:

$$\begin{cases} x'_1 = 2x_1 + x_2 \\ x'_2 = x_1 + 2x_2 \end{cases}.$$

(3.27)

Перейдем к отображениям:

$$x_1 \to X1(p);$$
$$x_2 \to X2(p);$$
$$x'_1 \to pX1(p) - x_1(0);$$

(3.28)

$$x'_2 \to pX2(p) - x_2(0);$$
$$x'_1 \to pX1(p) - 1;$$
$$x'_2 \to pX2(p) - 3.$$

Подставляем в (3.27):

$$\begin{cases} pX1(p) - 1 = 2X1(p) + X2(p); \\ pX2(p) - 3 = X1(p) + 2X2(p); \end{cases}$$

(3.29)

$$\begin{cases} (p-2)X1(p) - X2(p) = 1; \\ -X1(p) + (p-2)X2(p) = 3; \end{cases}.$$

(3.30)

Умножим первое уравнение на $(p-2)$ и сложим уравнения:

$$\begin{gathered} ((p-2)^2 - 1)X1(p) = p - 2 + 3; \\ (p^2 - 4p + 3)X1(p) = p + 1; \end{gathered}$$

(3.31)

$$X1(p) = \frac{p+1}{(p-1)(p-3)}.$$

Разложим на простейшие дроби:

$$\frac{p+1}{(p-1)(p-3)} = \frac{A}{p-1} + \frac{B}{p-3} = \frac{A(p-3) + B(p-1)}{(p-1)(p-3)}.$$

(3.32)

Пусть $p = 1$, тогда $-2A = 2$, $A = -1$.

Пусть $p = 3$, тогда $2B = 4$, $B = 2$.

$$X1(p) = \frac{p+1}{(p-1)(p-3)} = -\frac{1}{p-1} + \frac{2}{p-3}.$$

(3.33)

Отображение для $x_1(t)$ найдено. Теперь можем найти оригинал по таблицам[5]: $x_1(t) = 2e^{3t} - e^t$.

Найдем $X2(p)$. Подставим $X1(p)$ в первое уравнение системы (3.30):

$$\frac{(p-2)(p+1)}{(p-1)(p-3)} - 1 = X2(p).$$

(3.34)

Приведем к общему знаменателю:

$$X2(p) = \frac{3p-5}{(p-1)(p-3)}.$$

(3.35)

Разложим на простейшие дроби:

$$\frac{3p-5}{(p-1)(p-3)} = \frac{A}{p-1} + \frac{B}{p-3} = \frac{A(p-3)+B(p-1)}{(p-1)(p-3)}.$$

(3.36)

Пусть $p=1$, тогда $-2A=-1$; $A=1$.

Пусть $p=3$, тогда $2B=4$; $B=2$ и отображение запишется следующим образом:

$$X2(p) = \frac{3p-5}{(p-1)(p-3)} = \frac{1}{p-1} + \frac{2}{p-3}.$$

(3.37)

Тогда оригинал равен

$$x_2(t) = e^t + 2e^{3t}.$$

(3.38)

3.3 Программная реализация аналитических моделей

```
function [X,DX,D2X] = DSolveXDXD2X
%-- ФУНКЦИЯ ДЛЯ  НАХОЖДЕНИЯ X(t)
%-- И ЕЕ ПРОИЗВОДНЫХ DX(t) и D2X(t):
%-- 1.Нахождение X(t):
    syms x t
    X = dsolve('D2x-2*Dx-3*x=exp(4*t)','x(0)=1','Dx(0)=0');
%-- 2.Нахождение DX(t):
    DX = diff(X);
%-- 3.Нахождение D2X(t):
     D2X = diff(DX);
%-- 4.Векторизация и подстановка X,DX,D2X и t:
    X = vectorize(X)
    X = subs(X,{t},{0:0.1:1});
    DX = vectorize(DX);
    DX = subs(DX,{t},{0:0.1:1});
    D2X = vectorize(D2X);
```

```
      D2X = subs(D2X,{t},{0:0.1:1})
      t = 0:0.1:1;
%-- 5.Визуализация U(t):
      U = exp(4.*t);
      subplot(4,2,1)
      plot(t,U,'r')
      xlabel('t')
      ylabel('U')
%-- 6.Визуализация X(t):
      subplot(4,2,3)
      plot(t,X,'r')
      xlabel('t')
      ylabel('X')
%-- 7.Визуализация DX(t):
      subplot(4,2,2)
      plot(t,DX,'r')
      xlabel('t')
      ylabel('DX')
%-- 8.Визуализация D2X(t):
      subplot(4,2,4)
      plot(t,D2X,'r')
      xlabel('t')
      ylabel('D2X')
%-- 9.Конец функции DSolveXDXD2X
function [X,DX,D2X] = EvalXDXD2X
%-- ФУНКЦИЯ ДЛЯ ВЫЧИСЕНИЯ ЗНАЧЕНИЙ U(t),X(t) =
%--     1/5e(4t)+4/5e(-t) И ЕЕ ПРОИЗВОДНЫХ:
%-- 1.Задание вектора t:
      t  =  0:0.1:1;
%-- 2.Вычисление U(t):
      U  = exp(4.*t);
%-- 3.Вычисление X(t):
      X  = 1./5.*exp(4.*t)+4./5.*exp(-t);
%-- 4.Вычисление DX(t):
      DX  =  4./5.*exp(4.*t)-4./5.*exp(-t);
%-- 5.Вычисление D2X(t):
      D2X  = 16./5.*exp(4.*t)+4./5.*exp(-t);
%-- 6.Визуализация U(t):
      subplot(4,2,1)
      plot(t,U,'r')
      xlabel('t')
      ylabel('U')
%-- 7.Визуализация X(t):
      subplot(4,2,3)
      plot(t,X,'r')
```

```
    xlabel('t')
    ylabel('X')
%-- 8.Визуализация DX(t):
    subplot(4,2,2)
    plot(t,DX,'r')
    xlabel('t')
    ylabel('DX')
%-- 9.Визуализация D2X(t):
    subplot(4,2,4)
    plot(t,D2X,'r')
    xlabel('t')
    ylabel('D2X')
%-- 10.Конец функции EvalXDXD2X
```

3.4 Построение имитационных моделей

В соответствии с математическим описанием объекта управления и поставленными задачами имитационная модель содержит два интегрирующих блока, необходимые генераторы сигналов, дисплеи, осциллографы, сумматоры и другие элементы (см. рис. 5.2).

Требуется построить эту модель, используя библиотеки блоков пакета Simulink, и настроить параметры блоков в соответствии с условиями задачи. Проверить работу модели можно путем её многократного запуска при изменении времени окончания работы. Модифицируя состав модели и изменяя режим её работы, можно получить все требуемые характеристики объекта управления.

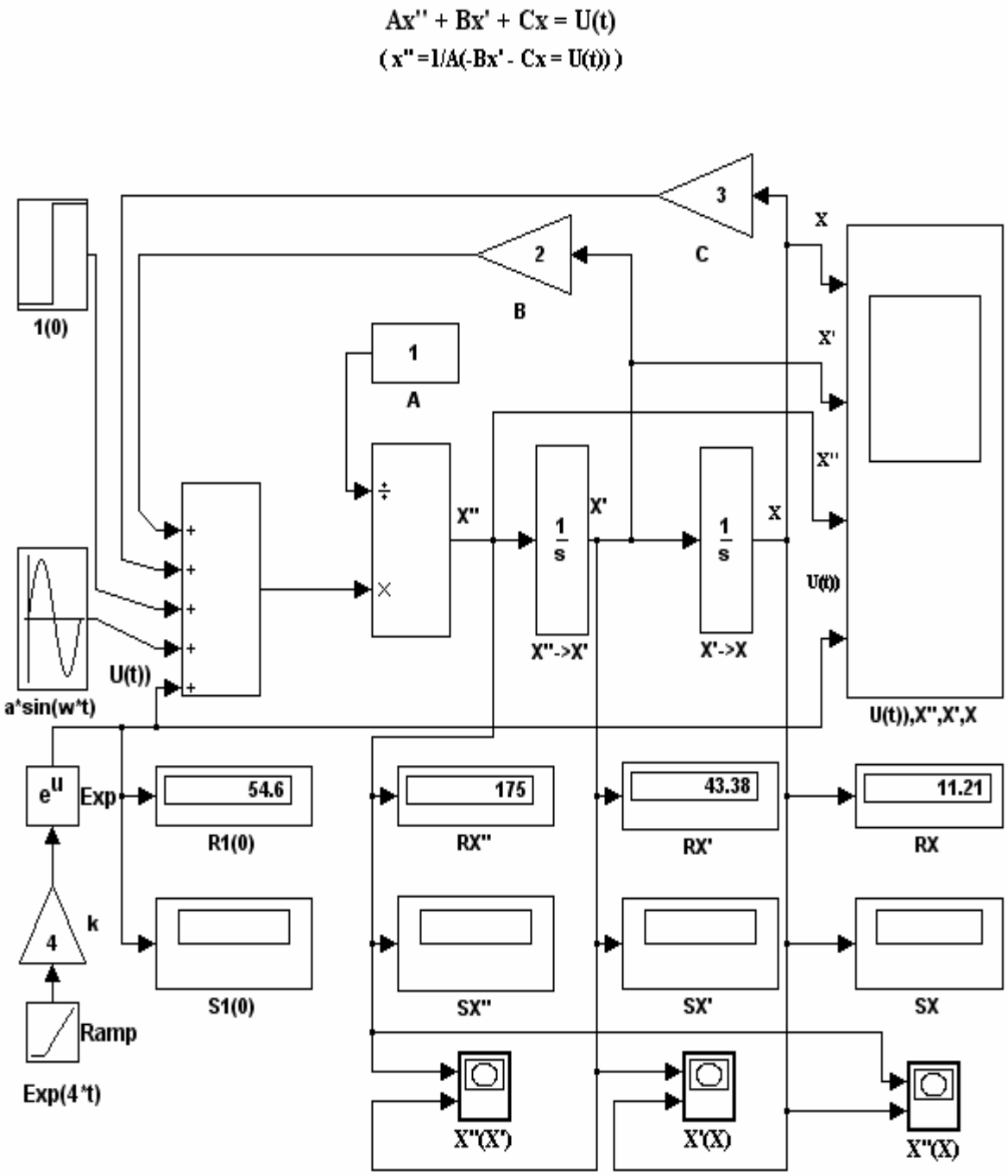

Рис. 3.1 Имитационная модель для линейной динамической системы 2-го порядка с постоянными коэффициентами

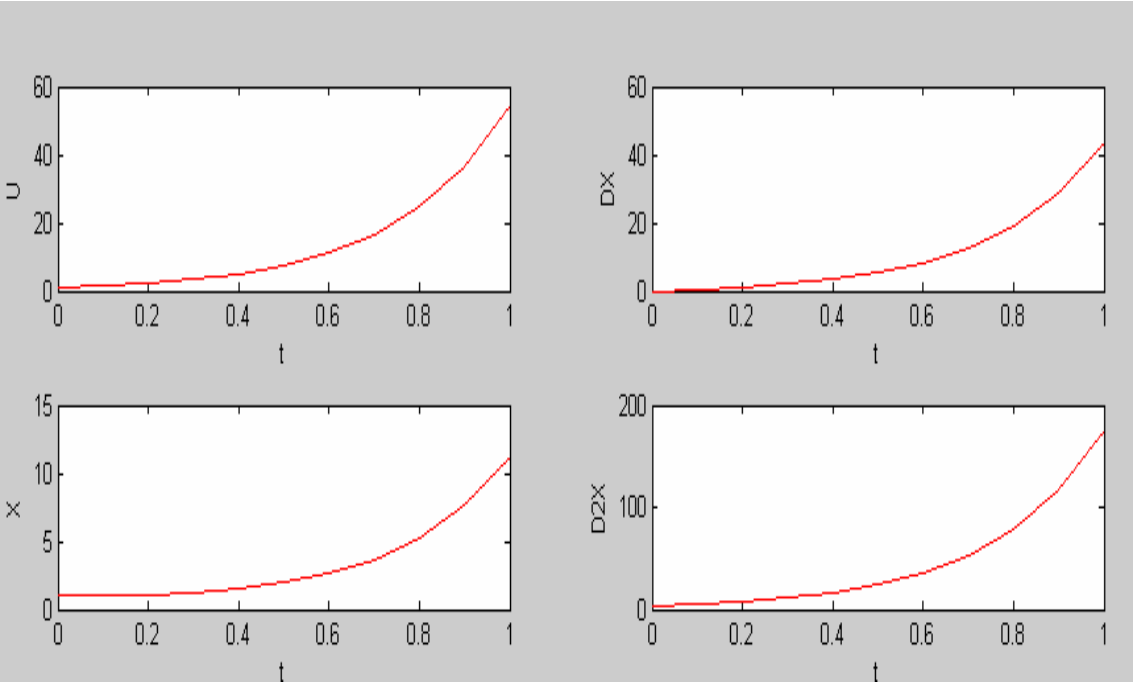

Рис. 3.2 Графики решения уравнения $x''-2x'-3x=e^{4t}$ с помощью преобразований Лапласа

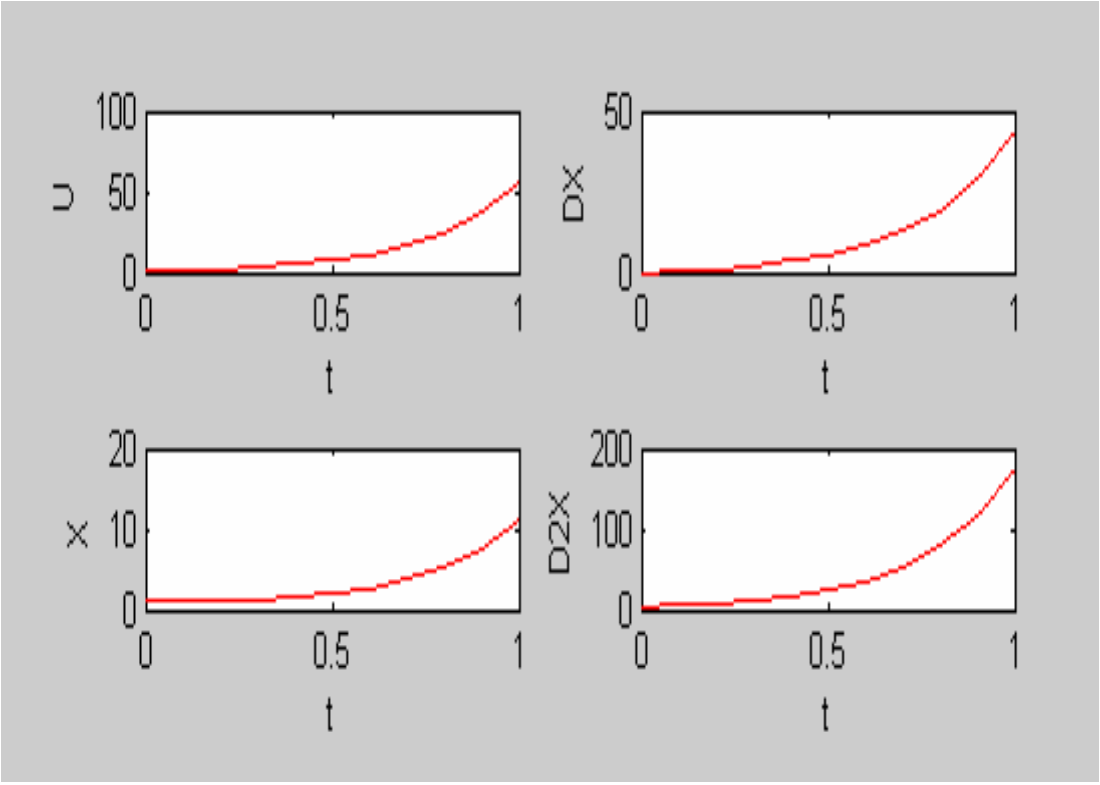

Рис. 3.3 Графики решения уравнения $x''-2x'-3x=e^{4t}$ с помощью символических вычислений

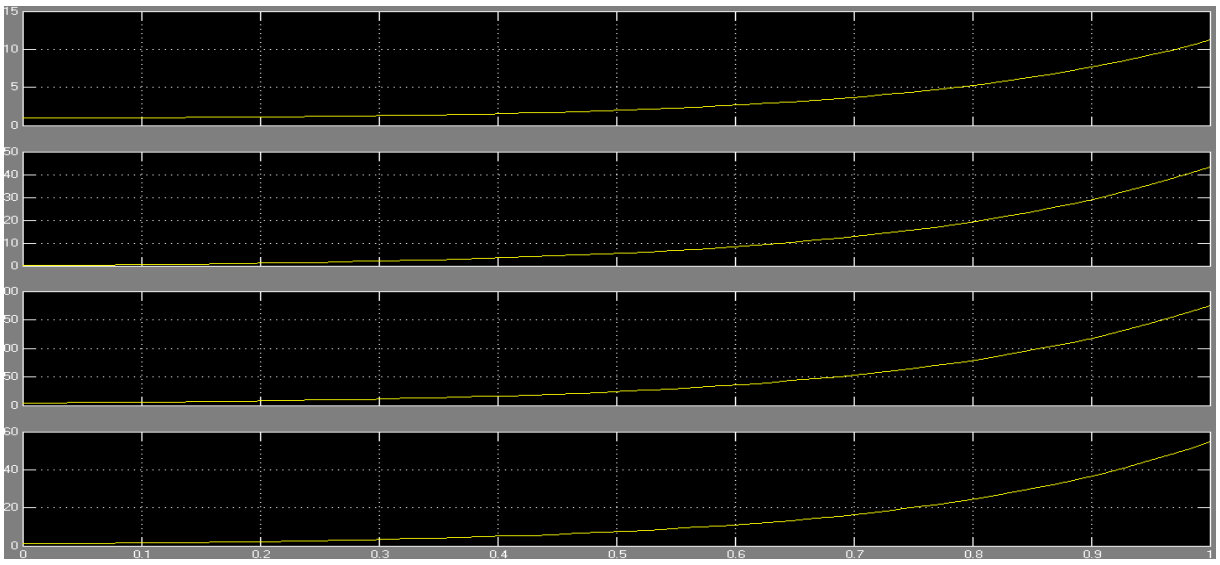

Рис. 3.4 Осциллограммы модели $x''-2x'-3x=e^{4t}$.

3.5 Верификация математических моделей

Верификацию аналитической, программной и имитационной моделей динамической системы, описываемой дифференциальным уравнением $x''-2x'-3x=e^{4t}$, произведем с помощью сопоставления переходных процессов, протекающих в этих моделях при заданном возмущающем воздействии в диапазоне изменения t от 0 до 1 с шагом 0.1.

Таблица 3.1

Результаты расчета и моделирования поведения динамической системы Ax''+Bx'+Cx=U(t)

Текущее время t	Возмущающее воздействие u(t)	Выходная величина x(t)		Скорость изменения выходной величины x'(t)	
		Расчетное значение	Модельное значение	Расчетное значение	Модельное значение
0.0	1.0000	1.0000	1.0000	0.0000	0.0000
0.1	1.4918	1.0222	1.0222	0.4696	0.4696
0.2	2.2255	1.1001	1.1001	1.1254	1.1254
0.3	3.3201	1.2567	1.2567	2.0634	2.0634
0.4	4.9530	1.5269	1.5269	3.4262	3.4262
0.5	7.3891	1.9630	1.9630	5.4260	5.4260
0.6	11.0232	2.6437	2.6437	8.3795	8.3795
0.7	16.4446	3.6862	3.6862	12.7584	12.7584
0.8	24.5325	5.2660	5.2660	19.2666	19.2666
0.9	36.5982	7.6449	7.6449	28.9533	28.9533
1.0	54.5982	11.2139	11.2139	43.3842	43.3842

3.6 Варианты заданий и порядок их выполнения

1. Используя операционное исчисление и функции прямого и обратного преобразования Лапласа laplace и ilaplace, решить уравнения 7-8 и системы уравнений 9-11 из табл. 3.1.

2. С помощью функции DSolveXDXD2X, приведенной в разделе 3.3, решить все дифференциальные уравнения из табл. 3.1 и построить графики найденных функций и их производных.

3. Построить для рассматриваемых динамических систем имитационные модели и произвести верификацию полученных математических моделей. Результаты верификации занести в таблицу.

4. Исследовать характер переходных процессов для динамических систем, описываемых линейным дифференциальным уравнением второго порядка Ax"+Bx'+Cx=U(t) с постоянными коэффициентами, задавая им различные значения.

3.7 Оформление отчета по результатам исследований

Для завершения лабораторной работы необходимо сгенерировать отчет в формате HTML, затем преобразовать его в формат RTF с помощью текстового редактора, включит в него теоретические результаты, отформатировать текст и графические объекты, записать на дискету и в электронном виде предъявить преподавателю. Обосновать достоверность полученных результатов.

Лабораторная работа № 4

ОПРЕДЕЛЕНИЕ ХАРАКТЕРИСТИК И МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ СИСТЕМ АВТОМАТИЧЕСКОГО УПРАВЛЕНИЯ

Цель работы: ознакомление с классификацией систем автоматического управления, их структурой и математическими определениями характеристик, а также исследование устойчивости и переходных процессов в этих системах с использованием математической системы MATLAB и программного пакета имитационного моделирования Simulink.

4.1 Постановка задач исследования

При поддержании регулярного хода производственного процесса обычно применяются три вида управления: жесткое управление, регулирование и настройка.

Жесткое управление включает в себя наиболее простые функции управления такие, как включение и выключение агрегатов, передачу импульсов по определенной программе и т.д. Все эти функции управления представляют собой фиксированные заранее воздействия или сигналы, не учитывающие дополнительные факторы, которые могут изменить процесс и действие которых зачастую предусмотреть совершенно невозможно. Такими факторами могут стать:

а)неточность поддержания заданных входных параметров;

б)неучтенные внешние воздействия на управляемый объект;

в)изменение состояния или характеристик управляемого объекта.

Структура системы жесткого автоматического управления представлена на рисунке 4.1. Управляющие устройство УУ оказывает управляющее воздействие U на управляемый объект О, чтобы обеспечить заданное значение выходного параметра X. Воздействие U не зависит от хода процесса в объекте О и от возможных отклонений параметра X.

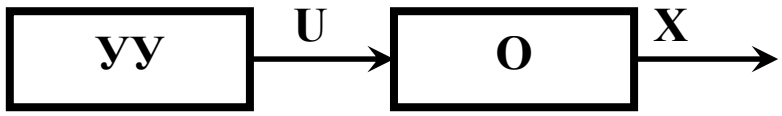

Рис. 4.1.

При регулировании управляющие сигналы, действующие на управляемый объект, заранее не определены. Их вид определяется конкретным ходом производственного процесса. Задача регулирования заключается в том, чтобы поддерживать требуемые значения показателей какого-либо процесса. Эти показатели называются регулируемыми величинами. Для каждой регулируемой величины X должно быть определено установленное или задающее значение X0. Оно может быть:

а)постоянной величиной, которую надо строго выдерживать с заданной точностью;

б)заранее известной функцией времени, которую надо реализовать также с заданной точностью;

в)заранее неизвестной функцией времени, которую надо определять в зависимости от хода производственного процесса и внешних факторов.

Операция установления и поддержания равенства

$$X=X0 \qquad (4.1)$$

называется регулированием. Это – основная задача системы автоматического регулирования (рис. 4.2).

В реальной системе это равенство реализуется с некоторой ошибкой или погрешностью x.

$$x=X0 - X \qquad (4.2)$$

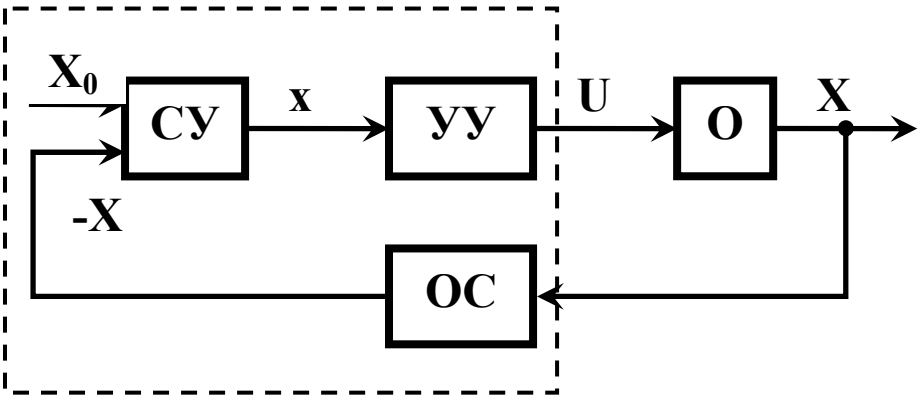

Рис. 4.2

Общий путь решения данной задачи состоит в применении обратной связи. Регулируемая величина X, поступающая от регулируемого объекта O, сравниваются с дающим воздействием X0. Определяется погрешность x, и по величине, знаку и тенденции её изменения управляющее устройство УУ автоматически определяет значение регулирующего воздействия U, поступающего на вход регулируемого объекта O. Блок обратной связи ОС, суммирующее устройство СУ и устройство выработки управляющего сигнала УУ объединяется в один блок управления системы автоматического регулирования, который связан с объектом управления прямой и обратной связью. Таким образом, система автоматического регулирования – это замкнутая система с обратной связью.

Иногда требуется регулировать несколько связанных величин X1, X2,..., Xn, являющихся параметрами процесса, происходящего в управляемом объекте. Эти величины должны равняться установочным значениям X01, X02,..., X0n соответственно. В этом случае задачей системы автоматического регулирования является установление векторного равенства

$$\mathbf{X=X0} \qquad (4.3)$$

с допустимой погрешностью

$$\mathbf{x=X0 - X} \qquad (4.4)$$

Регулирование нескольких величин принципиально не отличается от регулирования одной величины, хотя осуществляется часто более сложными

техническими приемами. Поэтому в лабораторных работах рассматривается лишь регулирование одной величины.

Третьим видом управления является настройка. Операции настройки могут заключаться в эпизодической или периодической настройки устройства, регулировке процессов на оптимум, а также в непрерывном корректировании параметров автоматической управляемой системы таким образом, чтобы обеспечить наилучший технологический режим при постоянно меняющихся характеристиках управляемого объекта и среды. Основной задачей при этом является поиск наилучшего управления. Системы, которые реализуют этот вид управления, называются самонастраивающимися, или системами автоматической настройки. Структурная схема таких систем помимо устройства управления УУ, блока обратной связи ОС и самого объекта управления О содержит ещё блок настройки УН, который связан с объектом управления О и устройством управления УУ. Назначение блока УН состоит в том, чтобы анализировать ход процесса регулирования и, изменяя характеристики УУ, добиваться лучшего регулирования.

При анализе и синтезе системы автоматического управления необходимо определить её структуру, руководствуясь следующими приложениями теории таких систем:

а)любую структуру можно представить в виде соединения между собой элементарных звеньев;

б)каждое элементарное звено обладает направленным действием – от входа к выходу звена;

в)имеется небольшое число различных типов элементарных звеньев;

г)единственной основой классификации элементарных звеньев является их динамические свойства, т.е. характер математической зависимости выходной величины x от входной величины x_0.

После определения структуры и математического описания системы автоматического управления производят расчеты процессов в её отдельных звеньях и в системе в целом. При этом решаются три группы взаимосвязанных задач:

1)Исследование погрешностей в системе и обеспечение малого значения их величин, в том числе и в установившихся режимах (установившиеся погрешности xs).

2)Исследование устойчивости системы и её обеспечение, с тем чтобы динамические погрешности xd(t) со временем стремились к нулю.

3)Исследование переходных процессов, т.е. функций xd(t) в устойчивой системе и обеспечение условий надлежащего качества управления.

4)Оптимизация характеристик и повышение качества работы системы.

На примере систем автоматического регулирования рассмотрим, как решаются эти задачи аналитически с помощью аппарата дифференциальных уравнений и операционного исчисления, а также с помощью построения имитационных моделей в системе Simulink.

4.2 Разработка аналитических моделей для систем автоматического регулирования

По виду изменения задающего воздействия или характеру паразитных возмущений системы автоматического регулирования делятся на три класса.

Если воздействие X0 постоянно, то система автоматического регулирования называется системой автоматической стабилизации.

Если воздействие X0 меняется по заранее заданной (запрограммированной) кривой, то система называется системой программного регулирования.

Наконец, если X0 является произвольной функцией времени, то система называется следящей системой.

В любом случае задачей системы автоматического регулирования является поддержание равенства между действительным X и предписанным X0 значениями регулируемой величины.

Однако в реальных системах эта задача выполняется с некоторой погрешностью

$$\Delta x = X_0 - X, \tag{4.5}$$

которая должна быть настолько мала, насколько это требуется условиями работы системы. Эта погрешность носит название ошибки системы регулирования. Лишь в некоторых частных случаях величина $\Delta x = 0$.

Изменение регулируемой величины вызывается возмущающими воздействиями ω, приложенными к системе, которые нарушают равенство между X и X0. С другой стороны, управляющее воздействие регулятора изменяет X таким образом, чтобы было соблюдено условие $X \approx X_0$.

Движение системы регулирования определяется указанными воздействиями обоих видов. Если обозначить через y регулируемую величину, то это движение в общем случае может быть описано уравнением

$$F_1(y, y', y'', ...) = F_2(w, w', ..., s, s', ...). \tag{4.6}$$

Очень часто это уравнение является линейным относительно y, ω и S, а также всех их производных при этом входящие в уравнение коэффициенты постоянны. Такие системы называются линейными. Существенным свойством линейных систем является принцип суперпозиции: если к линейной системе приложено одновременно несколько возмущающих воздействий, то их совместный эффект равен сумме эффектов, вызванных каждым воздействием в отдельности. Это позволяет записать рассматриваемое уравнение в следующем виде:

$$F_1(y, y', y'', ...) = F_{21}(s, s', ...) + F_{22}(w, w', ...). \tag{4.7}$$

Если ввести оператор $p = \frac{d}{dt}$, то уравнение примет операторную форму:

$$F_1(p) \cdot y = F_{21}(p) \cdot S + F_{22}(p) \cdot w, \tag{4.8}$$

где $F_1(p)$, $F_{21}(p)$, $F_{22}(p)$ в силу линейности уравнения и постоянства коэффициентов является многочленами p.

Задающее воздействие s прикладывается к системе через чувствительный элемент, а возмущающее воздействие w может быть приложено к любому звену системы и чаще всего к объекту регулирования. Если привести w к чувствительному элементу, решив уравнение

$$F_{21}(p)w_p = F_{22}(p)w, \tag{4.9}$$

где w_p - приведенное возмущающее воздействие, то уравнение движения системы примет следующий операторный вид:

$$F_1(p)\cdot y = F_{21}(p)(S + w_p). \tag{4.10}$$

Наконец, если ввести безразмерные величины

$$\lambda = \frac{y}{y_б} \text{ и } \varphi = \frac{s + \omega_p}{s_б},$$

где $y_б$ и $s_б$ - базисные значения величин y и s, то в безразмерной операторной форме с приведенным возмущающим воздействием уравнение движения линейной системы автоматического регулирования примет следующий окончательный вид:

$$D(p)\lambda = U(p)\varphi \tag{4.11}$$

в этом уравнении $D(p)$ и $U(p)$ - операторные полиномы.

Если система автоматического регулирования имеет звенья ненаправленного действия (одновременно от входа к выходу и от выхода ко входу) или является нелинейной, то она преобразуется следующим образом:

а)звено ненаправленного действия заменяется соответствующим комплексом направленных звеньев;

б)уравнения нелинейных звеньев заменяется приближенными линейными дифференциальными уравнениями с постоянными коэффициентами, используя разложения функций в ряд Тейлора в окрестности рассматриваемой точки движения системы и отбрасывая члены второго и высшего порядка, т.е. производится линеаризация нелинейной системы, при этом допустимость такого преобразования устанавливается теоремой А.М. Ляпунова об устойчивости линеаризованных систем.

Таким образом, всякую системы автоматического регулирования можно разложить на простейшие звенья, движения которых описываются дифференциальными уравнениями не выше второго порядка, линейными относительно неизвестных функций и их производных и имеющими постоянные коэффициенты. Физические величины, от которых зависят эти коэффициенты, называются параметрами. Составленная из таких простейших звеньев схема системы называется структурной схемой в отличие от функциональной, на которой указываются функциональные устройства и приборы системы.

Переходной процесс всякого звена, являющегося частным случаем линейной системы, может быть описан в общем случае дифференциальным уравнением того же вида, что и для системы в целом:

$$D(p)\lambda(t) = U(p)\varphi(t), \tag{4.12}$$

где $\varphi(t)$ - входная функция звена, $\lambda(t)$ - выходная функция звена, $D(p)$ и $U(p)$ - дифференциальные операторные полиномы выхода и входа звена;

$p = \frac{d}{dt}$ - символ дифференцирования.

Это уравнение легко решается с помощью преобразования Лапласа для заданных начальных условий. Исследование переходного процесса производится для наиболее характерных видов возмущающих воздействий: единичной толчкообразной функции 1(t), функции единичного импульса 1-го порядка 1`(t), функции единичного импульса 2-го порядка 1``(t), периодической функции возмущающего воздействия и др. При этом выявляется устойчивость звена и характер изменения погрешности, а также её величина. Подавая на вход синусоидальное возмущающее воздействие, получают амплитудно-фазовую характеристику выхода, что позволяет выявить влияние отдельных параметров звена на переходной процесс и значительно упрощает расчеты. Для анализа звена используется также передаточная функция звена

$$K(p) = \frac{U(p)}{D(p)}, \tag{4.13}$$

где $p = \delta + i\tau$ - комплексное число;

U(p) и D(p) – изображения соответственно выходной и входной функций, полученные с помощью преобразования по Лапласу при нулевых начальных условиях.

При определении передаточной функции всей системы используются следующие свойства этой функции:

а)передаточная функция последовательно соединенных звеньев равна произведению передаточных функций отдельных звеньев;

б)передаточная функция параллельно соединенных звеньев равна сумме передаточных функций отдельных звеньев.

Для теоретического расчета амплитудно-фазовой характеристики звена используется частотная функция

$$K(i\omega) = \frac{U(i\omega)}{D(i\omega)}, \tag{4.14}$$

где U(iω) и D(iω) – выходная и входная величины звена, выраженные в символической форме, если входная величина совершает синусоидальные колебания. Формально K(iω) получается из передаточной функции заменой p на iw.

Простейшие звенья могут иметь следующие варианты дифференциального операторного полинома выхода D(iω):

$$T_2^2 p^2 \pm T_1 p \pm 1; \quad T_2^2 p^2 \pm 1; \quad T_2^2 p^2 \pm T_1 p; \quad T_1\, p \pm 1; \quad p^2; \quad p. \tag{4.15}$$

Звено с оператором p^2 может быть разложено на два последовательных звена типа p. Аналогично звено $T_2^2 p^2 - T_1 p$ может быть представлено как последовательная комбинация из двух звеньев $T_1\, p$ и $\frac{T_2^2}{T_1} p + 1.$

Звенья, операторы которых имеют положительный свободный член, равный 1, называют статическими. Если он равен 0, то звено называется астатическим. Если он равен -1, то звено имеет отрицательный статизм.

Классификация простейших звеньев может производиться по их передаточным функциям K(iω). Перечислим типовые простейшие звенья:

1. Усилительное безынерционное звено:

$$\delta\lambda(t) = \varphi(t). \tag{4.16}$$

2. Апериодическое звено 1-го порядка:

$$(T_1 p + \delta)\lambda(t) = \varphi(t) \qquad при \qquad \delta > 0 \tag{4.17}$$

3. Интегрирующие звено.

$$T_1 p\lambda(t) = \varphi(t) \tag{4.18}$$

4. Апериодическое звено 2-го порядка:

$$(T_2^2 p^2 + T_1 p + \delta)\lambda(t) = \varphi(t) \qquad при \qquad \frac{1}{4\delta}\left(\frac{T_1}{T_2}\right)^2 \geq 1. \tag{4.19}$$

5. Колебательное звено:

$$(T_2^2 p^2 + T_1 p + \delta)\lambda(t) = \varphi(t) \tag{4.20}$$

$$при \qquad \frac{T_1}{2T_2\sqrt{\delta}} < 1 \qquad или \quad \left(\frac{T_1}{T_2}\right)^2 < 4\delta.$$

6. Астатическое звено 2-го порядка:

$$(T_2^2 p^2 + T_1 p)\lambda(t) = \varphi(t) \tag{4.21}$$

7. Изодромное дифференцирующее звено со статизмом:

$$(T_1 p + \delta)\lambda(t) = (\hat{T}_1 p + 1)\varphi(t) \tag{4.22}$$

8. Изодромное дифференцирующее звено без статизма:

$$(T_1 p + \delta)\lambda(t) = \hat{T}_1 p\varphi(t). \tag{4.23}$$

9. Дифференцирующее звено:

$$\lambda(t) = \hat{T}_1 p\varphi(t). \tag{4.24}$$

10. Звенья с отрицательным статизмом:

$$(T_1 - \delta)\lambda(t) = \varphi(t);\ (T_2^2 p^2 + T_1 p - \delta)\lambda(t) = \varphi(t);\ (T_2^2 p^2 - \delta)\lambda(t) = \varphi(t): \tag{4.25}$$

В табл. 4.1 приведены передаточные функции основных простейших звеньев. Знание передаточной функции звена позволяет по изображению возмущающего воздействия легко определить изображение функции регулируемого параметра при движении системы, а переходя к оригиналу, и действительное поведение этого параметра. С помощью передаточных функций отдельных звеньев можно также вычислить передаточную функцию любой системы автоматического регулирования, так как при последовательном соединении звеньев их передаточные функции умножаются, а при параллельном - суммируются. Если в системе ряд звеньев с передаточной функцией $K_1(p)$ охвачен обратной связью с передаточной функцией $K_2(p)$, то передаточная функция такого комплекса звеньев K(p) будет вычисляться по формуле:

$$K(p) = \frac{K_1(p)}{1 + K_1(p) \cdot K_2(p)}. \tag{4.26}$$

Таблица 4.1

Основные характеристики простейших звеньев

№ п/п	Передаточная функция K*(p)*	Переходная проводимость $\Lambda(t)$	Функция веса W*(t)*
1.	$\frac{1}{\delta}$	$\frac{1}{\delta}1(t)$	$\frac{1}{\delta}1'(t)$
2.	$\frac{1}{T_1 p+\delta}$	$\frac{1}{\delta}(1-e^{-\frac{t\delta}{T_1}})$	$\frac{1}{T_1}e^{-\frac{t\delta}{T_1}}$
3.	$\frac{1}{T_1 p}$	$\frac{1}{T_1}t$	$\frac{1}{T_1}$
4.	$\frac{1}{T_2^2 p^2+T_1 p+\delta}$	$\frac{1}{\delta}+A_1 e^{-t\alpha}-A_2 e^{-t\beta}$	$A_2\beta e^{-t\beta}-A_1\alpha e^{-t\alpha}$
5.	$\frac{1}{T_2^2 p^2+T_1 p+\delta}$	$\frac{1}{\delta}+Be^{-t\nu}\sin(w_c t-\psi_c)$	$\frac{e^{-t\nu}}{w_c T_2^2}\sin w_c t$
6.	$\frac{1}{p(T_2^2 p+T_1)}$	$-A(1-e^{-t\alpha})+Bt$	$B-A\alpha e^{-t\alpha}$
7.	$\frac{\hat{T}_1 p+1}{T_1 p+\delta}$	$A+Be^{-t\alpha}$	$\frac{e^{-t\alpha}}{T_1}\left(1-\delta\frac{\hat{T}_1}{T_1}\right)$
8.	$\frac{\hat{T}_1 p}{T_1 p+1}$	$\frac{\hat{T}_1}{T_1}e^{-t\alpha}$	$-\frac{\hat{T}_1}{T_1^2}e^{-t\alpha}$
9.	$\hat{T}_1 p$	$\hat{T}_1\cdot 1'(t)$	$\hat{T}_1\cdot 1''(t)$

В частности если вся система охвачена обратной связью и таким образом является замкнутой, то для замкнутой системы передаточная функция $K_3(p)$ будет вычисляться по формуле:

$$K_3(p) = \frac{K(p)}{1 + K(p)}, \tag{4.27}$$

где K(p) – передаточная функция разомкнутой системы, а коэффициент обратной связи равен 1.

Анализ системы автоматического регулирования может быть произведен на основе частотной функции, которая получается из передаточной замены p на iw, где w – круговая частота. Разлагается возмущающую функцию в ряд Фурье, можно получить выходную функцию как сумму входных гармоник, умноженных на |K(iw)| и сдвинутых по фазе на угол

$$\psi = arctg \frac{\mathrm{Im}(K(i\omega))}{\mathrm{Re}(K(i\omega))}. \tag{4.28}$$

Построив далее амплитудно-фазовую характеристику, можно оценить устойчивость системы по критерию Найквиста – Михайлова, который формулируется следующим образом: замкнутая система автоматического регулирования устойчива, если её амплитудно-фазовая характеристика в разомкнутом состоянии при изменении частоты ω от -∞ до +∞ не охватывает точку с координатами (-1, i=0).

В табл. 4.1 приведены также функция проводимости $\Lambda(t)$ для каждого звена и функция веса $W(t)$, являющаяся производной от функции $\Lambda(t)$. По смыслу переходная проводимость $\Lambda(t)$ является реакцией звена или системы на единичную толчкообразную функцию 1(t), поступившую на вход:

$$1(t) = \begin{cases} 0 & при \quad t < 0; \\ 1 & при \quad t \geq 0. \end{cases} \tag{4.29}$$

Зная $\Lambda(t)$, можно с помощью интеграла Дюамеля рассчитать поведение регулируемой величины $\lambda(t)$ при любом входном воздействии $\varphi(t)$:

$$\lambda(t) = \frac{d}{dt} \int_0^t \varphi(\tau) \Lambda(t - \tau) d\tau. \tag{4.30}$$

Функция веса $W(t)$ является реакцией звена или системы на единичное импульсное воздействие:

$$\delta(t) = \begin{cases} \dfrac{1}{\tau} & при \quad 0 < t < \tau; \\ 0 & при \quad \mathrm{t} > \tau \text{ и } \mathrm{t} < 0. \end{cases} \tag{4.31}$$

С использованием этой функции регулируемая величина рассчитывается по формуле:

$$\lambda(t) = \int_0^t \varphi(\tau) W(t - \tau) d\tau. \tag{4.32}$$

Её лапласовское изображение является передаточной функцией рассматриваемого звена K(p).

Таблица 4.2

Варианты трехзвенных соединений

первое звено \ второе звено	1. Усилительное безинерционное звено	4. Апериодическое звено первого порядка	3. Интегрирующее звено	4. Апериодическое звено второго порядка	5. Колебательное звено	6. Астатическое звено второго порядка	7. Изодромное дифференцирующее со статизмом	8. Изодромное дифференцирующее без статизма	9. Дифференцирующее звено
1. Усилительное безынерционное звено	0-9	0-9	0-9	0-9	0-9	0-9	0-9	0-9	0-9
4. Апериодическое звено первого порядка	0-9	0-9	0-9	0-9	0-9	0-9	0-9	0-9	0-9
3. Интегрирующее звено	0-9	0-9	0-9	0-9	0-9	0-9	0-9	0-9	0-9
4. Апериодическое звено второго порядка	0-9	0-9	0-9	0-9	0-9	0-9	0-9	0-9	0-9
5. Колебательное звено	0-9	0-9	0-9	0-9	0-9	0-9	0-9	0-9	0-9
6. Астатическое звено второго порядка	0-9	0-9	0-9	0-9	0-9	0-9	0-9	0-9	0-9
7. Изодромное дифференцирующее со статизмом	0-9	0-9	0-9	0-9	0-9	0-9	0-9	0-9	0-9
8. Изодромное дифференцирующее без статизма	0-9	0-9	0-9	0-9	0-9	0-9	0-9	0-9	0-9
9. Дифференцирующее звено	0-9	0-9	0-9	0-9	0-9	0-9	0-9	0-9	0-9

Известно, что интеграл Дюмеля может быт записан и в другой форме:

$$\lambda(t) = \varphi(0)\Lambda(t) + \int_0^t \varphi'(\tau)\Lambda(t-\tau)\partial\tau \tag{4.33}$$

Дифференциальное уравнение движения линеаризованной системы автоматического регулирования в безразмерной форме было выведено ранее (см. формулу 4.11). В операторном виде оно имеет вид:

$$D(p)\lambda = U(p)\varphi \tag{4.34}$$

где λ – регулируемая величина;

φ – возмущающее воздействие;

$D(p)$ и $U(p)$ – выходной и входной операторные полиномы;

$p\dfrac{d}{dt}$ – оператор дифференцирования;

λ и φ – функции времени.

Таким образом, чтобы система автоматического регулирования была устойчивой, необходимо, чтобы корни характеристического уравнения были вещественными отрицательными или комплексными сопряженными с отрицательной вещественной частью.

Для оценки качества регулирования необходимо знать величину погрешности регулирования, которая определяется как разность

$$\Delta\lambda = \lambda_p - \lambda \tag{4.35}$$

где λ_p-заданное значение в безразмерной форме;

λ-действительное значение регулируемой величины в безразмерной форме;

Заменяя λ выражением $\lambda = \lambda_{пер} + \lambda_b$ можно представить значение погрешности $\Delta\lambda$ в другом виде:

$$\Delta\lambda = (\lambda_p - \lambda_b) - \lambda_{пер} \tag{4.36}$$

или

$$\Delta\lambda = \Delta\lambda_{пер} + \Delta\lambda_b \tag{4.37}$$

где $\Delta\lambda = \lambda_p - \lambda_b$ - установившаяся погрешность регулирования;

$\Delta\lambda_{пер} = -\lambda_{пер}$ - переходная или динамическая погрешность.

При постоянстве возмущающих сил погрешность $\Delta\lambda_b$ характеризует вид системы регулирования

а) если $\Delta\lambda_b = 0$, то система регулирования называется астатической;

б) если $\Delta\lambda_b \neq 0$,то система регулирования называется статической.

Для систем регулирования величина $|\Delta\lambda_{пер}|$ определяет дополнительную максимальную погрешность, которая накладывается на величину погрешности $\Delta\lambda_b$ при переходном процессе. Если $|\Delta\lambda_{пер}| \to \infty$ при $t \to \infty$, то система называется устойчивой.

При изменении возмущающих сил всякая система автоматического регулирования отклоняется от заданного ей закона движения на величину $\Delta\lambda$. Задачей регулятора является возвращение системы к заданному движению.

Под воздействием возмущающих сил с одной стороны, и восстанавливающего действия регулятора, с другой, возникает переходной процесс в системе, которой может закончиться одним из следующих способов:

1.Система регулирования не может восстановить требуемого движения после его нарушения возмущающим воздействиями, и действительное движение будет всё дальше удаляться от требуемого. Такой переходной процесс называется расходящимся, а система регулирования – неустойчивой.

2.Система регулирования после нарушения движения с течением времени возвращается к заданному движению с точностью, отвечающей статической погрешности системы. Здесь возмущенное движение системы асимптотически приближается к заданному. Система регулирования называется устойчивой.

3.Система регулирования получает дополнительное установившееся периодическое движение. Такой переходной процесс называется незатухающим колебательным, а система – находящаяся на границе асимптотической устойчивости, или имеющая устойчивые автоколебания.

Имеется несколько методов для определения устойчивости как линейных, так и нелинейных систем регулирования (см. выше критерий Найквиста – Михайлова).

4.3 Разработка аналитических моделей для простейших звеньев второго порядка

В качестве примера рассмотрим простейшее звено наиболее общего вида, описываемое следующим дифференциальным уравнением второго порядка:

$$T_2^2 \frac{d^2\lambda}{dt^2} + T_1 \frac{d\ \lambda}{dt} + \delta\lambda(t) = \varphi(t) \tag{4.38}$$

или в операторной форме

$$(T_2^2 p^2 + T_1\ p + \delta)\lambda(t) = \varphi(t) \tag{4.39}$$

Величина $\rho = \dfrac{T_1}{2T_2\sqrt{\delta}}$ называется коэффициентом демпфирования. Когда $\rho^2 \geq 1$ или $\rho \geq 1$, корни характеристического уравнения

$$T_2^2 p^2 + T_1\ p + \delta = 0 \tag{4.40}$$

являются вещественными и отрицательными, и такое звено называется апериодическим звеном 2-го порядка. Когда $\rho^2 < 1$, корни являются комплексными сопряженными, а звено называется колебательным.

Из уравнения (4.39) следует, что для обоих звеньев передаточная и частотная функции имеют вид:

$$K(p) = \frac{1}{T_2^2 p^2 + T_1\ p + \delta}\ ; \tag{4.41}$$

$$K(i\omega) = \frac{1}{\delta - T_2^2\omega^2 + iT_1\ \omega} \quad . \tag{4.42}$$

Функции переходной проводимости звеньев находятся как решения дифференциального уравнения

$$T_2^2 \frac{d^2\lambda}{dt^2} + T_1 \ \frac{d\ \lambda}{dt} + \delta\lambda(t) = 1(t) \tag{4.43}$$

Используя прямое и обратное преобразование Лапласа, получаем:

$$\Lambda(t) = \frac{1}{\delta} + A_1 e^{-\alpha t} - A_2 e^{-\beta t} \tag{4.44}$$

для апериодического звена и

$$\Lambda(t) = \frac{1}{\delta} + Be^{-\nu t}\sin(w_c t - \psi_c) \tag{4.45}$$

для колебательного звена.

Функции веса звеньев получаются путем дифференцирования соответствующих функций переходной проводимости и имеют вид:

$$W(t) = A_2\beta e^{-\beta t} - A_1\alpha e^{-\alpha t} \tag{4.46}$$

для апериодического звена и

$$W(t) = \frac{e^{-\nu t}}{\omega_c T_2^2}\sin\omega_c t \tag{4.47}$$

для колебательного звена.

В этих формулах были использованы следующие обозначения:

$$\alpha = \frac{T_1 + 2T_2\sqrt{\delta(\rho^2 - 1)}}{2T_2^2}\ ; \tag{4.48}$$

$$\beta = \frac{T_1 - 2T_2\sqrt{\delta(\rho^2 - 1)}}{2T_2^2}\ ; \tag{4.49}$$

$$\mathrm{A}_1 = \frac{T_2\beta}{2\delta\sqrt{\delta(\rho^2 - 1)}}\ ; \tag{4.50}$$

$$\mathrm{A}_2 = \frac{T_2\alpha}{2\delta\sqrt{\delta(\rho^2 - 1)}}\ ; \tag{4.51}$$

$$\nu = \frac{T_1}{2T_2^2}\ ; \tag{4.52}$$

$$\omega_c = \mu = \frac{\sqrt{\delta(1 - \rho^2)}}{T_2}\ ; \tag{4.53}$$

$$\psi_c = arctg\frac{\omega_c}{-\nu}\ ; \tag{4.54}$$

$$B = \frac{1}{\delta\sin\psi_c}\ . \tag{4.55}$$

В зависимости от назначения коэффициента демпфирования ρ различают следующие разновидности рассматриваемых звеньев:

1) $\rho > 1$ – сильно-демпфированное звено. Переходная составляющая $\lambda_{пер}$ имеет апериодический характер при любом возмущающем воздействии. Модуль частотной функции монотонно убывает с ростом частоты. Наибольшее значение модуля равно $\frac{1}{\delta}$ при $\omega = 0$. Явление резонанса здесь исключено.

2) $\rho = 1$ – апериодическое предельно-демпфированное звено. Реакция звена имеет апериодическую составляющую при любом возмущении. Это предельный случай. При малейшем уменьшении отношения $\frac{T_1}{T_2}$ или увеличении δ характер переходного процесса изменяется и становится колебательным.

3) $1 > \rho > \frac{\sqrt{2}}{2}$ – нормально-демпфированное звено. Переходная составляющая $\lambda_{пер}$ реакции представляет собой функцию, имеющую колебательный затухающий характер при любом возмущающем воздействии. Колебания происходят с частотой

$$\omega_c = \frac{\sqrt{\delta(1-\rho^2)}}{T_2}, \qquad (4.56)$$

называемой собственной частотой колебаний, которая не зависит от частоты возмущающих воздействий. Явление резонанса не имеет места. Модуль частотной функции с ростом ω монотонно убывает, начиная с величины $\frac{1}{\delta}$ при $\omega = 0$.

4) $\rho = \frac{\sqrt{2}}{2}$ – критически демпфированное звено. Модуль частотной функции имеет экстремум, равный $\frac{1}{\delta}$ при $\omega = 0$. В остальном звено ведет себя как нормально-демпфированное.

5) $\frac{\sqrt{2}}{2} > \rho > 0$ – слабо-демпфированное звено. Модуль частотной функции в диапазоне частот от 0 до ∞ имеет максимум, равный $\frac{T_2}{T_1\sqrt{\delta(1-\rho^2)}}$. Имеет место резонанс при $\omega_b^2 = \omega_c^2 - \nu^2$ или при $\omega_b = \frac{\sqrt{\delta}}{T_2}$. При малых значениях отношения $\frac{T_1}{T_2}$, а также малых значениях δ модуль частотной функции может принять очень большое значение.

6) $\rho = 0$ – недемпфированное звено. Этот случай соответствует значению $T_1 = 0$. Переходной процесс имеет незатухающий колебательный характер с частотой колебаний $\omega_c = \frac{\sqrt{\delta}}{T_2}$. При равенстве $\omega_b = \omega_c$ наступает явление резонанса.

Таким образом, звено демпфировано, если в его математическом описании имеется составляющая, пропорциональная первой производной $\lambda(t)$. Составляющая, пропорциональная второй производной, характеризует инерционные свойства звена. Для уравнения (4.38) эти составляющие определяются соответственно коэффициентами T_1 и T_2^2.

4.4 Программная реализация аналитических моделей

М-функция для построения модуля, фазы и годографов частотной функции $K(i\omega)$ на комплексной плоскости и в пространстве имеет вид:

```
function var= RKiw(A,B,C)
%
%-- ГРАФИКИ ЧАТОТНОЙ ФУНКЦИИ K(iw) :
%
%-- A = T2^2;
%-- B = T1 ;
%-- C = δ;
%-- R = ρ = T1/(2*T2*sqrt(δ)) ;
%
%-- 1.Диапазон частот:
%
    w=0.01:0.001:10;
%
%-- 2.Коэффициент демпфирования:
%
    R=B/(2*sqrt(A)*sqrt(C));
%
%-- 3.Вид колебательного звена:
%
    if (R>1)
      disp('Сильно-демпфированное звено')
    end
    if (R==1)
      disp('Апериодическое предельно-демпфированное звено')
    end
    if (1>R && R>sqrt(2)./2)
      disp('Нормально-демпфированное звено')
    end
    if (R==sqrt(2)./2)
      disp('Критически-демпфированное звено')
    end
```

```
    if (R<sqrt(2)./2 && R>0)
      disp('Слабо-демпфированное звено')
    end
    if (R==0)
      disp('Недемпфированное звено')
    end
%
%-- 4.Частотная функция:
%
    Kiw=1./(A.*(i.*w).^2+B.*i.*w+C);
%
%-- 5.Модуль частотной функции:
%
    subplot(2,2,1)
    plot(w,abs(Kiw),'r')
    xlabel('w')
    ylabel('abs(Kiw)')
%
%-- 6.Фаза частотной функции:
%
    subplot(2,2,2)
    plot(w,angle(Kiw),'r')
    xlabel('w')
    ylabel('angle(Kiw)')
%
%-- 7.Годограф на плоскости:
%
    subplot(2,2,3)
    plot(real(Kiw),imag(Kiw),'r')
    xlabel('real(Kiw)')
    ylabel('image(Kiw)')
%
%-- 8.Годограф в трехмерном пространстве(комета):
%
    subplot(2,2,4)
    comet3(real(Kiw),imag(Kiw),w)
    xlabel('real(Kiw)')
    ylabel('image(Kiw)')
    zlabel('w')
    pause
%
%-- 9.Конец функции RKiw.
```

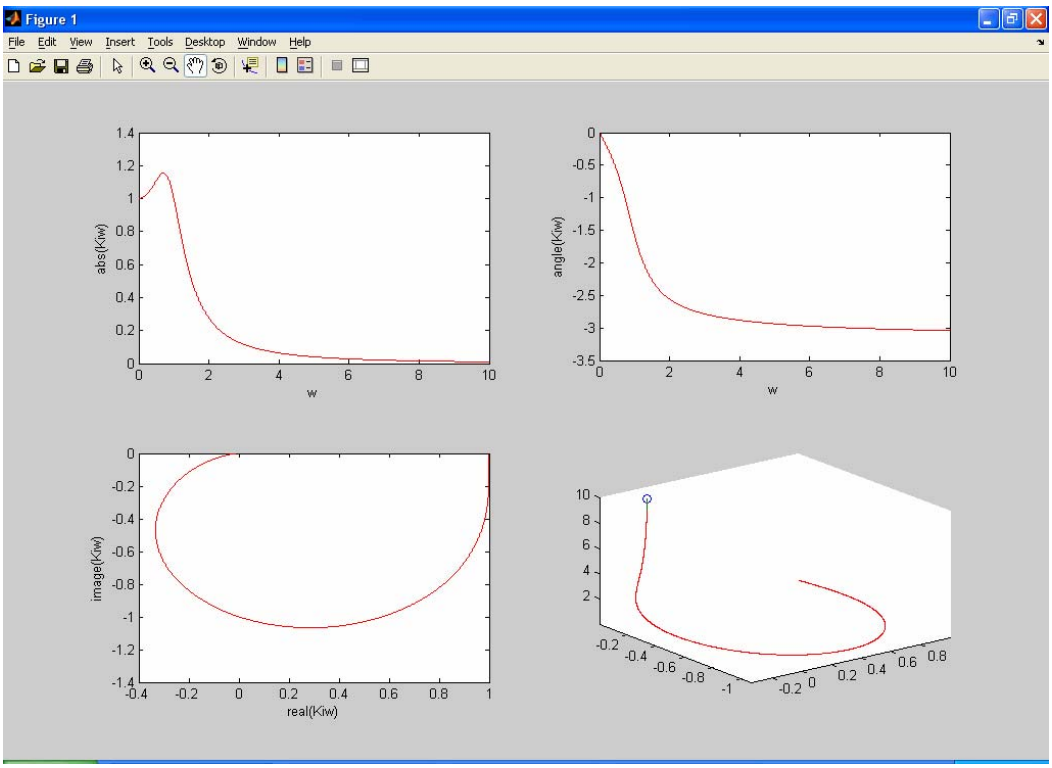

Рис. 4.3 Частотные характеристики простейшего звена второго порядка при A=1,B=1,C=1

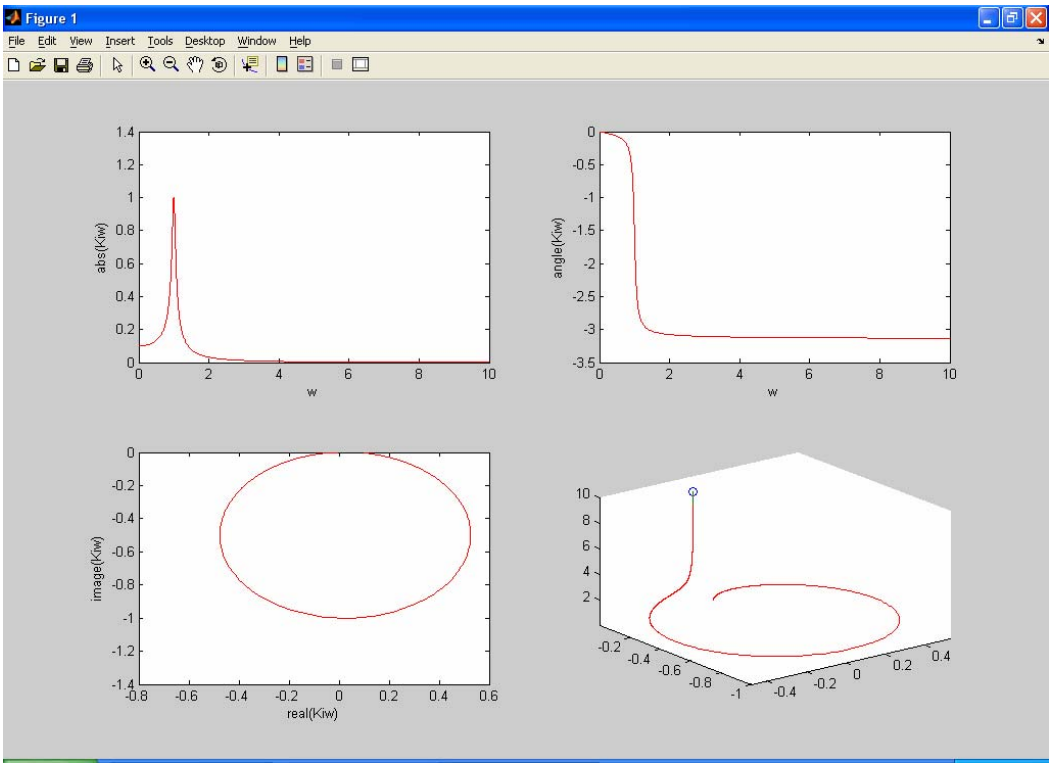

Рис. 4.4 Частотные характеристики простейшего звена второго порядка при A=10,B=1,C=10

М-функция для построения графиков переходного процесса имеет вид:

```
function [Lambda, Weight,XSin]= LWSin(A,B,C,a0,w)

%
%-- ГРАФИКИ ПЕРЕХОДНЫХ ФУНКЦИЙ: Lambda(t), Weight(t) и XSin(t)
%

%
%-- 1.Коэффициент демпфирования:
%

    R = B/(2*sqrt(A)*sqrt(C));

%
%-- 2.Вычисляемые промежуточные параметры:
%

    Alpha  = (B + 2*(sqrt(a*c*(R^2-1))))/2*A
    Beta   = (B - 2*(sqrt(a*c*(R^2-1))))/2*A
    A1     = ((sqrt(A))*Beta)/2*C(sqrt(C*(R^2-1))
    A2     = ((sqrt(A))*Alpha)/2*C(sqrt(C*(R^2-1))

%
%-- 3.Вид демпфированного  звена:
%

    if (R>1)
      disp('Сильно-демпфированное звено')
    end
    if (R==1)
      disp('Апериодическое предельно-демпфированное звено')
    end
    if (1>R && R>sqrt(2)./2)
      disp('Нормально-демпфированное звено')
    end
    if (R==sqrt(2)./2)
      disp('Критически-демпфированное звено')
    end
    if (R<sqrt(2)./2 && R>0)
      disp('Слабо-демпфированное звено')
    end
    if (R==0)
      disp('Недемпфированное звено')
    end
```

```
%
%-- 4.Временной интервал:
%

    t = 0.000:0.001:10.00;

%
%-- 5.Функция переходной проводимости:
%
    if R >= 1   %-- Апериодическое звено:
      Lambda  = 1./C-A2.*exp(-Beta.*t)+A1.*exp(-Alpha.*t);
    else        %-- Колебательное звено:
      Lambda  = -1./2.*exp(-1./2.*(B-(B.^2-4.*C.*A).^(1./2))./...
                   A.*t).*(B+(B.^2-4.*C.*A).^(1./2))./(B.^2-...
                   4.*C.*A).^(1./2)./C+1./2.*exp(-1./2.*(B+(B.^2-...
                   4.*C.*A).^(1./2))./A.*t).*(B-(B.^2-4.*C.*A).^...
                   (1./2))./(B.^2-4.*C.*A).^(1./2)./C+1./C;
    end

    subplot(4,2,1)
    plot(t,Lambda,'r')
    xlabel('t')
    ylabel('Lambda')

%
%-- 6.Функция веса:
%

    if R >= 1   %-- Апериодическое звено:
      Weight  = A2.*Beta.*exp(-Beta.*t)-A1.*Alpha.*exp(-Alpha.*t);
    else        %-- Колебательное звено:
      Weight  = 1./4.*(B-(B.^2-4.*C.*A).^(1./2))./A.*exp(-1./2.*...
                  (B-(B.^2-4.*C.*A).^(1./2))./A.*t).*(B+(B.^2-...
                  4.*C.*A).^(1./2))./(B.^2-4.*C.*A).^(1./2)./C-...
                  1./4.*(B+(B.^2-4.*C.*A).^(1./2))./A.*...
                  exp(-1./2.*(B+(B.^2-4.*C.*A).^(1./2))./A.*t).*(B-...
                  (B.^2-4.*C.*A).^(1./2))./(B.^2-4.*C.*A).^(1./2)./C;
    end

    subplot(4,2,2)
    plot(t,Weight,'r')
    xlabel('t')
    ylabel('Weight')
```

```
%
%-- 7.Реакция звеньев на синусоидальные воздействия:
%

      XSin  =

            1./2.*exp(-1./2.*(B-(B.^2-4.*C.*A).^(1./2))...
            ./A.*t).*a0.*w.*(B.^2+B.*(B.^2-4.*C.*A).^(1./2)-...
            2.*C.*A+2.*w.^2.*A.^2)./(B.^2-...
            4.*C.*A).^(1./2)./(B.^2.*w.^2+C.^2-...
            2.*C.*A.*w.^2+w.^4.*A.^2)-...
            1./2.*exp(-1./2.*(B+(B.^2-...
            4.*C.*A).^(1./2))./A.*t).*...
            (B.^2-B.*(B.^2-4.*C.*A).^(1./2)-...
            2.*C.*A+2.*w.^2.*A.^2).*a0.*w./(B.^2-...
            4.*C.*A).^(1./2)./(B.^2.*w.^2+C.^2-...
            2.*C.*A.*w.^2+w.^4.*A.^2)-...
            ((-C+w.^2.*A).*sin(w.*t)+w.*cos(w.*t).*B).*...
            a0./(w.^4.*A.^2+(B.^2-2.*C.*A).*w.^2+C.^2);

    subplot(4,2,3)
    plot(t,X,'r')
    xlabel('t')
    ylabel('X')

%
%-- 8.Конец функции LWSin(A,B,C,a0,w).
%
```

4.5 Построение имитационных моделей

В соответствии с математическим описанием объекта(4.38) управления и поставленными задачами имитационная модель содержит два интегрирующих блока, необходимые генераторы сигналов, дисплеи, осциллографы, сумматоры и другие элементы.

Требуется построить эту модель, используя библиотеки блоков пакета Simulink, и установить параметры блоков, значения которых . Проверить работу модели можно путем её многократного запуска при изменении времени окончания работы. Модифицируя состав и изменяя режим работы модели, можно получить все требуемые характеристики объекта управления.

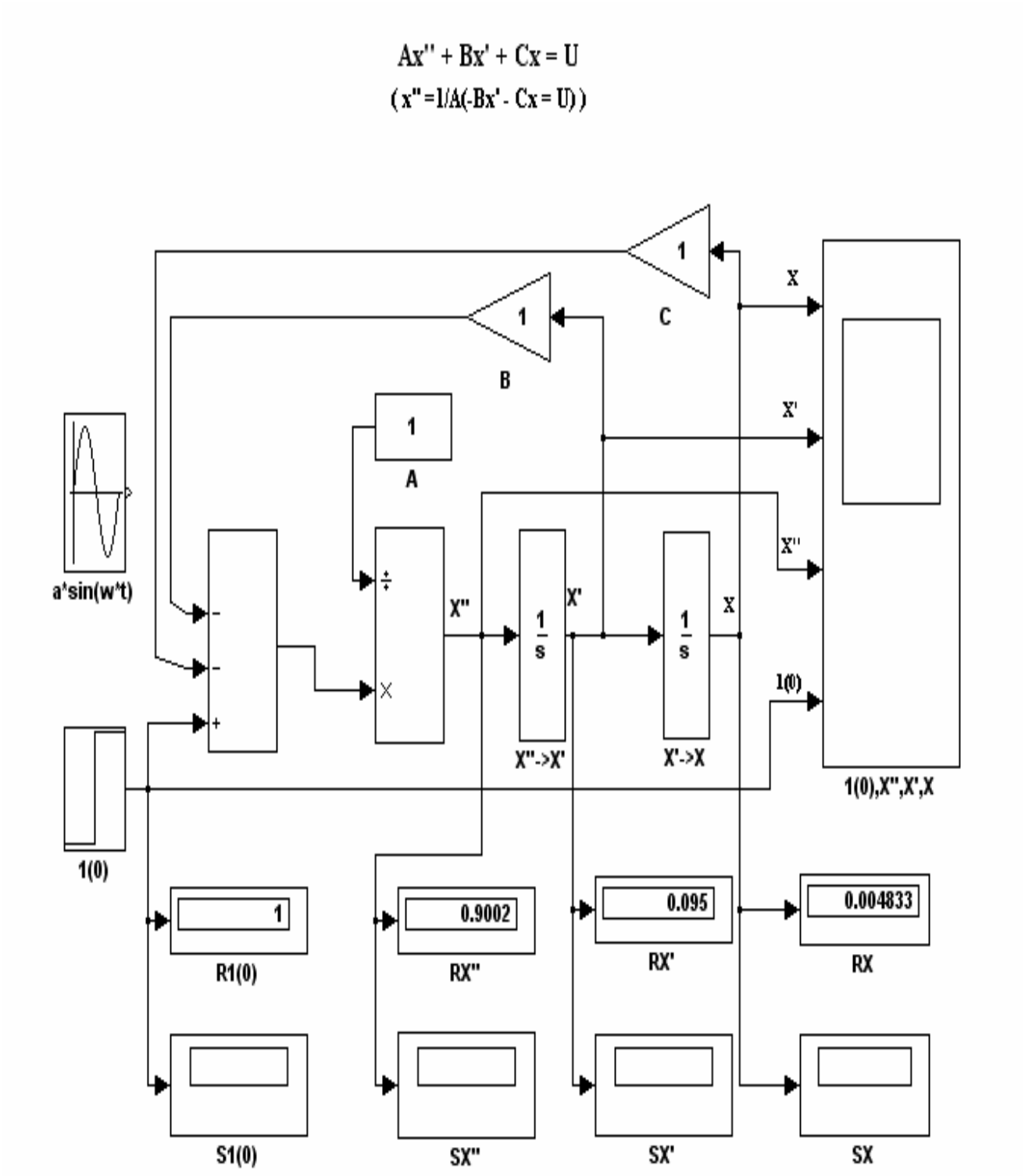

Рис. 4.5 Имитационная модель для простейших звеньев второго порядка

$$T_2^2 \frac{d^2\lambda}{dt^2} + T_1 \frac{d\ \lambda}{dt} + \delta\lambda(t) = \varphi(t)\ :$$

где $x = \lambda(t)$; $A = T_2^2$; $B = T_1 > 0$; $C = \delta > 0$; $U = \varphi(t)$

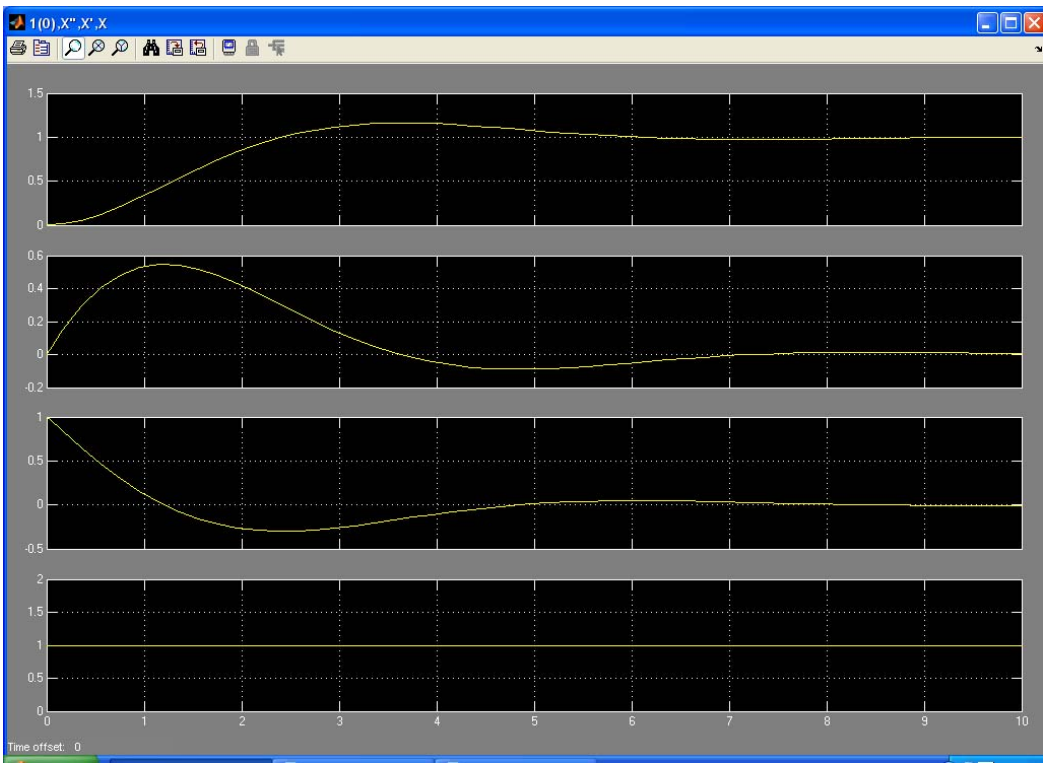

Рис. 4.6 Переходная проводимость и функция веса для простейшего звена второго порядка при A=1,B=1,C=1

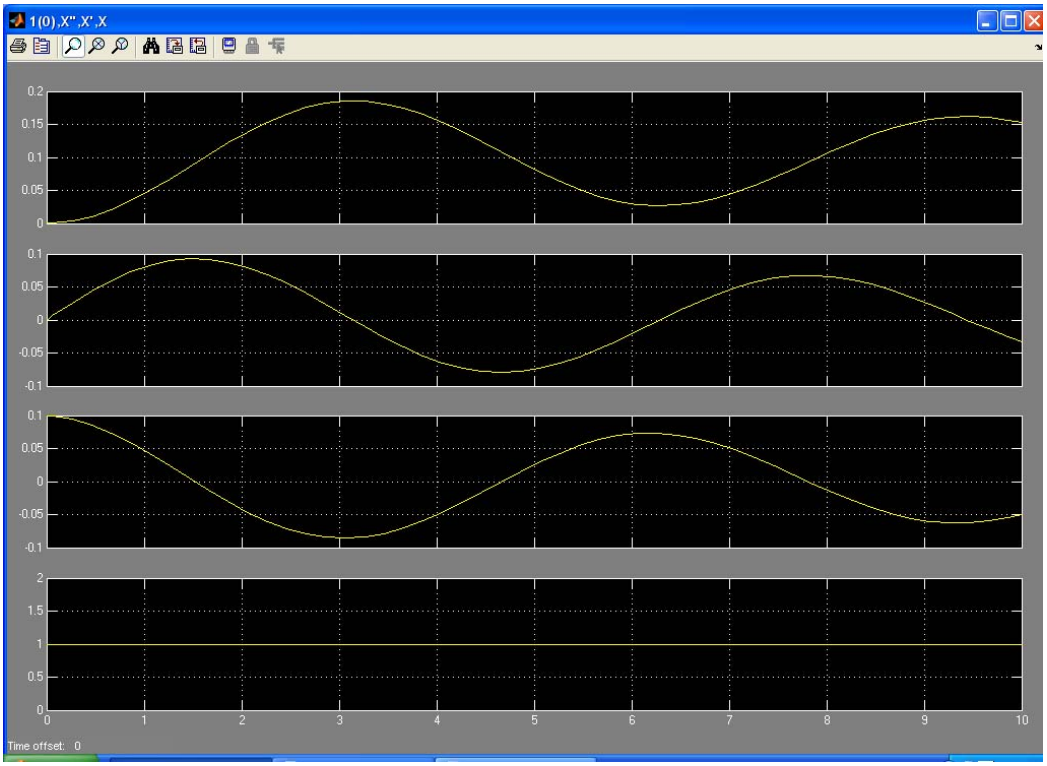

Рис. 4.7 Переходная проводимость и функция веса для простейшего звена второго порядка при A=10,B=1,C=10

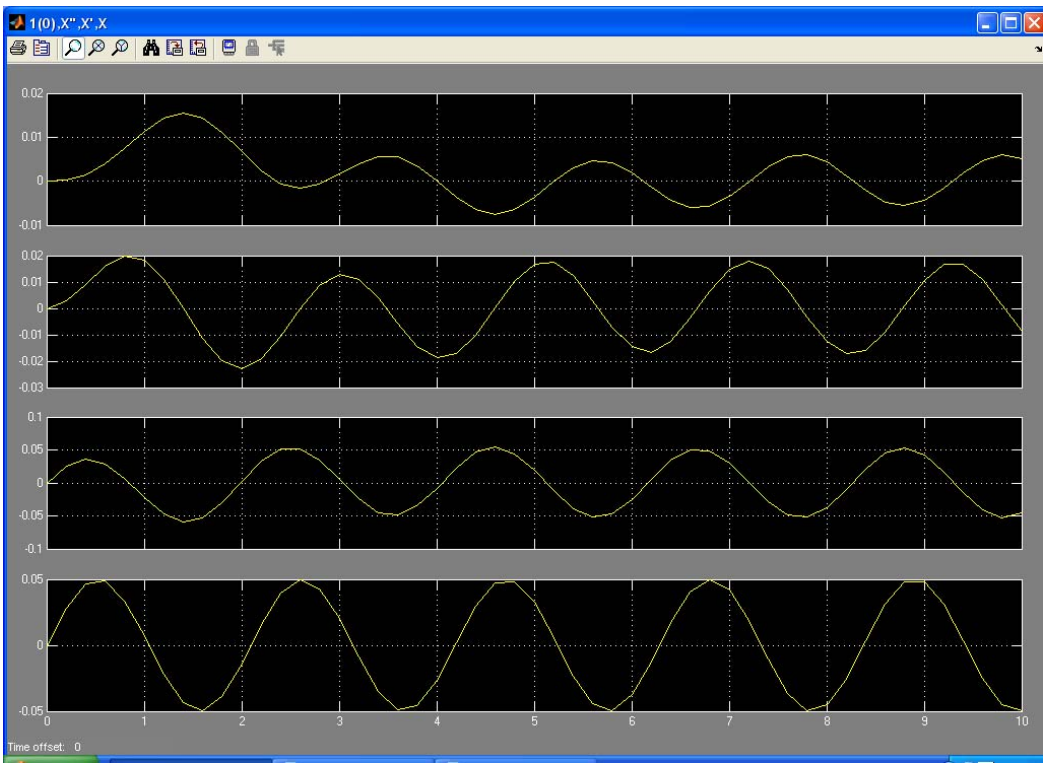

Рис. 4.8 Реакция простейшего звена второго порядка на синусоидальное возмущающее воздействие при A=1,B=1,C=1

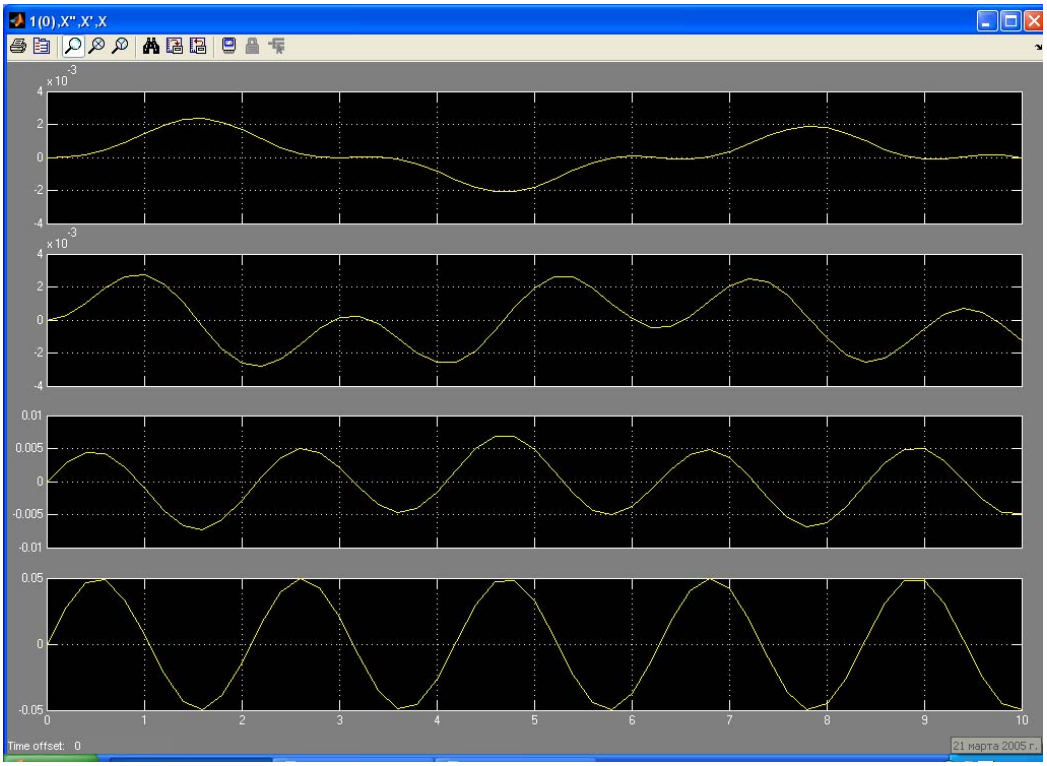

Рис. 4.9 Реакция простейшего звена второго порядка на синусоидальное возмущающее воздействие при A=10,B=1,C=10

4.6 Верификация математических моделей

Результаты верификации разработанных аналитических и имитационных моделей представлены в следующих таблицах:

Таблица 4.3

Результаты расчета и моделирования переходной проводимости и функции веса для апериодического звена второго порядка

Текущее время t	Возмущающее воздействие U(t)	Переходная проводимость Λ(t)		Функция веса W(t)	
		Расчетное значение	Модельное значение	Расчетное значение	Модельное значение
0	1	0	0	0	0
1	1	0.3403	0.3403	0.5335	0.5335
2	1	0.8449	0.8449	0.4193	0.4193
3	1	1.1244	1.1244	0.1332	0.1332
4	1	1.1531	1.1531	0.0495	0.0495
5	1	1.0746	1.0746	0.0879	0.0879
6	1	1.0023	1.0023	0.0509	0.0509
7	1	0.9744	0.9744	0.0076	0.0076
8	1	0.9790	0.9790	0.0127	0.0127
9	1	0.9929	0.9929	0.0128	0.0128
10	1	1.0022	1.0022	0.0054	0.0054

Таблица 4.4

Результаты расчета и моделирования переходной проводимости и функции веса для колебательного звена второго порядка

8	1	0.1058	0.1058	0.0665	0.0665
9	1	0.1564	0.1564	0.0270	0.0270
10	1	0.1529	0.1529	-0.0324	-0.0324

Управляемая величина — Скорость изменения управляемой величины

Текущее время t	Возмущающее воздействие U(t)	Λ(t)		W(t)	
		Расчетное значение	Модельное значение	Расчетное значение	Модельное значение
0	1	0	0	0	0
1	1	0.0445	0.0445	0.0801	0.0801
2	1	0.1333	0.1333	0.0825	0.0825
3	1	0.1845	0.1845	0.0125	0.0125
4	1	0.1569	0.1569	-0.0618	-0.0618
5	1	0.0821	0.0821	-0.0749	-0.0749
6	1	0.0301	0.0301	-0.0213	-0.0213
7	1	0.0442	0.0442	0.0459	0.0459

Таблица 4.5

Результаты расчета и моделирования переходного процесса для простейшего звена второго порядка при синусоидальном возмущающем воздействии

Текущее время t	Возмущающее воздействие U(t)	Управляемая величина X(t) при A=1,B=1,C=1		Управляемая величина X(t) при A=10,B=1,C=10	
		Расчетное значение	Модельное Значение	Расчетное значение	Модельное значение
0	0.05*sin(3*0)	0	0	0	0
1	0.05*sin(3*1)	0.0114	0.0114	0.0014	0.0014
2	0.05*sin(3*2)	0.0068	0.0068	0.0017	0.0017
3	0.05*sin(3*3)	0.0015	0.0015	-0.0000	-0.0000
4	0.05*sin(3*4)	0.0001	0.0001	-0.0009	-0.0009
5	0.05*sin(3*5)	-0.0036	-0.0036	-0.0018	-0.0018
6	0.05*sin(3*6)	0.0019	0.0019	0.0001	0.0001
7	0.05*sin(3*7)	-0.0035	-0.0035	0.0004	0.0004
8	0.05*sin(3*8)	0.0043	0.0043	0.0018	0.0018
9	0.05*sin(3*9)	-0.0044	-0.0044	-0.0001	-0.0001
10	0.05*sin(3*10)	0.0052	0.0052	-0.0000	-0.0000

4.7 Варианты заданий и порядок их выполнения

1. По табл. 4.2 выбрать набор простейших звеньев для построения вариантов их соединения в системе автоматического регулирования. Например, набор 5-4-3 определяет следующие звенья: колебательное звено (1-й столбец таблицы), апериодическое звено 2-го порядка (4-я строка) и интегрирующее звено (пересечение 4-й строки и 5-го столбца).

2. Построить все варианты соединений, которые можно получить из трех звеньев: последовательное, параллельное, последовательно-паралельное, последовательное с обратной связью, параллельное с обратной связью, звено с двумя последовательными звеньями в качестве обратной связи, звено с двумя параллельными звеньями в качестве обратной связи и соединения, в которых одно звено используется в качестве обратной связи для какой-либо другого звена – всего порядка 25 соединений.

3. Для всех вариантов соединений выбранного набора простейших звеньев вывести аналитические выражения для вычисления передаточных функций этих соединений $K(p)$, учитывая, что при последовательном соединении звеньев их передаточные функции умножаются, при параллельном – складываются, а при наличии звеньев обратной связи – вычисляются по формуле

$K(p) = K_1(p)/(1 + K_1(p) * K_2(p))$, где $K_1(p)$ – передаточная функция для основного соединения и $K_2(p)$ – передаточная функция для звеньев обратной связи.

4. Для всех вариантов соединений выбранного набора простейших звеньев вывести аналитические выражения для вычисления частотных функций этих соединений, а также их модулей и аргументов, учитывая, что частотная функция $K(i\omega)$ соединения равна передаточной функции $K(p)$ того же соединения при $p = i\omega$.

5. Задать произвольно или с соблюдением некоторых условий значения параметров звеньев, построить на комплексной плоскости аплитудно-фазовые характеристики – годографы векторов $K(i\omega)$ для всех вариантов соединений, и оценить их устойчивость используя М-функцию Godograph.

6. Для одного из вариантов соединения простейших звеньев найти аналитическое выражение для реакции соединения на синусоидальное возмущающее воздействие и выявить наличие собственных колебаний, а также возможность возникновения резонанса.

7. С помощью преобразований Лапласа для всех вариантов соединений выбранного набора простейших звеньев вывести аналитические выражения для переходных проводимостей и функций веса этих соединений.

8. Построить имитационные модели для выбранного набора простейших звеньев и всех вариантов их соединений, произвести их моделирование при нулевых начальных условиях и отсутствии возмущающих сил, построить графики выходных величин, используя осциллографы, и оценить устойчивость соединений, а также характеристики переходного процесса.

9. Используя построенные имитационные модели, определить реакции соединений на ступенчатое воздействие и оценить характеристики переходного процесса.

10. С помощью имитационных моделей соединений и синусоидального входного сигнала, имеющего переменную частоту, построить амплитудную характеристику этого соединения.

11. Используя переходную проводимость или функцию веса одного соединения, рассчитать реакцию соединения на заданный входной сигнал.

12. Оформить отчет по лабораторной работе, применяя средства генерирования HTML-описания пакета Simulink и используя результаты работы имитационных моделей.

4.8 Оформление отчета по результатам исследований

Для завершения лабораторной работы необходимо сгенерировать отчет в формате HTML, затем преобразовать его в формат RTF с помощью текстового редактора, включит в него теоретические результаты, отформатировать текст и графические объекты, записать на дискету и в электронном виде предъявить преподавателю. Обосновать достоверность полученных результатов.

Лабораторная работа № 5

ОПТИМАЛЬНОЕ УПРАВЛЕНИЕ ПРОСТЕЙШИМИ ЗВЕНЬЯМИ

Цель работы: разработка аналитических моделей для определения характеристик и нахождения оптимального управления объектами, состоящими из простейших звеньев, реализация этих моделей в программной среде математической системы MATLAB и построение имитационных моделей с помощью пакета Simulink, верификация разработанных моделей и определение с их использованием характеристик объектов управления, а также нахождение оптимального управления для обеспечения изменения выходной величины на заданное значение за минимальное время.

5.1 Постановка задач исследования

В данной лабораторной работе рассматриваются объекты, движение которых описывается обыкновенными дифференциальными уравнениями с постоянными коэффициентами, порядок которых не ниже второго. С точки зрения структуры это – либо отдельные простейшие звенья, либо несложные соединения таких звеньев. Целью управления является минимизация времени изменения выходной величины объекта на заданное значение за счет рационального выбора ограниченного по модулю управляющего воздействия U(t). Прежде чем приступить к поиску оптимального управления, необходимо определить характеристики объекта и провести всесторонние исследования динамики его поведения при различных возмущающих воздействиях, в частности оценить устойчивость звена или соединения, так как только для устойчивых объектов имеет смысл поиск оптимального управления. Аналитическое выражение для оптимального управления U(t) следует искать с помощью функции Гамильтона (гамильтониана) и принципа максимума Понтрягина. Расчет характеристик переходного процесса должен производиться с помощью системы MATLAB. Имитационная модель должна подтвердить расчеты по программе. Необходимо также разработать имитационную модель для автоматического определения параметров оптимального управления и имитационную модель для нахождения управлений, улучшающих динамические характеристики целевой системы при оптимальном управлении. Необходимо также с помощью интеграла Дюамеля рассчитать реакцию системы на оптимальное управляющее воздействие и проверить это на имитационной модели.

5.2 Разработка аналитических моделей

В качестве прототипа рассмотрим объект управления, который описывается дифференциальным уравнением

$$T \cdot \ddot{x} + \dot{x} = k \cdot u\,, \qquad (5.1)$$

где T и k – положительные постоянные. Данное уравнение характеризует объект, состоящий из интегрирующего и инерционного звеньев, соединенных последовательно.

Уравнением (5.1) приближенно описываются многие объекты управления: маломощные электрические следящие системы постоянного и переменного тока, двигатели которых управляются электронными усилителями, некоторые тепловые объекты, у которых регулирующий орган имеет интегрирующий электрический, гидравлический или пневматический привод; транспортные механизмы, двигатели которых управляются напряжением сети, и т. д.

Требуется найти алгоритм управления, переводящий объект из положения $x = 0$, $\dot{x} = 0$ при $t = 0$ в положение $x = x_n$, $\dot{x} = 0$ за минимальное время t_2; на управляющее воздействие наложено ограничение $|u| \le u_{\max}$. Определить момент переключения t_1, оптимальный переходной процесс $\vec{x}(t) = (x_1(t), x_2(t))$ и время перехода t_2. По точкам построить графики $x_1(t)$ и $x_2(t)$. Для этого интервал времени от 0 до t_1 и интервал от t_1 до t_2 разделить на 5 равных частей и вычислить значения $x_1(t)$ и $x_2(t)$ в соответствующих точках. В точке $t = t_1$ $x_1(t)$ и $x_2(t)$ должны рассчитываться дважды по разным формулам и эти значения должны совпадать.

Прежде чем приступить к поиску оптимального решения, необходимо оценить характеристики объекта и устойчивость его состояния.

Передаточная и частотная функции объекта имеют вид:

$$K(p) = \frac{1}{p(Tp+1)}, \qquad (5.2)$$

$$K(i\omega) = \frac{1}{i\omega(Ti\omega+1)}. \qquad (5.3)$$

Функция переходной проводимости объекта находится как решение дифференциального уравнения (5.1) при толчкообразном внешнем воздействии

$$T\ddot{x} + \dot{x} = 1(t). \qquad (5.4)$$

Используя прямое и обратное преобразование Лапласа, получаем

$$\Lambda(t) = -T + t + Te^{-\frac{t}{T}}, \qquad (5.5)$$

или

$$\Lambda(t) = t + T(e^{-\frac{t}{T}} - 1) \qquad (5.6)$$

Функция веса объекта получается путем дифференцирования функции переходной проводимости и имеет вид

$$W(t) = 1 - e^{-\frac{t}{T}}. \qquad (5.7)$$

Реакция объекта на синусоидальное возмущающее воздействие определяется путем решения дифференциального уравнения

$$T\ddot{x} + \dot{x} = A_0 \sin \omega_0 t \qquad (5.8)$$

и имеет следующее аналитическое выражение:

$$x(t) = \frac{A_0\omega_0}{T}\left(\frac{T}{\omega_0^2} - \frac{T^3}{(1+k_0^2T^2}e^{-\frac{t}{T}} - \frac{1}{2\omega_0^2}\left(\frac{1}{-i\omega_0 + \frac{1}{T}}e^{-i\omega_0 t} + \frac{1}{i\omega_0 + \frac{1}{T}}e^{i\omega_0 t}\right)\right). \qquad (5.9)$$

Рассмотрим порядок вывода этого выражения.

Используя преобразования Лапласа при нулевых начальных условиях, получаем

$$(Tp^2 + p)\cdot x(p) = \frac{A_0\omega_0}{p^2 + \omega_0^2}. \qquad (5.10)$$

Отсюда находим изображение регулируемой величины

$$x(p) = \frac{A_0\omega_0}{p(Tp+1)(p^2 + \omega_0^2)}. \qquad (5.11)$$

Корни характеристического уравнения легко находятся и равны: $p_1 = 0$; $p_2 = -\frac{1}{T}$; $p_3 = -i\omega_0$; $p_1 = i\omega_0$.

Разложим дробь на простейшие дроби:

$$\frac{1}{(p-0)(p-(-\frac{1}{T}))(p-(-i\omega_0))(p-i\omega_0)} = \frac{A}{p-0} + \frac{B}{p+\frac{1}{T}} + \frac{C}{p+i\omega_0} + \frac{D}{p-i\omega_0}. \qquad (5.12)$$

Приведя это уравнение к общему знаменателю и отбросив его, получим:

$$A(p+\frac{1}{T})(p^2+\omega_0^2) + Bp(p^2+\omega_0^2) + Cp(p+\frac{1}{T})(p-i\omega_0) + Dp(p+\frac{1}{T})(p+i\omega_0) = 1. \qquad (5.13)$$

Подставим в это уравнение корни и найдем коэффициенты разложения:

$$A = \frac{T}{\omega_0^2}; \qquad (5.14)$$

$$B = -\frac{T^3}{1+\omega_0 T^2}; \qquad (5.15)$$

$$D = -\frac{1}{2\omega_0^2(i\omega_0 + \frac{1}{T})}; \qquad (5.16)$$

$$C = -\frac{1}{2\omega_0^2(-i\omega_0 + \frac{1}{T})} \cdot \tag{5.17}$$

Переходя с помощью таблиц от изображения к оригиналу, получаем следующее выражение для вынужденных колебаний объекта:

$$x(t) = \left(\frac{A_0\omega_0}{T}\right)\left(\frac{T}{\omega_0^2} - \frac{T^3}{1+\omega_0^2T^2}\right)e^{-\frac{t}{T}} - \left(\frac{1}{\omega_0^2(\omega_0^2 + \frac{1}{T^2})}\right)\left(\omega_0 \sin\omega_0 t + \frac{1}{T}\cos\omega_0 t\right) \tag{5.18}$$

Теперь необходимо найти алгоритм оптимального управления.

Обозначим: $x_1(t) = x(t)$, $x_2(t) = \dot{x}_1(t)$. Тогда вместо уравнения (5.1) будем иметь следующую систему дифференциальных уравнений:

$$\begin{cases} \dot{x}_1(t) = x_2(t), \\ T \cdot \dot{x}_2(t) + x_2(t) = k \cdot u. \end{cases} \tag{5.18}$$

Разделим все члены второго уравнения на T и запишем систему в следующем виде:

$$\begin{cases} \dot{x}_1(t) = x_2(t), \\ \dot{x}_2(t) = \frac{1}{T} \cdot (k \cdot u - x_2). \end{cases} \tag{5.19}$$

Функция Гамильтона для системы двух дифференциальных уравнений конструируется следующим образом:

$$H(\vec{\psi}, \vec{x}, u) = \psi_1(t) \cdot f_1(x_1, x_2, t, u) + \psi_2(t) \cdot f_2(x_1, x_2, t, u), \tag{5.20}$$

где $\psi_1(t)$ и $\psi_2(t)$- вспомогательные функции,

$f_1(x_1, x_2, t, u)$ и $f_2(x_1, x_2, t, u)$- правые части дифференциальных уравнений.

В нашем случае $f_1(x_1, x_2, t, u) = x_2(t)$, $f_2(x_1, x_2, t, u) = \frac{1}{T} \cdot (k \cdot u - x_2(t))$.

Тогда гамильтониан запишется в виде

$$H(\vec{\psi}, \vec{x}, u) = \psi_1(t) \cdot x_2 + \psi_2(t) \cdot \frac{1}{T} \cdot (k \cdot u - x_2(t)). \tag{5.21}$$

Вспомогательные функции $\psi_1(t)$ и $\psi_2(t)$ должны удовлетворять следующей системе дифференциальных уравнений:

$$\begin{cases} \dot{\psi}_1(t) = -\frac{\partial H}{\partial x_1}, \\ \dot{\psi}_2(t) = -\frac{\partial H}{\partial x_2}. \end{cases} \tag{5.22}$$

В нашем случае эта система будет иметь вид:

$$\begin{cases} \dot{\psi}_1(t) = 0, \\ \dot{\psi}_2(t) = -\psi_1(t) + \dfrac{1}{T} \cdot \psi_2(t). \end{cases} \tag{5.23}$$

Из первого уравнения следует $\psi_1(t) = c_1$, где c_1 - постоянная величина. Тогда второе уравнения можно записать в виде:

$$\dot{\psi}_2(t) - \frac{1}{T} \cdot \psi_2(t) = -c_1 . \tag{5.24}$$

Это линейное неоднородное уравнение. Решением линейного однородного уравнения является функция $c_2 \cdot \exp\left(\frac{t}{T}\right)$, а частное решение неоднородного уравнения равна константе c_3. Подставляя эту константу в дифференциальное уравнение, получим $c_3 = c_1 \cdot T$, тогда можно записать

$$\psi_2(t) = c_1 T + c_2 \cdot \exp\left(\frac{t}{T}\right) \quad . \tag{5.25}$$

Функция Гамильтона примет вид

$$H = c_1 \cdot x_2(t) + \left(c_1 \cdot T + c_2 \cdot \exp\left(\frac{t}{T}\right)\right) \cdot \frac{1}{T} \cdot \left[k \cdot u - x_2(t)\right]. \tag{5.26}$$

На основании принципа максимума Понтрягина управление выбирается таким образом, чтобы H принимала наибольшее значение. Для этого максимальное значение должно принять слагаемое функции H, которое зависит от управления u. Обозначим это слагаемое H^*. В нашем случае оно имеет вид

$$H^* = \frac{k}{T}\left(c_1 \cdot T + c_2 \cdot \exp\left(\frac{t}{T}\right)\right) \cdot u . \tag{5.27}$$

Величина H^* принимает наибольшее значение при $k = sign\left(c_1 \cdot T + c_2 \cdot \exp\left(\frac{t}{T}\right)\right) \cdot u_{\max}$. Так как $\left(c_1 \cdot T + c_2 \cdot \exp\left(\frac{t}{T}\right)\right)' = \frac{c_2}{T} \cdot \exp\left(\frac{t}{T}\right)$, то мы видим, что производная в нуль не обращается. Поэтому функция $c_1 \cdot T + c_2 \cdot \exp\left(\frac{t}{T}\right)$ может менять знак не более одного раза. Так как $\frac{k}{T} > 0$, то

$$\max_{u \in U} H^* = \frac{k}{T}\left| c_1 \cdot T + c_2 \cdot \exp\left(\frac{t}{T}\right)\right| \cdot u_{\max} . \tag{5.28}$$

Обозначим через t_1 - время переключения управления, через t_2 - время перехода системы из начальной точки $\vec{x}_0 = (0;0)$ в конечную точку $\vec{x}_1(x_n;0)$.

Оптимальное управление будет таким:

$$u = \begin{cases} u_{\max}, & \text{при} \quad 0 \le t < t_1, \\ -u_{\max}, & \text{при} \quad t_1 < t \le t_2. \end{cases} \tag{5.29}$$

График функции представлен на рис 5.1.

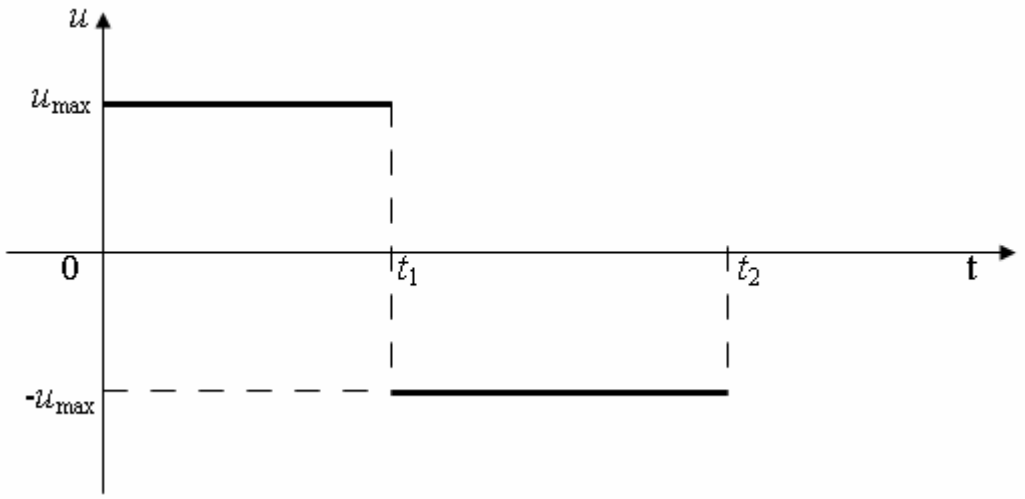

Рис 5.1 Оптимальное управление для простейшего звена

Теперь найдем время переключения t_1, время управления t_2, а также $x_1(t)$ и $x_2(t)$. Решим дифференциальное уравнение (5.1) сначала для $u = u_{\max}$, при $0 \le t < t_1$, а затем для $u = -u_{\max}$, при $t_1 \le t < t_2$.

Пусть $u = u_{\max}$. Тогда уравнение (5.1) запишется в виде

$$T \cdot \ddot{x}_1 + \dot{x}_1 = k \cdot u_{\max}. \tag{5.30}$$

Это линейное неоднородное уравнение с постоянными коэффициентами. Соответствующее линейное однородное уравнение имеет вид:

$$T \cdot \ddot{x}_1 + \dot{x}_1 = 0. \tag{5.31}$$

Характеристическое уравнение $T \cdot r^2 + r = 0$ имеет корни $r_1 = 0$, $r_2 = -\frac{t}{T}$. Тогда общее решение однородного уравнения будет иметь вид $c_4 + c_5 \cdot \exp\left(-\frac{t}{T}\right)$. Так как нуль является корнем характеристического уравнения, а правая часть неоднородного уравнения равна константе, то частное решение неоднородного уравнения будем искать в виде $c_6 \cdot t$. Подставляя эту функцию в неоднородное уравнение, получим $c_6 = k \cdot u_{\max}$. Поэтому общее решение неоднородного уравнения запишется в виде

$$x_1(t) = k \cdot u_{\max} \cdot t + c_4 + c_5 \cdot \exp\left(-\frac{t}{T}\right). \qquad (5.32)$$

Постоянные интегрирования c_4 и c_5 найдем из начальных условий: $x_1(0) = 0$, $\dot{x}_1(0) = 0$. В нашем случае $\dot{x}_1(t) = k \cdot u_{\max} - \frac{c_5}{T}\exp\left(-\frac{t}{T}\right)$. Тогда будем иметь следующую систему:

$$\begin{cases} x_1(0) = c_4 + c_5 = 0, \\ \dot{x}_1(0) = k \cdot u_{\max} - \frac{c_5}{T} = 0. \end{cases} \qquad (5.33)$$

Решая полученную систему, получим $c_4 = -k \cdot u_{\max} \cdot T$, $c_5 = k \cdot u_{\max} \cdot T$. Тогда при $0 \le t < t_1$:

$$x_1(t) = k \cdot u_{\max} \cdot t + k \cdot u_{\max} \cdot T \cdot \left(\exp\left(-\frac{t}{T}\right) - 1\right), \qquad (5.34)$$

$$\dot{x}_1(t) = k \cdot u_{\max} - k \cdot u_{\max} \cdot \exp\left(-\frac{t}{T}\right). \qquad (5.35)$$

Теперь решим уравнение (5.1) при $u = -u_{\max}$. Решение однородного уравнения остается без изменения, а частным решением неоднородного уравнения теперь будет функция $-k \cdot u_{\max} \cdot t$. Общим решением неоднородного уравнения является функция

$$x_1(t) = -k \cdot u_{\max} \cdot t + c_6 + c_7 \cdot \exp\left(-\frac{t}{T}\right). \qquad (5.36)$$

Используем условия в конце управления: $x_1(t_2) = x_n$, $\dot{x}_1(t_2) = 0$. Так как $\dot{x}_1(t) = -k \cdot u_{\max} - \frac{c_7}{T}\exp\left(-\frac{t}{T}\right)$, то получим следующие уравнения

$$x_1(t_2) = -k \cdot u_{\max} \cdot t_2 + c_6 + c_7 \cdot \exp\left(-\frac{t_2}{T}\right) = x_n, \qquad (5.37)$$

$$\dot{x}_1(t_2) = -k \cdot u_{\max} - \frac{c_7}{T}\exp\left(-\frac{t_2}{T}\right) = 0. \qquad (5.38)$$

Из последних двух уравнений выразим c_6 и c_7 через неизвестную t_2. Имеем $c_7 = -k \cdot u_{\max} \cdot T \cdot \exp\left(\frac{t_2}{T}\right)$, $c_6 = x_n + k \cdot u_{\max} \cdot t_2 + k \cdot u_{\max} \cdot T$. Тогда при $t_1 \le t < t_2$ будем иметь

$$x_1(t) = x_n + k \cdot u_{\max} \cdot \left(t_2 - t + T - T \cdot \exp\left(\frac{t_2 - t}{T}\right)\right), \qquad (5.39)$$

$$\dot{x}_1(t) = k \cdot u_{\max} \cdot \left(\exp\left(\frac{t_2 - t}{T} \right) - 1 \right). \qquad (5.40)$$

Теперь мы имеем две неизвестные t_1 и t_2. Для их определения применим метод стыковывания уравнений. В точке $t = t_1$ $x_1(t)$, вычисленные по формулам (5.16) и (5.18), должны совпадать. В этой точке значения $\dot{x}_1(t)$, вычисленные по формулам (5.17) и (5.19), также должны совпадать. Имеем следующую систему уравнений

$$\begin{cases} k \cdot u_{\max} \cdot t_1 + k \cdot u_{\max} \cdot T \cdot \left(\exp\left(-\frac{t_1}{T} \right) - 1 \right) = x_n + k \cdot u_{\max} \cdot \left(t_2 - t_1 + T - T \cdot \exp\left(\frac{t_2 - t_1}{T} \right) \right) \\ k \cdot u_{\max} - k \cdot u_{\max} \cdot \exp\left(-\frac{t_1}{T} \right) = k \cdot u_{\max} \cdot \left(\exp\left(\frac{t_2 - t_1}{T} \right) - 1 \right). \end{cases}, \qquad (5.41)$$

В первом уравнении раскроем скобки и приведем подобные члены, а правую и левую часть второго уравнения разделим на $k \cdot u_{\max}$, получим

$$\begin{cases} 2 \cdot k \cdot u_{\max} \cdot t_1 + k \cdot u_{\max} \cdot T \cdot \left(\exp\left(-\frac{t_1}{T} \right) - 2 \right) = x_n + k \cdot u_{\max} \cdot t_2 - k \cdot u_{\max} \cdot T \cdot \exp\left(\frac{t_2 - t_1}{T} \right), \\ \exp\left(\frac{t_2 - t_1}{T} \right) + \exp\left(-\frac{t_1}{T} \right) - 2 = 0. \end{cases} \qquad (5.42)$$

Перепишем первое уравнение в виде:

$$2 \cdot k \cdot u_{\max} \cdot t_1 + k \cdot u_{\max} \cdot T \cdot \left(\exp\left(\frac{t_2 - t_1}{T} \right) + \exp\left(-\frac{t_1}{T} \right) - 2 \right) = x_n + k \cdot u_{\max} \cdot t_2. \qquad (5.43)$$

Так как выражение в скобках равно 0, то из последнего уравнения находим

$$t_2 = 2t_1 - \frac{x_n}{k \cdot u_{\max}}. \qquad (5.44)$$

Второе уравнение сначала умножим на $\exp\left(\frac{t_1}{T} \right)$, а вместо t_2 подставим его найденное значение, получим

$$2\exp\left(\frac{t_1}{T} \right) - \exp\left(\frac{2t_1 - \frac{x_n}{k \cdot u_{\max}}}{T} \right) - 1 = 0. \qquad (5.45)$$

Обозначим $\exp\left(\frac{t_1}{T} \right) = y$, $(y > 1)$. Тогда следует

$$2y - y^2 \cdot \exp\left(-\frac{x_n}{k \cdot u_{\max} \cdot T} \right) - 1 = 0. \qquad (5.46)$$

Правую и левую части последнего уравнения умножим на $-\exp\left(\frac{x_n}{k\cdot u_{\max}\cdot T}\right)$, получим $y^2-2^{\frac{x_n}{k\cdot u_{\max}\cdot T}}\cdot y+e^{\frac{x_n}{k\cdot u_{\max}\cdot T}}=0$. Обозначим $z=\exp\left(\frac{x_n}{k\cdot u_{\max}\cdot T}\right)$, $(z>1)$. Тогда уравнение примет вид $y^2-2z\cdot y+z=0$. Решая квадратное уравнение, получим $y_1=z-\sqrt{z^2-z}$, $y_2=z+\sqrt{z^2-z}$. Так как $z>1$, то $z-1<\sqrt{z^2-z}<z$, а тогда $-z<-\sqrt{z^2-z}<1-z$. Прибавим ко всем частям неравенства z, получим $0<z-\sqrt{z^2-z}<1$. Следовательно, $y_1<1$ является посторонним корнем. Таким образом $y=z+\sqrt{z^2-z}$. Произведем расчеты для заданных исходных данных: T = 0.62; k = 0.0023; U_{max} = 220; x_n = 2.09.

Подставляя эти значения в выражения для z и y, получим:

$$z=\exp\left(\frac{2.09}{0.0023\cdot 220\cdot 0.62}\right)=782.107, \qquad (5.47)$$

$$y=782.107+\sqrt{782.107^2-782/107}=1563.714. \qquad (5.48)$$

Из уравнения $\exp\left(\frac{t_1}{T}\right)=y$, находим $t_1=T\cdot\ln y=0.62\cdot\ln(1563.714)=4.5600$. Теперь по формуле (5.20) найдем время управления. $t_2=2\cdot 4.5600-\frac{2.09}{0.0023\cdot 220}=4.9895$. Траектория движения точки в фазовом пространстве $x_1(t)$ и её скорость $x_2(t)$ должны рассчитываться по формулам:

$$x_1(t)=\begin{cases}k\cdot u_{\max}\cdot t+k\cdot u_{\max}\cdot T\cdot\left(\exp\left(-\frac{t}{T}\right)-1\right), & при \quad 0\le t<t_1,\\ x_n+k\cdot u_{\max}\cdot\left(t_2-t+T-T\cdot\exp\left(\frac{t_2-t}{T}\right)\right), & при \quad t_1<t\le t_2.\end{cases} \qquad (5.49)$$

$$x_2(t)=\begin{cases}k\cdot u_{\max}\cdot\left(1-\exp\left(-\frac{t}{T}\right)\right), & при \quad 0\le t<t_1,\\ k\cdot u_{\max}\cdot\left(\exp\left(\frac{t_2-t}{T}\right)-1\right), & при \quad t_1<t\le t_2.\end{cases} \qquad (5.50)$$

5.3 Программная реализация аналитических моделей

```
function  RKiw(T)
%
%-- ГРАФИКИ ЧАТОТНОЙ ФУНКЦИИ K(iw):
%
%-- 1.Диапазон частот:
%
```

```
    w=0.01:0.001:1;
%
%-- 2.Частотная функция:
%
    Kiw=1./((i.*w).*(T.*i.*w+1));
%
%-- 3.Модуль частотной функции:
%
    subplot(2,2,1)
    plot(w,abs(Kiw),'r')
    xlabel('w')
    ylabel('abs(Kiw)')
%
%-- 4.Фаза частотной функции:
%
    subplot(2,2,2)
    plot(w,angle(Kiw),'r')
    xlabel('w')
    ylabel('angle(Kiw)')
%
%-- 5.Годограф на плоскости:
%
    subplot(2,2,3)
    plot(real(Kiw),imag(Kiw),'r')
    xlabel('real(Kiw)')
    ylabel('image(Kiw)')
%
%-- 6.Годограф в трехмерном пространстве (комета):
%
    subplot(2,2,4)
    comet3(real(Kiw),imag(Kiw),w)
    xlabel('real(Kiw)')
    ylabel('image(Kiw)')
    zlabel('w')
%
%-- Конец функции RKiw(T).

function [t1,t2] = Param(T,k,Umax,Xn)
%
%-- ФУНКЦИЯ ДЛЯ РАСЧЕТА ПАРАМЕТРОВ МАТЕМАТИЧЕСКИХ
%--                          МОДЕЛЕЙ:
%
%-- T    = 0.62;          %-- постоянная при x";
%-- k    = 0.0023;        %-- постоянная при управляющем воздействии u(t);
%-- Umax = 220;           %-- максимальное значение управляющего
```

```
                              %-- воздействия u(t);
%-- Xn   = 2.09;              %-- конечное значение управляемой величина x(t);
%
%-- 1.Расчет параметров математических моделей:
%
    z    = exp(Xn/(k*Umax*T))%-- промежуточный расчетный параметр;
    y    = z + sqrt(z^2-z)      %-- промежуточный расчетный параметр;
    t1   = T*log(y)             %-- время переключения управления;
    t2   = 2*t1-Xn/(k*Umax)     %-- время перехода системы в конечную
                                %-- точку;
%
%-- 2.Конец функции Param(T,k,Umax,Xn).

function [Lambda, Weight,XSin]= LWSin(T,a0,w)
%
%-- ГРАФИКИ ПЕРЕХОДНЫХ ФУНКЦИЙ  Lambda(t), Weight(t) и
%--                              Sin(A, B, C):
%
%-- 1.Временной интервал:
%
    t = 0:1:10;
%
%-- 2.Функция переходной проводимости:
%
    Lambda = -T+t+T.*exp(-t./T);
   subplot(4,2,1)
    plot(t,Lambda,'r')
    xlabel('t')
    ylabel('Lambda')
%
%-- 3.Функция веса:
%
    Weight = 1-exp(-t./T);
    subplot(4,2,2)
    plot(t,Weight,'r')
    xlabel('t')
    ylabel('Weight')
%
%-- 4.Реакция звеньев на синусоидальные воздействия:
%
     XSin = (a0.*w./T).*(T./w.^2-(T.^3./...
                (1+w.^2.*T.^2)).*exp(-t./T)-...
                (1./(w.^2.*(w.^2+1./T.^2))).*...
                (w.*sin(w.*t)+1./T.*cos(w.*t)));
    subplot(4,2,3)
```

```
    plot(t,XSin,'r')
    xlabel('t')
    ylabel('X')
%
%-- 8.Конец функции LWSin(T,a0,w).

function [X,DX] = OptUpr(T,k,Umax,Xn)
%
%-- ФУНКЦИЯ ДЛЯ РАСЧЕТА ПЕРЕХОДНОГО ПРОЦЕССА
%--            ОПТИМАЛЬНОГО УПРАВЛЕНИЯ:
%
%-- 1.Расчет параметров математической модели:
%
   z   = exp(Xn/(k*Umax*T))  %-- промежуточный расчетный параметр;
   y   = z + sqrt(z^2-z)     %-- промежуточный расчетный параметр;
   t1  = T*log(y)            %-- время переключения управления;
   t2  = 2*t1-Xn/(k*Umax)    %-- время перехода системы в конечную точку;
%
%-- 2.Задание временных точек:
%
   t = [0.0000 0.9120 1.8240 2.7360 3.6480 4.5600 4.6459 4.7318 4.8177 …
        4.9036 4.9895];
%
%-- 3.Расчет управляемой величины:
%
   for i = 1:1:11
      if t(i) <= t1
        X(i) = k*Umax*t(i)+k*Umax*T*(exp(-t(i)/T)-1);
      else
        X(i) = Xn+k*Umax*(t2-t(i)+T-T*exp((t2-t(i))/T));
      end
   end
%
%-- 4.Расчет скорости изменения управляемой величины:
%
   for i = 1:1:11
      if t(i) <= t1
        DX(i) = k*Umax*(1-exp(-(t(i)/T)));
      else
        DX(i) = k*Umax*(exp((t2-t(i))/T)-1);
      end
   end
%
%-- 5.Визуализация управляемой величины и скорости ее изменения:
%
```

```
    subplot(2,2,1)
    plot(t,X,'r')
    xlabel('t')
    ylabel('X')
    subplot(2,2,2)
    plot(t,DX,'r')
    xlabel('t')
    ylabel('DX')
%-- 6.Конец функции OptUpr(T,k,Umax,Xn).
```

5.4 Построение имитационных моделей

В соответствии с математическим описанием объекта управления и поставленными задачами имитационная модель содержит два интегрирующих блока, необходимые генераторы сигналов, дисплеи, осциллографы, сумматоры и другие элементы (см. рис. 5.2).

Требуется построить эту модель, используя библиотеки блоков пакета Simulink, и настроить параметры блоков в соответствии с условиями задачи. Проверить работу модели можно путем её многократного запуска при изменении времени окончания работы. Модифицируя состав модели и изменяя режим её работы, можно получить все требуемые характеристики объекта управления.

$$Tx''+x' = ku(t)$$

$$(x''=1/T(-x'+ku(t)))$$

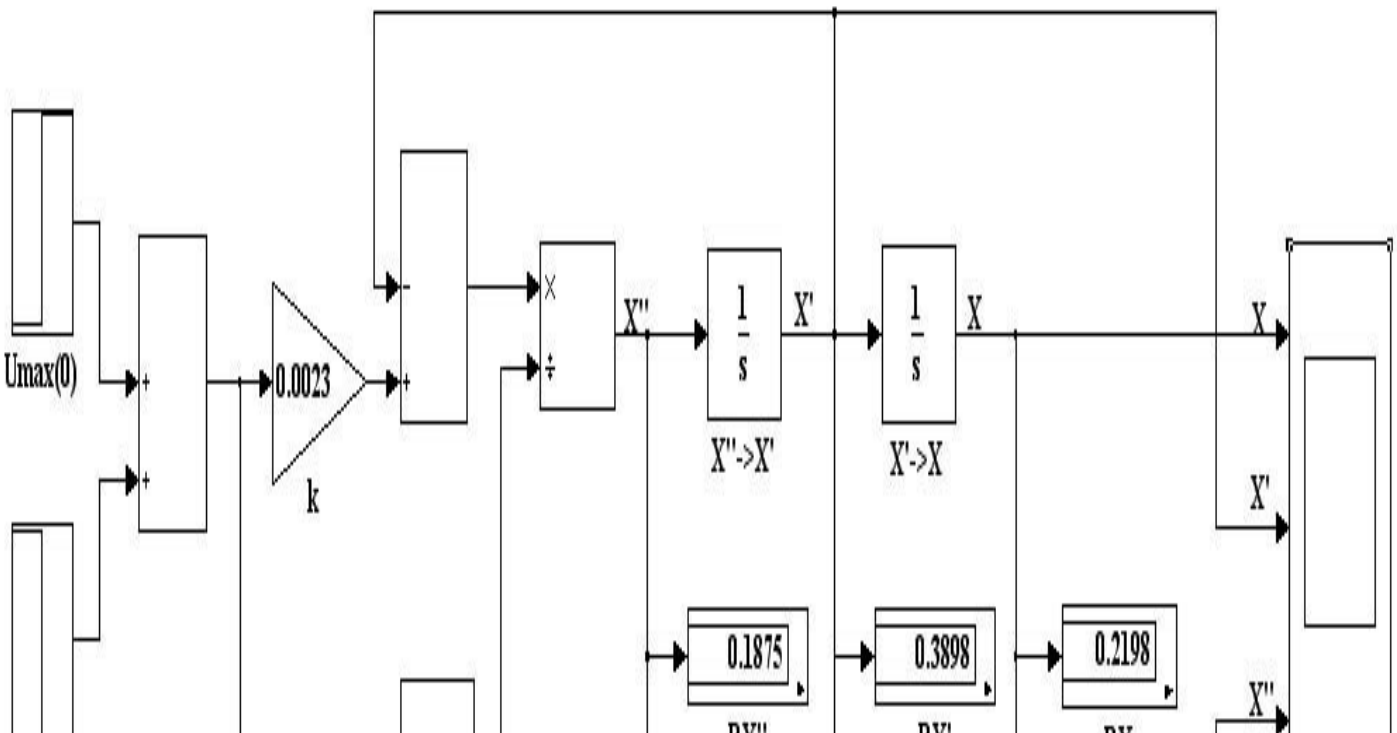

Рис. 5.2 Имитационная модель оптимального управления

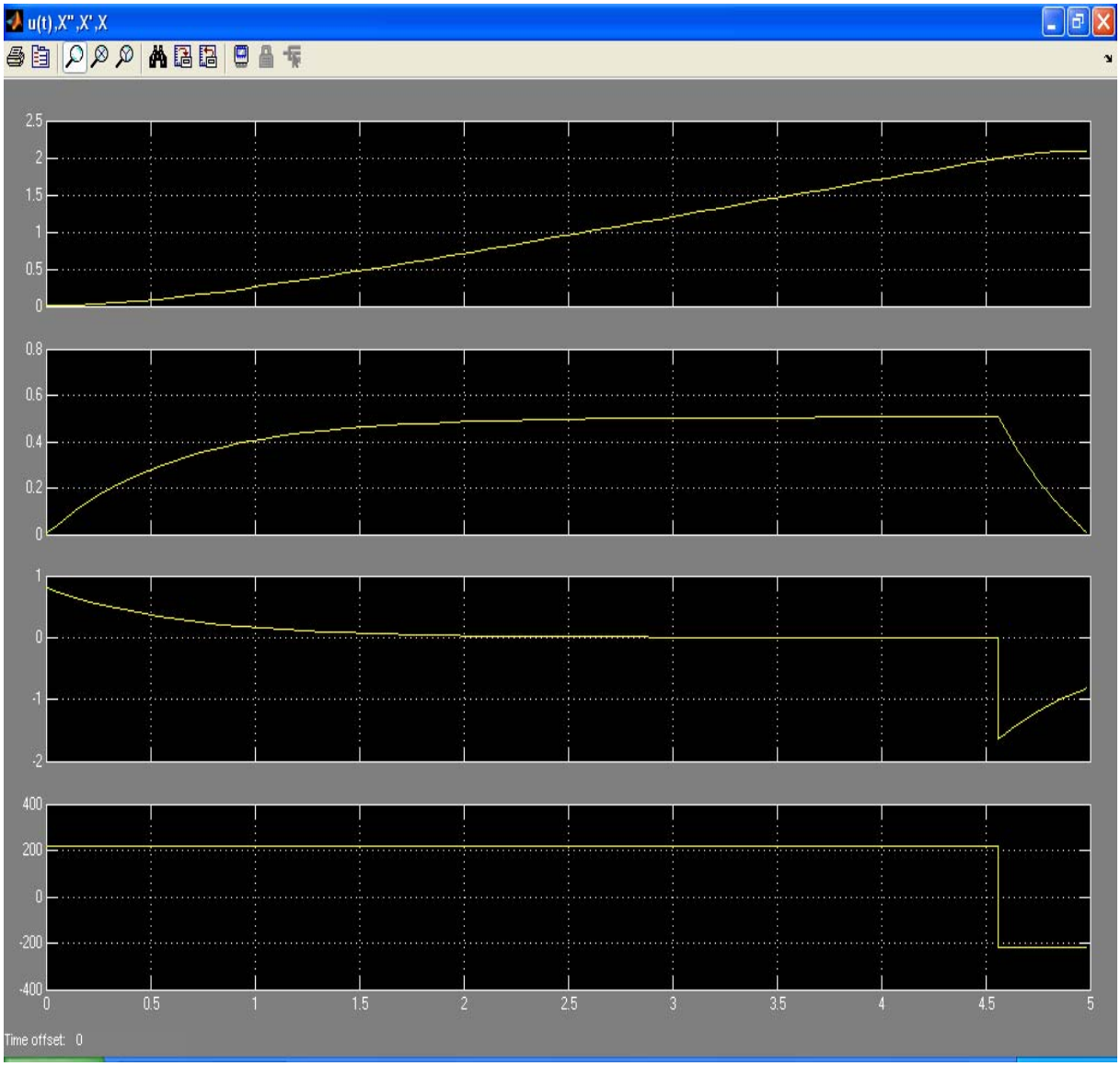

Рис. 5.3 Осциллограммы оптимального управления

5.5 Верификация математических моделей

Верификацию аналитической и имитационной моделей объекта управления произведем с помощью сопоставления переходных процессов, протекающих в этих моделях при оптимальном управляющем воздействии с ограничением U_{max} = 220 для следующих значений параметров объекта T = 0.6200 и k = 0.0023, когда конечное значение выходной величины x_n=2.09.

Таблица 5.1

Результаты расчета и моделирования переходного процесса для объекта управления при оптимальном управляющем воздействии u(t)

Текущее время t	Оптимальное управляющее воздействие u(t)	Управляемая величина x(t)		Скорость изменения управляемой величины x'(t)	
		Расчетное значение	Модельное значение	Расчетное значение	Модельное значение
0.0000	220	0.0000	0.0000	0.0000	0.0000
0.9120	220	0.2198	0.2198	0.3898	0.3898
1.8240	220	0.6258	0.6258	0.4793	0.4793
2.7360	220	1.0745	1.0745	0.4999	0.4999
3.6480	220	1.5330	1.5330	0.5046	0.5046
4.5600	-220	1.9338	1.9338	0.5057	0.5057
4.6459	-220	2.0315	2.0315	0.3748	0.3748
4.7318	-220	2.0587	2.0587	0.2608	0.2608
4.8177	-220	2.0768	2.0768	0.1616	0.1616
4.9036	-220	2.0868	2.0868	0.0752	0.0752
4.9895	-220	2.0900	2.0900	0.0000	0.0000

5.6 Варианты заданий и порядок их выполнения

1. Для рассматриваемого простейшего звена с помощью функции RKiw построить частотные графики.

2. Используя функцию LWSin , построить графики переходных функций при отсутствии возмущающих воздействий и нулевых начальных условиях, при толчкообразном и синусоидальном возмущениях, сравнить их с осциллограммами имитационной модели для таких же режимов и заполнить таблицы значений переходных функций.

3. По табл. 4.2 лабораторной работы № 4 выбрать два простейших звена и образовать из них систему, движение которой должно описываться обыкновенным дифференциальным уравнением порядка не ниже второго.

4. Для выбранной целевой системы вывести самостоятельно или получить с помощью компьютера аналитические выражения для вычисления передаточной и частной функций, а также функций переходной проводимости и веса.

5. С помощью пакета символьных вычислений Symbolic Math найти вид оптимального управления, обеспечивающего изменение выходной величины на заданное значение за минимальное время, используя функцию Гамильтона и принцип максимума Понтрягина, и вывести аналитические выражения для переходных функций системы, работающей в этом режиме.

6. Написать программы для вычисления амплитуды и фазы частот-ной функции, а также для расчёта переходного процесса системы при толчкообразном входном сигнале, используя выражения для функций переходной проводимости и веса.

7. На комплексной плоскости построить амплитудно-фазовую характеристику - годограф вектора K(iw) и оценить устойчивость целевой системы.

8. Для целевой системы найти аналитическое выражение для её ре-акции на синусоидальное возмущающее воздействие и выявить наличие собственных колебаний, а также возможность возникновения резонанса.

9. Построить имитационную модель целевой системы и произвести её моделирование при нулевых начальных условиях и отсутствии возмущающих сил, регистрируя переходной процесс с помощью соответствующих осциллографов.

10. Используя синусоидальный входной сигнал с переменной частотой, произвести моделирование системы и проверить её амплитудно-фазовые характеристики.

11. Используя ступенчатый входной сигнал Step, произвести моделирование системы и проверить её функции переходной проводимости и веса.

12. С помощью интеграла Дюамеля рассчитать реакцию целевой системы на заданное возмущающее воздействие и проверить результаты расчёта на имитационной модели этой системы.

13. Построить имитационную модель оптимальной целевой систе-мы, произвести её моделирование и регистрацию динамических процессов с помощью соответствующих осциллографов.

14. Построить имитационную модель для автоматического опре-деления параметров оптимального управления.

15. Произвести верификацию всех построенных моделей.

16. Построить имитационные модели для нахождения управлений, улучшающих динамические характеристики целевой системы при оптимальном управлении.

17. Оформить отчёт по лабораторной работе, применяя средства генерирования описания и результатов работы имитационных моделей, встроенные в пакет Simulink.

18. Если при выполнении какого-либо этапа исследования встре-тятся затруднения, рекомендуется сначала выполнить этот этап для системы-прототипа, описанный в лабораторной работе.

5.7 Оформление отчета по результатам исследований

Для завершения лабораторной работы необходимо сгенерировать отчет в формате HTML, затем преобразовать его в формат RTF с помощью текстового редактора, включит в него теоретические результаты, отформатировать текст и графические объекты, записать на дискету и в электронном виде предъявить преподавателю. Обосновать достоверность полученных результатов.

Лабораторная работа № 6

ОПТИМАЛЬНОЕ УПРАВЛЕНИЕ ЗВЕНЬЯМИ С ПЕРЕМЕННЫМИ ПАРАМЕТРАМИ

Цель работы: разработка аналитических моделей для определения характеристик и нахождения оптимального управления объектами, состоящими из простейших звеньев с переменными параметрами, реализация этих моделей в программной среде математической системы MATLAB и построение имитационных моделей с помощью пакета Simulink, верификация разработанных моделей и определение с их использованием характеристик объектов управления, а также нахождение оптимального управления для обеспечения изменения выходной величины на заданное значение за минимальное время.

6.1 Постановка задач исследования

В данной лабораторной работе рассматриваются объекты, движение которых описывается обыкновенными дифференциальными уравнениями с переменными коэффициентами, порядок которых не ниже второго. С точки зрения структуры это – либо отдельные простейшие звенья, либо несложные соединения таких звеньев. Целью управления является минимизация времени изменения выходной величины объекта на заданное значение за счет рационального выбора ограниченного по модулю управляющего воздействия U(t). Прежде чем приступить к поиску оптимального управления, необходимо определить характеристики объекта и провести всесторонние исследования динамики его поведения при различных возмущающих воздействиях, в частности оценить устойчивость звена или соединения, так как только для устойчивых объектов имеет смысл поиск оптимального управления. Аналитическое выражение для оптимального управления U(t) следует искать с помощью функции Гамильтона (гамильтониана) и принципа максимума Понтрягина. Расчет характеристик переходного процесса должен производиться с помощью системы MATLAB. Имитационная модель должна подтвердить расчеты по программе. Необходимо также разработать имитационную модель для автоматического определения параметров оптимального управления и имитационную модель для нахождения управлений, улучшающих динамические характеристики целевой системы при оптимальном управлении. Желательно также с помощью интеграла Дюамеля рассчитать реакцию системы на оптимальное управляющее воздействие и проверить это на имитационной модели.

6.2 Разработка аналитических моделей

В качестве прототипа рассмотрим объект управления, который описывается дифференциальным уравнением

$$T_1 \cdot \ddot{x} + \dot{x} = k \cdot u, \text{ при } 0 \le t < t_1, \quad (6.1)$$

$$T_2 \cdot \ddot{x} + \dot{x} = k \cdot u, \text{ при } t_1 < t \le t_2, \quad (6.2)$$

где T_1, T_2 и k - положительные постоянные.

Данная система уравнений возникает, например, при описании торможения двигателя противовключением, когда в его якорь включается добавочное сопротивление.

Алгоритм управления, переводящий объект из положения $x = 0$, $\dot{x} = 0$ при $t = 0$ в положение $x = x_n$, $\dot{x} = 0$ за минимальное время состоит из двух интервалов управления $\pm u_{\max}$. На управляющее воздействие наложено ограничение $|u| \le u_{\max}$. Определить момент переключения t_1, оптимальный переходной процесс $\vec{x}(t) = (x_1(t), x_2(t))$ и время перехода t_2. По точкам построить графики $x_1(t)$ и $x_2(t)$. В точке $t = t_1$ $x_1(t)$ и $x_2(t)$ должны рассчитываться дважды по разным формулам и эти значения должны совпадать.

Прежде чем приступить к поиску оптимального решения, необходимо исследовать характеристики объекта и устойчивость его состояния. Для этих целей следует воспользоваться формулами для передаточной, частотной и переходной функций и функции веса, которые были выведены в лабораторной работе № 5. В эти формулы надо подставить сначала T_1, а затем T_2. Построить графики частотных функций, а также графики переходных функций при толчкообразном и синусоидальном внешних воздействиях, используя М-функции RKiw и LWSin.

Теперь необходимо найти алгоритм оптимального управления для следующих исходных данных: $T_1 =$ 0.6200; $T_2 =$ 1.2000; $k =$ 0.0023; $u_{max} =$ 220; $x_n =$ 2.0900.

Сначала найдем решение уравнения на первом интервале. Обозначим: $x_1 = x$. Так как на нем $u = u_{\max}$, то дифференциальное уравнение будет иметь вид

$$T_1 \cdot \ddot{x}_1 + \dot{x}_1 = k \cdot u_{\max} \ . \quad (6.3)$$

Общим решением этого уравнения является функция

$$x_1(t) = k \cdot u_{\max} \cdot t + c_1 + c_2 \cdot \exp\left(-\frac{t}{T_1}\right). \quad (6.4)$$

Тогда

$$\dot{x}_1(t) = k \cdot u_{\max} - \frac{c_2}{T_1} \cdot \exp\left(-\frac{t}{T_1}\right). \quad (6.5)$$

Используя начальные условия $x_1 = 0$, $\dot{x}_1 = 0$ определим c_1 и c_2. Имеем следующую систему уравнений:

$$\begin{cases} c_1 + c_2 = 0, \\ k \cdot u_{\max} - \dfrac{c_2}{T_1} = 0. \end{cases} \quad (6.6)$$

Решая систему, получим $c_1 = -k \cdot u_{\max} \cdot T_1$, $c_2 = k \cdot u_{\max} \cdot T_1$. Подставляя найденные значения c_1 и c_2 в формулы (6.4) и (6.5), получим

$$x_1(t) = k \cdot u_{\max} \cdot t + k \cdot u_{\max} \cdot T_1 \cdot \left(\exp\left(-\frac{t}{T_1} \right) - 1 \right), \tag{6.7}$$

$$\dot{x}_1(t) = k \cdot u_{\max} \cdot \left(1 - \exp\left(-\frac{t}{T_1} \right) \right). \tag{6.8}$$

На втором интервале $(t_1 < t \le t_2)$ управление $u = -u_{\max}$. На этом интервале дифференциальное уравнение будет иметь вид

$$T_2 \cdot \ddot{x}_1 + \dot{x}_1 = -k \cdot u_{\max}. \tag{6.9}$$

Общим решением является функция

$$x_1(t) = -k \cdot u_{\max} \cdot t + c_3 + c_4 \cdot \exp\left(-\frac{t}{T_2} \right). \tag{6.10}$$

Тогда

$$\dot{x}_1(t) = -k \cdot u_{\max} - \frac{c_4}{T_2} \cdot \exp\left(-\frac{t}{T_2} \right). \tag{6.11}$$

Имеется четыре неизвестных c_3, c_4, t_1 и t_2. Для их определения используем два условия $x_1(t_2) = x_n$, $\dot{x}_1(t_2) = 0$. Метод стыкования дает еще уравнения. Тогда получим четыре уравнения с четырьмя неизвестными.

Сначала используем условия в точке t_2

$$x_1(t_2) = -k \cdot u_{\max} \cdot t_2 + c_3 + c_4 \cdot \exp\left(-\frac{t_2}{T_2} \right) = x_n, \tag{6.12}$$

$$\dot{x}_1(t_2) = -k \cdot u_{\max} - \frac{c_4}{T_2} \exp\left(-\frac{t_2}{T_2} \right) = 0. \tag{6.13}$$

Решая эту систему относительно c_3 и c_4, получим

$$c_4 = -k \cdot u_{\max} \cdot T_2 \cdot \exp\left(\frac{t_2}{T_2} \right), \tag{6.14}$$

$$c_3 = x_n + k \cdot u_{\max} \cdot t_2 + k \cdot u_{\max} \cdot T_2. \tag{6.15}$$

Подставляя эти значения в формулы (6.10) и (6.11), получим

$$x_1(t) = x_n + k \cdot u_{\max} \cdot (t_2 - t) + k \cdot u_{\max} \cdot T_2 \cdot \left(1 - \exp\left(\frac{t_2 - t}{T} \right) \right), \tag{6.16}$$

$$\dot{x}_1(t) = k \cdot u_{\max} \cdot \left(\exp\left(\frac{t_2 - t}{T} \right) - 1 \right). \tag{6.17}$$

Теперь будем считать, что $x_1(t)$ и $\dot{x}_1(t)$ в точке переключения управления t_1 непрерывны. Приравняем значения $x_1(t)$, найденное по формулам (6.7) и (6.16), а также значения $\dot{x}_1(t)$, найденное по формулам (6.8) и (6.17),

$$\begin{aligned} & k \cdot u_{\max} \cdot t_1 + k \cdot u_{\max} \cdot T_1 \cdot \left(\exp\left(-\frac{t_1}{T_1} \right) - 1 \right) = \\ & = x_n + k \cdot u_{\max} \cdot (t_2 - t_1) + k \cdot u_{\max} \cdot T_2 \cdot \left(1 - \exp\left(\frac{t_2 - t_1}{T} \right) \right), \end{aligned} \tag{6.18}$$

получим

$$k \cdot u_{\max} \cdot \left(1 - \exp\left(-\frac{t_1}{T_1}\right)\right) = k \cdot u_{\max} \cdot \left(\exp\left(\frac{t_2 - t_1}{T_2}\right) - 1\right). \qquad (6.19),$$

Первое уравнение, с учетом второго, можно записать в виде

$$k \cdot u_{\max} \cdot t_1 + k \cdot u_{\max} \cdot T_1 \cdot \left(\exp\left(-\frac{t_1}{T_1}\right) - 1\right) = x_n + k \cdot u_{\max} \cdot (t_2 - t_1) + k \cdot u_{\max} \cdot T_2 \cdot \left(\exp\left(-\frac{t_1}{T_1}\right) - 1\right).$$

Приведем подобные члены.

$$2k \cdot u_{\max} \cdot t_1 + k \cdot u_{\max} \cdot (T_1 - T_2) \cdot \left(\exp\left(-\frac{t_1}{T_1}\right) - 1\right) = x_n + k \cdot u_{\max} \cdot t_2. \qquad (6.20)$$

Второе уравнение системы можно записать в виде

$$\exp\left(\frac{t_2 - t_1}{T_2}\right) + \exp\left(-\frac{t_1}{T_1}\right) - 2 = 0. \qquad (6.21)$$

Найдем из уравнений (6.20) и (6.21) t_2 и приравняем их. Из уравнения (6.20) имеем

$$t_2 = 2t_1 + (T_1 - T_2) \cdot \left(\exp\left(-\frac{t_1}{T}\right) - 1\right) - \frac{x_n}{k \cdot u_{\max}}. \qquad (6.22)$$

Уравнение (6.21) умножим на $\exp\left(\frac{t_1}{T_2}\right)$, получим

$$\exp\left(\frac{t_2}{T_2}\right) = 2 \cdot \exp\left(\frac{t_1}{T_2}\right) - \exp\left(\frac{t_1}{T_2} - \frac{t_1}{T_1}\right). \qquad (6.23),$$

Тогда

$$\frac{t_2}{T_2} = \ln\left[\exp\left(\frac{t_1}{T_2}\right) \cdot \left(2 - \exp\left(-\frac{t_1}{T_1}\right)\right)\right]. \qquad (6.24),$$

Из последнего соотношения имеем

$$t_2 = T_2 \cdot \ln\left(2 - \exp\left(-\frac{t_1}{T_1}\right)\right) + t_1. \qquad (6.25)$$

Приравнивая правые части (6.22) и (6.25) после некоторых преобразований получим нелинейное уравнение относительно t_1

$$f(t_1) = T_2 \cdot \ln\left(2 - \exp\left(-\frac{t_1}{T_1}\right)\right) + \frac{x_n}{k \cdot u_{\max}} + (T_2 - T_1) \cdot \left(\exp\left(-\frac{t_1}{T_1}\right) - 1\right) - t_1 = 0. \qquad (6.26)$$

Применяя метод половинного деления для решения уравнения (6.26) с помощью компьютера для заданных исходных данных находим $t_1 = 4.3822$. Тогда по формуле (6.25) имеем $t_2 = 5.2135$. Обозначим $x_2(t) = \dot{x}_1(t)$. Тогда формулы оптимального процесса можно записать в виде

$$x_1(t) = \begin{cases} k \cdot u_{\max} \cdot \left[t + T_1 \cdot \left(\exp\left(-\frac{t}{T_1} \right) - 1 \right) \right], & при \quad 0 \le t < t_1 , \\ x_n + k \cdot u_{\max} \cdot \left[t_2 - t + T_2 \left(1 - \exp\left(\frac{t_2 - t}{T_2} \right) \right) \right], & при \quad t_1 < t \le t_2 , \end{cases} \quad (6.27),$$

$$x_2(t) = \begin{cases} k \cdot u_{\max} \cdot \left(1 - \exp\left(-\frac{t}{T_1} \right) \right), & при \quad 0 \le t < t_1 , \\ k \cdot u_{\max} \cdot \left(\exp\left(\frac{t_2 - t}{T_2} \right) - 1 \right), & при \quad t_1 < t \le t_2 . \end{cases} \quad (6.28),$$

Для заданных значений параметров и найденных значений t_1 и t_2 с помощью компьютера вычислены значения $x_1(t)$ и $x_2(t)$ в некоторых точках.

6.3 Программная реализация аналитических моделей

Расчет переходного процесса для оптимального управления производится с помощью следующей М-функции:

```
function [X,DX] = OptUpr(T1,T2,k,Umax,Xn)
%
%-- ФУНКЦИЯ ДЛЯ РАСЧЕТА ПЕРЕХОДНОГО ПРОЦЕССА
%--            ОПТИМАЛЬНОГО УПРАВЛЕНИЯ:
%

%-- 1.Расчет параметров математической модели:
%

   t1  =  4.3822        %-- время переключения управления;
   t2  =   5.2135       %-- время перехода системы в конечную точку;

%
%-- 2.Задание временных точек:
%

   t = [0.0000 0.8764 1.7529 2.6293 3.5058 4.3822 4.5484 4.7147 4.8810…
        5.0472 5.2135];
%
%-- 3.Расчет управляемой величины:
%

   for i = 1:1:11
      if t(i) <= t1
        X(i) = k*Umax*t(i)+k*Umax*T1*(exp(-t(i)/T1)-1);
      else
        X(i) = Xn+k*Umax*(t2-t(i)+T2-T2*exp((t2-t(i))/T2));
      end
```

```
   end

%
%-- 4.Расчет скорости изменения управляемой величины:
%

   for i = 1:1:11
      if t(i) <= t1
        DX(i) = k*Umax*(1-exp(-(t(i)/T1)));
      else
        DX(i) = k*Umax*(exp((t2-t(i))/T2)-1);
      end
   end
%

%-- 5.Визуализация управляемой величины и скорости ее изменения:
%

    subplot(2,2,1)
     plot(t,X,'r')
     xlabel('t')
     ylabel('X')

     subplot(2,2,2)
     plot(t,DX,'r')
     xlabel('t')
     ylabel('DX')

%-- 6.Конец функции OptUpr(T,k,Umax,Xn).
```

6.4 Построение имитационных моделей

В соответствии с математическим описанием объекта управления и поставленными задачами имитационная модель содержит два интегрирующих блока, необходимые генераторы сигналов, дисплеи, осциллографы, сумматоры и другие элементы (см. рис. 6.1).

Требуется построить эту модель, используя библиотеки блоков пакета Simulink, и настроить параметры блоков в соответствии с условиями задачи. Проверить работу модели можно путем её многократного запуска при изменении времени окончания работы. Модифицируя состав модели и изменяя режим её работы, можно получить все требуемые характеристики объекта управления.

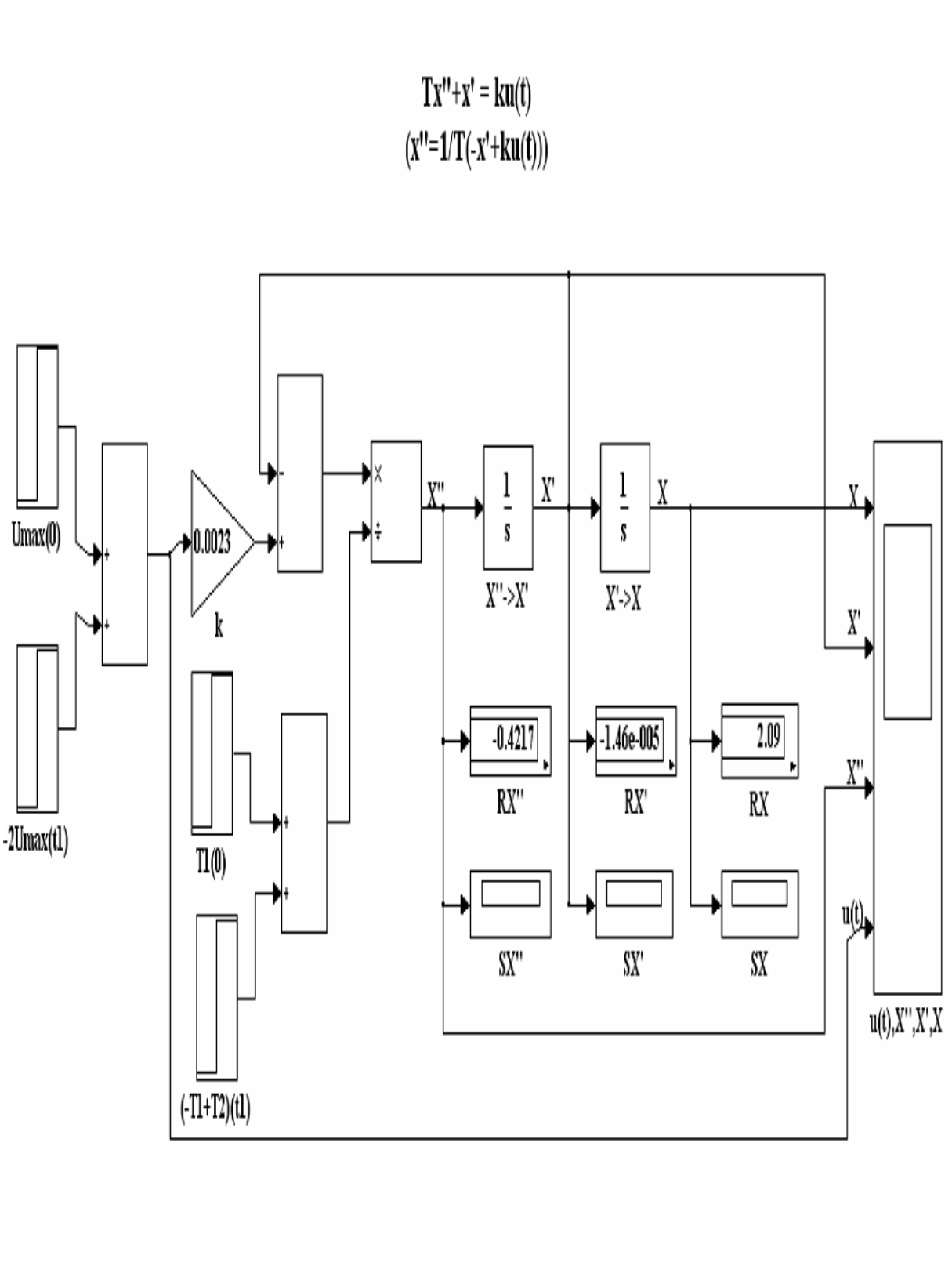

Рис 6.1 Имитационная модель оптимального управления

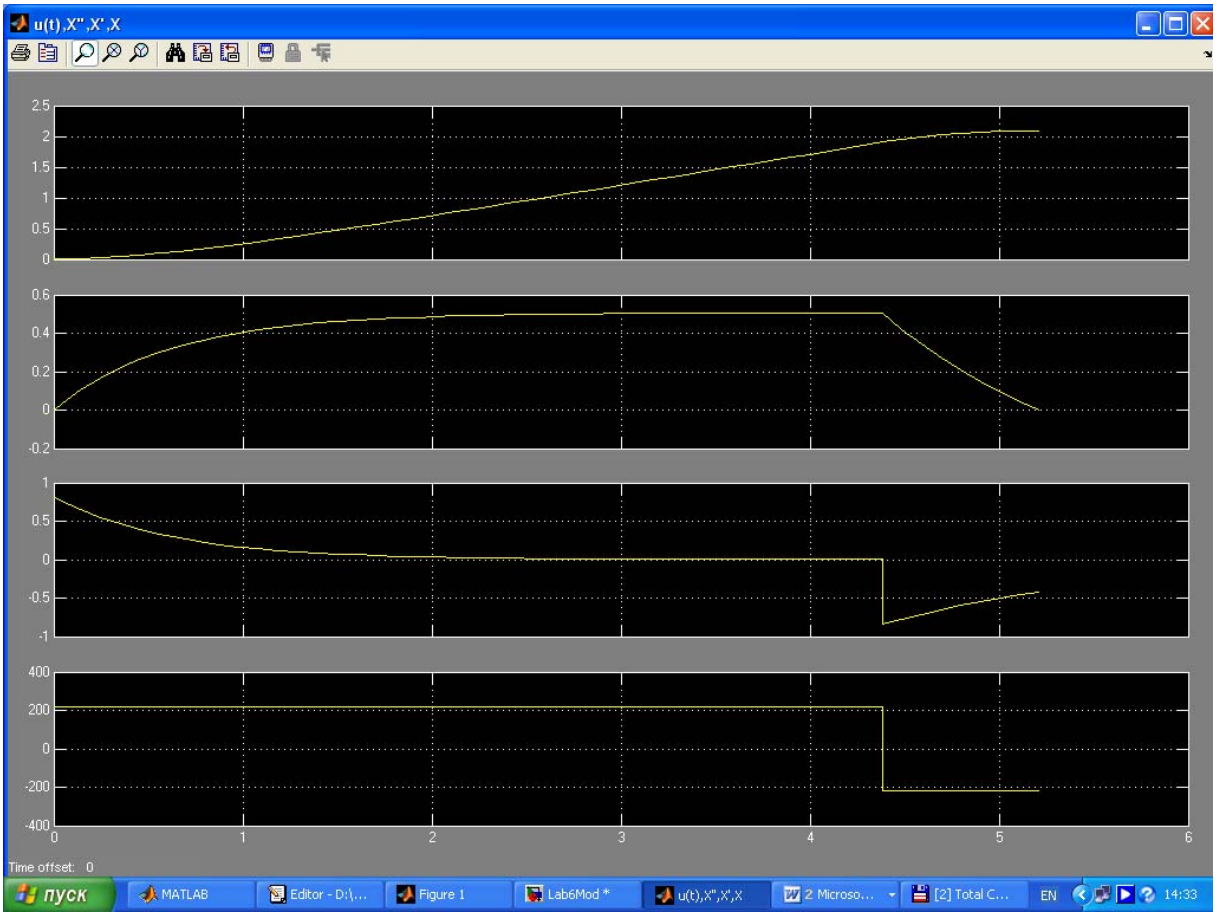

Рис. 6.2 Осциллограммы оптимального управления

6.5 Верификация математических моделей

Верификацию аналитической и имитационной моделей объекта управления произведем с помощью сопоставления переходных процессов, протекающих в этих моделях при оптимальном управляющем воздействии с ограничением U_{max} = 220, для следующих значений параметров объекта T = 0.6200 и k=0.0023, когда конечное значение выходной величины x_n равно 2.09. Расчетные значения времени переключения и времени перехода в конечную точку соответственно равны: t_1 = 4.3822; t_2 = 5.2135.

Таблица 6.1

Результаты расчета и моделирования переходного процесса для объекта управления при оптимальном управляющем воздействии u(t)

Текущее время t	Оптимальное управляющее воздействие u(t)	Управляемая величина x(t)		Скорость изменения управляемой величины x'(t)	
		Расчетное значение	Модельное значение	Расчетное значение	Модельное значение
0.0000	220	0.0000	0.0000	0.0000	0.0000
0.8764	220	0.2061	0.2061	0.3829	0.3829
1.7529	220	0.5918	0.5918	0.4761	0.4761
2.6293	220	1.0212	1.0212	0.4987	0.4987
3.5058	220	1.4613	1.4613	0.5042	0.5042
4.3822	-220	1.9039	1.9039	0.5056	0.5056
4.5484	-220	1.9769	1.9769	0.3747	0.3747
4.7147	-220	2.0295	2.0295	0.2608	0.2608
4.8810	-220	2.0644	2.0644	0.1616	0.1616
5.0472	-220	2.0839	2.0839	0.0752	0.0752
5.2135	-220	2.0900	2.0900	0.0000	0.0000

6.6 Варианты заданий и порядок их выполнения

1. Для рассматриваемого простейшего звена с помощью функции RKiw построить частотные графики.

2. Используя функцию LWSin , построить графики переходных функций при отсутствии возмущающих воздействий и нулевых начальных условиях, при толчкообразном и синусоидальном возмущениях, сравнить их с осциллограммами имитационной модели для таких же режимов и заполнить таблицы значений переходных функций.

3. По табл. 4.2 лабораторной работы № 4 выбрать два простейших звена и образовать из них систему, движение которой должно описываться обыкновенным дифференциальным уравнением порядка не ниже второго.

4. Для выбранной целевой системы вывести самостоятельно или получить с помощью компьютера аналитические выражения для вычисления передаточной и частной функций, а также функций переходной проводимости и веса.

5. С помощью пакета символьных вычислений Symbolic Math найти вид оптимального управления, обеспечивающего изменение выходной величины на заданное значение за минимальное время, используя функцию Гамильтона и принцип максимума Понтрягина, и вывести аналитические выражения для переходных функций системы, работающей в этом режиме.

6. Написать программы для вычисления амплитуды и фазы частот-ной функции, а также для расчёта переходного процесса системы при толчкообразном входном сигнале, используя выражения для функций переходной проводимости и веса.

7. На комплексной плоскости построить амплитудно-фазовую характеристику - годограф вектора K(iw) и оценить устойчивость целевой системы.

8. Для целевой системы найти аналитическое выражение для её ре-акции на синусоидальное возмущающее воздействие и выявить наличие собственных колебаний, а также возможность возникновения резонанса.

9. Построить имитационную модель целевой системы и произвести её моделирование при нулевых начальных условиях и отсутствии возмущающих сил, регистрируя переходной процесс с помощью соответствующих осциллографов.

10. Используя синусоидальный входной сигнал с переменной частотой, произвести моделирование системы и проверить её амплитудно-фазовые характеристики.

11. Используя ступенчатый входной сигнал Step, произвести моделирование системы и проверить её функции переходной проводимости и веса.

12. С помощью интеграла Дюамеля рассчитать реакцию целевой системы на заданное возмущающее воздействие и проверить результаты расчёта на имитационной модели этой системы.

13. Построить имитационную модель оптимальной целевой систе-мы, произвести её моделирование и регистрацию динамических процессов с помощью соответствующих осциллографов.

14. Построить имитационную модель для автоматического опре-деления параметров оптимального управления.

15. Произвести верификацию всех построенных моделей.

16. Построить имитационные модели для нахождения управлений, улучшающих динамические характеристики целевой системы при оптимальном управлении.

17. Оформить отчёт по лабораторной работе, применяя средства генерирования описания и результатов работы имитационных моделей, встроенные в пакет Simulink.

18. Если при выполнении какого-либо этапа исследования встре-тятся затруднения, рекомендуется сначала выполнить этот этап для системы-прототипа, описанный в лабораторной работе.

6.7 Оформление отчета по результатам исследований

Для завершения лабораторной работы необходимо сгенерировать отчет в формате HTML, затем преобразовать его в формат RTF с помощью текстового редактора, включит в него теоретические результаты, отформатировать текст и графические объекты, записать на дискету и в электронном виде предъявить преподавателю. Обосновать достоверность полученных результатов.

ОПТИМАЛЬНОЕ УПРАВЛЕНИЕ ПРОСТЕЙШИМИ ЗВЕНЬЯМИ С ДИНАМИЧЕСКИМИ ОГРАНИЧЕНИЯМИ

Цель работы: разработка аналитических моделей для определения характеристик и нахождения оптимального управления объектами с динамическими ограничениями, состоящими из простейших звеньев, реализация этих моделей в программной среде математической системы MATLAB и построение имитационных моделей с помощью пакета Simulink, верификация разработанных моделей и определение с их использованием характеристик объектов управления, а также нахождение оптимального управления для обеспечения изменения выходной величины на заданное значение за минимальное время.

7.1 Постановка задач исследования

В данной лабораторной работе рассматриваются объекты, движение которых описывается обыкновенными дифференциальными уравнениями с постоянными коэффициентами, порядок которых не ниже второго. С точки зрения структуры это – либо отдельные простейшие звенья, либо несложные соединения таких звеньев. Целью управления является минимизация времени изменения выходной величины объекта на заданное значение за счет рационального выбора ограниченного по модулю управляющего воздействия U(t) и при ограничении скорости изменения управляемой величины. Прежде чем приступить к поиску оптимального управления, необходимо определить характеристики объекта и провести всесторонние исследования динамики его поведения при различных возмущающих воздействиях, в частности оценить устойчивость звена или соединения, так как только для устойчивых объектов имеет смысл поиск оптимального управления. Аналитическое выражение для оптимального управления U(t) следует искать с помощью функции Гамильтона (гамильтониана) и принципа максимума Понтрягина. Расчет характеристик переходного процесса должен производиться с помощью системы MATLAB. Имитационная модель должна подтвердить расчеты по программе. Необходимо также разработать имитационную модель для автоматического определения параметров оптимального управления и имитационную модель для нахождения управлений, улучшающих динамические характеристики целевой системы при оптимальном управлении. Желательно также с помощью интеграла Дюамеля рассчитать реакцию системы на оптимальное управляющее воздействие и проверить это на имитационной модели.

7.2 Разработка аналитических моделей

В качестве прототипа рассмотрим объект управления, который описывается дифференциальным уравнением

$$T \cdot \ddot{x} + \dot{x} = k \cdot u, \qquad (7.1)$$

где T и k – положительные постоянные. Требуется перевести объект из положения $x = 0$, $\dot{x} = 0$ при $t = 0$ в положение $x = x_n$, $\dot{x} = 0$ за минимальное время при ограничении скорости $\dot{x} \le \dot{x}_{\max}$. На управляющее воздействие наложено ограничение $|u| \le u_{\max}$. Известно, что алгоритм управления должен состоять из интервала разгона, когда $u = +u_{\max}$, интервала, на котором u пропорционально скорости ограничения $\dot{x}_{\max}$, а скорость движения равна $\dot{x}_{\max}$, и интервала торможения, на котором $u = -u_{\max}$. Определить время разгона t_1, время движения с постоянной скоростью t_2, время торможения t_3, оптимальный переходной процесс $\vec{x}(t) = (x_1(t), x_2(t))$ и время перехода $t_4 = t_1 + t_2 + t_3$. По точкам построить графики $x_1(t)$ и $x_2(t)$. Для этого интервал времени от 0 до t_1, от t_1 до $t_1 + t_2$, от $t_1 + t_2$ до $t_1 + t_2 + t_3$ разделить на 5 равных частей и вычислить значения $x_1(t)$ и $x_2(t)$ в соответствующих точках. В точках $t = t_1$ и $t = t_1 + t_2$ $x_1(t)$ и $x_2(t)$ должны рассчитываться дважды по разным формулам и эти значения должны совпадать.

Прежде чем приступить к поиску оптимального решения, необходимо оценить характеристики объекта и устойчивость его состояния.

Передаточная и частотная функции объекта имеют вид:

$$K(p) = \frac{1}{p(Tp + 1)}, \qquad (7.2)$$

$$K(i\omega) = \frac{1}{i\omega(Ti\omega + 1)}. \qquad (7.3)$$

Функция переходной проводимости объекта находится как решение дифференциального уравнения (7.1) при толчкообразном внешнем воздействии

$$T\ddot{x} + \dot{x} = 1(t). \qquad (7.4)$$

Используя прямое и обратное преобразование Лапласа, получаем

$$\Lambda(t) = -T + t + Te^{-\frac{t}{T}}, \qquad (7.5)$$

или

$$\Lambda(t) = t + T(e^{-\frac{t}{T}} - 1) \qquad (7.6)$$

Функция веса объекта получается путем дифференцирования функции переходной проводимости и имеет вид

$$W(t) = 1 - e^{-\frac{t}{T}}. \qquad (7.7)$$

Реакция объекта на синусоидальное возмущающее воздействие определяется путем решения дифференциального уравнения

$$T\ddot{x} + \dot{x} = A_0 \sin \omega_0 t \qquad (7.8)$$

и имеет следующее аналитическое выражение:

$$x(t)=\frac{A_0\omega_0}{T}\left(\frac{T}{\omega_0^2}-\frac{T^3}{(1+k_0^2T^2}e^{-\frac{t}{T}}-\frac{1}{2\omega_0^2}\left(\frac{1}{-i\omega_0+\frac{1}{T}}e^{-i\omega_0 t}+\frac{1}{i\omega_0+\frac{1}{T}}e^{i\omega_0 t}\right)\right). \qquad (7.9)$$

Рассмотрим порядок вывода этого выражения.

Используя преобразования Лапласа при нулевых начальных условиях, получаем

$$(Tp^2+p)\cdot x(p)=\frac{A_0\omega_0}{p^2+\omega_0^2}. \qquad (7.10)$$

Отсюда находим изображение регулируемой величины

$$x(p)=\frac{A_0\omega_0}{p(Tp+1)(p^2+\omega_0^2)}. \qquad (7.11)$$

Корни характеристического уравнения легко находятся и равны: $p_1=0$; $p_2=-\frac{1}{T}$; $p_3=-i\omega_0$; $p_1=i\omega_0$.

Разложим дробь на простейшие дроби:

$$\frac{1}{(p-0)(p-(-\frac{1}{T}))(p-(-i\omega_0))(p-i\omega_0)}=\frac{A}{p-0}+\frac{B}{p+\frac{1}{T}}+\frac{C}{p+i\omega_0}+\frac{D}{p-i\omega_0}. \qquad (7.12)$$

Приведя это уравнение к общему знаменателю и отбросив его, получим:

$$A(p+\frac{1}{T})(p^2+\omega_0^2)+Bp(p^2+\omega_0^2)+Cp(p+\frac{1}{T})(p-i\omega_0)+Dp(p+\frac{1}{T})(p+i\omega_0)=1. \qquad (7.13)$$

Подставим в это уравнение корни и найдем коэффициенты разложения:

$$A=\frac{T}{\omega_0^2};\quad B=-\frac{T^3}{1+\omega_0T^2};$$
$$D=-\frac{1}{2\omega_0^2(i\omega_0+\frac{1}{T})}; \qquad (7.14)$$
$$C=-\frac{1}{2\omega_0^2(-i\omega_0+\frac{1}{T})}\cdot$$

Переходя с помощью таблиц от изображения к оригиналу, получаем следующее выражение для вынужденных колебаний объекта:

$$x(t)=\left(\frac{A_0\omega_0}{T}\right)\left(\frac{T}{\omega_0^2}-\frac{T^3}{1+\omega_0^2T^2}\right)e^{-\frac{t}{T}}-\left(\frac{1}{\omega_0^2(\omega_0^2+\frac{1}{T^2})}\right)\left(\omega_0\sin\omega_0t+\frac{1}{T}\cos\omega_0t\right) \qquad (7.15)$$

Теперь необходимо найти алгоритм оптимального управления. Расчеты произвести для следующих данных: $T=0.62$; $k=0.0023$; $u_{\max}=220$; $x_n=2.09$; $\dot{x}_{\max}=0.4$.

Обозначим $x_1(t)=x(t)$, $x_2(t)=\dot{x}_1(t)$. На первом участке $(0\le t\le t_1)$ $u=u_{\max}$. Уравнение (20) имеет вид

$$T \cdot \ddot{x}_1 + \dot{x}_1 = k \cdot u_{\max}.$$

Общим решением этого уравнения является функция

$$x_1(t) = k \cdot u_{\max} \cdot t + c_1 + c_2 \cdot \exp\left(-\frac{t}{T}\right). \quad (7.16)$$

Для определения c_1 и c_2 используем начальные условия: $x_1(0) = 0$; $\dot{x}_1(0) = 0$. Так как

$$x_1(t) = k \cdot u_{\max} - \frac{c_2}{T} \cdot \exp\left(-\frac{t}{T}\right), \quad (7.17)$$

то получим следующую систему уравнений:

$$\begin{cases} c_1 + c_2 = 0, \\ k \cdot u_{\max} - \dfrac{c_2}{T} = 0. \end{cases}$$

Решая полученную систему, найдем $c_1 = -k \cdot u_{\max} \cdot T$; $c_2 = k \cdot u_{\max} \cdot T$. Подставляя эти значения в (7.16) и (7.17) получим:

$$x_1(t) = k \cdot u_{\max} \cdot t + k \cdot u_{\max} \cdot T \cdot \left(\exp\left(-\frac{t}{T}\right) - 1\right), \quad (7.18)$$

$$\dot{x}_1(t) = k \cdot u_{\max} - k \cdot u_{\max} \cdot \exp\left(-\frac{t}{T}\right), \quad (7.19)$$

Определим время разгона объекта t_1 из условия: $\dot{x}_1(t_1) = \dot{x}_{\max}$. В нашем случае уравнение имеет вид $k \cdot u_{\max} - k \cdot u_{\max} \cdot \exp\left(-\frac{t_1}{T}\right) = \dot{x}_{\max}$; отсюда следует $k \cdot u_{\max} \cdot \exp\left(-\frac{t}{T}\right) = k \cdot u_{\max} - \dot{x}_{\max}$. Тогда

$$t_1 = -T \cdot \ln\left(1 - \frac{\dot{x}_{\max}}{k \cdot u_{\max}}\right). \quad (7.20)$$

Для наших исходных данных $t_1 = -0.62 \cdot \ln\left(1 - \dfrac{0.4}{0.0023 \cdot 220}\right) = 0.9691$.

Теперь найдем время торможения объекта. На третьем участке, а уравнение (7.1) будет иметь вид:

$$T \cdot \ddot{x}_1 + \dot{x}_1 = -k \cdot u_{\max}. \quad (7.21)$$

Общим решением этого уравнения является функция

$$x_1(t) = -k \cdot u_{\max} \cdot t + c_3 + c_4 \cdot \exp\left(-\frac{t}{T}\right). \quad (7.22)$$

В формуле (7.22) будем считать, что $(0 \le t \le t_3)$. При расчете $x_1(t)$ на всем интервале управления будем учитывать путь, пройденный на первых двух участках. Начальными условиями для уравнения (7.21) будут $x_1(0) = 0$; $\dot{x}_1(0) = \dot{x}_{\max}$. Так как

$$\dot{x}_1(t) = -k \cdot u_{\max} - \frac{c_4}{T} \cdot \exp\left(-\frac{t}{T}\right), \quad (7.23)$$

то для определения постоянных интегрирования c_3 и c_4 имеем следующую систему уравнений

$$\begin{cases} c_3 + c_4 = 0, \\ -k \cdot u_{\max} - \dfrac{c_4}{T} = \dot{x}_{\max}. \end{cases}$$

Решая систему, находим $c_3 = T \cdot \left(k \cdot u_{\max} + \dot{x}_{\max}\right)$, $c_4 = -T \cdot \left(k \cdot u_{\max} + \dot{x}_{\max}\right)$. Подставляя найденные значения в формулы (7.22) и (7.23), получим

$$x_1(t) = -k \cdot u_{\max} \cdot t + T \cdot \left(k \cdot u_{\max} + \dot{x}_{\max}\right) \cdot \left(1 - \exp\left(-\frac{t}{T}\right)\right), \tag{7.24}$$

$$\dot{x}_1(t) = -k \cdot u_{\max} + \left(k \cdot u_{\max} + \dot{x}_{\max}\right) \cdot \exp\left(-\frac{t}{T}\right). \tag{7.25}$$

Найдем время торможения объекта из условия $\dot{x}_1(t_3) = 0$. Из уравнения (7.25) имеем

$$-k \cdot u_{\max} + \left(k \cdot u_{\max} + \dot{x}_{\max}\right) \cdot \exp\left(-\frac{t_3}{T}\right) = 0.$$

Отсюда получаем $\exp\left(-\frac{t_3}{T}\right) = \frac{k \cdot u_{\max}}{k \cdot u_{\max} + \dot{x}_{\max}}$, а тогда

$$t_3 = T \cdot \ln\left(\frac{k \cdot u_{\max} + \dot{x}_{\max}}{k \cdot u_{\max}}\right). \tag{7.26}$$ Для

наших исходных данных имеем $t_3 = 0.62 \cdot \ln\left(\frac{0.0023 \cdot 220 + 0.4}{0.0023 \cdot 220}\right) = 0.3612$.

По формуле (7.18) найдем путь, пройденный объектом во время разгона

$$s_1 = k \cdot u_{\max}\left[t_1 + T \cdot \left(\exp\left(-\frac{t_1}{T}\right) - 1\right)\right]. \tag{7.27}$$

Подставим в формулу (32) исходные и найденные данные

$s_1 = 0.0023 \cdot 220\left[0.9691 + 0.62 \cdot \left(e^{\frac{-0.9691}{0.62}} - 1\right)\right] = 0.2424$.

По формуле (7.24) найдем путь, пройденный объектом за время торможения

$$s_3 = -k \cdot u_{\max} \cdot t_3 + T \cdot \left(k \cdot u_{\max} + \dot{x}_{\max}\right) \cdot \left(1 - \exp\left(-\frac{t_3}{T}\right)\right). \tag{7.28}$$

Для наших данных получим

$s_3 = -0.0023 \cdot 220 \cdot 0.3612 + 0.62 \cdot \left(0.0023 \cdot 220 + 0.4\right) \cdot \left(1 - e^{\frac{-0.3612}{0.62}}\right) = 0.0653$. Путь, пройденный на втором участке

$$s_2 = x_n - s_1 - s_3. \tag{7.29}$$

Подставляя исходные и полученные данные, имеем $s_2 = 2.09 - 0.2424 - 0.0653 = 1.7824$.

Время движения объекта на втором участке управления

$$t_2 = \frac{s_2}{\dot{x}_{\max}}. \tag{7.30}$$ Для

наших данных $t_2 = \frac{1.7824}{0.4} = 4.4559$. Общее время оптимального управления $t_4 = t_1 + t_2 + t_3$. В нашем случае $t_4 = 0.9691 + 4.4559 + 0.3612 = 5.4250$.

На втором участке уравнение (20) будет иметь вид:

$$\dot{x}_1 = k \cdot u, \tag{7.31}$$

так как здесь $\ddot{x}_1 = 0$. Но на этом участке $\dot{x}_1 = \dot{x}_{\max}$. Тогда управление на этом участке

$$u = \frac{\dot{x}_{\max}}{k}. \qquad (7.32)$$

В нашем случае $u = \dfrac{0.4}{0.0023} = 173.9100$.

Обозначим $x_2(t) = \dot{x}_1(t)$. Тогда полученные результаты можно записать в виде

$$u = \begin{cases} 220, & \text{при } 0 \le t \le t_1, \\ 173.91, & \text{при } t_1 < t < t_1 + t_2, \\ -220, & \text{при } t_1 + t_2 < t < t_1 + t_2 + t_3, \end{cases} \qquad (7.33)$$

$$x_1(t) = \begin{cases} k \cdot u_{\max}\left[t + T \cdot \left(\exp\left(-\dfrac{t}{T} \right) - 1 \right) \right], & \text{при } 0 \le t \le t_1, \\ s_1 + \dot{x}_{\max} \cdot (t - t_1), & \text{при } t_1 \le t \le t_1 + t_2, \\ s_1 + s_2 - k \cdot u_{\max} \cdot (t - t_1 - t_2) + T \cdot (k \cdot u_{\max} + \dot{x}_{\max}) \cdot \left(1 - \exp\left(-\dfrac{t - t_1 - t_2}{T} \right) \right), & \\ & \text{при } t_1 + t_2 \le t \le t_1 + t_2 + t_3. \end{cases}$$

$$x_2(t) = \begin{cases} k \cdot u_{\max} \cdot \left(1 - \exp\left(-\dfrac{t}{T} \right) \right), & \text{при } 0 \le t \le t_1, \\ \dot{x}_{\max}, & \text{при } t_1 \le t \le t_1 + t_2, \\ -k \cdot u_{\max} + (k \cdot u_{\max} + \dot{x}_{\max}) \cdot \exp\left(-\dfrac{t - t_1 - t_2}{T} \right), & \text{при } t_1 + t_2 \le t \le t_1 + t_2 + t_3. \end{cases}$$

Ниже приведены результаты расчетов на компьютере всех искомых величин. Результаты показывают, что функции $x_1(t)$ и $x_2(t)$ непрерывны на всем интервале управления, в том числе и в точках переключения управления $t = t_1$ и $t = t_1 + t_2$.

7.3 **Программная реализация аналитических моделей**

Расчет переходного процесса для оптимального управления производится с помощью следующей М-функции:

```
function [X,DX] = OptUpr(T,k,Umax,Vmax,Xn)
%
%-- Функция для расчета переходного процесса оптимального управления
%
%
%-- 1.Задание параметров математической модели:
%

   t1 = 0.9691;        %-- время разгона объекта управления;
   t2 = 4.4559;        %-- время движения со скоростью Vmax;
   t3 = 0.3612;        %-- время торможения;
   t4 = 5.7862;        %-- время перехода системы в конечную точку;
```

```
   S1 = 0.2424;        %-- путь разгона;
   S2 = 1.7824;        %-- путь движения со скоростью Vmax;
   S3 = 0.0653;        %-- путь торможеня;
   S4 = 2.0900;        %-- полный путь,равный S1+S2+S3;

%
%-- 2.Задание временных точек:
%

   t = [0.0000 0.1938 0.3876 0.5815 0.7753 0.9691 1.8603 2.7515...
      3.6427 4.5339 5.4250 5.4973 5.5695 5.6417 5.7140 5.7862];


%
%-- 3.Расчет управляемой величины:
%
   for i = 1:1:16
      if t(i) <= t1
        X(i) = k*Umax*(t(i)+T*(exp(-t(i)/T)-1));
      end
      if t1 < t(i) & t(i) <= t1+t2
         X(i) = S1+Vmax*(t(i)-t1);
      end
      if t1+t2 < t(i) & t(i) <= t1+t2+t3
        X(i) = S1+S2-k*Umax*(t(i)-t1-t2)+T*(k*Umax+Vmax)*(1-exp(-(t(i)-…
                t1-t2)/T));
      end
   end

%
%-- 4.Расчет скорости изменения управляемой величины:
%
   for i = 1:1:16
      if t(i) <= t1
        DX(i) = k*Umax*(1-exp(-(t(i)/T)));
      end
      if t1 < t(i) & t(i) <= t1+t2
        DX(i) = Vmax;
      end
      if t1+t2 < t(i) & t(i) <= t1+t2+t3
        DX(i) = -k*Umax+(k*Umax+Vmax)*exp(-(t(i)-t1-t2)/T);
      end
   end

%
%-- 5.Визуализация управляемой величины и скорости ее изменения:
%
```

```
subplot(2,2,1)
plot(t,X,'r')
xlabel('t')
ylabel('X')

subplot(2,2,2)
plot(t,DX,'r')
xlabel('t')
ylabel('DX')

%
%-- 6.Конец функции OptUpr(T,k,Umax,Vmax,Xn).
                                                    %
```

7.4 Построение имитационных моделей

В соответствии с математическим описанием объекта управления и поставленными задачами имитационная модель содержит два интегрирующих блока, необходимые генераторы сигналов, дисплеи, осциллографы, сумматоры и другие элементы (см. рис. 7.1).

Требуется построить эту модель, используя библиотеки блоков пакета Simulink, и настроить параметры блоков в соответствии с условиями задачи. Проверить работу модели можно путем её многократного запуска при изменении времени окончания работы. Модифицируя состав модели и изменяя режим её работы, можно получить все требуемые характеристики объекта управления.

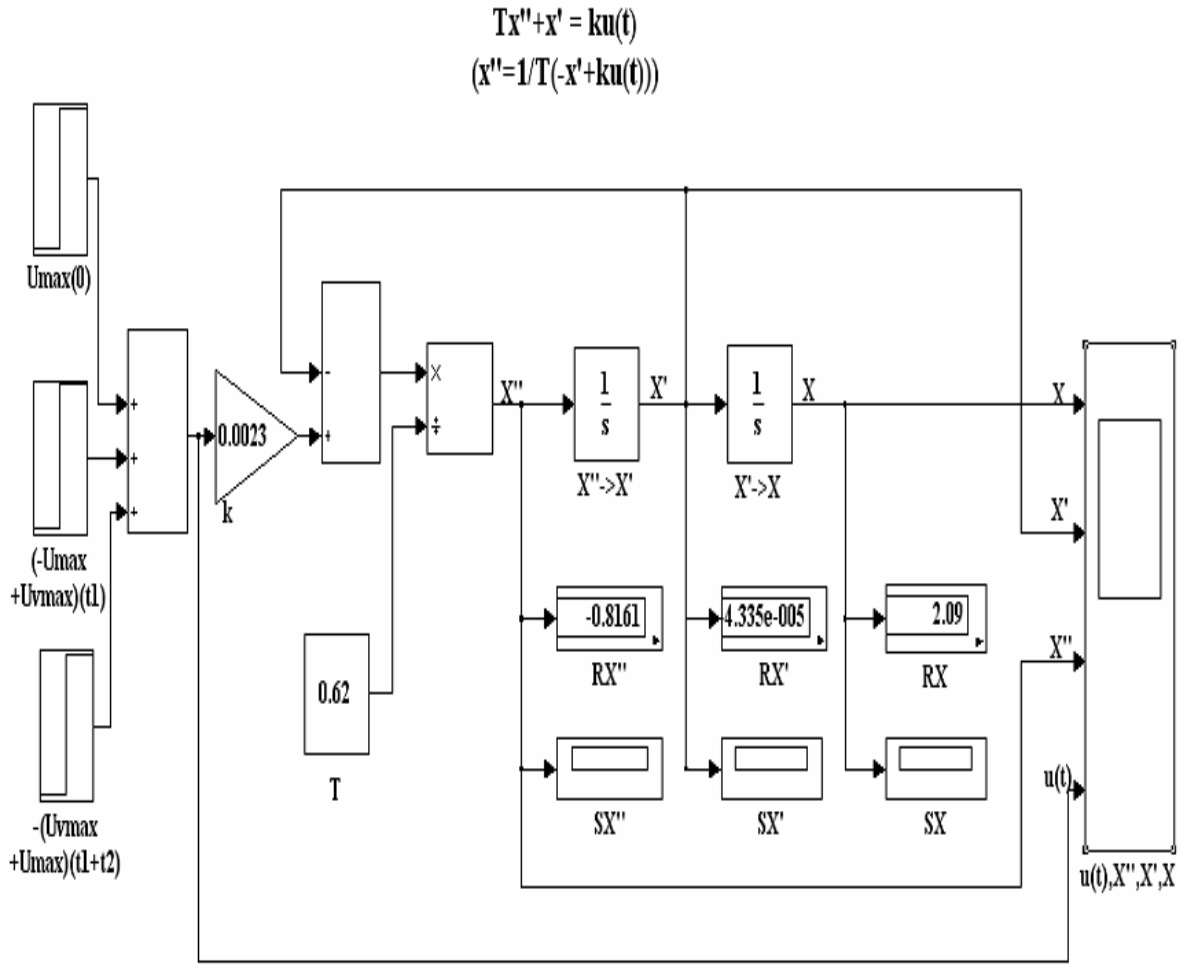

Рис 7.1 Имитационная модель оптимального управления

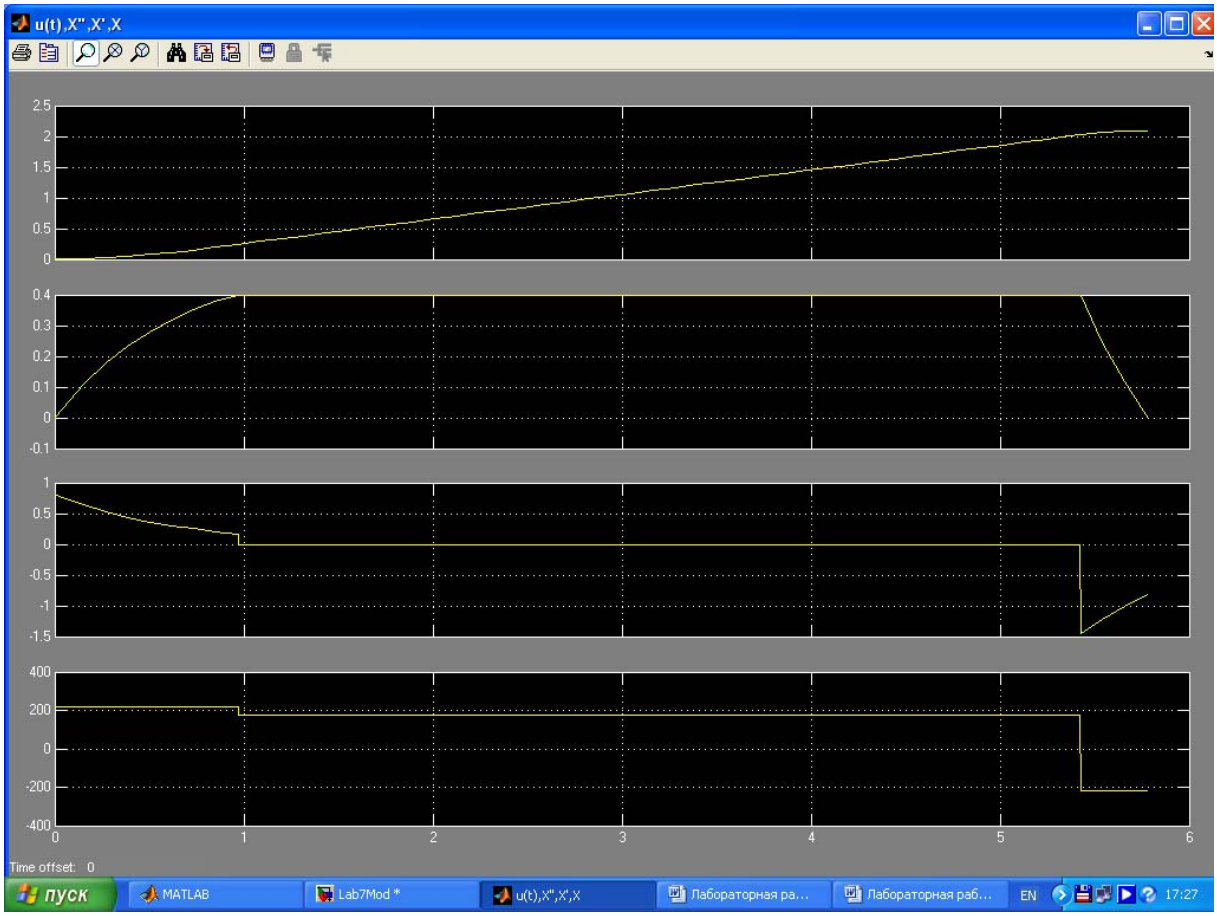

Рис. 7.2 Осциллограммы оптимального управления

7.5 Верификация математических моделей

Верификацию аналитической и имитационной моделей объекта управления произведем с помощью сопоставления переходных процессов, протекающих в этих моделях при оптимальном управляющем воздействии с ограничениями U_{max} = 220 и Vmax = 0.4, для следующих значений параметров объекта T = 0.6200 и k=0.0023, когда конечное значение выходной величины x_n равно 2.09. Расчетные значения времени переключения и времени перехода в конечную точку соответственно равны: $t_1 =$ 4.3822; t_2 = 5.2135. Расчетные параметры задачи:

$$t_1 = 0.9691;\quad t_2 = 4.4559;\quad t_3 = 0.3612;\quad t_4 = 5.7862;$$
$$s_1 = 0.2424;\ s_2 = 1.7824;\quad s_3 = 0.0653;$$

Таблица 7.1

Результаты расчета и моделирования переходного процесса для объекта управления при оптимальном управляющем воздействии u(t)

Текущее время t	Оптимальное управляющее воздействие u(t)	Управляемая величина x(t)		Скорость изменения управляемой величины x'(t)	
		Расчетное значение	Модельное Значение	Расчетное значение	Модельное значение
0.0000	220.0000	0.0000	0.0000	0.0000	0.0000
0.1938	220.0000	0.0138	0.0138	0.1358	0.1358
0.3876	220.0000	0.0503	0.0503	0.2352	0.2352
0.5815	220.0000	0.1033	0.1033	0.3079	0.3079
0.7753	220.0000	0.1684	0.1684	0.3611	0.3611
0.9691	220.0000	0.2424	0.2424	0.4000	0.4000
1.8603	173.9130	0.5989	0.5989	0.4000	0.4000
2.7515	173.9130	0.9554	0.9554	0.4000	0.4000
3.6427	173.9130	1.3118	1.3118	0.4000	0.4000
4.5339	173.9130	1.6683	1.6683	0.4000	0.4000
5.4250	173.9130	2.0248	2.0248	0.4000	0.4000
5.4973	-220.0000	2.0500	2.0500	0.3003	0.3003
5.5695	-220.0000	2.0685	2.0685	0.2116	0.2116
5.6417	-220.0000	2.0808	2.0808	0.1328	0.1328
5.7140	-220.0000	2.0878	2.0878	0.0625	0.0625
5.7862	-220.0000	2.0900	2.0900	0.0000	0.0000

7.6 Варианты заданий и порядок их выполнения

1. Для рассматриваемого простейшего звена с помощью функции RKiw построить частотные графики.

2. Используя функцию LWSin , построить графики переходных функций при отсутствии возмущающих воздействий и нулевых начальных условиях, при толчкообразном и синусоидальном возмущениях, сравнить их с осциллограммами имитационной модели для таких же режимов и заполнить таблицы значений переходных функций.

3. По табл. 4.2 лабораторной работы № 4 выбрать два простейших звена и образовать из них систему, движение которой должно описываться обыкновенным дифференциальным уравнением порядка не ниже второго.

4. Для выбранной целевой системы вывести самостоятельно или получить с помощью компьютера аналитические выражения для вычисления передаточной и частной функций, а также функций переходной проводимости и веса.

5. С помощью пакета символьных вычислений Symbolic Math найти вид оптимального управления, обеспечивающего изменение выходной величины на заданное значение за минимальное время, используя функцию Гамильтона и принцип максимума Понтрягина, и вывести аналитические выражения для переходных функций системы, работающей в этом режиме.

6. Написать программы для вычисления амплитуды и фазы частот-ной функции, а также для расчёта переходного процесса системы при толчкообразном

входном сигнале, используя выражения для функций переходной проводимости и веса.

7. На комплексной плоскости построить амплитудно-фазовую характеристику - годограф вектора K(iw) и оценить устойчивость целевой системы.

8. Для целевой системы найти аналитическое выражение для её ре-акции на синусоидальное возмущающее воздействие и выявить наличие собственных колебаний, а также возможность возникновения резонанса.

9. Построить имитационную модель целевой системы и произвести её моделирование при нулевых начальных условиях и отсутствии возмущающих сил, регистрируя переходной процесс с помощью соответствующих осциллографов.

10. Используя синусоидальный входной сигнал с переменной частотой, произвести моделирование системы и проверить её амплитудно-фазовые характеристики.

11. Используя ступенчатый входной сигнал Step, произвести моделирование системы и проверить её функции переходной проводимости и веса.

12. С помощью интеграла Дюамеля рассчитать реакцию целевой системы на заданное возмущающее воздействие и проверить результаты расчёта на имитационной модели этой системы.

13. Построить имитационную модель оптимальной целевой систе-мы, произвести её моделирование и регистрацию динамических процессов с помощью соответствующих осциллографов.

14. Построить имитационную модель для автоматического опре-деления параметров оптимального управления.

15. Произвести верификацию всех построенных моделей.

16. Построить имитационные модели для нахождения управлений, улучшающих динамические характеристики целевой системы при оптимальном управлении.

17. Оформить отчёт по лабораторной работе, применяя средства генерирования описания и результатов работы имитационных моделей, встроенные в пакет Simulink.

18. Если при выполнении какого-либо этапа исследования встре-тятся затруднения, рекомендуется сначала выполнить этот этап для системы-прототипа, описанный в лабораторной работе.

7.7 Оформление отчета по результатам исследований

Для завершения лабораторной работы необходимо сгенерировать отчет в формате HTML, затем преобразовать его в формат RTF с помощью текстового редактора, включит в него теоретические результаты, отформатировать текст и графические объекты, записать на дискету и в электронном виде предъявить преподавателю. Обосновать достоверность полученных результатов.

Лабораторная работа № 8

ОПТИМАЛЬНОЕ УПРАВЛЕНИЕ СОЕДИНЕНИЯМИ НЕСКОЛЬКИХ ПРОСТЕЙШИХ ЗВЕНЬЕВ

Цель работы: разработка аналитических моделей для определения характеристик и нахождения оптимального управления объектами, состоящими из простейших звеньев, реализация этих моделей в программной среде математической системы MATLAB и построение имитационных моделей с помощью пакета Simulink, верификация разработанных моделей и определение с их использованием характеристик объектов управления, а также нахождение оптимального управления для обеспечения изменения выходной величины на заданное значение за минимальное время.

8.1 Постановка задач исследования

В данной лабораторной работе рассматриваются объекты, движение которых описывается обыкновенными дифференциальными уравнениями с постоянными коэффициентами, порядок которых не ниже второго. С точки зрения структуры это – несложные соединения таких звеньев. Целью управления является минимизация времени изменения выходной величины объекта на заданное значение за счет рационального выбора ограниченного по модулю управляющего воздействия U(t). Прежде чем приступить к поиску оптимального управления, необходимо определить характеристики объекта и провести всесторонние исследования динамики его поведения при различных возмущающих воздействиях, в частности оценить устойчивость звена или соединения, так как только для устойчивых объектов имеет смысл поиск оптимального управления. Аналитическое выражение для оптимального управления U(t) следует искать с помощью функции Гамильтона (гамильтониана) и принципа максимума Понтрягина. Расчет характеристик переходного процесса должен производиться с помощью системы MATLAB. Имитационная модель должна подтвердить расчеты по программе. Необходимо также разработать имитационную модель для автоматического определения параметров оптимального управления и имитационную модель для нахождения управлений, улучшающих динамические характеристики целевой системы при оптимальном управлении. Желательно также с помощью интеграла Дюамеля рассчитать реакцию системы на оптимальное управляющее воздействие и проверить это на имитационной модели.

8.2 Разработка аналитических моделей

В качестве прототипа рассмотрим объект управления, который описывается дифференциальным уравнением

$$T_1 T_2 \ddot{x} + (T_1 + T_2) \cdot \dot{x} + x = k \cdot u, \qquad (8.1)$$

где T_1, T_2 и k – положительные постоянные.

Данное уравнение характеризует объект, состоящий из двух последовательно соединенных инерционных звеньев. Подобным уравнением приближенно описываются многие объекты управления: двигатели постоянного тока, управляемые генераторами, магнитными усилителями, электромашинными усилителями; маломощные двигатели переменного тока, управляемые магнитными усилителями, теплообменники и т. д.

Требуется найти алгоритм управления, переводящий объект из положения $x = 0$, $\dot{x} = 0$ при $t = 0$ в положение $x = x_n$, $\dot{x} = 0$ за минимальное время; на управляющее воздействие наложено ограничение $|u| \le u_{\max}$. Определить момент переключения t_1, оптимальный переходной процесс $\vec{x}(t) = (x(t), \dot{x}(t))$ и время перехода t_2. По точкам построить графики $x(t), \dot{x}(t)$. Для этого интервал времени от 0 до t_1 и интервал времени от t_1 до t_2 разделить на 5 равных частей и вычислить значения $x(t)$ и $\dot{x}(t)$ в соответствующих точках. В точке $t = t_1$ $x(t)$ и $\dot{x}(t)$ должны рассчитываться дважды по разным формулам и эти значения должны совпадать.

Прежде чем приступить к поиску оптимального решения, необходимо оценить характеристики объекта и устойчивость его состояния.

Теперь необходимо найти алгоритм оптимального управления для следующих исходных данных: $T_1 = 0.5$; $T_2 = 0.3$; $k = 2.15$; $u_{\max} = 127$; $x_n = 236$.

Обозначим

$$x_1 = T_2 \cdot \dot{x} + x. \qquad (8.2)$$

Тогда

$$\dot{x}_1 = T_2 \cdot \ddot{x} + \dot{x}. \qquad (8.3)$$

Умножая правую и левую части уравнения (8.3) на T_1, получим

$$T_1 \cdot \dot{x}_1 = T_1 \cdot T_2 \cdot \ddot{x} + T_1 \cdot \dot{x}. \qquad (8.4)$$

Из уравнения (8.1) имеем

$$k \cdot u - x - T_2 \cdot \dot{x} = T_1 \cdot T_2 \cdot \ddot{x} + T_1 \cdot \dot{x}. \qquad (8.5)$$

Приравнивая левые части уравнений (8.5) и (8.6), получим

$$T_1 \cdot \dot{x}_1 = k \cdot u - x - T_2 \cdot \dot{x} = k \cdot u - x_1. \qquad (8.6)$$

Систему уравнений (8.2) и (8.6) запишем в нормальной форме

$$\begin{cases} \dot{x} = \dfrac{1}{T_2}(x_1 - x) = f_1(x, x_1, t, u), \\ \dot{x}_1 = \dfrac{1}{T_1}(ku - x_1) = f_2(x, x_1, t, u). \end{cases} \tag{8.7}$$

Для решения поставленной задачи применим метод максимума Понтрягина. Составим систему уравнений для вспомогательных функций

$$\begin{cases} \dot{\psi}_1 = -\dfrac{\partial H}{\partial x}, \\ \dot{\psi}_2 = -\dfrac{\partial H}{\partial x_1}. \end{cases} \tag{8.8}$$

Функция Гамильтона $H(x, x_1, t, u) = \psi_1 \cdot f_1 + \psi_2 \cdot f_2$. В нашем случае

$$H(x, x_1, t, u) = \psi_1 \cdot \frac{1}{T_2} \cdot (x_1 - x) + \psi_2 \cdot \frac{1}{T_1} \cdot (ku - x_1) \ . \tag{8.9}$$

Тогда $\dfrac{\partial H}{\partial x} = -\dfrac{\psi_1}{T_2}$, $\dfrac{\partial H}{\partial x_1} = \dfrac{\psi_1}{T_2} - \dfrac{\psi_2}{T_1}$. В этом случае система (8.8) запишется в виде

$$\begin{cases} \dot{\psi}_1 = \dfrac{\psi_1}{T_2}, \\ \dot{\psi}_2 = \dfrac{\psi_2}{T_1} - \dfrac{\psi_1}{T_2}. \end{cases} \tag{8.10}$$

Из первого уравнения системы (8.10) находим $\psi_1(t) = c_0 \cdot \exp\left(\dfrac{t}{T_2}\right)$. Тогда второе уравнение системы (8.10) можно записать в виде

$$\dot{\psi}_2 - \frac{\psi_2}{T_1} = -\frac{c_0}{T_2} \cdot \exp\left(\frac{t}{T_2}\right). \tag{8.11}$$

Соотношение (8.11) является линейным неоднородным дифференциальным уравнением. Его общее решение состоит из суммы общего решения соответствующего однородного уравнения и частного решения неоднородного уравнения. Общее решение однородного уравнения равно $c_1 \cdot \exp\left(\dfrac{t}{T_1}\right)$. Частное решение будем искать в виде: $\mathrm{A} \cdot \exp\left(\dfrac{t}{T_2}\right)$. Подставляя эту функцию в уравнение (8.11), получим $\dfrac{\mathrm{A}}{T_2} \cdot \exp\left(\dfrac{t}{T_2}\right) - \dfrac{\mathrm{A}}{T_1} \cdot \exp\left(\dfrac{t}{T_2}\right) = -\dfrac{c_0}{T_2} \cdot \exp\left(\dfrac{t}{T_2}\right)$. Тогда из этого уравнения следует $\mathrm{A} \cdot \dfrac{T_1 - T_2}{T_1 \cdot T_2} = -\dfrac{c_0}{T_2}$. Из последнего уравнения находим $\mathrm{A} = \dfrac{c_0 \cdot T_1}{T_2 - T_1}$. Общим

решением уравнения (8.11) является функция $\psi_2(t) = c_1 \cdot \exp\left(\frac{t}{T_1}\right) + \frac{c_0 T_1}{T_2 - T_1} \cdot \exp\left(\frac{t}{T_2}\right)$.

Функция Гамильтона имеет вид

$$\mathrm{H} = \frac{c_0}{T_2}(x_1 - x) \cdot \exp\left(\frac{t}{T_2}\right) + \frac{ku - x_1}{T_1}\left(c_1 \cdot \exp\left(\frac{t}{T_1}\right) + \frac{c_0 T_1}{T_2 - T_1} \cdot \exp\left(\frac{t}{T_2}\right)\right). \qquad (8.12)$$

Рассмотрим слагаемое H, которое зависит от u.

$$\mathrm{H}^* = \frac{ku}{T_1}\left(c_1 \cdot \exp\left(\frac{t}{T_1}\right) + \frac{c_0 T_1}{T_2 - T_1} \cdot \exp\left(\frac{t}{T_2}\right)\right) \quad . \qquad (8.13)$$

Обозначим $c_2 = -\frac{c_0 T_1}{T_2 - T_1}$. Тогда $\mathrm{H}^* = \frac{ku}{T_1}\left(c_1 \cdot \exp\left(\frac{t}{T_1}\right) - c_2 \cdot \exp\left(\frac{t}{T_2}\right)\right)$. В таком случае $\max \mathrm{H}^* = \frac{ku_{\max}}{T_1}\left|c_1 \cdot \exp\left(\frac{t}{T_1}\right) + \frac{c_0 T_1}{T_2 - T_1} \cdot \exp\left(\frac{t}{T_2}\right)\right|$. Управление определяется формулой $u(t) = sign\left(c_1 \cdot \exp\left(\frac{t}{T_1}\right) - c_2 \cdot \exp\left(\frac{t}{T_2}\right)\right) \cdot u_{\max}$, где

$$sign(x) = \begin{cases} 1, & \text{при } x > 0, \\ 0, & \text{при } x = 0, \\ -1, & \text{при } x < 0. \end{cases} \qquad (8.14)$$

Функция $c_1 \cdot \exp\left(\frac{t}{T_1}\right) - c_2 \cdot \exp\left(\frac{t}{T_2}\right)$ меняет знак не более одного раза. Поэтому управление задается формулой

$$u(t) = \begin{cases} u_{\max}, & при \quad 0 \le t < t_1, \\ -u_{\max}, & при \quad t_1 < t \le t_2. \end{cases} \qquad (8.15)$$

где t_1 - время переключения управления, t_2 - время оптимального управления.

Теперь найдем траекторию оптимального процесса и моменты переключения управления. На первом интервале управления уравнение (8.1) имеет вид

$$T_1 T_2 \ddot{x} + (T_1 + T_2) \cdot \dot{x} + x = k \cdot u_{\max}, \qquad (8.16)$$

Это линейное неоднородное уравнение с постоянными коэффициентами. Соответствующее линейное однородное уравнение будет таким

$$T_1 T_2 \ddot{x} + (T_1 + T_2) \cdot \dot{x} + x = 0. \qquad (8.17)$$

Характеристическое уравнение $T_1 T_2 r^2 + (T_1 + T_2) \cdot r + 1 = 0$ имеет корни:

$r_1 = -\frac{1}{T_1}$, $r_2 = -\frac{1}{T_2}$. Общим решением однородного уравнения является функция $c_1 \cdot \exp\left(-\frac{t}{T_1}\right) + c_2 \cdot \exp\left(-\frac{t}{T_2}\right)$. Частным решением неоднородного уравнения является величина $k \cdot u_{\max}$. Тогда общим решением уравнения (8.16) будет функция

$$x(t) = k \cdot u_{\max} + c_1 \cdot \exp\left(-\frac{t}{T_1}\right) + c_2 \cdot \exp\left(-\frac{t}{T_2}\right). \qquad (8.18)$$

Постоянные интегрирования c_1 и c_2 определим из начальных условий: $x = 0$, $\dot{x} = 0$. Так как

$$\dot{x}(t) = -\frac{c_1}{T_1} \cdot \exp\left(-\frac{t}{T_1}\right) - \frac{c_2}{T_2} \cdot \exp\left(-\frac{t}{T_2}\right), \qquad (8.19)$$

то получим следующую систему:

$$\begin{cases} k \cdot u_{\max} + c_1 + c_2 = 0, \\ -\frac{c_1}{T_1} - \frac{c_2}{T_2} = 0. \end{cases} \qquad (8.20)$$

Решая систему, получим $c_1 = \frac{k \cdot u_{\max} \cdot T_1}{T_2 - T_1}$, $c_2 = \frac{k \cdot u_{\max} \cdot T_2}{T_1 - T_2}$. Подставляя эти значения в формулы (8.18) и (8.19), получим

$$x(t) = k \cdot u_{\max} \cdot \left(1 + \frac{T_1}{T_2 - T_1} \cdot \exp\left(-\frac{t}{T_1}\right) + \frac{T_2}{T_1 - T_2} \cdot \exp\left(-\frac{t}{T_2}\right)\right), \qquad (8.21)$$

$$\dot{x}(t) = \frac{k \cdot u_{\max}}{T_1 - T_2} \cdot \left(\exp\left(-\frac{t}{T_1}\right) - \exp\left(-\frac{t}{T_2}\right)\right). \qquad (8.22)$$

На втором этапе управления уравнение (8.1) будет иметь вид

$$T_1 T_2 \ddot{x} + (T_1 + T_2) \cdot \dot{x} + x = -k \cdot u_{\max}, \qquad (8.23)$$

Общим решением этого уравнения является функция

$$x(t) = -k \cdot u_{\max} + c_3 \cdot \exp\left(-\frac{t}{T_1}\right) + c_4 \cdot \exp\left(-\frac{t}{T_2}\right). \qquad (8.24)$$

Тогда

$$\dot{x}(t) = -\frac{c_3}{T_1} \cdot \exp\left(-\frac{t}{T_1}\right) - \frac{c_4}{T_2} \cdot \exp\left(-\frac{t}{T_2}\right). \qquad (8.25)$$

Для определения постоянных интегрирования c_3 и c_4 используем условия $x(t_2) = x_n, \quad \dot{x}(t_2) = 0.$ Тогда получим следующую систему

$$\begin{cases} -k \cdot u_{\max} + c_3 \cdot \exp\left(-\frac{t_2}{T_1}\right) + c_4 \cdot \exp\left(-\frac{t_2}{T_2}\right) = x_n, \\ -\frac{c_3}{T_1} \cdot \exp\left(-\frac{t_2}{T_1}\right) - \frac{c_4}{T_2} \cdot \exp\left(-\frac{t_2}{T_2}\right) = 0. \end{cases} \qquad (8.26)$$

Второе уравнение умножим на T_1. Тогда систему можно переписать в виде

$$\begin{cases} c_3 \cdot \exp\left(-\frac{t_2}{T_1}\right) + c_4 \cdot \exp\left(-\frac{t_2}{T_2}\right) = x_n + k \cdot u_{\max}, \\ -c_3 \cdot \exp\left(-\frac{t_2}{T_1}\right) - c_4 \cdot \frac{T_1}{T_2} \cdot \exp\left(-\frac{t_2}{T_2}\right) = 0. \end{cases} \qquad (8.27)$$

Сложив эти уравнения после небольших преобразований, найдем $c_4 = \frac{(x_n + k \cdot u_{\max}) \cdot T_2}{T_2 - T_1} \cdot \exp\left(-\frac{t_2}{T_2}\right)$. Затем из второго уравнения системы имеем $c_3 = \frac{(x_n + k \cdot u_{\max}) \cdot T_1}{T_1 - T_2} \cdot \exp\left(-\frac{t_2}{T_1}\right)$. Подставляя найденные значения c_3 и c_4 в формулы (8.24) и (8.25), получим

$$x(t) = -k \cdot u_{\max} + \frac{x_n + k \cdot u_{\max}}{T_1 - T_2} \cdot \left(T_1 \cdot \exp\left(-\frac{t_2 - t}{T_1}\right) - T_2 \cdot \exp\left(-\frac{t_2 - t}{T_2}\right)\right), \qquad (8.28)$$

$$\dot{x}(t) = \frac{x_n + k \cdot u_{\max}}{T_2 - T_1} \cdot \left(\exp\left(-\frac{t_2 - t}{T_1}\right) - \exp\left(-\frac{t_2 - t}{T_2}\right)\right). \qquad (8.29)$$

Теперь определим моменты переключения управления t_1 и t_2. Для этого применим метод стыкования функций. В точке $t = t_1$ функции $x(t)$ и $\dot{x}(t)$ должны быть непрерывны. Приравняем значения $x(t)$, найденные по формулам (8.21) и (8.28), а также значения $\dot{x}(t)$, найденные по формулам (8.22) и (8.29), в точке $t = t_1$, получим следующую систему

$$\begin{cases} k\cdot u_{\max}\cdot\left(1+\dfrac{T_1}{T_2-T_1}\cdot\exp\left(-\dfrac{t_1}{T_1}\right)+\dfrac{T_2}{T_1-T_2}\cdot\exp\left(-\dfrac{t_1}{T_2}\right)\right)= \\ \qquad\qquad = -k\cdot u_{\max}+\dfrac{x_n+k\cdot u_{\max}}{T_1-T_2}\cdot\left(T_1\cdot\exp\left(\dfrac{t_2-t_1}{T_1}\right)-T_2\cdot\exp\left(\dfrac{t_2-t_1}{T_2}\right)\right), \\ \dfrac{k\cdot u_{\max}}{T_1-T_2}\cdot\left(\exp\left(-\dfrac{t_1}{T_1}\right)-\exp\left(-\dfrac{t_1}{T_2}\right)\right)=\dfrac{x_n+k\cdot u_{\max}}{T_2-T_1}\cdot\left(\exp\left(\dfrac{t_2-t_1}{T_1}\right)-\exp\left(\dfrac{t_2-t_1}{T_2}\right)\right). \end{cases} \quad (8.30)$$

Из второго уравнения системы находим

$$\frac{x_n+k\cdot u_{\max}}{T_2-T_1}\cdot\exp\left(\frac{t_2-t_1}{T_2}\right)=\frac{x_n+k\cdot u_{\max}}{T_2-T_1}\cdot\exp\left(\frac{t_2-t_1}{T_1}\right)-\frac{k\cdot u_{\max}}{T_1-T_2}\cdot\left(\exp\left(-\frac{t_1}{T_1}\right)-\exp\left(-\frac{t_1}{T_2}\right)\right), \quad (8.31)$$

Подставим это значение в первое уравнение и раскроем скобки:

$$\begin{aligned} 2\cdot k\cdot u_{\max}+\frac{k\cdot u_{\max}}{T_2-T_1}\cdot\exp\left(-\frac{t_1}{T_1}\right)+\frac{k\cdot u_{\max}\cdot T_2}{T_1-T_2}\cdot\exp\left(-\frac{t_1}{T_2}\right)= \\ =\frac{\left(x_n+k\cdot u_{\max}\right)\cdot T_1}{T_1-T_2}\cdot\exp\left(\frac{t_2-t_1}{T_1}\right)+\frac{\left(x_n+k\cdot u_{\max}\right)\cdot T_2}{T_2-T_1}\cdot\exp\left(\frac{t_2-t_1}{T_1}\right)- \\ -\frac{k\cdot u_{\max}\cdot T_2}{T_1-T_2}\cdot\exp\left(-\frac{t_1}{T_1}\right)+\frac{k\cdot u_{\max}\cdot T_2}{T_1-T_2}\cdot\exp\left(-\frac{t_1}{T_2}\right). \end{aligned} \quad (8.32)$$

Выполнив некоторые упрощения, получим

$$2\cdot k\cdot u_{\max}-k\cdot u_{\max}\cdot\exp\left(-\frac{t_1}{T_1}\right)=\left(x_n+k\cdot u_{\max}\right)\cdot\exp\left(\frac{t_2-t_1}{T_1}\right). \quad (8.33)$$

Все члены последнего уравнения разделим на $k\cdot u_{\max}$ и, обозначив $\frac{x_n}{k\cdot u_{\max}}=z$, получим следующую систему уравнений:

$$\begin{cases} \left(1+z\right)\cdot\exp\left(\dfrac{t_2-t_1}{T_1}\right)+\exp\left(-\dfrac{t_1}{T_1}\right)-2=0, \\ -\left(1+z\right)\cdot\exp\left(\dfrac{t_2-t_1}{T_1}\right)+\left(1+z\right)\cdot\exp\left(\dfrac{t_2-t_1}{T_2}\right)-\exp\left(-\dfrac{t_1}{T_1}\right)+\exp\left(-\dfrac{t_1}{T_2}\right)=0. \end{cases} \quad (8.34)$$

Сложив уравнения, получим

$$\begin{cases} \left(1+z\right)\cdot\exp\left(\dfrac{t_2-t_1}{T_1}\right)+\exp\left(-\dfrac{t_1}{T_1}\right)-2=0, \\ \left(1+z\right)\cdot\exp\left(\dfrac{t_2-t_1}{T_2}\right)+\exp\left(-\dfrac{t_1}{T_2}\right)-2=0. \end{cases} \quad (8.35)$$

Найдем t_2 из первого и второго уравнений системы, а затем их приравняем. Последовательно получаем

$$\begin{cases} (1+z)\cdot\exp\left(\dfrac{t_2}{T_1}\right)=\left(2-\exp\left(-\dfrac{t_1}{T_1}\right)\right)\cdot\exp\left(\dfrac{t_1}{T_1}\right), \\ (1+z)\cdot\exp\left(\dfrac{t_2}{T_2}\right)=\left(2-\exp\left(-\dfrac{t_1}{T_2}\right)\right)\cdot\exp\left(\dfrac{t_1}{T_2}\right), \end{cases} \tag{8.36}$$

$$\begin{cases} \exp\left(\dfrac{t_2}{T_1}\right)=\dfrac{1}{1+z}\cdot\left(2\cdot\exp\left(\dfrac{t_1}{T_1}\right)-1\right), \\ \exp\left(\dfrac{t_2}{T_2}\right)=\dfrac{1}{1+z}\cdot\left(2\cdot\exp\left(\dfrac{t_1}{T_2}\right)-1\right), \end{cases} \tag{8.37}$$

$$\begin{cases} t_2=T_1\cdot\ln\left[\dfrac{1}{1+z}\cdot\left(2\cdot\exp\left(\dfrac{t_1}{T_1}\right)-1\right)\right], \\ t_2=T_2\cdot\ln\left[\dfrac{1}{1+z}\cdot\left(2\cdot\exp\left(\dfrac{t_1}{T_2}\right)-1\right)\right]. \end{cases} \tag{8.38}$$

Приравняв правые части последней системы, получим нелинейное уравнение относительно t_1:

$$F(t_1)=T_1\cdot\ln\left[\frac{1}{1+z}\cdot\left(2\cdot\exp\left(\frac{t_1}{T_1}\right)-1\right)\right]-T_2\cdot\ln\left[\frac{1}{1+z}\cdot\left(2\cdot\exp\left(\frac{t_1}{T_2}\right)-1\right)\right]=0. \tag{8.39}$$

Уравнение (8.39) решаем на компьютере методом половинного деления. Для наших исходных данных получаем $t_1=1.3990$. Тогда $t_2=1.4187$. Значения $x(t)$ и $\dot{x}(t)$ можно записать в виде

$$x(t)=\begin{cases} k\cdot u_{\max}\cdot\left(1+\dfrac{T_1}{T_2-T_1}\cdot\exp\left(-\dfrac{t}{T_1}\right)+\dfrac{T_2}{T_1-T_2}\cdot\exp\left(-\dfrac{t}{T_2}\right)\right), & \text{при } 0\le t\le t_1, \\ -k\cdot u_{\max}+\dfrac{x_n+k\cdot u_{\max}}{T_1-T_2}\left(T_1\cdot\exp\left(\dfrac{t_2-t}{T_1}\right)-T_2\cdot\exp\left(\dfrac{t_2-t}{T_2}\right)\right), & \text{при } t_1\le t\le t_2, \end{cases} \tag{8.40}$$

$$\dot{x}(t)=\begin{cases} \dfrac{k\cdot u_{\max}}{T_1-T_2}\cdot\left(\exp\left(-\dfrac{t}{T_1}\right)-\exp\left(-\dfrac{t}{T_2}\right)\right), & \text{при } 0\le t\le t_1, \\ \dfrac{x_n+k\cdot u_{\max}}{T_2-T_1}\left(\exp\left(\dfrac{t_2-t}{T_1}\right)-\exp\left(\dfrac{t_2-t}{T_2}\right)\right), & \text{при } t_1\le t\le t_2. \end{cases} \tag{8.41}$$

При решении нелинейного уравнения было вычислено значение функции y в точке $t = t_1$. Оно равно $1.3 \cdot 10^{-10}$, что подтверждает правильность решения нелинейного уравнения. Исходные данные и результаты расчетов приведены ниже. В точке $t = t_1 = 1.3990$ значения $x(t)$ и $\dot{x}(t)$, вычисленные по разным формулам, совпадают. Это подтверждает правильность приведенных формул.

8.3 Программная реализация аналитических моделей

```
function [X,DX] = OptUpr(T1,T2,k,Umax,Xn)
%-- ФУНКЦИЯ ДЛЯ РАСЧЕТА ПЕРЕХОДНОГО ПРОЦЕССА
 %-- ОПТИМАЛЬНОГО УПРАВЛЕНИЯ:
%-- 1.Расчет параметров математической модели:
   z    = 0.8643;       %-- промежуточный расчетный параметр;
   y    = 1.309672e-10; %-- промежуточный расчетный параметр;
   t1   = 1.3990;       %-- время переключения управления;
   t2   = 1.4187;       %-- время перехода системы в конечную точку;
%-- 2.Задание временных точек:
   t = [0.0000 0.2798 0.5596 0.8394 1.1192...
        1.3990 1.4029 1.4069 1.4108 1.4147 1.4187];
%-- 3.Расчет управляемой величины:
     for i = 1:1:11
      if t(i) <= t1
        X(i) = k*Umax*(1+T1/(T2-T1)*exp(-t(i)/T1)+...
                T2/(T1-T2)*exp(-t(i)/T2));
      else
        X(i) = -k*Umax+(Xn+k*Umax)/(T1-T2)*(T1*exp((t2-...
                t(i))/T1)-T2*exp((t2-t(i))/T2));
      end
   end
%-- 4.Расчет скорости изменения управляемой величины:
    for i = 1:1:11
      if t(i) <= t1
        DX(i) = k*Umax/(T1-T2)*(exp(-t(i)/T1)-...
                  exp(-t(i)/T2));
      else
        DX(i) = (Xn+k*Umax)/(T2-T1)*(exp((t2-...
                  t(i))/T1)-exp((t2-t(i))/T2));
      end
   end
%-- 5.Визуализация управляемой величины и скорости ее изменения:
     subplot(2,2,1)
     plot(t,X,'r')
     xlabel('t')
     ylabel('X')
     subplot(2,2,2)
     plot(t,DX,'r')
```

```
    xlabel('t')
    ylabel('DX')

%-- 6.Конец функции OptUpr(T1,T2,k,Umax,Xn)
```

8.4 Построение имитационных моделей

В соответствии с математическим описанием объекта управления и поставленными задачами имитационная модель содержит два интегрирующих блока, необходимые генераторы сигналов, дисплеи, осциллографы, сумматоры и другие элементы (см. рис. 8.2).

Требуется построить эту модель, используя библиотеки блоков пакета Simulink, и настроить параметры блоков в соответствии с условиями задачи. Проверить работу модели можно путем её многократного запуска при изменении времени окончания работы. Модифицируя состав модели и изменяя режим её работы, можно получить все требуемые характеристики объекта управления.

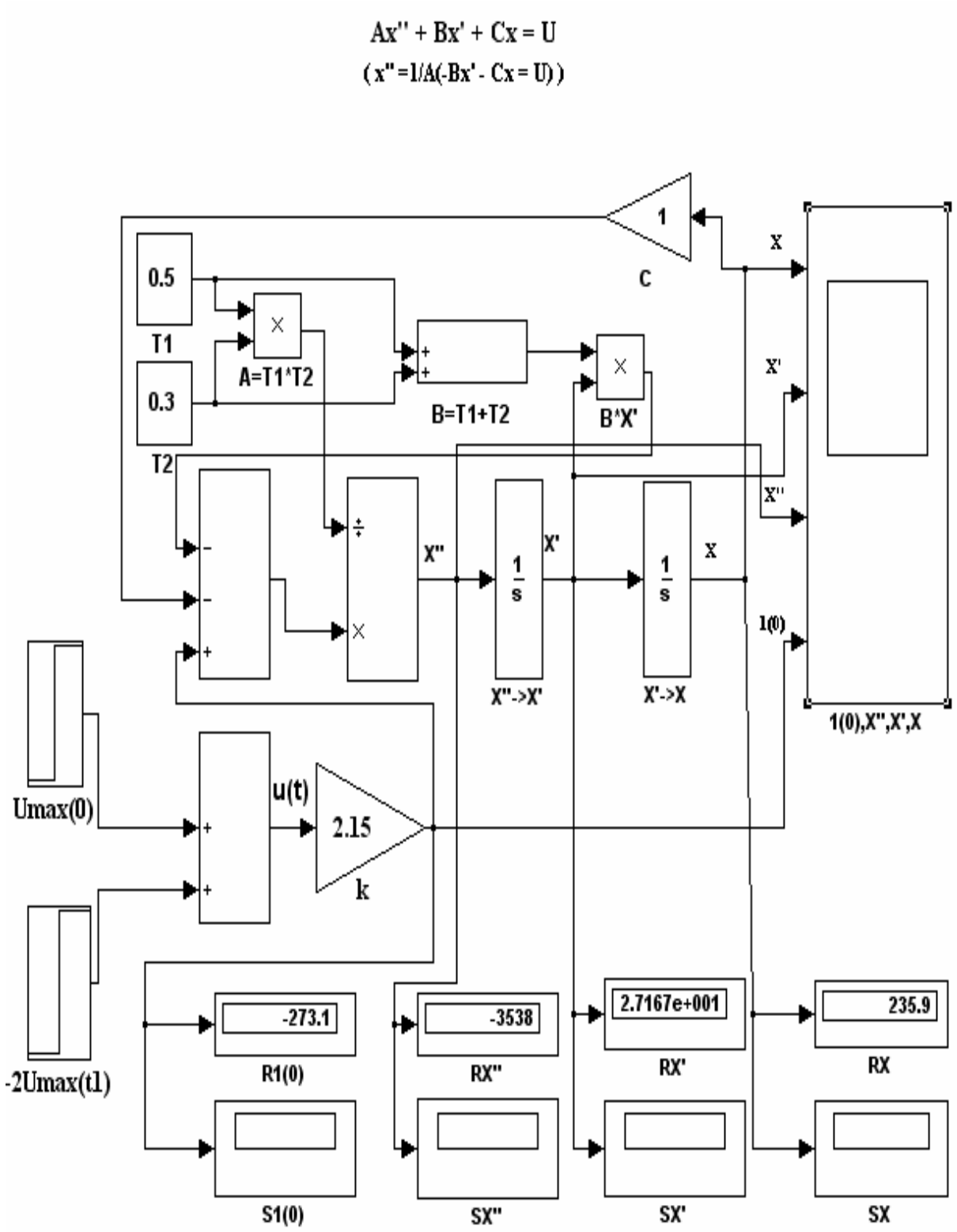

Рис 8.1 Имитационная модель оптимального управления

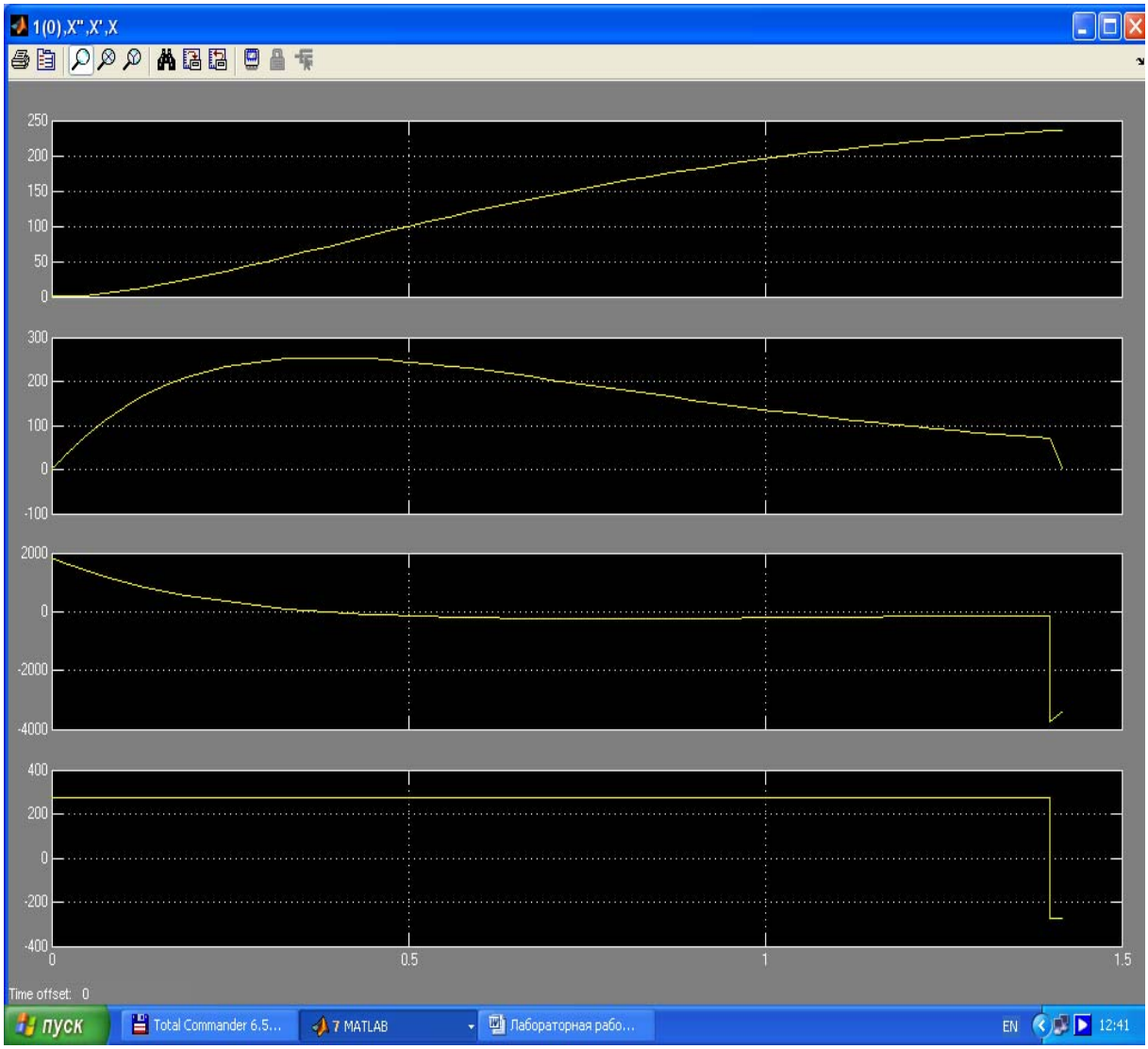

Рис. 8.2 Осциллограммы оптимального управления

8.5 Верификация математических моделей

Верификацию аналитической и имитационной моделей объекта управления произведем с помощью сопоставления переходных процессов, протекающих в этих моделях при оптимальном управляющем воздействии с ограничением U_{max} = 127 для следующих значений параметров объекта T = 0.5000, T2 = 0.3000 и k=2.15, когда конечное значение выходной величины x_n равно 236. Расчетные параметры моделей:

z = 0.8643; t_1 = 1.3990; y = 1.309672E-10; t_2 = 1.4187.

Таблица 8.1

Результаты расчета и моделирования переходного процесса для объекта управления при оптимальном управляющем воздействии u(t)

Текущее время t	Оптимальное управляющее воздействие u(t)	Управляемая величина x(t)		Скорость изменения управляемой величины x'(t)	
		Расчетное значение	Модельное значение	Расчетное значение	Модельное Значение
0.0000	127	0.0000	0.0000	0.0000	0.0000
0.2798	127	44.1414	44.2748	242.9252	243.0518
0.5596	127	113.5655	113.6942	234.4083	234.3057
0.8394	127	170.6298	170.7241	171.5654	171.4379
1.1192	127	210.0827	210.1447	112.8408	112.7408
1.3990	127	235.3208	235.3594	70.3061	70.2383
1.4029	-127	235.5643	235.6049	55.9284	55.7070
1.4069	-127	235.7587	235.7984	41.3258	41.1080
1.4108	-127	235.8926	235.9314	27.3809	27.1665
1.4147	-127	235.9727	235.0166	13.7203	13.5093
1.4187	-127	236.0000	236.0370	0.0000	-0.2075

8.6 Варианты заданий и порядок их выполнения

1. Для рассматриваемого простейшего звена с помощью функции RKiw построить частотные графики.

2. Используя функцию LWSin , построить графики переходных функций при отсутствии возмущающих воздействий и нулевых начальных условиях, при толчкообразном и синусоидальном возмущениях, сравнить их с осциллограммами имитационной модели для таких же режимов и заполнить таблицы значений переходных функций.

3. По табл. 4.2 лабораторной работы № 4 выбрать два простейших звена и образовать из них систему, движение которой должно описываться обыкновенным дифференциальным уравнением порядка не ниже второго.

4. Для выбранной целевой системы вывести самостоятельно или получить с помощью компьютера аналитические выражения для вычисления передаточной и частной функций, а также функций переходной проводимости и веса.

5. С помощью пакета символьных вычислений Symbolic Math найти вид оптимального управления, обеспечивающего изменение выходной величины на заданное значение за минимальное время, используя функцию Гамильтона и принцип максимума Понтрягина, и вывести аналитические выражения для переходных функций системы, работающей в этом режиме.

6. Написать программы для вычисления амплитуды и фазы частот-ной функции, а также для расчёта переходного процесса системы при толчкообразном входном сигнале, используя выражения для функций переходной проводимости и веса.

7. На комплексной плоскости построить амплитудно-фазовую характеристику - годограф вектора K(iw) и оценить устойчивость целевой системы.

8. Для целевой системы найти аналитическое выражение для её ре-акции на синусоидальное возмущающее воздействие и выявить наличие собственных колебаний, а также возможность возникновения резонанса.

9. Построить имитационную модель целевой системы и произвести её моделирование при нулевых начальных условиях и отсутствии возмущающих сил, регистрируя переходной процесс с помощью соответствующих осциллографов.

10. Используя синусоидальный входной сигнал с переменной частотой, произвести моделирование системы и проверить её амплитудно-фазовые характеристики.

11. Используя ступенчатый входной сигнал Step, произвести моделирование системы и проверить её функции переходной проводимости и веса.

12. С помощью интеграла Дюамеля рассчитать реакцию целевой системы на заданное возмущающее воздействие и проверить результаты расчёта на имитационной модели этой системы.

13. Построить имитационную модель оптимальной целевой систе-мы, произвести её моделирование и регистрацию динамических процессов с помощью соответствующих осциллографов.

14. Построить имитационную модель для автоматического опре-деления параметров оптимального управления.

15. Произвести верификацию всех построенных моделей.

16. Построить имитационные модели для нахождения управлений, улучшающих динамические характеристики целевой системы при оптимальном управлении.

17. Оформить отчёт по лабораторной работе, применяя средства генерирования описания и результатов работы имитационных моделей, встроенные в пакет Simulink.

18. Если при выполнении какого-либо этапа исследования встре-тятся затруднения, рекомендуется сначала выполнить этот этап для системы-прототипа, описанный в лабораторной работе.

8.7 Оформление отчета по результатам исследований

Для завершения лабораторной работы необходимо сгенерировать отчет в формате HTML, затем преобразовать его в формат RTF с помощью текстового редактора, включит в него теоретические результаты, отформатировать текст и графические объекты, записать на дискету и в электронном виде предъявить преподавателю. Обосновать достоверность полученных результатов.

Лабораторная работа № 9

ВАРИАЦИОННЫЕ МЕТОДЫ РЕШЕНИЯ ЗАДАЧ ОПТИМАЛЬНОГО УПРАВЛЕНИЯ С ПРИМЕНЕНИЕМ ИНСТРУМЕНТАЛЬНЫХ ПАКЕТОВ СИСТЕМЫ MATLAB

Цель работы: освоение вариационных методов решения задач оптимального управления с помощью точных и приближенных аналитических моделей, а также с использованием средств символьной обработки математической системы MATLAB и имитационного моделирования инструментального пакета Simulink.

9.1 Постановка задач исследования

В данной лабораторной работе рассматриваются задачи оптимального управления, решаемые классическими вариационными методами. Для предлагаемых функционалов требуется построить точные и приближенные аналитические модели нахождения экстремалей этих функционалов, а также точные и приближенные значения их экстремумов. Используя средства символьной обработки математической системы MATLAB, необходимо разработать программную модель для решения указанных задач. С помощью инструментального пакета Simulink построить имитационные модели нахождения экстремалей и значений функционалов, произвести верификацию всех разработанных моделей и оценку влияния помех на качество управления.

9.2 Разработка аналитических моделей

Формализация многих задач оптимального управления приводит к интегральным функционалам классического вариационного исчисления, отображающих функцию или набор функций в число. Решение задачи в этом случае заключается в нахождении таких функций, которые обеспечивают минимальное или максимальное значение соответствующего функционала. Эти функции называются экстремалями задачи. В вариационном исчислении доказывается, что экстремали должны удовлетворять некоторым дифференциальным уравнениям. Однако удовлетворение этим уравнениям представляет собой лишь необходимое условие экстремума функционала. Поэтому, решив дифференциальные уравнения и найдя экстремали, необходимо тем или иным способом определить, что они обеспечивают экстремум рассматриваемого функционала.

В зависимости от вида функционала необходимые условия записываются следующим образом:

а) для функционала $J[y(x)] = \int_{x_0}^{x_1} F(x, y, y')dx$:

$$F_y - \frac{d}{dx} F_{y'} = 0; \quad y(x_0) = y_0; \quad y(x_1) = y_1, \tag{9.1}$$

где F_y - частная производная по y от функции F;

$F_{y'}$ - частная производная по y' от той же функции F (это условие называется уравнением Эйлера);

б) для функционала $J[y(x),z(x)]=\int_{x_0}^{x_1} F(x,y,y',z,z')dx$:

$$F_y-\frac{d}{dx}F_{y'}=0;\qquad F_z-\frac{d}{dx}F_{z'}=0; \tag{9.2}$$

$$y(x_0)=y_0;\ y(x_1)=y_1;\ z(x_0)=z_0;\ z(x_1)=z_1, \tag{9.3}$$

где F_z и $F_{z'}$ - соответствующие частные производные;

в) для функционала

$$J[y_1(x),y_2(x),...,y_n(x)]=\int_{x_0}^{x_1} F(x,y_1,y_1',y_2,y_2',...,y_{n-1},y_{n-1}',y_n,y_n')dx: \tag{9.4}$$

$$F_{y_k}-\frac{d}{dx}F_{y_k'}=0\ \text{(k=1,2,…,n);} \tag{9.5}$$

$$y_k(x_0)=y_{0k};\ y_k(x_1)=y_{1k}\ \text{(k=1,2,…,n);} \tag{9.6}$$

г) для функционала $J[y(x)]=\int_{x_0}^{x_1} F(x,y,y',...,y^{(n)})dx$:

$$F_y-\frac{d}{dx}F_{y'}+...+(-1)^n\frac{d^n}{dx^n}F_{y(n)}=0; \tag{9.7}$$

$$y(x_0)=y_0;\ y(x_1)=y_1;\ y'(x_0)=y_0';\ y^{(n-1)}(x_0)=y^{(n-1)}{}_0;\ y^{(n-1)}(x_1)=y^{(n-1)}1; \tag{9.8}$$

д) для функционала $J[u(x,y)]=\iint_B F(x,y,u,u_x,u_y)dxdy$:

$$F_u-\frac{\partial}{\partial x}F_{ux}-\frac{\partial}{\partial y}F_{uy}=0;\ u_l(x,y), \tag{9.9}$$

где u_x и u_y - частные производные от искомой функции $u(x,y)$ соответственно по x и y.

F_u, F_{ux}, F_{uy} - частные производные от подынтегрального выражения;

$u_l(x,y)$ - граничные условия, т.е. значения функции $u(x,y)$ на контуре области интегрирования B (это дифференциальное уравнение называется уравнением Остроградского);

е) для функционала $J[u(x,y,z)]=\iiint_V F(x,y,z,u,u_x,u_y,u_z)dxdydz$:

$$F_u-\frac{\partial}{\partial x}F_{ux}-\frac{\partial}{\partial y}F_{uy}-\frac{\partial}{\partial z}F_{uz}=0;\ \text{и}\ u_s(x,y,z), \tag{9.10}$$

где $u_s(x,y,z)$ - значения функции $u(x,y,z)$ на поверхности S объема интегрирования V;

ж) если в выражение функционала $J[u(x,y,z)]$ входят производные функции $u(x,y,z)$ до порядка n, то уравнение Остроградского имеет вид:

$$F_u-\frac{\partial}{\partial x}F_{ux}-\frac{\partial}{\partial y}F_{uy}-\frac{\partial}{\partial z}F_{uz}+\frac{\partial^2}{\partial x^2}F_{uxx}+...=0. \tag{9.11}$$

В табл. 9.1 и 9.2 приведено несколько примеров вариационных задач для функционалов различных видов.

Нахождение экстремалей позволяет вычислять относительные, или локальные экстремумы. Математическое определение понятия локального экстремума интегрального функционала производится с помощью ε-окрестностей функций и их классов. Например, для простейшего функционала

$$J[y(x)] = \int_{x_0}^{x_1} F(x, y, y')dx, \qquad (9.12)$$

в котором F является непрерывной функцией всех трех аргументов вместе с ее производными до второго порядка в некоторой области B плоскости (x,y) и при любых значениях y', экстремаль ищется в классе функций C_1, имеющих в промежутке $[x_0, x_1]$ непрерывную производную. В этом классе определяется ε-окрестность кривой $y = y(x)$ как множество кривых $y_i(x)$, которые во всем промежутке $[x_0, x_1]$ удовлетворяют неравенству $|y_i(x) - y(x)| \le \varepsilon$. Иногда, кроме этого неравенства, добавляют еще одно: $|y_i'(x) - y'(x)| \le \varepsilon$. В первом случае говорят об ε-близости нулевого порядка, а во втором случае, при наличии двух неравенств, говорят об ε-близости первого порядка. Если окажется, что величина этого функционала для $y(x)$, лежащей внутри упомянутой области B, принадлежащей классу C_1 и удовлетворяющей предельным условиям.

$$y(x_0) = y_0; \;\; y(x_1) = y_1; \qquad (9.13)$$

не меньше (или не больше) его величины для любых других кривых класса C_1, находящихся в некоторой ε-близости к $y(x)$ и удовлетворяющих тем же предельным условиям, то говорят, что функционал $J[y(x)]$ достигает локального экстремума для кривой $y(x)$.

Наряду с понятием локального экстремума вводится понятие абсолютного экстремума для некоторого класса функций D, для которых интеграл $J[y(x)]$ имеет смысл. Говорят, что функционал J[y(x)] достигает в классе D абсолютного экстремума для кривой y(x), если величина этого функционала для y(x) не меньше (или не больше) его величины для всех других кривых класса D.

Аналогичным образом определяются локальные абсолютные экстремумы для функционалов других видов.

Таблица 9.1

Варианты задач для функционала типа $J[y[x]]$

№ п/п	Вид функционала	Функционал	Граничные условия
1	$J[y[x]] = \int_{x_0}^{x_1} F(x, y, y')dx$	$\int_0^2 y'^3 dx$	$y(0) = 0; y(2) = 1$
2	$J[y[x]] = \int_{x_0}^{x_1} F(x, y, y')dx$	$\int_0^1 (xy + y^2 - 2y^2 y')dx$	$y(0) = 1; y(1) = 2$
3	$J[y[x]] = \int_{x_0}^{x_1} F(x, y, y')dx$	$\int_0^1 (y'^2 + 12xy)dx$	$y(0) = 1; y(1) = 1$

4	$J[y[x]] = \int_{x_0}^{x_1} F(x, y, y')dx$	$\int_0^{\pi/2} (y'^2 - y^2)dx$	$y(0) = 0; y(\frac{\pi}{2}) = 1$
5	$J[y[x]] = \int_{x_0}^{x_1} F(x, y, y', y'')dx$	$\int_0^1 (y''^2 - 48y)dx$	$y(0) = 1; y'(0) = 4;$ $y(1) = 0; y'(1) = 0$
6	$J[y[x]] = \int_{x_0}^{x_1} F(x, y, y', y'')dx$	$\int_0^1 (2xy - y''^2)dx$	$y(0) = 0; y'(0) = 0;$ $y(1) = 0; y'(1) = 0.1$
7	$J[y[x]] = \int_{x_0}^{x_1} F(x, y, y', y'')dx$	$\int_0^1 e^{-x} y''^2 dx$	$y(0) = 0; y'(0) = 1;$ $y(1) = e; y'(1) = 2e$
8	$J[y[x]] = \int_{x_0}^{x_1} F(x, y, y', y'')dx$	$\int_0^1 e^{x} y''^2 dx$	$y(0) = 0; y'(0) = 1;$ $y(1) = e; y'(1) = e$
9	$J[y[x]] = \int_{x_0}^{x_1} F(x, y, y', y'')dx$	$\int_0^1 (y''^2 - 24xy)dx$	$y(0) = 0; y'(0) = 0;$ $y(1) = 0.2; y'(1) = 1$
10	$J[y[x]] = \int_{x_0}^{x_1} F(x, y, y')dx$	$\int_0^1 (y'^2 - y^2 - 2xy)dx$	$y(0) = 0; y(1) = 0$
11	$J[y[x]] = \int_{x_0}^{x_1} F(x, y, y')dx$	$\int_0^1 (y'^2 - y^2 - 16xy)dx$	$y(0) = 0; y(1) = 0$
12	$J[y[x]] = \int_{x_0}^{x_1} F(x, y, y')dx$	$\int_0^1 (y'^2 - y^2 + 2xy)dx$	$y(0) = 0; y(2) = 0$
13	$J[y[x]] = \int_{x_0}^{x_1} F(x, y, y')dx$	$\int_0^1 (y'^2 + y^2 + 16xy)dx$	$y(0) = 0; y(2) = 0$

Таблица 9.2

Варианты задач для функционалов типа $J[y[x],z(x)]$

5	$J[y[x],z(x)]=\int_{x_0}^{x_1}F(x,y,y',z,z')dx$	$\int_0^1(y'^2+z'^2+2y)dx$	$y(0)=1;y(1)=1.5;$ $z(0)=0;z(1)=1$
6	$J[y[x],z(x)]=\int_{x_0}^{x_1}F(x,y,y',z,z')dx$	$\int_0^1(y^2+z^2+2y'z')dx$	$y(0)=0y(1)=sh(1);$ $z(0)=0;z(1)=sh(1)$
7	$J[y[x],z(x)]=\int_{x_0}^{x_1}F(x,y,y',z,z')dx$	$\int_0^1(y'z'+2yz)dx$	$y(0)=0;y(1)=2;$ $z(0)=0;z(1)=-2$
8	$J[y[x],z(x)]=\int_{x_0}^{x_1}F(x,y,y',z,z')dx$	$\int_{-1}^1(2xy-y'^2+\frac{1}{3}z'^3)dx$	$y(-1)=2;y(1)=0;$ $z(-1)=-1;z(1)=1$
9	$J[y[x],z(x)]=\int_{x_0}^{x_1}F(x,y,y',z,z')dx$	$\int_0^1(2xy-2y^2+y'^2-z'^2+2y)dx$	$y(0)=1;y(1)=0;$ $z(0)=0;z(1)=1$

Следует отметить, что поиск экстремалей с помощью уравнений Эйлера-Остроградского сужает класс рассматриваемых функций до класса C_2, в котором функции должны иметь непрерывные производные до второго порядка. Поэтому если их решение не дало желаемого результата, то поиск экстремума функционала должен производиться с помощью других методов.

Для функционала аналогом дифференциала функции является его вариация ∂J, т.е. изменение функционала при замене функции из рассматриваемого класса на ε-близкую функцию из того же класса. Так, для функционала

$$J[y(x)]=\int_{x_0}^{x_1}F(x,y,y')dx \qquad (9.14)$$

вблизи экстремали $y(x)$ его первая вариация записывается следующим образом:

$$\delta J=[F_{y'}\delta y]_{x_0}^{x_1}+\int_{x_0}^{x_1}(F_y-\frac{d}{dx}F_{y'})\delta y dx, \qquad (9.15)$$

где $\delta y=\alpha\eta(x)$, так что ε-близкая к $y(x)$ функция будет $y(x)+\alpha\eta(x)$;

α - малый численный параметр;

$\eta(x)$ - произвольная функция класса C_2, принимающая нулевые значения на концах промежутка интегрирования.

В качестве примера найдем точное и приближенное решение задачи о минимуме функционала

$$\nu[y(x)]=\int_0^1\left(y'^2+y^2+12xy\right)dx,\quad y(0)=y(1)=0. \qquad (9.16)$$

Построим графики функций $y(x)$, $y_0(x)$ и $y_1(x)$,а также вычислим $v[y(x)]$, $v[y_0(x)]$ и $v[y_1(x)]$, где $y(x)$ – точное решение задачи.

Сначала найдем точное решение задачи. Функция $y(x)$ должна удовлетворять уравнению Эйлера

$$F_y - \frac{d}{dx}F_{y'} = 0. \qquad (9.17)$$

В нашем случае

$$F(x, y, y') = y'^2 + y^2 + 12xy, \qquad F_y = 2y + 12x, \qquad F'_y = 2y', \qquad \frac{d}{dx}F_y = 2y''. \qquad (9.18)$$

Тогда уравнение Эйлера будет иметь вид

$$y'' - y = 6x. \qquad (9.19)$$

Запишем уравнение в следующем виде

$$2y + 12x - 2y'' = 0. \qquad (9.20)$$

Это линейное неоднородное уравнение. Его общее решение состоит из суммы общего решения соответствующего однородного уравнения и частного решения неоднородного уравнения. Однородное уравнение имеет вид

$$y'' - y = 0. \qquad (9.21)$$

Характеристическое уравнение $r^2 - 1 = 0$ имеет корни $r_1 = 1, r_2 = -1.$ Общее решение однородного уравнения будет таким:

$$y_1 = c_1 e^x + c_2 e^{-x}. \qquad (9.22)$$

Частное решение неоднородного уравнения будем искать в виде $y_2 = Ax.$ Подставляя это значение в неоднородное уравнение, получим: $-Ax = 6x.$ Тогда $A = -6$, $y_2 = -6x.$

Таким образом, общее решение неоднородного уравнения принимает вид:

$$y(x) = c_1 e^x + c_2 e^{-x} - 6x. \qquad (9.23)$$

Используя краевые условия, получим следующую систему уравнений для определения c_1 и c_2

$$\begin{aligned} y(0) &= c_1 + c_2 = 0, \\ y(1) &= c_1 e + c_2 e^{-1} - 6 = 0. \end{aligned} \qquad (9.24)$$

Решая полученную систему, находим

$$c_1 = \frac{3}{sh\,1}, \qquad c_2 = -\frac{3}{sh\,1}. \qquad (9.25)$$

Тогда

$$y(x) = \frac{3}{sh1}\left(e^x - e^{-x}\right) - 6x. \qquad (9.26)$$

Учитывая, что

$$\frac{e^x - e^{-x}}{2} = shx, \qquad (9.27)$$

получим

$$y(x) = \frac{6shx}{sh1} - 6x. \qquad (9.28)$$

В таком случае

$$y'(x) = \frac{6chx}{sh1} - 6. \qquad (9.29)$$

С помощью MATLAB находим

$$v[y(x)] = \int_0^1 \left(y'^2 + y^2 + 12xy\right)dx \approx -0{,}730730. \qquad (9.30)$$

Теперь найдем первое приближение точного решения. Его будем искать в виде

$$y_0(x) = a_0\left(x - x^2\right). \qquad (9.31)$$

Тогда

$$y'_0(x) = a_0(1 - 2x). \qquad (9.32)$$

В таком случае

$$v[y_0(x)] = \int_0^1 \left(a_0{}^2(1-2x)^2 + a_0{}^2\left(x - x^2\right)^2 + 12a_0x\left(x - x^2\right)\right)dx = \varphi(a_0). \qquad (9.33)$$

Чтобы найти минимум $\varphi(a_0)$ возьмем производную и приравняем ее к нулю. Производную возьмем под знаком интеграла. Имеем

$$\varphi'(a_0) = \int_0^1 \left(2a_0(1-2x)^2 + 2a_0\left(x - x^2\right)^2 + 12x\left(x - x^2\right)\right)dx. \qquad (9.34)$$

После преобразований получим

$$\varphi'(a_0) = \int_0^1 \left(\left(-4x^3 + 2x^4 + 10x^2 - 8x + 2\right)a_0 + 12x^2 - 12x^3\right)dx. \qquad (9.35)$$

Взяв интеграл и приравняв его нулю, получим

$$\varphi'(a_0) = \left(\frac{2}{5} - 1 + \frac{10}{3} - 4 + 2\right)a_0 + (4 - 3) = 0. \qquad (9.36)$$

Отсюда находим $a_0 = -\dfrac{15}{11}$.

Тогда

$$y_0(x) = -\frac{15}{11}(x - x^2), \quad y'_0(x) = \frac{15}{11}(2x - 1). \qquad (9.37)$$

$$v[y_0(x)] = \int_0^1 \left(y_0'^2 + y_0{}^2 + 12xy_0\right)dx \approx -0{,}681818. \qquad (9.38)$$

Теперь найдем второе приближение точного решения. Будем его искать в виде

$$y_1(x) = (a_0 + a_1x)\left(x - x^2\right). \qquad (9.39)$$

Раскроем скобки и приведем подобные члены

$$y_1(x) = a_0x + (a_1 - a_0)x^2 - a_1x^3. \qquad (9.40)$$

Тогда

$$y'_1(x) = a_0 + 2(a_1 - a_0)x - 3a_1x^2. \qquad (9.41)$$

Функционал будет иметь вид

$$v[y_1(x)] = \int_0^1 \left\{\left[a_0 + 2(a_1 - a_0)x - 3a_1x^2\right]^2 + \left[a_0x + (a_1 - a_0)x^2 - a_1x^3\right]^2 + 12x\left[a_0x + (a_1 - a_0)x^2 - a_1x^3\right]\right\}dx = \varphi(a_0, a_1). \qquad (9.42)$$

Чтобы найти минимум $\varphi(a_0, a_1)$, возьмем частные производные по a_0 и a_1 и приравняем их к нулю. Решая полученную систему, определим a_0 и a_1 :

$$\frac{1}{2}\frac{d\varphi}{da_0} = \int_0^1 \left\{\left[a_0 + 2(a_1 - a_0)x - 3a_1x^2\right](1 - 2x) + \left[a_0x + (a_1 - a_0)x^2 - a_1x^3\right]\left(x - x^2\right) + 6x^2 - 6x^3\right\}dx. \qquad (9.43)$$

Взяв интеграл и выполнив некоторые преобразования, получим первое уравнение

$$22a_0 + 11a_1 = -30. \tag{9.44}$$

Аналогично

$$\frac{1}{2}\frac{d\varphi}{da_1} = \int_0^1 \{\left[a_0 + 2(a_1 - a_0)x - 3a_1x^2\right](2x - 3x^2) + \left[a_0x + (a_1 - a_0)x^2 - a_1x^3\right](x^2 - x^3) + \\ + 6x^3 - 6x^4\}dx. \tag{9.45}$$

После преобразований получим второе уравнение

$$77a_0 + 60a_1 = -126. \tag{9.46}$$

Решая систему уравнений, находим

$$a_0 = -\frac{417}{423}, \quad a_1 = -\frac{42}{43}. \tag{9.47}$$

Тогда второе приближение будет таким

$$y_1(x) = \left(\frac{417}{473} + \frac{42}{43}x\right)(x^2 - x). \tag{9.48}$$

В таком случае

$$y'_1(x) = \frac{42}{43}(x^2 - x) + (2x - 1)\left(\frac{417}{473} + \frac{42}{43}x\right). \tag{9.49}$$

Значение функционала

$$v[y_1(x)] = \int_0^1 \left(y_1'^2 + y_1^2 + 12xy_1\right)dx \approx -0{,}730641. \tag{9.50}$$

В нашем случае

$$v[y(x)] < v[y_1(x)] < v[y_0(x)], \tag{9.51}$$

что подтверждает правильность решения задачи.

Ниже приведено решение задачи с помощью MATLAB, построены графики функций $y(x), y_0(x), y_1(x)$, а также графики функций $z_1(x) = y(x) - y_0(x), z_2(x) = y(x) - y_1(x)$.

9.3 Программная реализация аналитических моделей

```
function TrueEvaluation
%-- Вычисление точного значения функционала
%
syms x yx Dyx JFunc
yx=sym('6*sinh(x)/sinh(1)-6*x')      %-- определение экстремали
                                     %   функционала;
Dyx=sym('6*cosh(x)/sinh(1)-6')       %-- определение производной от
                                     %    экстремали функционала;
JFunc=Dyx^2+yx^2+12*x*yx             %-- определение подынтегрального
                                     %    выражения функционала;
J=int(JFunc,x,0,1)                   %-- нахождение функционала в
```

```
                                    %   аналитическом виде;
J=vpa(J,6)                          %-- вычисление значения функционала.


function FirstApproximation
%-- Нахождение и вычисление первого приближения функционала
%
Syms y0 a0 x Dy0 DiffFi0 DiffDiffFi0 IntDiffDiffFi0
y0=sym('a0*(x-x^2)')                %-- вид первого приближения;
Dy0=diff(y0,x,1)                    %-- производная первого
                                    %   приближения;
DiffFi0=(Dy0^2+y0^2+12*x*y0)        %-- подынтегральное выражение
                                    %   функционала;
DiffDiffFi0=diff(DiffFi0,a0,1)      %-- дифференцирование
                                    %   подынтегрального выражения
                                    %   функционала;
IntDiffDiffFi0=int(DiffDiffFi0,x,0,1)  %-- нахождение производной
                                    %   функционала по a0;
[a0]=solve(IntDiffDiffFi0)          %-- нахождение коэффициента a0;
y0=subs(y0,a0,'a0')                 %-- нахождение выражения для первого
                                    %   приближения;
Dy0=subs(Dy0,a0,'a0')               %-- нахождение выражения для
                                    %   производной первого приближения;
J0=int(Dy0^2+y0^2+12*x*y0,x,0,1)    %-- нахождение выражения для первого
                                    %   приближения функционала;
J0=vpa(J0,6)                        %-- вычисление первого приближения
                                    %   функционала.


function SecondApproximation
%-- Нахождение и вычисление второго приближения функционала
%
Syms y1 a0 a1 x DiffFi1 DiffDiffFi11 DiffDiffFi12 IntDiffDiffFi11 IntDiffDiffFi12
y1=sym('(a0+a1*x)*(x-x^2)')         %-- вид второго приближения;
Dy1=diff(y1,x,1)                    %-- производная второго приближения;
DiffFi= (Dy1^2+y1^2+12*x*y1)        %-- подинтегральное выражение
                                    %   функционала;
DiffDiffFi11=(2*(Dy1*(1-2*x)+y1*(x-x^2)+6*x^2-6*x^3))
            %-- дифференцирование подинтегрального выражения
            %   функционала по a0;
```

```
DiffDiffFi12=(2*(Dy1*(2*x-3*x^2)+y1*(x^2-x^3)+6*x^3-6*x^4))
            %-- дифференцирование подинтегрального выражения
            %   функционала по a1;
IntDiffDiffFi11=int(DiffDiffFi11,x,0,1)
            %-- нахождение производной функционала по a0;
IntDiffDiffFi12=int(DiffDiffFi12,x,0,1)
            %-- нахождение производной функционала по a1;
[a0,a1]=solve(IntDiffDiffFi11,IntDiffDiffFi12,a0,a1)
            %-- нахождение коэффициентов a0 и a1;
y1=subs(subs(y1,a0,'a0'),a1,'a1')
            %-- нахождение выражения для второго приближения;
Dy1=subs(subs(Dy1,a0,'a0'),a1,'a1')
            %-- нахождение выражения для производной второго
            %   приближения;
J1=int(Dy1^2+y1^2+12*x*y1,x,0,1)
            %-- нахождение выражения для второго приближения
            %   функционала;
J1=vpa(J1,6)
            %-- вычисление второго приближения функционала.
```

9.4 Построение имитационных моделей

На рис 9.1 представлена имитационная модель для нахождения экстремали рассматриваемого функционала, а на рис. 9.2 – осциллограммы работы модели после нахождения экстремали. Имитационная модель для вычисления функционала и соответствующая осциллограмма приведены на рис. 9.3 и 9.4.

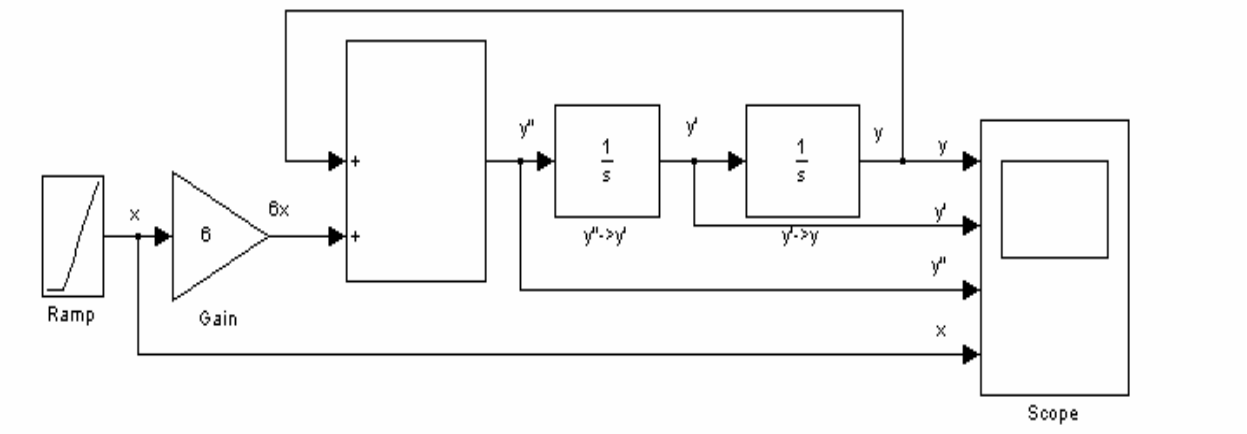

Рис.9.1 Имитационная модель для нахождения экстремали функционала

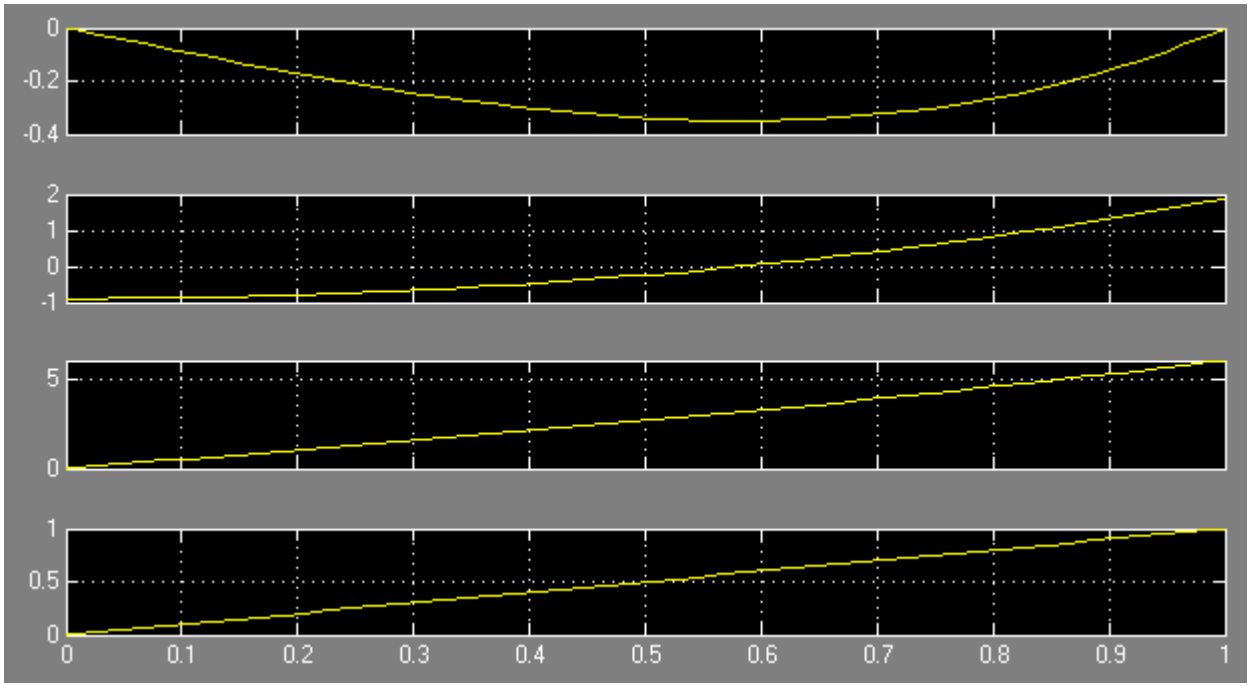

Рис.9.2 Осциллограммы работы модели

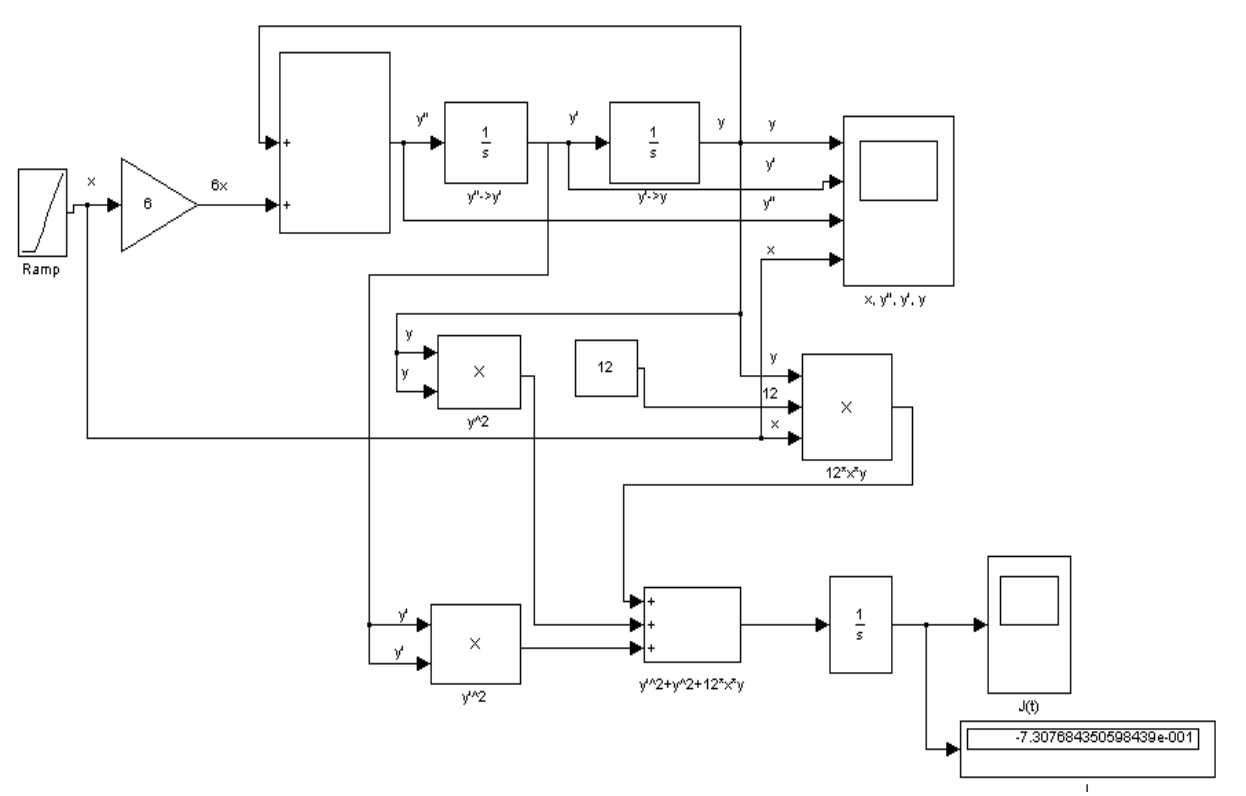

Рис. 9.3 Имитационная модель для нахождения функционала

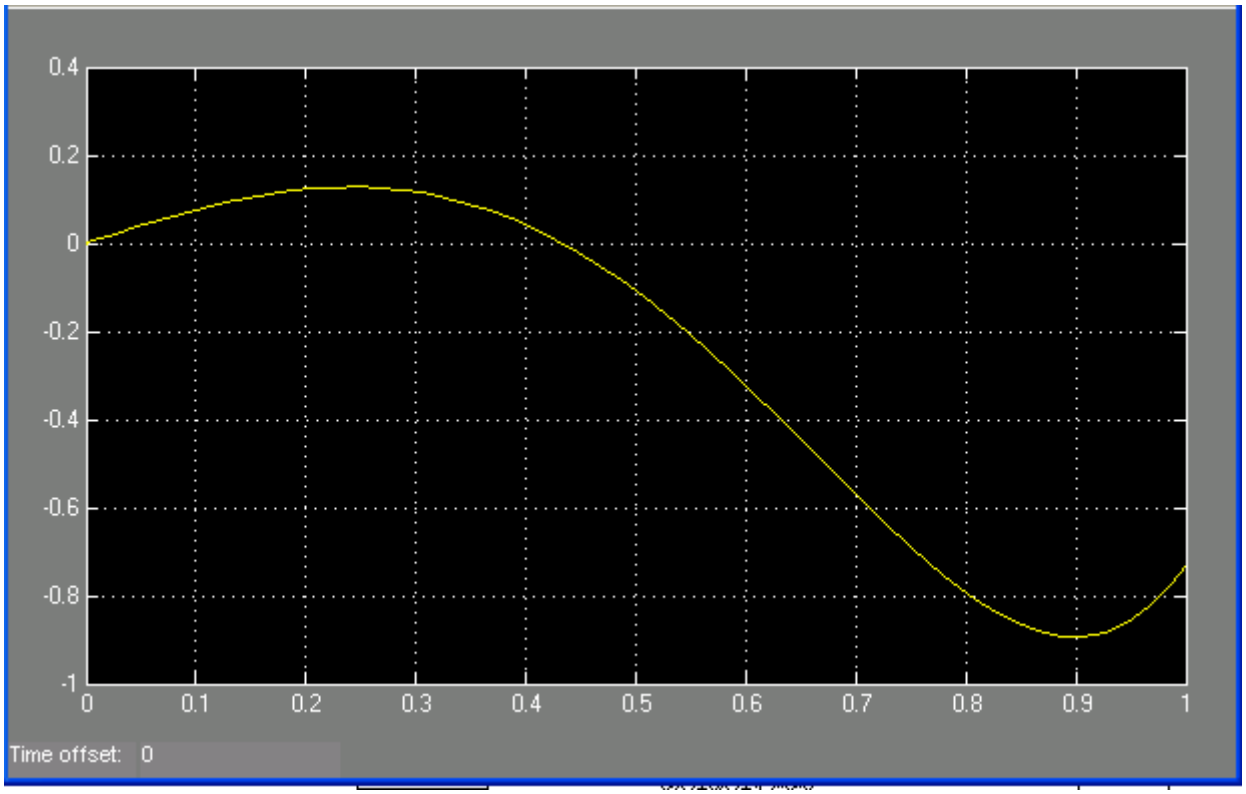

Рис. 9.4 Осциллограмма вычисления функционала

9.5 Верификация математических моделей

На рис.9.5, 9.6, 9.7 и 9.8 приведены графики экстремали рассматриваемого функционала и ее приближений, найденных по методу Ритца с использованием пакета символьной обработки Symbolic Math и математической системы MATLAB. Из рис. 9.6 видно, что графики точного решения и второго приближения практически совпадают.

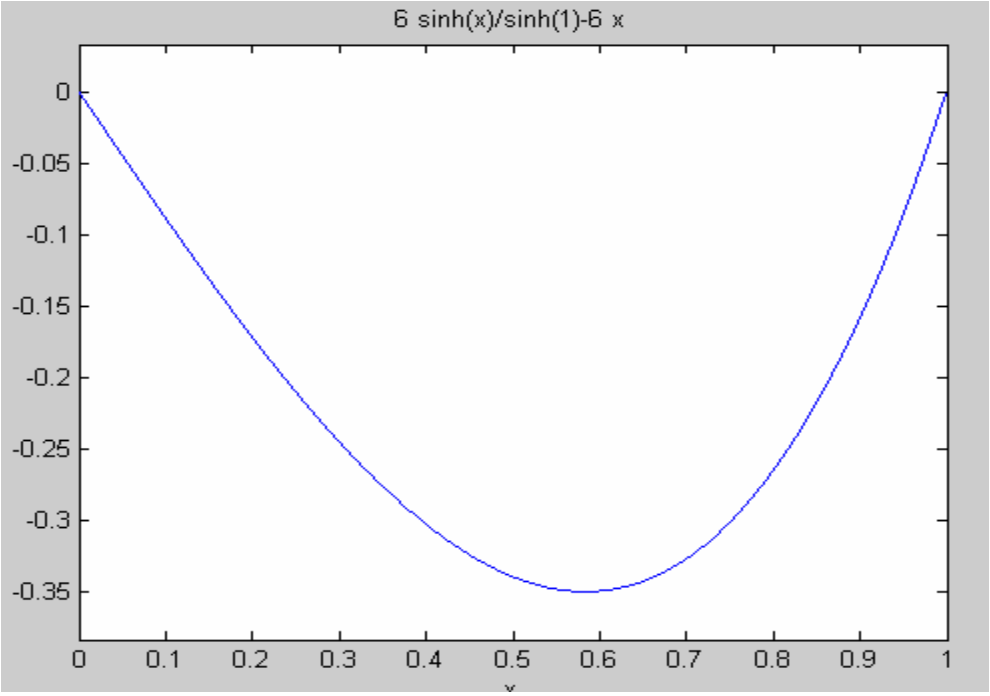

Рис.9.5 График экстремали функционала

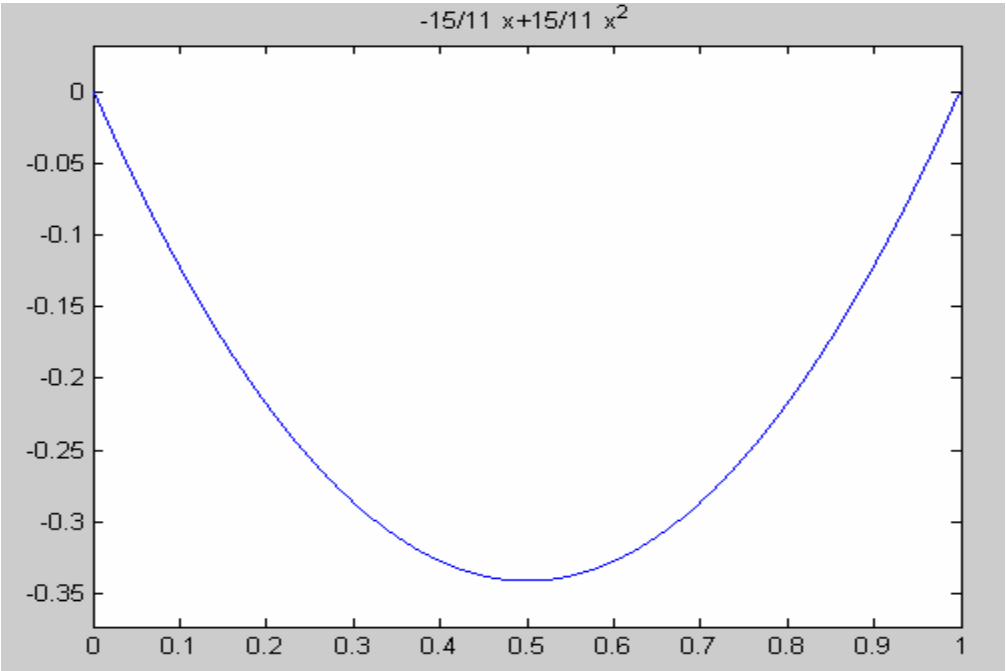

Рис.9.6 Первое приближение для экстремали

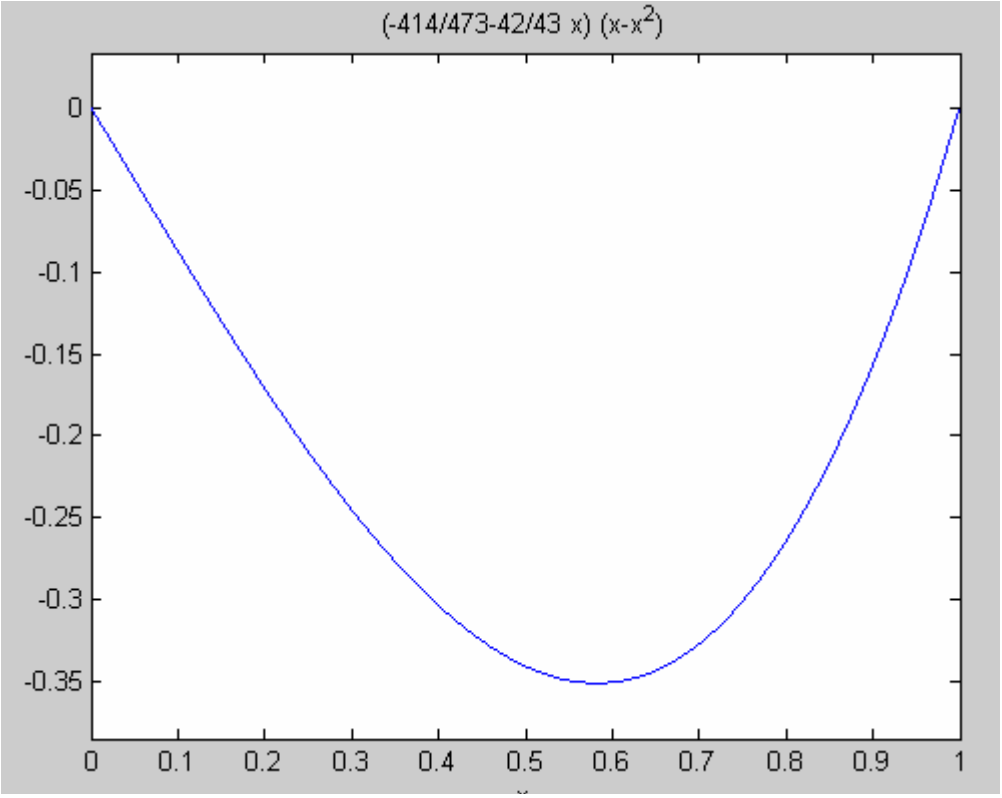

Рис.9.7 Второе приближение для экстремали

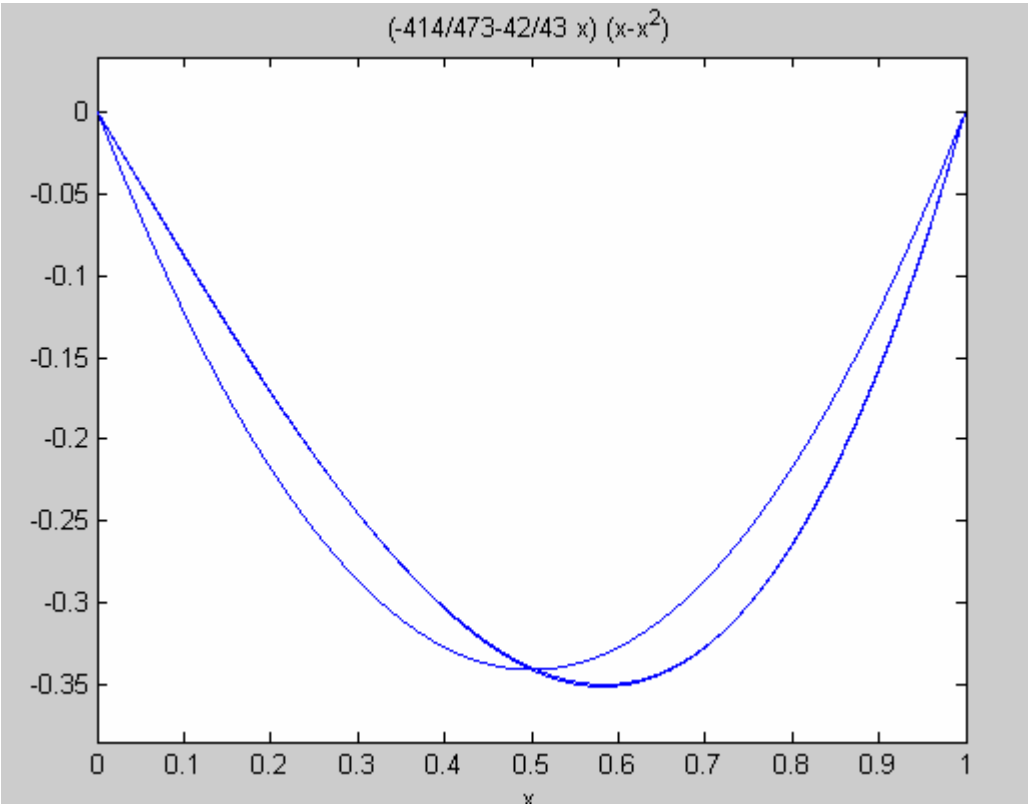

Рис.9.8 Экстремаль и ее первое и второе приближения

9.6 Варианты заданий и порядок их выполнения

1. По табл. 9.1 и 9.2 выбрать вариант функционала.
2. В зависимости от вида функционала определить формулы для вывода дифференциальных уравнений, определяющих экстремали функционала.
3. Найти выражения, определяющие экстремали функционала, решая аналитически составленные дифференциальные уравнения или используя для этих целей пакет символьной математики Symbolic Math Toolbox системы MATLAB.
4. Проинтегрировав найденное выражение в указанных пределах, найти экстремум функционала.
5. Используя метод Ритца или метод Б.Г.Галеркина, найти приближенные значения экстремума.
6. Построить график экстремали и ее приближений, а так же графики разности соответствующих функций – экстремали и ее приближений.
7. Составить имитационные модели для нахождения экстремали, ее приближений и экстремума, а так же приближенных значений экстремума.
8. Оценить устойчивость имитационных моделей.
9. Определить характеристики переходного процесса.
10. Исследовать реакцию моделей на синусоидальное возмущающее воздействие.
11. Оформить отчет, используя средства генерации описания моделей пакета Simulink.

9.7 Оформление отчета по результатам исследований

Для завершения лабораторной работы необходимо сгенерировать отчет в формате HTML, затем преобразовать его в формат RTF с помощью текстового редактора, включить в него теоретические результаты, отформатировать текст и графические объекты, записать на дискету и в электронном виде предъявить преподавателю.

Лабораторная работа № 10

ОПТИМАЛЬНОЕ УПРАВЛЕНИЕ ДИНАМИЧЕСКИМИ СИСТЕМАМИ С ПРИМЕНЕНИЕМ НЕЙРОРЕГУЛЯТОРОВ НА ОСНОВЕ ЭТАЛОННОЙ МОДЕЛИ

Цель работы: овладение методами идентификации динамических систем и проектирование искусственных нейронных сетей для оптимального управления такими системами на основе эталонной модели поведения.

10.1 Постановка задач исследования

Проектирование искусственных нейронных сетей для решения задач оптимального управления динамическими системами – нейронных регуля-торов осуществляется в два этапа.

На первом этапе производится идентификация динамической системы с помощью наборов входных и соответствующих выходных величин, разработка архитектуры нейронной модели динамической системы и настройка ее параметров с использованием идентифицирующих наборов. Созданная искусственная нейронная сеть с заданной точностью воспроизводит поведение динамической системы (идентифицирует ее) и используется для синтеза нейронного регулятора.

На втором этапе осуществляется поиск архитектуры искусственной нейронной сети для регулятора и требуемого закона управления динамической системой путем подбора параметров этой сети, соединенной с нейронной моделью системы, с тем, чтобы поведение объединенной сети с заданной точностью соответствовало некоторому эталонному поведению – поведению эталонной модели. Затем нейронная модель регулятора преобразуется в программный или аппаратный модуль для управления реальным объектом с заданной динамикой поведения.

В настоящее время используются и другие принципы построения нейрорегуляторов: нейрорегуляторы с предсказанием будущих реакций динамического процесса на случайные сигналы управления, нейрорегуляторы на основе модели авторегрессии со скользящим средним и т.д. И в этих случаях сначала производится нейронная идентификация управляемого объекта, а затем синтез искусственной нейронной сети для модели регулятора.

Рассмотрим проектирование нейронного регулятора на основе эталонной модели. Объектом управления является механическое звено робота, описываемое дифференциальным уравнением второго порядка с постоянными коэффициентами:

$$\frac{d^2\varphi}{dt^2} = -10\sin\varphi - 2\frac{d\varphi}{dt} + u\,, \qquad (10.1)$$

где φ – угол поворота звена;

u – момент, развиваемый двигателем постоянного тока.

Цель синтеза и обучения нейрорегулятора состоит в том, чтобы управлять объектом при любом внешнем воздействии r таким образом, что динамика поведения нейрорегулятора и объекта воспроизводит с заданной точностью поведение эталонной модели, определяемое следующим дифференциальным уравнением:

$$\frac{d^2 y_r}{dt^2} = -9y_r - 6\frac{dy_r}{dt} + 9r, \qquad (10.2)$$

где y_r – выход эталонной модели.

10.2 Исследование динамики объекта управления и эталонной модели

Аналитическое решение уравнений (10.1) и (10.2) для любых законов изменения внешних воздействий u(t) и r(t) не всегда возможно и не дает наглядного представления о динамике объекта управления и характере эталонного поведения. Поэтому исследования объекта и модели проведем с помощью их имитационного моделирования. На рис.10.1 представлена имитационная модель объекта управления, на рис.10.2 – эталонная модель, а на рис. 10.11 - модель для сравнения их динамики при воздействии Random Number. Свободные движения объекта при отсутствии вращающего момента u(t) и различных начальных условиях показаны на рис. 10.3, 10.4, 10.5 и 10.6, а модели – на рис. 10.7, 10.8, 10.9 и 10.10. График сравнения динамики объекта управления и эталонной модели при свободном движении изображен на рис. 10.12.

Задавая нулевые начальные условия на моделях и, подключая к ним одновременно различные источники сигналов: генератор наклонной линии Ramp, генератор перепада Step, генератор синусоиды Sine Wave и генератор случайных величин Random Number, можно оценить эти отличия в поведении при условиях, близких к реальным возмущениям на объект и модель. Графики на рис. 10.13, 10.14, 10.15 и 10.16 демонстрируют сравнение динамики объекта управления и эталонной модели для различных воздействий.

Таким образом, проектируемый нейронный регулятор должен свести к минимуму эти различия при любом внешнем воздействии на эталонную модель и на систему, состоящую из нейрорегулятора и объекта управления.

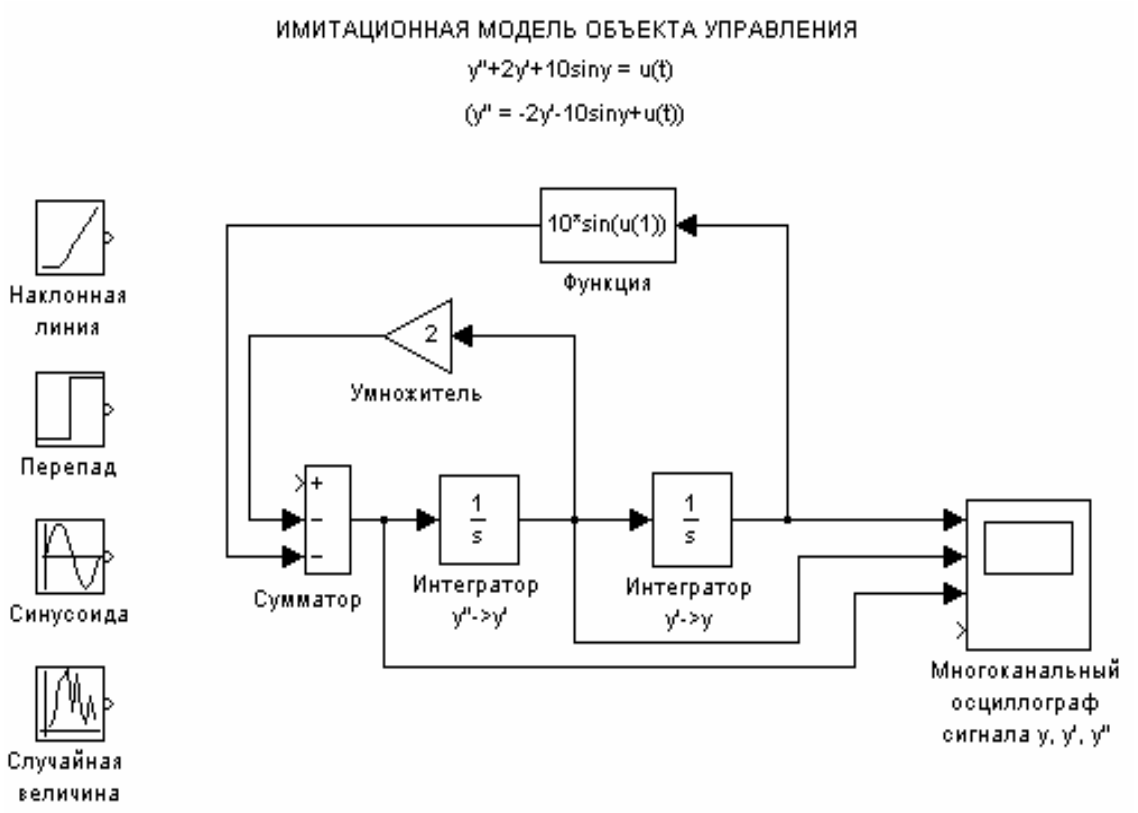

Рис. 10.1 Имитационная модель объекта управления

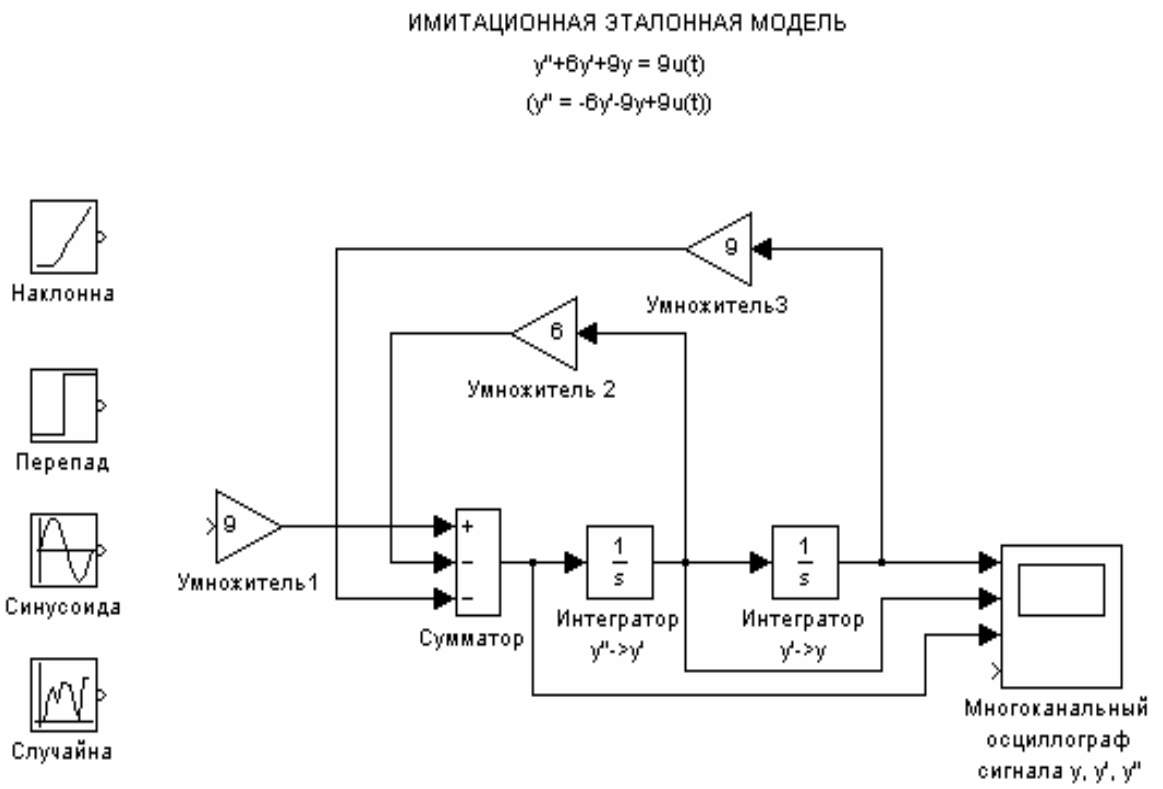

Рис. 10.2 Имитационная эталонная модель

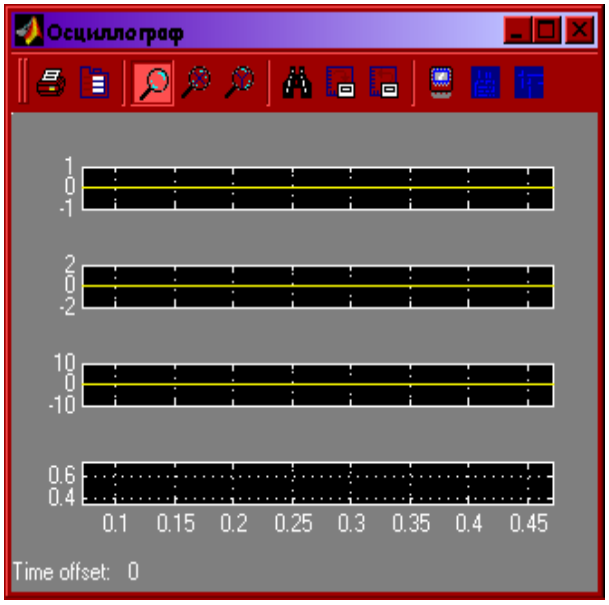

Рис 10.3 Свободные движения объекта управления при y'(0) = 0 и y(0) = 0

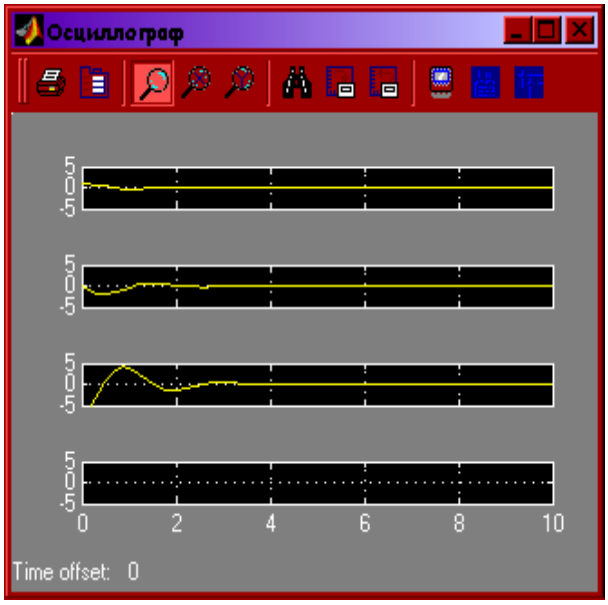

Рис 10.4 Свободные движения объекта управления при y'(0) = 0 и y(0) = 1

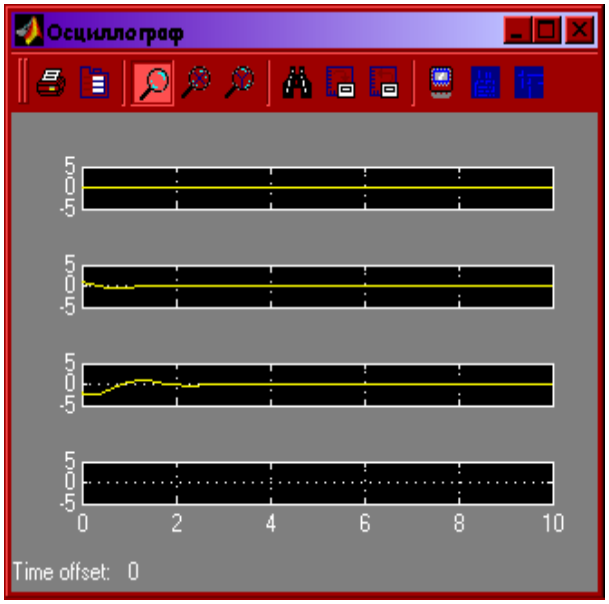

Рис 10.5 Свободные движения объекта управления при y'(0) = 1 и y(0) = 0

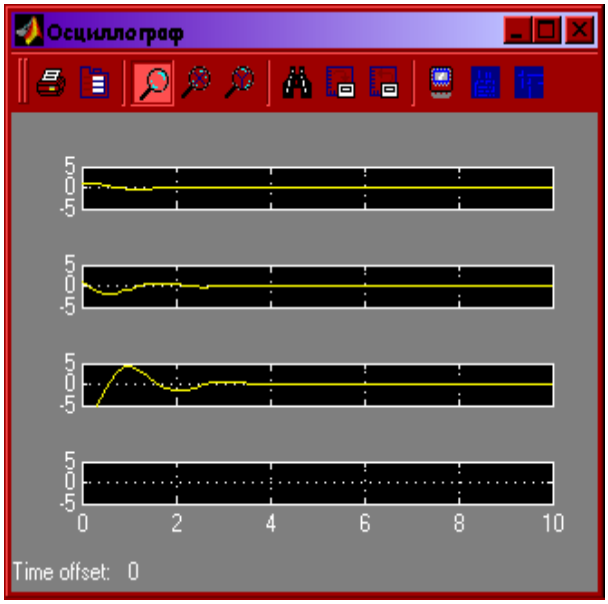

Рис 10.6 Свободные движения объекта управления при y'(0) = 1 и y(0) = 1

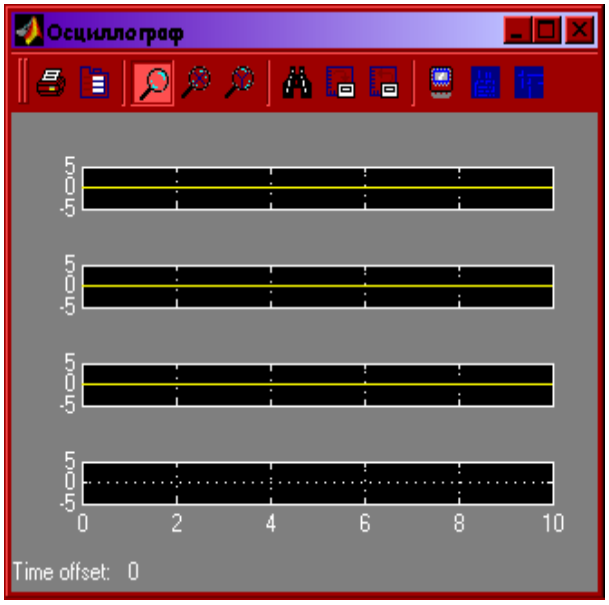

Рис 10.7 Свободные движения эталонной модели при y'(0) = 0 и y(0) = 0

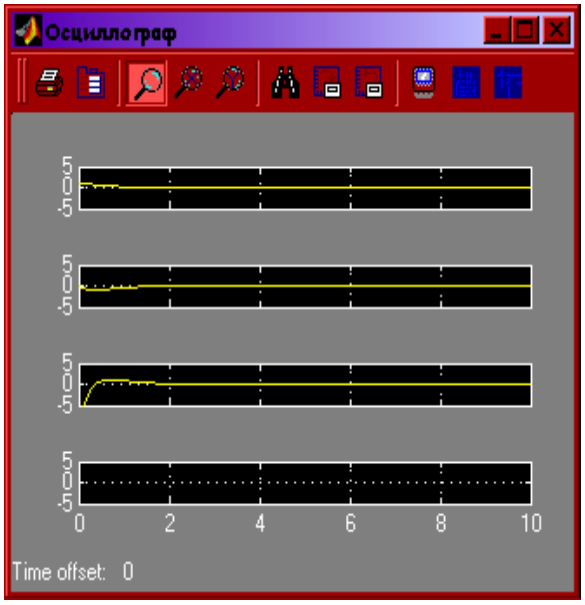

Рис 10.8 Свободные движения эталонной модели при y'(0) = 0 и y(0) = 1

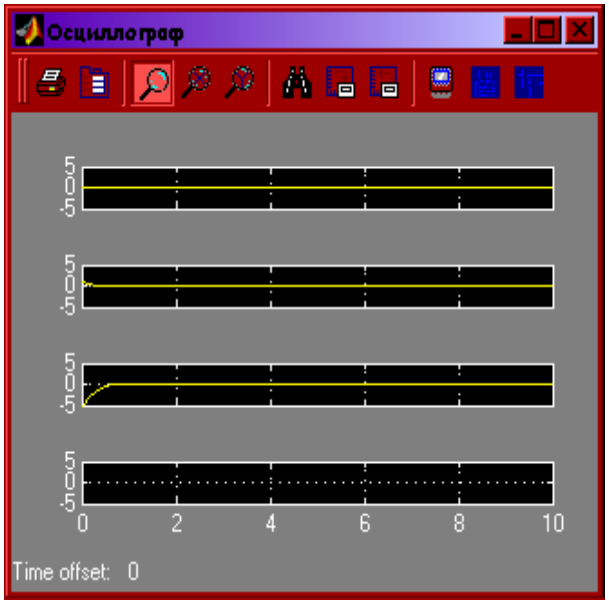

Рис 10.9 Свободные движения эталонной модели при y'(0) = 1 и y(0) = 0

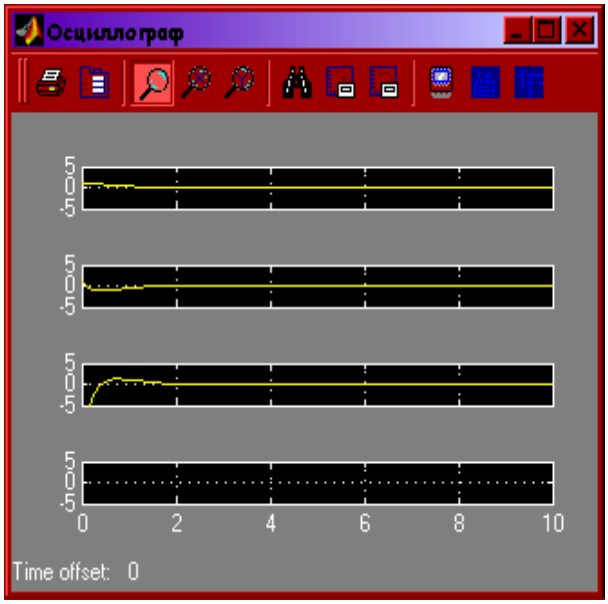

Рис 10.10 Свободные движения эталонной модели при y'(0) = 1 и y(0) = 1

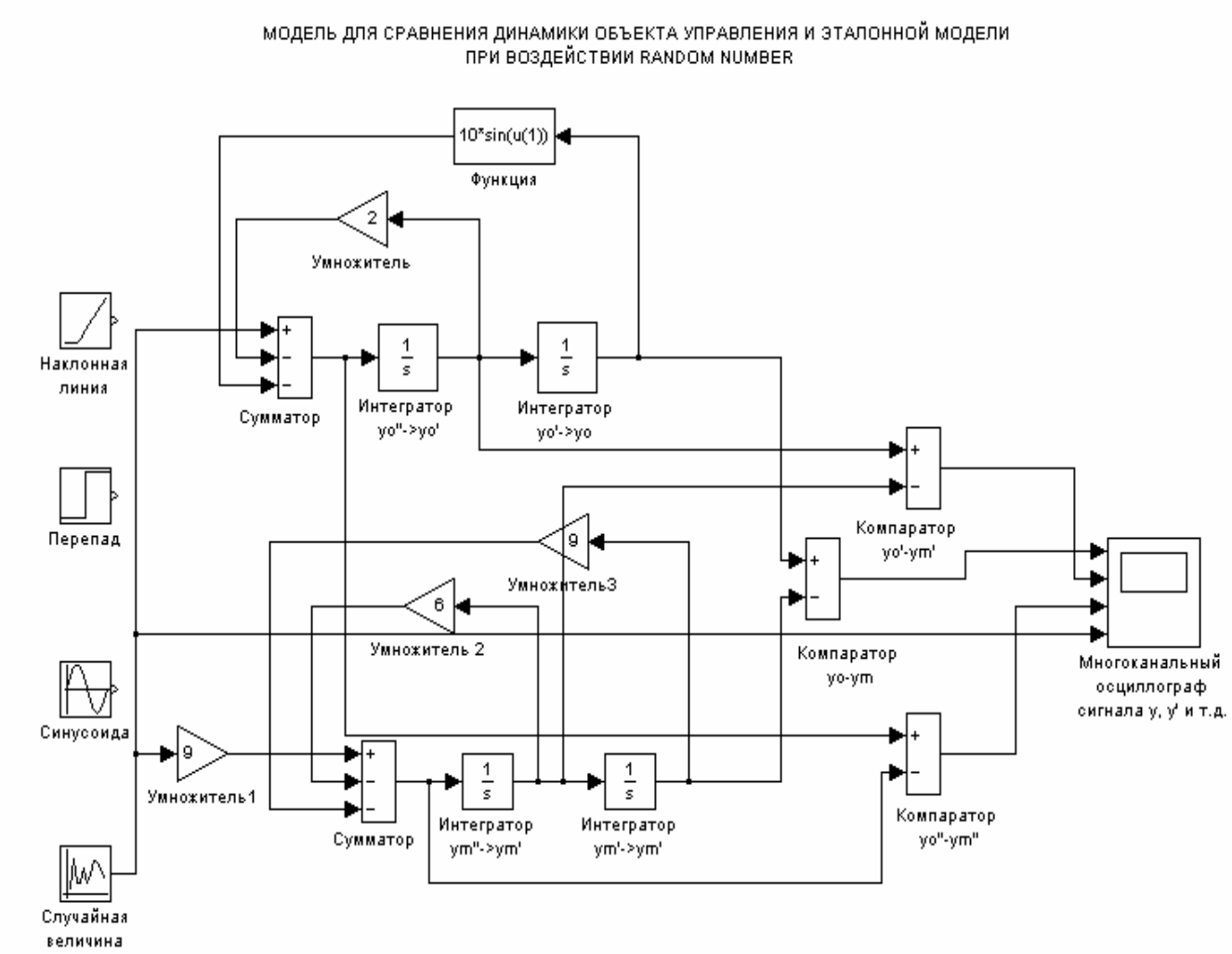

Рис. 10.11 Модель для сравнения динамики объекта управления и эталонной модели при воздействии Random Number

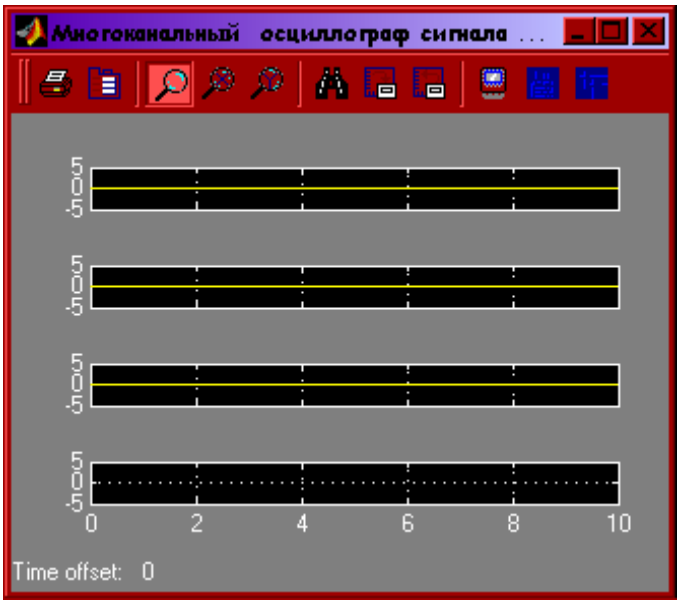

Рис. 10.12 Сравнение динамики объекта управления и эталонной модели при свободном движении

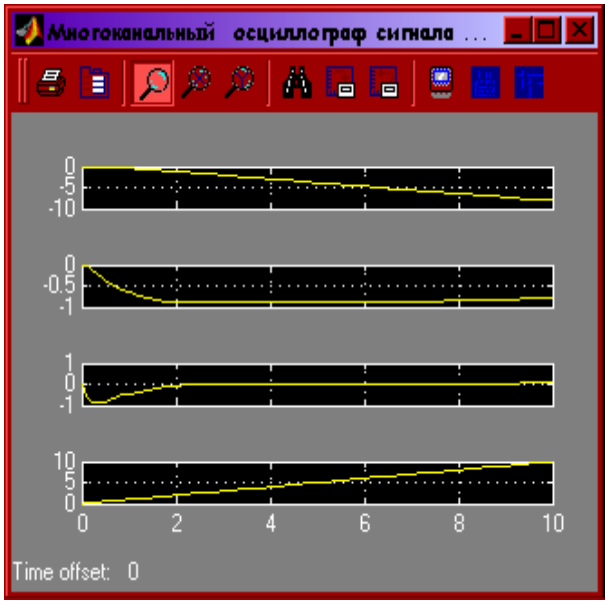

Рис. 10.13 Сравнение динамики объекта управления и эталонной модели при воздействии Ramp

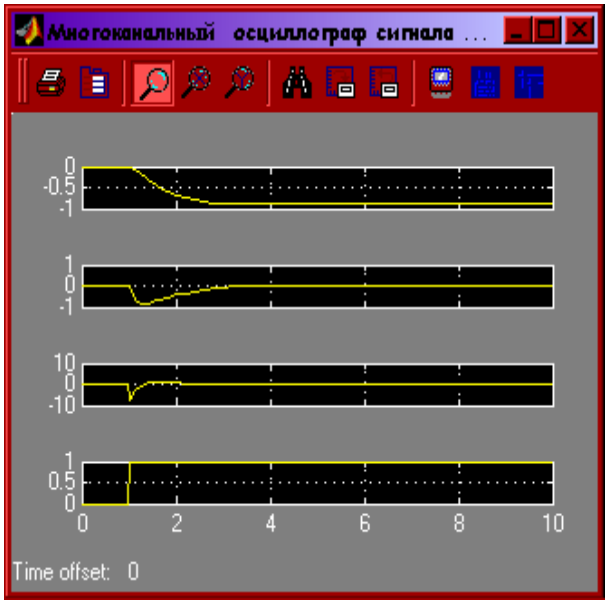

Рис. 10.14 Сравнение динамики объекта управления и эталонной модели при воздействии Step

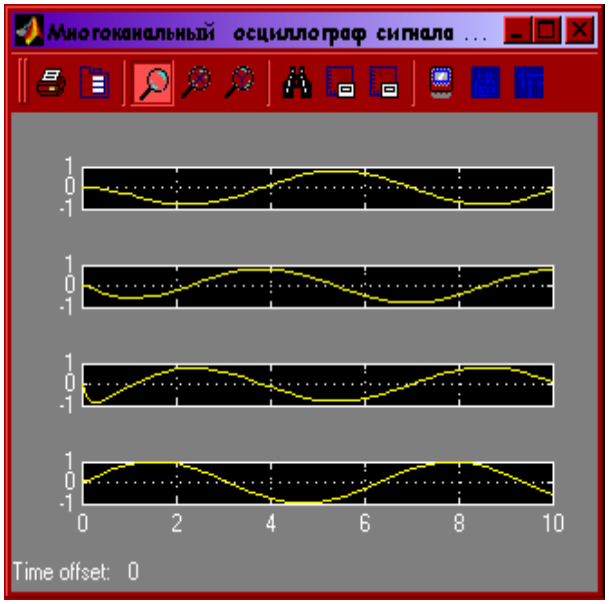

Рис. 10.15 Сравнение динамики объекта управления и эталонной модели при воздействии Sine Wave

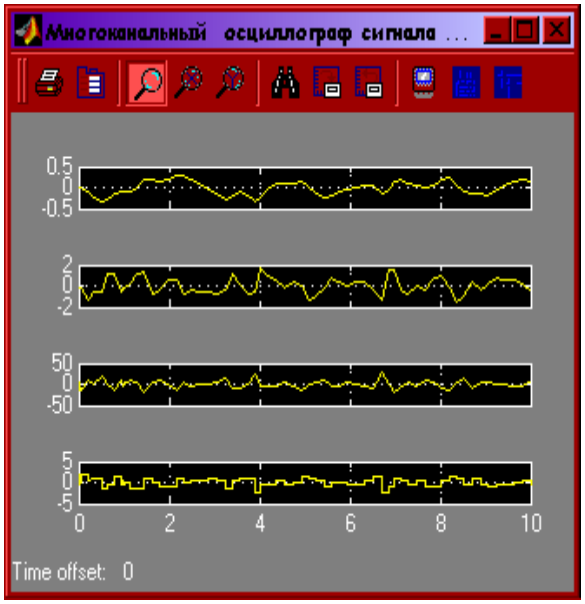

Рис. 10.16 Сравнение динамики объекта управления и эталонной модели при воздействии Random Number

10.3 Проектирование нейрорегулятора с использованием инструментальных средств системы MATLAB

Проектирование нейрорегуляторов для различных динамических систем значительно упрощается благодаря наличию в пакете NNT (Neural Networks Toolbox) специальных программных средств и системе имитационного моделирования Simulink.

Для проектирования нейронных регуляторов на основе эталонной модели запуск инструментальных средств производится исполнением команд mrefrobotarm (без нормализации данных) или mrefrobotarm2 (с нормализацией данных) и затем активизацией блока Model Reference Controller двойным щелчком левой кнопки мыши. В результате появляется диалоговая панель Model Reference Controller, которая позволяет определить архитектуру нейронной сети регулятора и произвести ее обучение. Однако, без идентификации объекта управления синтезировать нейрорегулятор невозможно. Поэтому необходимо открыть другую панель Plant Identification (идентификация механизма), щелкнув по кнопке Plant Identification.

Диалоговая панель для идентификации управляемого процесса Plant Identification входит в состав раздела Control Systems библиотеки нейронных блоков системы Simulink, является универсальным средством и может быть использована для построения нейросетевых моделей любых динамических объектов, которые могут быть представлены блоками этой системы.

С помощью управляющих элементов панели Plant Identification можно задать архитектуру нейронной сети, параметры обучающей последовательности и параметры обучения, а также управлять процессом идентификации и оценивать качество этого процесса.

Набор управляющих элементов для задания архитектурных параметров нейронной сети следующий:

1. Size of the Hiden Layer – количество нейронов во входном, или скрытом слое;
2. No. Delayed Plant Inputs – число задержек для входного слоя;
3. No. Delayed Plant Outputs – число задержек для выходного слоя;
4. Samling Interval – интервал квантования или шаг дискретности, в секундах, между двумя последовательными моментами отсчёта данных;
5. Notmalize Training Data – переключатель нормирования для преобразования обучающих данных к диапазону [0 1].

Набор управляющих элементов для задания характеристик обучающей последовательности таков:

1. Training Samples – число точек отсчёта для получения обучающей последовательности в виде пар значений вход-выход для управляемого процесса, определяемого моделью Simulink;
2. Maximum Plant Input – максимальное значение входного сигнала;
3. Minimum Plant Input – минимальное значение входного сигнала;
4. Maximum Interval Value (sec) – максимальный интервал идентификации (в секундах);
5. Minimum Interval Value (sec) – минимальный интервал идентификации (в секундах);
6. Limit Output Data – переключатель для ограничения значений выходного сигнала;
7. Maximum Plant Output – максимальное значение выходного сигнала, задаваемое при включённом переключателе Limit Output Data;

8. Minimum Plant Output – максимальное значение выходного сигнала, задаваемое при включённом переключателе Limit Output Data;

9. Simulink Plant Model – для задания модели управляемого процесса, реализованной с помощью блоков Simulink, имеющей порты входа и выхода и сохранённой в файле *.mdl; выбор модели производится с помощью кнопки Browse; имя модели отображается в специальном окне.

Параметры обучения задаются следующим образом:

1. Training Epochs – количество циклов обучения;

2. Training Function – для задания обучающей функции;

3. Use Current Weights – переключатель для использования текущих весов нейронной сети;

4. Use Validation Data – переключатель для использования контрольного множества в объёме 25**%** от обучающего множества;

5. Use Testing Data – переключатель для использования тестового множества в объёме 25**%** от обучающего множества.

Для идентификации управляемого процесса необходимо выполнить следующие действия:

1. Задать архитектуру нейронной сети, которая будет моделью управляемого процесса.

2. Задать параметры обучения.

3. Выбрать модель Simulink для управляемого процесса.

4. Сгенерировать обучающую последовательность заданного объёма, запустив модель Simulink с помощью кнопки Generate Training Data. Генерация обучающей последовательности производится с помощью воздействия ряда ступенчатых сигналов на модель управляемого процесса и снятия значений на входе и выходе модели через каждый шаг квантования. Графики входного и выходного сигнала отображаются в окне Plant Input-Output Data.

5. По завершении генерации обучающей последовательности необходимо либо принять эти данные, нажав на кнопку Accept Data, и тогда они будут использованы для обучения нейронной сети, либо отвергнуть их, нажав кнопку Reject Data, и повторить процесс идентификации управляемого процесса, представленного моделью Simulink.

6. После получения обучающей последовательности необходимо установить требуемые параметры обучения и с помощью кнопки Train Network запустить процесс обучения нейронной сети.

7. После завершения обучения его результаты отображаются на графиках изменения ошибки сети для обучающей, контрольной и тестирующей последовательностей, а также в окне выходных значений модели и сети при подаче на вход указанных последовательностей.

8. Если результаты обучения приемлемы, то надо сохранить параметры нейросетевой модели управляемого процесса и приступить к синтезу регулятора того или иного класса, нажав кнопки Apply и Ok.

9. Если результаты обучения неприемлемы, то следует нажать кнопку Cancel и повторить процесс идентификации сначала, изменяя архитектуру сети и параметры обучающей последовательности.

10. Обучающую последовательность можно импортировать из рабочей области или из файла, нажав на кнопку Import Data. Если необходимо обучающую последовательность сохранить в рабочей области или в файле для подбора параметров архитектуры нейронной сети, то следует после получения данных нажать на кнопку Export Data.

11. Если необходимо удалить только что сгенерированные данные, то следует нажать кнопку Erase Generated Data.

Таким образом, диалоговая панель Plant Identification позволяет идентифицировать управляемый процесс, представленный в виде имитационной модели Simulink, построить двухслойную нейронную сеть прямой передачи сигналов с необходимым числом нейронов и линий задержки, обучить эту сеть для получения нейронной модели управляемого процесса, оценить качество обучения и работу нейронной сети.

Архитектура нейронной модели регулятора аналогична архитектуре нейронной модели объекта, поэтому управляющие элементы на панели Model Reference Control такие же, что и на панели Plant Identification за небольшим исключением. Так, на панели имеется кнопка для инициирования процесса идентификации управляемого объекта Plant Identification, отсутствуют управляющие элементы для задания характеристик выходного сигнала, так как он непосредственно поступает на вход модели объекта, а обучающая последовательность разбивается на сегменты, для чего имеется специальное поле Controller Training Segments**.**

Для синтеза регулятора необходимо определить все требуемые параметры на панели, сгенерировать обучающие последовательности, нажав на кнопку Training Data и обучить нейронную сеть с помощью кнопки Train Controller, используя текущие веса и режим обучения с накоплением (если необходимо). Затем нажать на кнопку Apply для завершения процесса синтеза регулятора.

По окончании построения регулятора необходимо нажать на кнопку OK, вернуться в окно Simulink и выполнить моделирование работы системы нейронного регулирования для оценки характеристик регулятора.

На рис. 10.17 представлена модель для оценки управляющего сигнала и качества регулирования, на рис. 10.18 – сигнал внешнего воздействия и управляющий сигнал, а на рис. 10.19 – сравнение выходных сигналов эталонной модели и управляемого объекта.

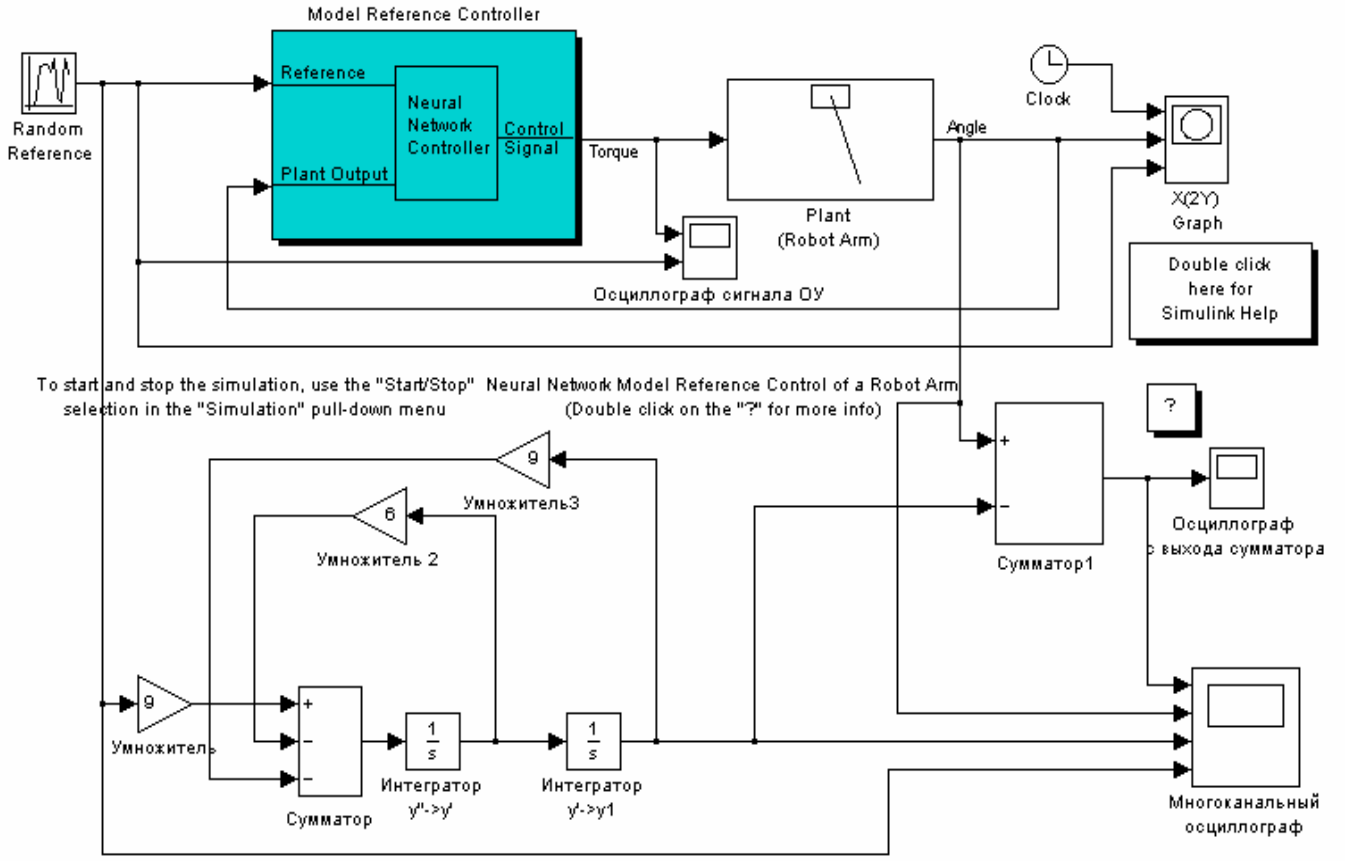

Рис. 10.17 Модель для оценки управляющего сигнала и качества регулирования

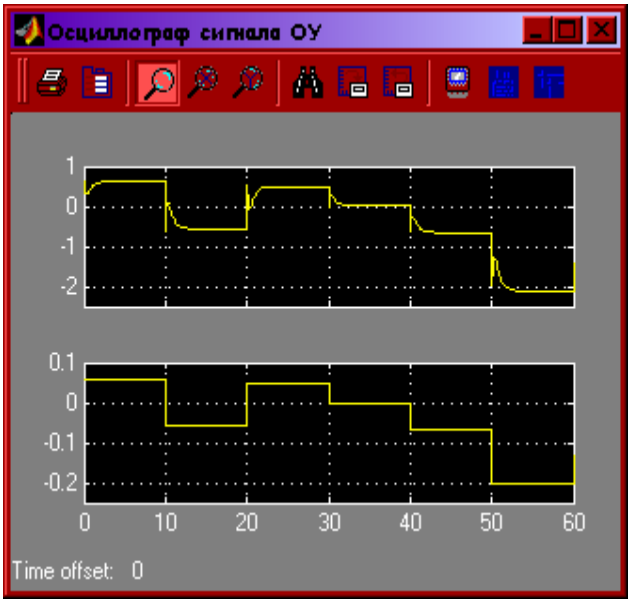

Рис. 10.18 Сигнал внешнего воздействия и управляющий сигнал

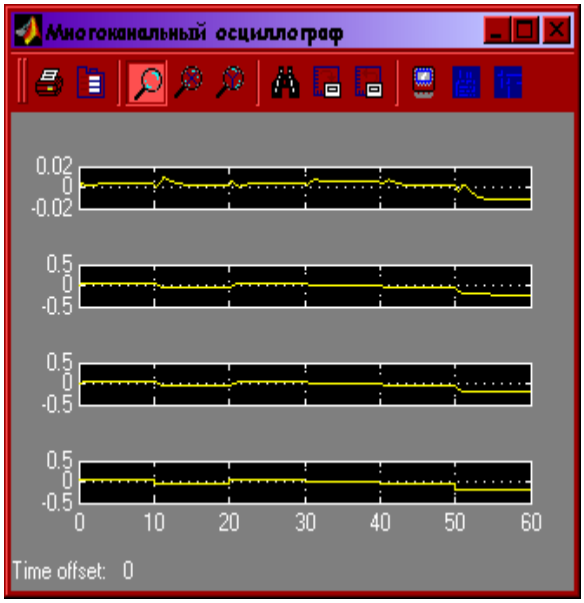

Рис. 10.19 Сравнение выходных сигналов эталонной модели и управляемого объекта

10.4 Программная реализация процедуры синтеза нейрорегулятора

Для программной реализации этапов проектирования нейрорегулятора в качестве нейронной модели, как для регулятора, так и для объекта будем использовать двухслойные нейронные сети с прямой передачей сигналов. Входной слой каждой сети имеет несколько нейронов, а функциями взвешивания, накопления и активации являются соответственно dotprod, netsum и logsig. Выходные слои для обоих случаев имеют такие же функции взвешивания и накопления, а функцией активации является для них линейная функция purelin. Сети с такой архитектурой могут воспроизводить весьма сложные нелинейные зависимости между входом и выходом.

Затем эти сети объединяются в одну сеть. Программно объединенная сеть создается одной функцией newff. Два первых слоя исключаются из циклов обучения с помощью признака learn, которому в этих слоях задается значение 0, и вся сеть, а точнее последние два слоя обучаются на наборах вход – выход, полученных на этапе идентификации объекта с помощью имитационной модели и блоков To Workspace. В результате обучения формируется модель объекта управления. Для формирования модели регулятора объединенную сеть вновь обучают, но уже на наборах вход – выход для эталонной модели и выключенными из процесса обучения двумя последними нейронными слоями. Так осуществляется синтез нейрорегулятора.

Изменяя число нейронов в слоях, количество линий задержки на входе, между слоями и в обратной связи, а также интервал квантования и объем обучающих последовательностей, добиваются требуемой точности моделей.

Если в нейронной сети используются линии задержки, то для ее обучения надо задавать функцию обучения, основанную на динамическом варианте метода обратного распространения ошибки trainbfgc (см. m-функции в файлах …\toolbox\nnet\nncontrol\trainbfgc.m и …\toolbox\nnet\nncontrol \srchbacxc.m).

По умолчанию для сетей с прямой передачей сигналов в качестве критерия обучения используется функционал, представляющий собой сумму квадратов ошибок между выходами сети и их целевыми значениями:

$$J = \frac{1}{2}\sum_{q=1}^{Q}\sum_{i=1}^{S}(t_i^q - d_i^q)^2, \qquad (10.3)$$

где Q – объём выборки;

q – номер выборки;

i – номер выхода;

t_i^q – целевое значение для i-го выхода выборки q;

d_i^q – сигнал на i-м выходе при подаче входных сигналов q-й выборки. Целью обучения сети является минимизация этого функционала с помощью изменения весов и смещений.

В настоящее время разработано несколько методов минимизации функционала ошибки на основе известных методов определения экстремумов функций нескольких переменных. Все эти методы можно разделить на три класса:

а) методы нулевого порядка, в которых для нахождения минимума используется только информация о значениях функционала в заданных точках;

б) методы первого порядка, в которых используется градиент функционала ошибки по настраиваемым параметрам, использующий частные производные функционала;

в) методы второго порядка, в которых используются вторые производные функционала.

Для линейных сетей задача нахождения минимума функционала (параболоида) сводится к решению системы линейных уравнений, включающих веса, смещения, входные обучающие значения и целевые выходы и, таким образом, может быть решена без использования итерационных методов. Во всех остальных случаях надо использовать методы первого или второго порядка.

Если используется градиент функционала ошибки, то

$$X_{k+1} = X_k - \alpha_k g_k, \qquad (10.4)$$

где X_k и X_{k+1} – векторы параметров на k-й и k+1-й итерациях;

α_k – параметр скорости обучения;

g_k – градиент функционала, соответствующий k-й итерации.

Если используется сопряжённый градиент функционала, то на первой итерации направление движения p_0 выбирают против градиента g_0 этой итерации:

$$p_0 = -g_0 . \qquad (10.5)$$

Для следующих итераций направление p_k выбирают как линейную комбинацию векторов g_k и p_{k-1}:

$$p_k = -g_k + \beta_k p_{k-1}, \qquad (10.6)$$

а вектор параметров рассчитывают по формуле:

$$X_{k+1} = X_k + \alpha_k p_k, \qquad (10.7)$$

Для методов второго порядка расчет параметров на *k*-м шаге производят по формуле (метод Ньютона):

$$X_{k+1} = X_k - H_k^{-1} g_k, \qquad (10.8)$$

где H_k – матрица вторых частных производных целевой функции (матрица Гессе);

g_k – вектор градиента на *k*-й итерации. Вычисление матрицы Гессе требует больших затрат машинного времени, поэтому её заменяют приближенными выражениями (квазиньютоновские алгоритмы).

Программа синтеза нейрорегулятора на основе эталонной модели универсальна и имеет следующий вид:

```
function = RefArm
%
%-- Программа построения нейронных моделей объекта управления и
%-- регулятора:
%
RefArmNet=newff([-0.5 0.5],[13 1 10 1],{'tansig' 'purelin' 'tansig' 'purelin'},'traingdx');
%
%-- Обучение нейронной модели объекта управления:
%
RefArmNet.inputWeights{1,1}.learn=0;
RefArmNet.layerWeights{2,1}.learn=0;
RefArmNet.layerWeights{3,2}.learn=1;
RefArmNet.layerWeights{4,3}.learn=1;
RefArmNet.trainParam.epochs=300;
RefArmNet.trainParam.goal=1e-5;
RefArmNet.trainParam.min_grad=1e-45;
```

```
RefArmNet.trainParam.mu_max=1e50;
RefArmNet=train(RefArmNet, Parm', Tarm');
Yarm=sim(RefArmNet,Parm');
hold off
plot(0:0.01:100,Tarm','r');
hold on
plot(0:0.01:100,Yarm,'b');
hold off
%
%-- Обучение нейронной модели регулятора:
%
RefArmNet.layerWeights{1,1}.learn=1;
RefArmNet.layerWeights{2,1}.learn=1;
RefArmNet.layerWeights{3,2}.learn=0;
RefArmNet.layerWeights{4,3}.learn=0;
RefArmNet.trainParam.epochs=300;
RefArmNet.trainParam.goal=1e-5;
RefArmNet.trainParam.min_grad=1e-45;
RefArmNet.trainParam.mu_max=1e50;
RefArmNet=train(RefArmNet, Preg', Treg');
Yreg=sim(RefArmNet,Preg');
hold off
plot(0:0.01:100,Treg','r');
hold on
plot(0:0.01:100,Yreg,'b');
```

10.5 Варианты заданий и порядок выполнения работы

Варианты объектов управления и эталонных моделей следует выбирать по табл. 4.2 лабораторной работы №4 или использовать звенья, которые исследовались в лабораторных работах №5, 6, 7 и 8. Для выбранного варианта необходимо построить имитационные модели и сохранить их в своей рабочей папке. Обе модели должны быть снабжены портами входа и выхода (см. блоки In1 и Out1 библиотеки Ports&Subsystems).

Далее необходимо исследовать динамику объекта управления и эталонной модели при изменении их параметров, зафиксировать несколько таких наборов, как для объекта, так и для модели и сгенерировать для них отчеты.

С помощью инструментальных средств системы MATLAB произвести идентификацию объекта robotarm и синтез нейрорегулятора для эталонной модели robotref, задавая различные алгоритмы обучения и варьируя объем обучающего множества. Затем выполнить эти действия, поменяв роли моделей и сделав robotref объектом управления, а robotarm – эталонной моделью. Для выбранных вариантов объектов и для ряда эталонных моделей синтезировать нейрорегуляторы и сохранить данные для отчета. Осциллограммы обучения и синтеза также вставить в отчет.

Используя программу и изучив реализацию квазиньютоновского алгоритма обучения trainbfgc, повторить анализ объекта и синтез нейрорегулятора для одного набора параметров. Синтезированную сеть регулятора преобразовать в исполняемый код и проверить ее работу в реальном масштабе времени, создав для этих целей программную модель объекта управления.

По промежуточным отчетам и результатам исследований оформить электронный отчет.

10.6 Оформление отчета по результатам исследований

Отчет должен содержать математическую постановку задач исследования, осциллограммы по динамике объектов управления и эталонных моделей, архитектуру искусственной нейронной сети и ее параметры после идентификации объекта управления и синтеза нейрорегулятора, анализ качества регулирования при различных внешних воздействиях и результаты испытаний для реального времени.

Лабораторная работа № 10

ОПТИМАЛЬНОЕ УПРАВЛЕНИЕ ДИНАМИЧЕСКИМИ СИСТЕМАМИ С ПРИМЕНЕНИЕМ НЕЙРОРЕГУЛЯТОРОВ НА ОСНОВЕ ЭТАЛОННОЙ МОДЕЛИ

Цель работы: овладение методами идентификации динамических систем и проектирование искусственных нейронных сетей для оптимального управления такими системами на основе эталонной модели поведения.

10.1 Постановка задач исследования

Проектирование искусственных нейронных сетей для решения задач оптимального управления динамическими системами – нейронных регуля-торов осуществляется в два этапа.

На первом этапе производится идентификация динамической системы с помощью наборов входных и соответствующих выходных величин, разработка архитектуры нейронной модели динамической системы и настройка ее параметров с использованием идентифицирующих наборов. Созданная искусственная нейронная сеть с заданной точностью воспроизводит поведение динамической системы (идентифицирует ее) и используется для синтеза нейронного регулятора.

На втором этапе осуществляется поиск архитектуры искусственной нейронной сети для регулятора и требуемого закона управления динамической системой путем подбора параметров этой сети, соединенной с нейронной моделью системы, с тем, чтобы поведение объединенной сети с заданной точностью соответствовало некоторому эталонному поведению – поведению эталонной модели. Затем нейронная модель регулятора преобразуется в программный или аппаратный модуль для управления реальным объектом с заданной динамикой поведения.

В настоящее время используются и другие принципы построения нейрорегуляторов: нейрорегуляторы с предсказанием будущих реакций динамического процесса на случайные сигналы управления, нейрорегуляторы на основе модели авторегрессии со скользящим средним и т.д. И в этих случаях сначала производится нейронная идентификация управляемого объекта, а затем синтез искусственной нейронной сети для модели регулятора.

Рассмотрим проектирование нейронного регулятора на основе эталонной модели. Объектом управления является механическое звено робота, описываемое дифференциальным уравнением второго порядка с постоянными коэффициентами:

$$\frac{d^2\varphi}{dt^2} = -10\sin\varphi - 2\frac{d\varphi}{dt} + u\,, \qquad (10.1)$$

где φ – угол поворота звена;

u – момент, развиваемый двигателем постоянного тока.

Цель синтеза и обучения нейрорегулятора состоит в том, чтобы управлять объектом при любом внешнем воздействии r таким образом, что динамика поведения нейрорегулятора и объекта воспроизводит с заданной точностью поведение эталонной модели, определяемое следующим дифференциальным уравнением:

$$\frac{d^2 y_r}{dt^2} = -9y_r - 6\frac{dy_r}{dt} + 9r, \qquad (10.2)$$

где y_r – выход эталонной модели.

10.2 Исследование динамики объекта управления и эталонной модели

Аналитическое решение уравнений (10.1) и (10.2) для любых законов изменения внешних воздействий u(t) и r(t) не всегда возможно и не дает наглядного представления о динамике объекта управления и характере эталонного поведения. Поэтому исследования объекта и модели проведем с помощью их имитационного моделирования. На рис.10.1 представлена имитационная модель объекта управления, на рис.10.2 – эталонная модель, а на рис. 10.11 - модель для сравнения их динамики при воздействии Random Number. Свободные движения объекта при отсутствии вращающего момента u(t) и различных начальных условиях показаны на рис. 10.3, 10.4, 10.5 и 10.6, а модели – на рис. 10.7, 10.8, 10.9 и 10.10. График сравнения динамики объекта управления и эталонной модели при свободном движении изображен на рис. 10.12.

Задавая нулевые начальные условия на моделях и, подключая к ним одновременно различные источники сигналов: генератор наклонной линии Ramp, генератор перепада Step, генератор синусоиды Sine Wave и генератор случайных величин Random Number, можно оценить эти отличия в поведении при условиях, близких к реальным возмущениям на объект и модель. Графики на рис. 10.13, 10.14, 10.15 и 10.16 демонстрируют сравнение динамики объекта управления и эталонной модели для различных воздействий.

Таким образом, проектируемый нейронный регулятор должен свести к минимуму эти различия при любом внешнем воздействии на эталонную модель и на систему, состоящую из нейрорегулятора и объекта управления.

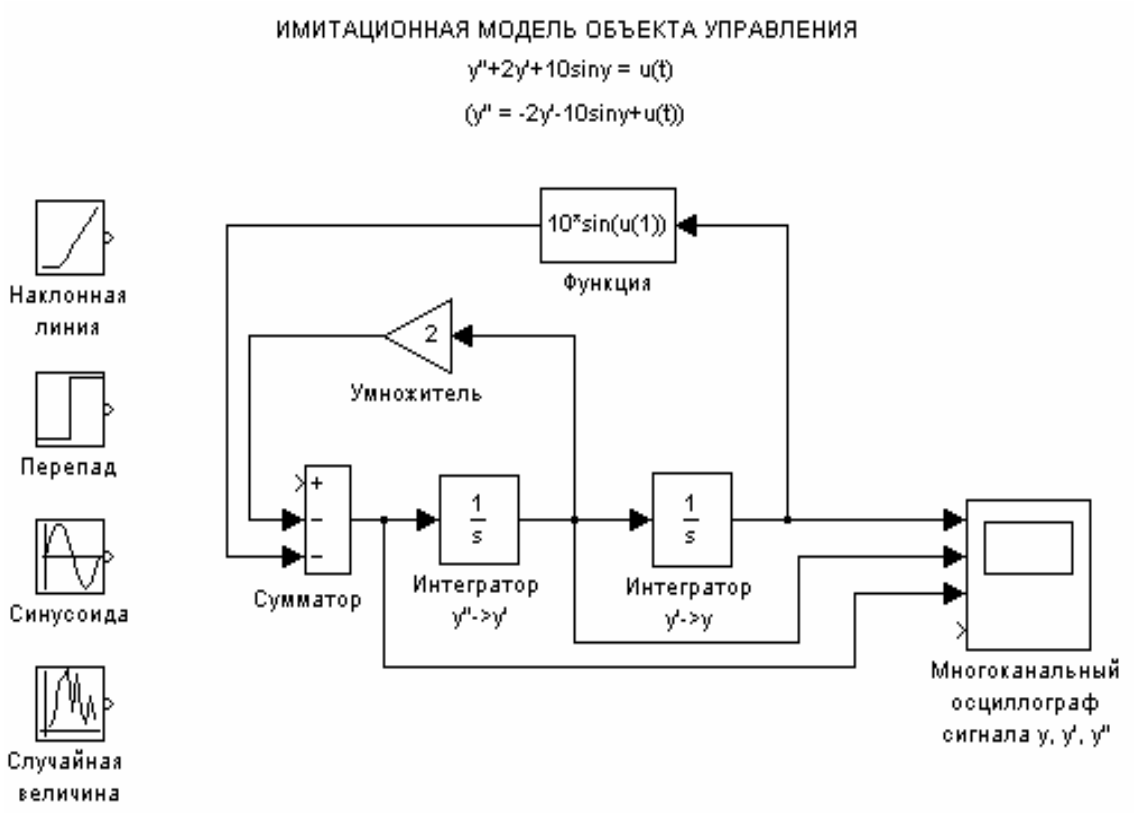

Рис. 10.1 Имитационная модель объекта управления

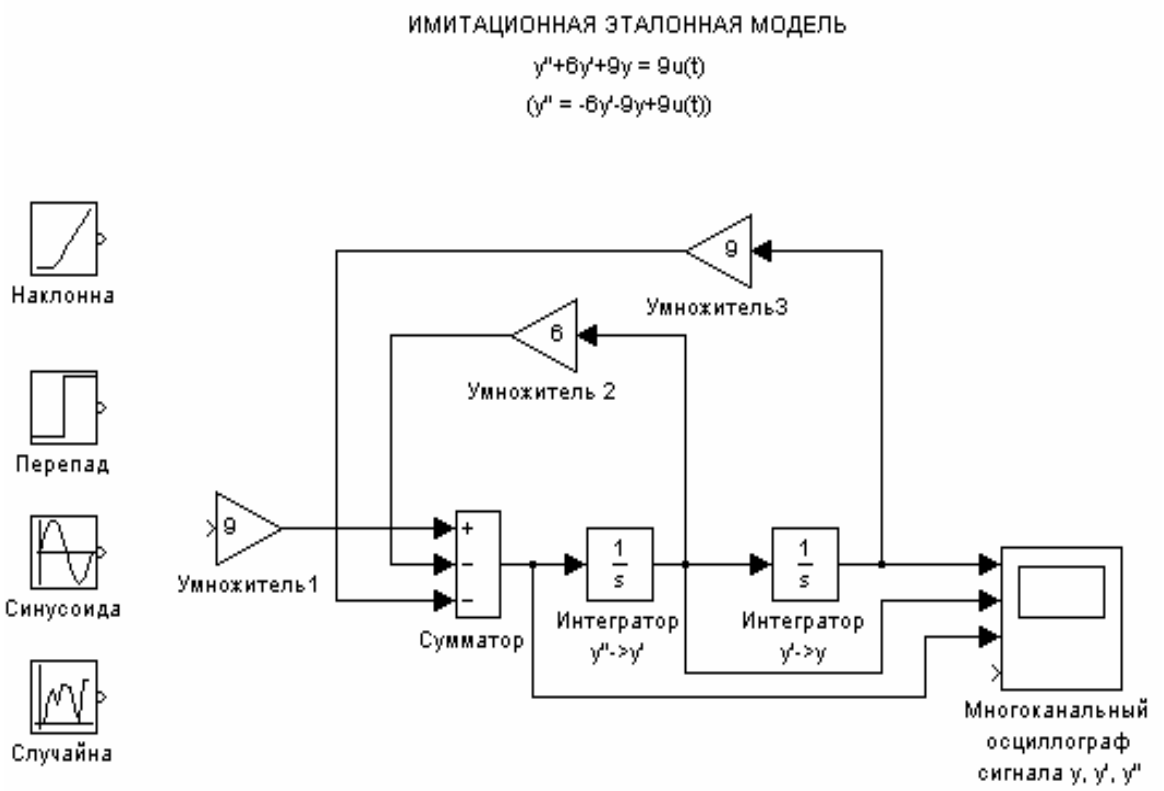

Рис. 10.2 Имитационная эталонная модель

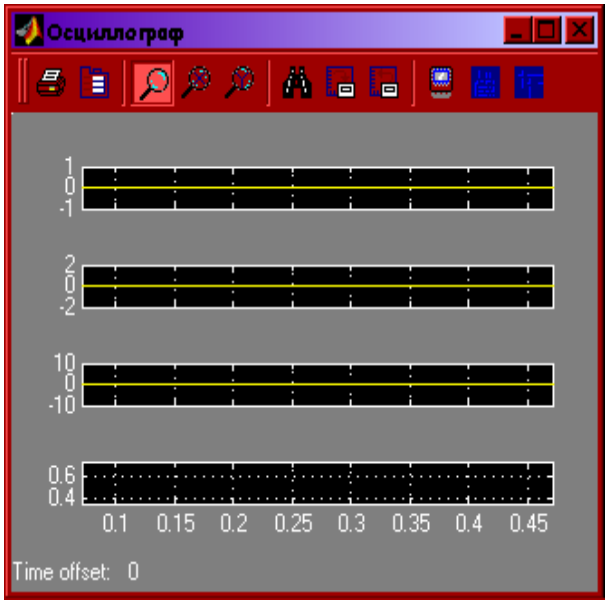

Рис 10.3 Свободные движения объекта управления при y'(0) = 0 и y(0) = 0

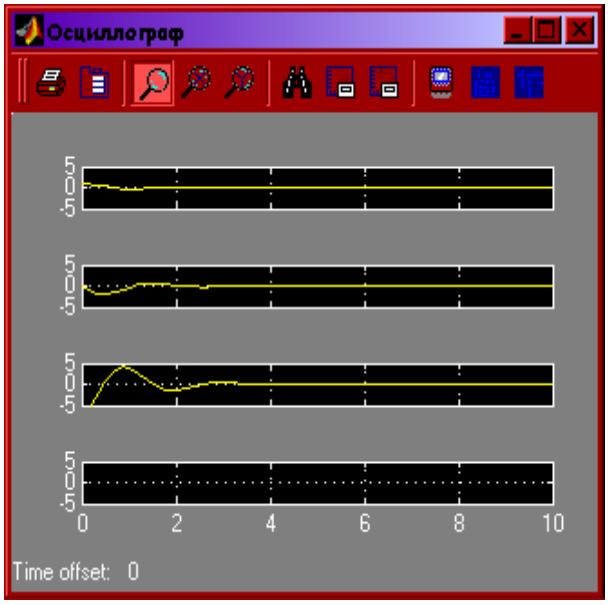

Рис 10.4 Свободные движения объекта управления при y'(0) = 0 и y(0) = 1

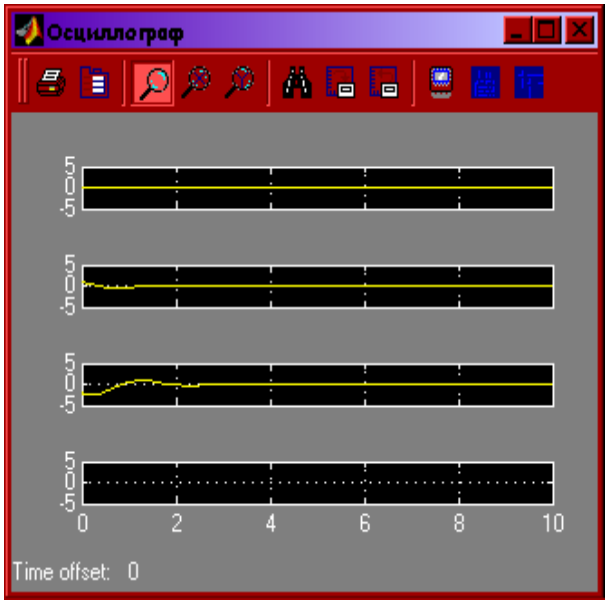

Рис 10.5 Свободные движения объекта управления при y'(0) = 1 и y(0) = 0

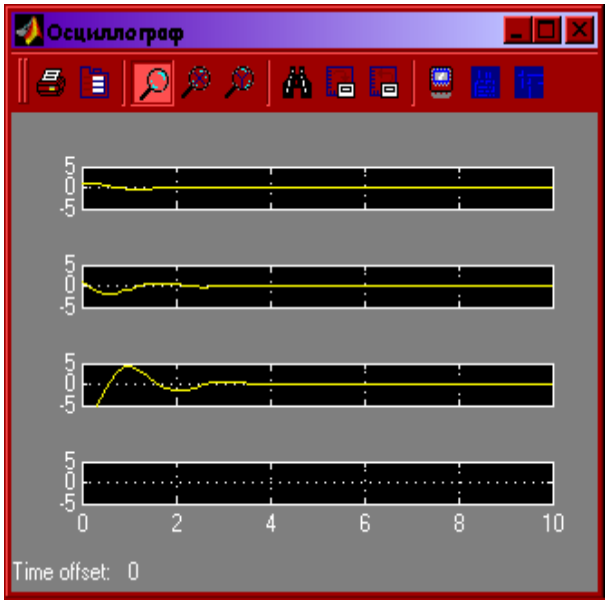

Рис 10.6 Свободные движения объекта управления при y'(0) = 1 и y(0) = 1

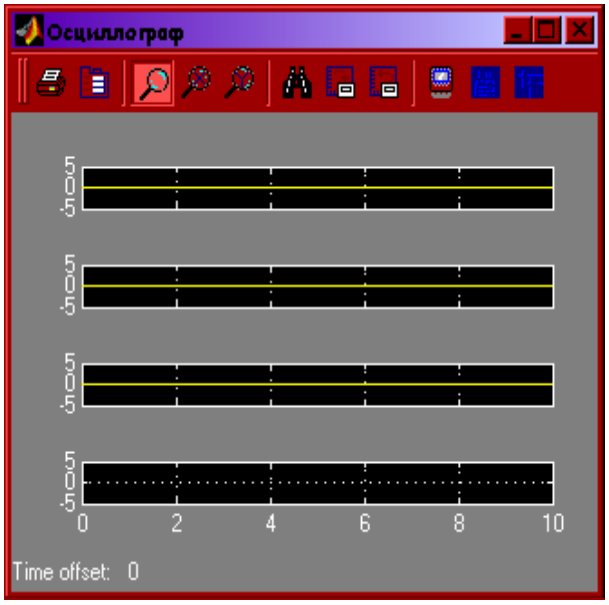

Рис 10.7 Свободные движения эталонной модели при y'(0) = 0 и y(0) = 0

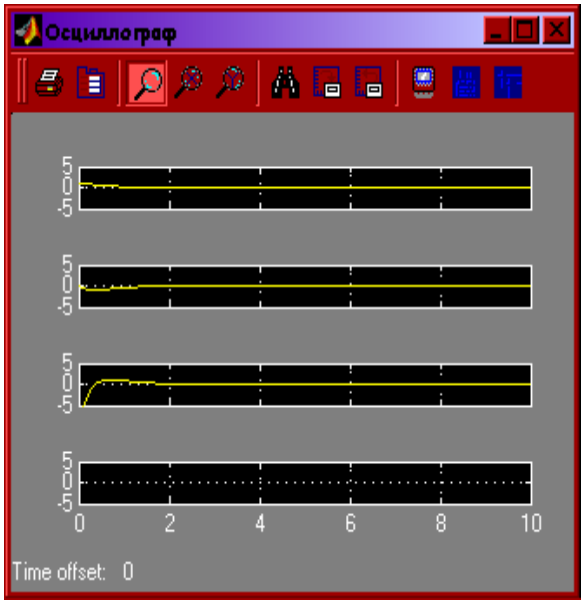

Рис 10.8 Свободные движения эталонной модели при y'(0) = 0 и y(0) = 1

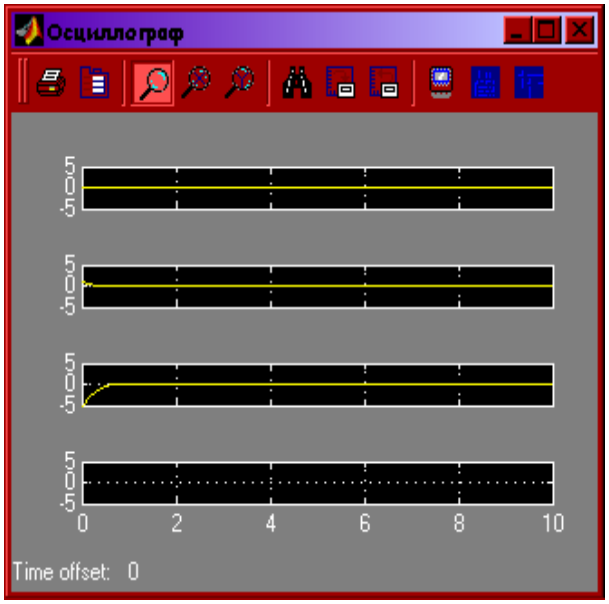

Рис 10.9 Свободные движения эталонной модели при y'(0) = 1 и y(0) = 0

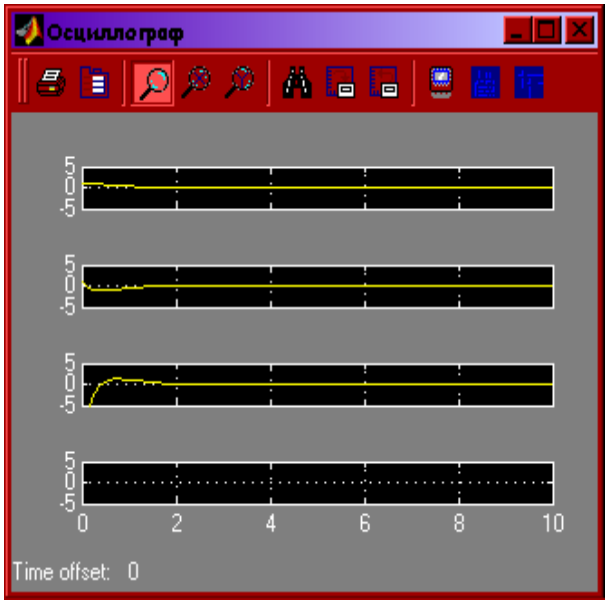

Рис 10.10 Свободные движения эталонной модели при y'(0) = 1 и y(0) = 1

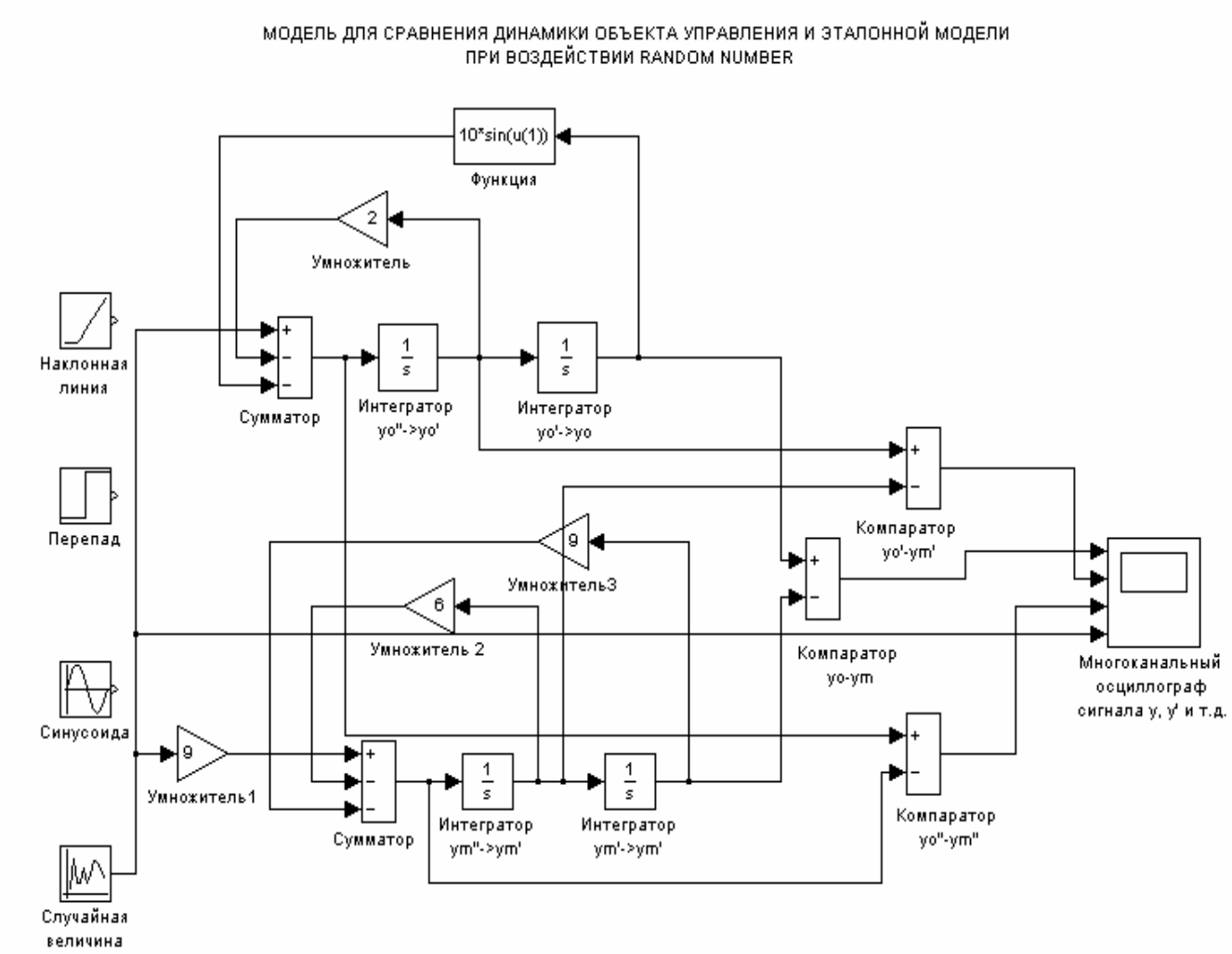

Рис. 10.11 Модель для сравнения динамики объекта управления и эталонной модели при воздействии Random Number

Рис. 10.12 Сравнение динамики объекта управления и эталонной модели при свободном движении

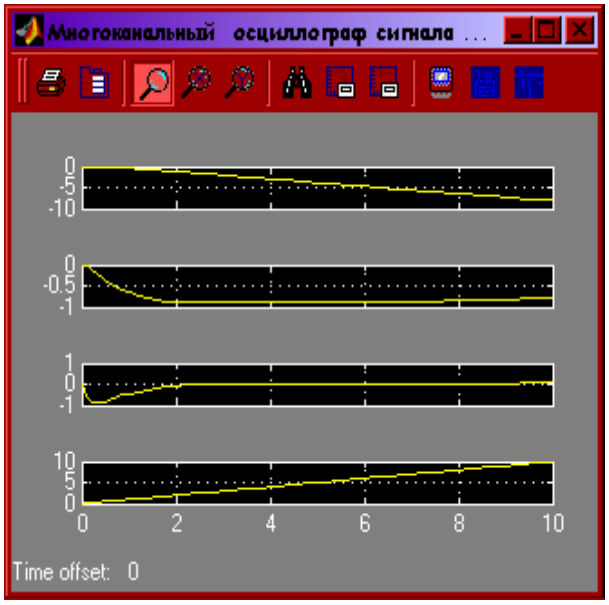

Рис. 10.13 Сравнение динамики объекта управления и эталонной модели при воздействии Ramp

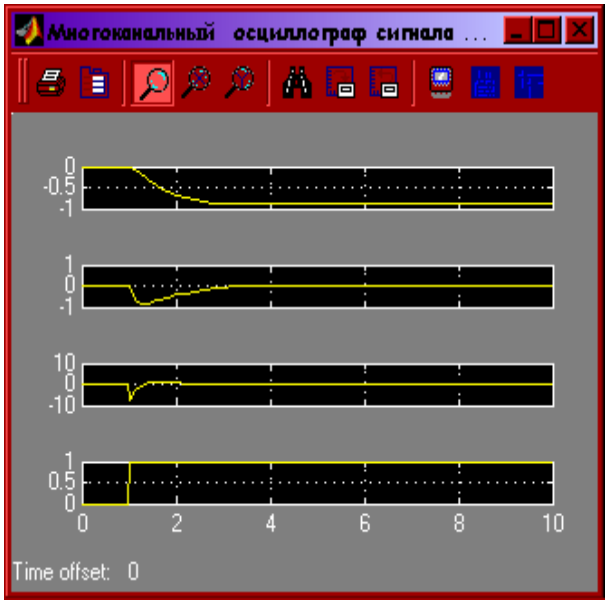

Рис. 10.14 Сравнение динамики объекта управления и эталонной модели при воздействии Step

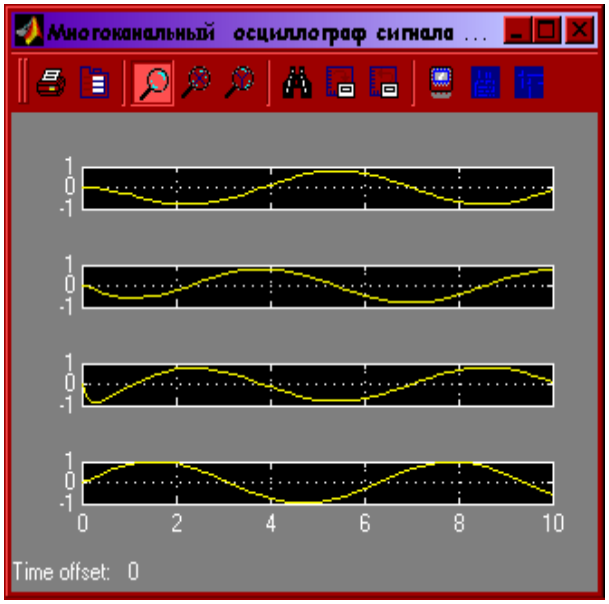

Рис. 10.15 Сравнение динамики объекта управления и эталонной модели при воздействии Sine Wave

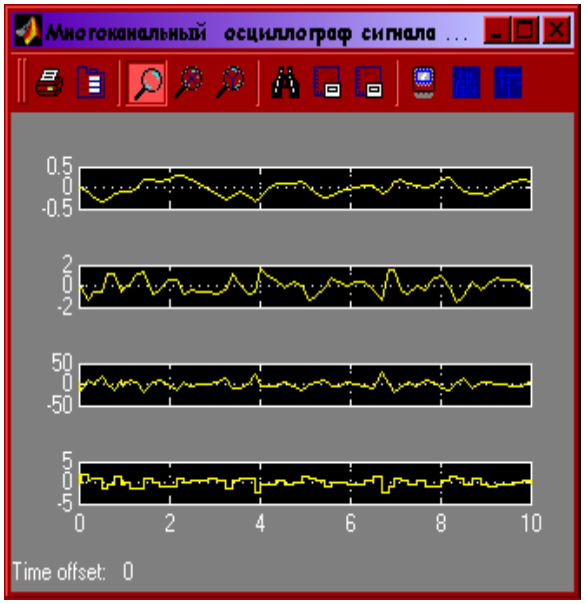

Рис. 10.16 Сравнение динамики объекта управления и эталонной модели при воздействии Random Number

10.3 Проектирование нейрорегулятора с использованием инструментальных средств системы MATLAB

Проектирование нейрорегуляторов для различных динамических систем значительно упрощается благодаря наличию в пакете NNT (Neural Networks Toolbox) специальных программных средств и системе имитационного моделирования Simulink.

Для проектирования нейронных регуляторов на основе эталонной модели запуск инструментальных средств производится исполнением команд mrefrobotarm (без нормализации данных) или mrefrobotarm2 (с нормализацией данных) и затем активизацией блока Model Reference Controller двойным щелчком левой кнопки мыши. В результате появляется диалоговая панель Model Reference Controller, которая позволяет определить архитектуру нейронной сети регулятора и произвести ее обучение. Однако, без идентификации объекта управления синтезировать нейрорегулятор невозможно. Поэтому необходимо открыть другую панель Plant Identification (идентификация механизма), щелкнув по кнопке Plant Identification.

Диалоговая панель для идентификации управляемого процесса Plant Identification входит в состав раздела Control Systems библиотеки нейронных блоков системы Simulink, является универсальным средством и может быть использована для построения нейросетевых моделей любых динамических объектов, которые могут быть представлены блоками этой системы.

С помощью управляющих элементов панели Plant Identification можно задать архитектуру нейронной сети, параметры обучающей последовательности и параметры обучения, а также управлять процессом идентификации и оценивать качество этого процесса.

Набор управляющих элементов для задания архитектурных параметров нейронной сети следующий:

6. Size of the Hiden Layer – количество нейронов во входном, или скрытом слое;

7. No. Delayed Plant Inputs – число задержек для входного слоя;

8. No. Delayed Plant Outputs – число задержек для выходного слоя;

9. Samling Interval – интервал квантования или шаг дискретности, в секундах, между двумя последовательными моментами отсчёта данных;

10. Notmalize Training Data – переключатель нормирования для преобразования обучающих данных к диапазону [0 1].

Набор управляющих элементов для задания характеристик обучающей последовательности таков:

10. Training Samples – число точек отсчёта для получения обучающей последовательности в виде пар значений вход-выход для управляемого процесса, определяемого моделью Simulink;

11. Maximum Plant Input – максимальное значение входного сигнала;

12. Minimum Plant Input – минимальное значение входного сигнала;

13. Maximum Interval Value (sec) – максимальный интервал идентификации (в секундах);

14. Minimum Interval Value (sec) – минимальный интервал идентификации (в секундах);

15. Limit Output Data – переключатель для ограничения значений выходного сигнала;

16. Maximum Plant Output – максимальное значение выходного сигнала, задаваемое при включённом переключателе Limit Output Data;

17. Minimum Plant Output – максимальное значение выходного сигнала, задаваемое при включённом переключателе Limit Output Data;

18. Simulink Plant Model – для задания модели управляемого процесса, реализованной с помощью блоков Simulink, имеющей порты входа и выхода и сохранённой в файле *.mdl; выбор модели производится с помощью кнопки Browse; имя модели отображается в специальном окне.

Параметры обучения задаются следующим образом:

6. Training Epochs – количество циклов обучения;

7. Training Function – для задания обучающей функции;

8. Use Current Weights – переключатель для использования текущих весов нейронной сети;

9. Use Validation Data – переключатель для использования контрольного множества в объёме 25**%** от обучающего множества;

10. Use Testing Data – переключатель для использования тестового множества в объёме 25**%** от обучающего множества.

Для идентификации управляемого процесса необходимо выполнить следующие действия:

12. Задать архитектуру нейронной сети, которая будет моделью управляемого процесса.

13. Задать параметры обучения.

14. Выбрать модель Simulink для управляемого процесса.

15. Сгенерировать обучающую последовательность заданного объёма, запустив модель Simulink с помощью кнопки Generate Training Data. Генерация обучающей последовательности производится с помощью воздействия ряда ступенчатых сигналов на модель управляемого процесса и снятия значений на входе и выходе модели через каждый шаг квантования. Графики входного и выходного сигнала отображаются в окне Plant Input-Output Data.

16. По завершении генерации обучающей последовательности необходимо либо принять эти данные, нажав на кнопку Accept Data, и тогда они будут использованы для обучения нейронной сети, либо отвергнуть их, нажав кнопку Reject Data, и повторить процесс идентификации управляемого процесса, представленного моделью Simulink.

17. После получения обучающей последовательности необходимо установить требуемые параметры обучения и с помощью кнопки Train Network запустить процесс обучения нейронной сети.

18. После завершения обучения его результаты отображаются на графиках изменения ошибки сети для обучающей, контрольной и тестирующей последовательностей, а также в окне выходных значений модели и сети при подаче на вход указанных последовательностей.

19. Если результаты обучения приемлемы, то надо сохранить параметры нейросетевой модели управляемого процесса и приступить к синтезу регулятора того или иного класса, нажав кнопки Apply и Ok.

20. Если результаты обучения неприемлемы, то следует нажать кнопку Cancel и повторить процесс идентификации сначала, изменяя архитектуру сети и параметры обучающей последовательности.

21. Обучающую последовательность можно импортировать из рабочей области или из файла, нажав на кнопку Import Data. Если необходимо обучающую последовательность сохранить в рабочей области или в файле для подбора параметров архитектуры нейронной сети, то следует после получения данных нажать на кнопку Export Data.

22. Если необходимо удалить только что сгенерированные данные, то следует нажать кнопку Erase Generated Data.

Таким образом, диалоговая панель Plant Identification позволяет идентифицировать управляемый процесс, представленный в виде имитационной модели Simulink, построить двухслойную нейронную сеть прямой передачи сигналов с необходимым числом нейронов и линий задержки, обучить эту сеть для получения нейронной модели управляемого процесса, оценить качество обучения и работу нейронной сети.

Архитектура нейронной модели регулятора аналогична архитектуре нейронной модели объекта, поэтому управляющие элементы на панели Model Reference Control такие же, что и на панели Plant Identification за небольшим исключением. Так, на панели имеется кнопка для инициирования процесса идентификации управляемого объекта Plant Identification, отсутствуют управляющие элементы для задания характеристик выходного сигнала, так как он непосредственно поступает на вход модели объекта, а обучающая последовательность разбивается на сегменты, для чего имеется специальное поле Controller Training Segments**.**

Для синтеза регулятора необходимо определить все требуемые параметры на панели, сгенерировать обучающие последовательности, нажав на кнопку Training Data и обучить нейронную сеть с помощью кнопки Train Controller, используя текущие веса и режим обучения с накоплением (если необходимо). Затем нажать на кнопку Apply для завершения процесса синтеза регулятора.

По окончании построения регулятора необходимо нажать на кнопку OK, вернуться в окно Simulink и выполнить моделирование работы системы нейронного регулирования для оценки характеристик регулятора.

На рис. 10.17 представлена модель для оценки управляющего сигнала и качества регулирования, на рис. 10.18 – сигнал внешнего воздействия и управляющий сигнал, а на рис. 10.19 – сравнение выходных сигналов эталонной модели и управляемого объекта.

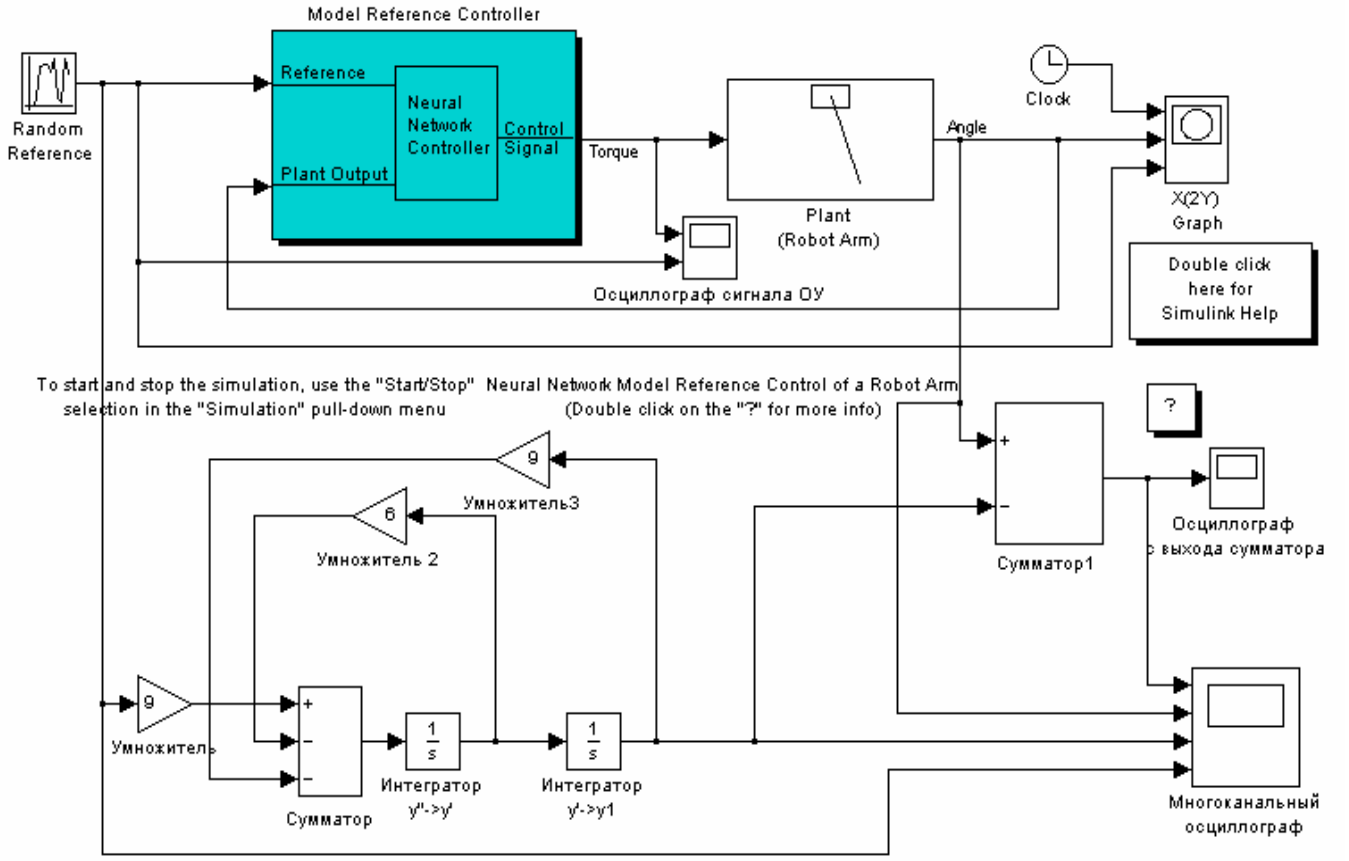

Рис. 10.17 Модель для оценки управляющего сигнала и качества регулирования

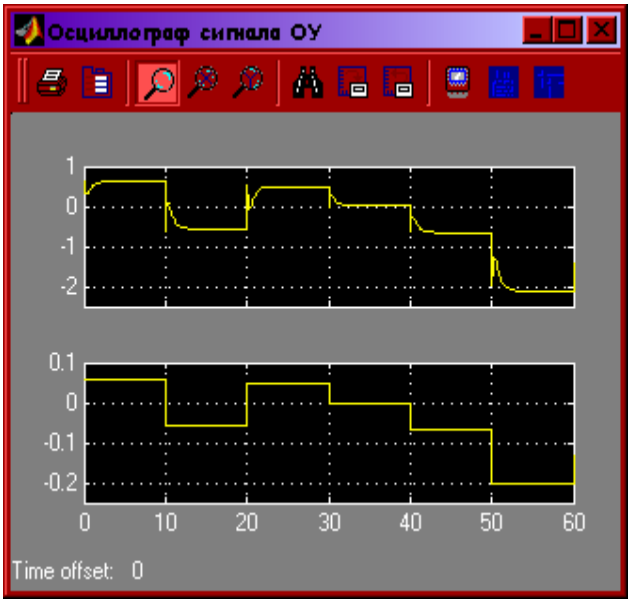

Рис. 10.18 Сигнал внешнего воздействия и управляющий сигнал

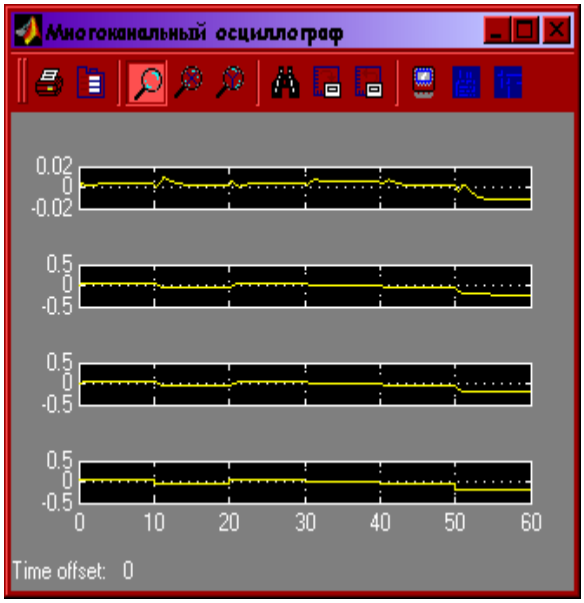

Рис. 10.19 Сравнение выходных сигналов эталонной модели и управляемого объекта

10.4 Программная реализация процедуры синтеза нейрорегулятора

Для программной реализации этапов проектирования нейрорегулятора в качестве нейронной модели, как для регулятора, так и для объекта будем использовать двухслойные нейронные сети с прямой передачей сигналов. Входной слой каждой сети имеет несколько нейронов, а функциями взвешивания, накопления и активации являются соответственно dotprod, netsum и logsig. Выходные слои для обоих случаев имеют такие же функции взвешивания и накопления, а функцией активации является для них линейная функция purelin. Сети с такой архитектурой могут воспроизводить весьма сложные нелинейные зависимости между входом и выходом.

Затем эти сети объединяются в одну сеть. Программно объединенная сеть создается одной функцией newff. Два первых слоя исключаются из циклов обучения с помощью признака learn, которому в этих слоях задается значение 0, и вся сеть, а точнее последние два слоя обучаются на наборах вход – выход, полученных на этапе идентификации объекта с помощью имитационной модели и блоков To Workspace. В результате обучения формируется модель объекта управления. Для формирования модели регулятора объединенную сеть вновь обучают, но уже на наборах вход – выход для эталонной модели и выключенными из процесса обучения двумя последними нейронными слоями. Так осуществляется синтез нейрорегулятора.

Изменяя число нейронов в слоях, количество линий задержки на входе, между слоями и в обратной связи, а также интервал квантования и объем обучающих последовательностей, добиваются требуемой точности моделей.

Если в нейронной сети используются линии задержки, то для ее обучения надо задавать функцию обучения, основанную на динамическом варианте метода обратного распространения ошибки trainbfgc (см. m-функции в файлах …\toolbox\nnet\nncontrol\trainbfgc.m и …\toolbox\nnet\nncontrol \srchbacxc.m).

По умолчанию для сетей с прямой передачей сигналов в качестве критерия обучения используется функционал, представляющий собой сумму квадратов ошибок между выходами сети и их целевыми значениями:

$$J = \frac{1}{2}\sum_{q=1}^{Q}\sum_{i=1}^{S}(t_i^q - d_i^q)^2, \qquad (10.3)$$

где Q – объём выборки;

q – номер выборки;

i – номер выхода;

t_i^q – целевое значение для i-го выхода выборки q;

d_i^q – сигнал на i-м выходе при подаче входных сигналов q-й выборки. Целью обучения сети является минимизация этого функционала с помощью изменения весов и смещений.

В настоящее время разработано несколько методов минимизации функционала ошибки на основе известных методов определения экстремумов функций нескольких переменных. Все эти методы можно разделить на три класса:

а) методы нулевого порядка, в которых для нахождения минимума используется только информация о значениях функционала в заданных точках;

б) методы первого порядка, в которых используется градиент функционала ошибки по настраиваемым параметрам, использующий частные производные функционала;

в) методы второго порядка, в которых используются вторые производные функционала.

Для линейных сетей задача нахождения минимума функционала (параболоида) сводится к решению системы линейных уравнений, включающих веса, смещения, входные обучающие значения и целевые выходы и, таким образом, может быть решена без использования итерационных методов. Во всех остальных случаях надо использовать методы первого или второго порядка.

Если используется градиент функционала ошибки, то

$$X_{k+1} = X_k - \alpha_k g_k, \qquad (10.4)$$

где X_k и X_{k+1} – векторы параметров на k-й и k+1-й итерациях;

α_k – параметр скорости обучения;

g_k – градиент функционала, соответствующий k-й итерации.

Если используется сопряжённый градиент функционала, то на первой итерации направление движения p_0 выбирают против градиента g_0 этой итерации:

$$p_0 = -g_0 . \qquad (10.5)$$

Для следующих итераций направление p_k выбирают как линейную комбинацию векторов g_k и p_{k-1}:

$$p_k = -g_k + \beta_k p_{k-1}, \qquad (10.6)$$

а вектор параметров рассчитывают по формуле:

$$X_{k+1} = X_k + \alpha_k p_k, \qquad (10.7)$$

Для методов второго порядка расчет параметров на *k*-м шаге производят по формуле (метод Ньютона):

$$X_{k+1} = X_k - H_k^{-1} g_k, \qquad (10.8)$$

где H_k – матрица вторых частных производных целевой функции (матрица Гессе);

g_k – вектор градиента на *k*-й итерации. Вычисление матрицы Гессе требует больших затрат машинного времени, поэтому её заменяют приближенными выражениями (квазиньютоновские алгоритмы).

Программа синтеза нейрорегулятора на основе эталонной модели универсальна и имеет следующий вид:

```
function = RefArm
%
%-- Программа построения нейронных моделей объекта управления и
%-- регулятора:
%
RefArmNet=newff([-0.5 0.5],[13 1 10 1],{'tansig' 'purelin' 'tansig' 'purelin'},'traingdx');
%
%-- Обучение нейронной модели объекта управления:
%
RefArmNet.inputWeights{1,1}.learn=0;
RefArmNet.layerWeights{2,1}.learn=0;
RefArmNet.layerWeights{3,2}.learn=1;
RefArmNet.layerWeights{4,3}.learn=1;
RefArmNet.trainParam.epochs=300;
RefArmNet.trainParam.goal=1e-5;
RefArmNet.trainParam.min_grad=1e-45;
```

```
RefArmNet.trainParam.mu_max=1e50;
RefArmNet=train(RefArmNet, Parm', Tarm');
Yarm=sim(RefArmNet,Parm');
hold off
plot(0:0.01:100,Tarm','r');
hold on
plot(0:0.01:100,Yarm,'b');
hold off
%
%-- Обучение нейронной модели регулятора:
%
RefArmNet.layerWeights{1,1}.learn=1;
RefArmNet.layerWeights{2,1}.learn=1;
RefArmNet.layerWeights{3,2}.learn=0;
RefArmNet.layerWeights{4,3}.learn=0;
RefArmNet.trainParam.epochs=300;
RefArmNet.trainParam.goal=1e-5;
RefArmNet.trainParam.min_grad=1e-45;
RefArmNet.trainParam.mu_max=1e50;
RefArmNet=train(RefArmNet, Preg', Treg');
Yreg=sim(RefArmNet,Preg');
hold off
plot(0:0.01:100,Treg','r');
hold on
plot(0:0.01:100,Yreg,'b');
```

10.5 Варианты заданий и порядок выполнения работы

Варианты объектов управления и эталонных моделей следует выбирать по табл. 4.2 лабораторной работы №4 или использовать звенья, которые исследовались в лабораторных работах №5, 6, 7 и 8. Для выбранного варианта необходимо построить имитационные модели и сохранить их в своей рабочей папке. Обе модели должны быть снабжены портами входа и выхода (см. блоки In1 и Out1 библиотеки Ports&Subsystems).

Далее необходимо исследовать динамику объекта управления и эталонной модели при изменении их параметров, зафиксировать несколько таких наборов, как для объекта, так и для модели и сгенерировать для них отчеты.

С помощью инструментальных средств системы MATLAB произвести идентификацию объекта robotarm и синтез нейрорегулятора для эталонной модели robotref, задавая различные алгоритмы обучения и варьируя объем обучающего множества. Затем выполнить эти действия, поменяв роли моделей и сделав robotref объектом управления, а robotarm – эталонной моделью. Для выбранных вариантов объектов и для ряда эталонных моделей синтезировать нейрорегуляторы и сохранить данные для отчета. Осциллограммы обучения и синтеза также вставить в отчет.

Используя программу и изучив реализацию квазиньютоновского алгоритма обучения trainbfgc, повторить анализ объекта и синтез нейрорегулятора для одного набора параметров. Синтезированную сеть регулятора преобразовать в исполняемый код и проверить ее работу в реальном масштабе времени, создав для этих целей программную модель объекта управления.

По промежуточным отчетам и результатам исследований оформить электронный отчет.

10.6 Оформление отчета по результатам исследований

Отчет должен содержать математическую постановку задач исследования, осциллограммы по динамике объектов управления и эталонных моделей, архитектуру искусственной нейронной сети и ее параметры после идентификации объекта управления и синтеза нейрорегулятора, анализ качества регулирования при различных внешних воздействиях и результаты испытаний для реального времени.

Лабораторная работа № 11

ИССЛЕДОВАНИЕ И ОПТИМИЗАЦИЯ ДИНАМИЧЕСКИХ ХАРАКТЕРИСТИК УПРАВЛЯЕМОГО ПОЛЕТА АЭРОКОСМИЧЕСКИХ АППАРАТОВ С ПОМОЩЬЮ МАТЕМАТИЧЕСКОГО МОДЕЛИРОВАНИЯ

Цель работы**:** изучение способов математического описания динамики аэрокосмических аппаратов для различных режимов управляемого полета и принципов автоматического управления этими аппаратами, исследование с помощью имитационного моделирования динамических характеристик простейших составляющих управляемого полета и овладение методами проектирования самонастраивающихся нейронных автопилотов на основе эталонных моделей.

11.1 Постановка задач исследования

Полет летательного аппарата характеризуется многими взаимосвязанными параметрами. Задание и поддержание режима полета возможно в том случае, когда его основные параметры известны и их можно изменять по желанию. Такими параметрами являются линейные и угловые координаты, скорости, ускорения и т.д. Задание режима полета сводится к заданию отдельных его параметров в таком сочетании, при котором обеспечиваются оптимальные условия полета.

Для поддержки режима полета неизменным или для изменения его по определенному закону в переменных внешних условиях при непрерывных возмущениях необходимо воздействовать на органы управления летательного аппарата. Это управление может быть как ручным, так и автоматическим. При ручном управлении ответные реакции летчика, как правило, оказываются недостаточно быстрыми и точными. Поэтому в настоящее время для всех режимов полета, за небольшим исключением, применяются системы автоматического управления полетом - автопилоты.

Для управления каждым параметром режима полета в отдельности необходимы свои независимые контуры, поскольку же число параметров велико, то система автоматического управления полетом получается многоконтурной. Отдельные контуры управления для частных видов движения (продольного,

бокового и т.п.) объединяются в каналы управления. Выходами каналов являются исполнительные механизмы (приводы), действующие на управляющие органы летательного аппарата.

Полет аэрокосмического летательного аппарата состоит из отдельных этапов: взлета, набора высоты и скорости, маршрутного полета, наведение на космические, воздушные и наземные цели, посадки и т.д. Система автоматического управления полетом должна быть комплексной и помимо функций управления параметрами режима полета должна одновременно обеспечить выполнение летательным аппаратом отдельных этапов его полета. При этом в различных условиях полета требуются различные управляющие воздействия на одни и те же возмущения. Другими словами, системы автоматического управления должны подстраиваться к условиям полета для получения оптимальных переходных процессов. Такие системы называются самонастраивающимися системами автоматического управления.

Полет летательного аппарата должен быть экономичным. Эта задача является экстремальной и для ее решения также требуются самонастраивающиеся системы. Одним из путей реализации самонастраивающихся систем является применение искусственных нейронных сетей.

11.2 Разработка аналитических моделей

11.2.1 Управляемый полет аэрокосмического аппарата

Движение аэрокосмического летательного аппарата как твердого тела складывается из двух движений: движение центра масс и движения вокруг центра масс. Так как в каждом из этих движений летательный аппарат обладает тремя степенями свободы, то в целом его движение характеризуется шестью степенями свободы: движения вдоль осей x, y, z и повороты вокруг тех же осей. Для определения движения летательного аппарата в любой момент времени t необходимо задать шесть координат x(t), y(t), z(t), φxx(t), φyy(t) и φzz(t) как функций времени. Учет деформации летательного аппарата усложняет

математическое описание динамики его полета и приводит к рассмотрению бесконечно большого числа степеней свободы.

Использование летательного аппарата возможно только в том случае, если его движение будет управляемым. Оно характеризуется определенными, заранее заданными координатами, скоростями, ускорениями и перегрузками и совершается для выполнения определенных тактических, оперативных или иных задач. Набор таких характеристик определяет режим полета. Чтобы обеспечить заданный режим полета, необходимо иметь средства и приборы для управления параметрами летательного аппарата: силой веса, силой тяги, силами трения и сопротивления, аэродинамическими моментами и т.д.

Движение летательного аппарата является единым процессом. Однако это сложное движение разбивают на простейшие виды: продольное движение, боковое движение, движение центра масс, угловые движения и т.д. Во многих случаях достаточно ограничиться только продольным и боковым движениями. Это, например, справедливо для самолетов.

11.2.2 Продольное движение самолета

Плоское движение самолета, при котором вектор скорости центра масс совпадает с плоскостью симметрии, называется продольным движением. Возможность осуществления продольного движения обусловлена симметрией самолета. Оно описывается дифференциальными уравнениями, устанавливающими связь между координатами, скоростями, ускорениями и действующими на самолет силами и моментами.

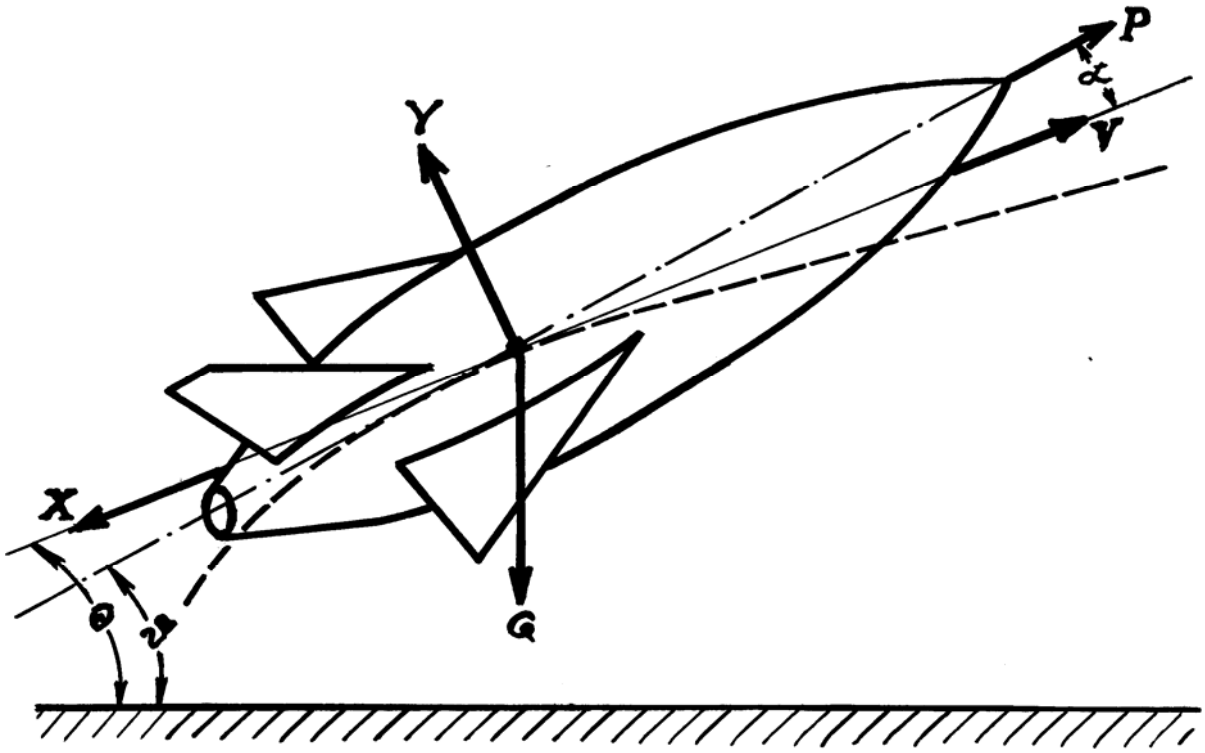

Рис. 11.1 Параметры продольного движения самолета:

V - скорость полета, направленная по касательной к траектории (пунктирная линия);

Y - подъемная сила, перпендикулярная касательной;

X - сила сопротивления, противоположная **V**;

G - сила веса;

υ - угол тангажа, т.е. угол между продольной осью самолета и горизонтальной плоскостью;

θ - угол наклона траектории;

α - угол атаки, т.е. угол между продольной осью самолета и проекцией вектора скорости на плоскость симметрии;

m = G/g - масса самолета;

P - сила тяги, направленная вдоль продольной оси самолета;

Mz - суммарный момент аэродинамических сил относительно поперечной оси z;

Jz - момент инерции самолета относительно z.

Используя обозначения величин, введенные на рис.11.1, спроектируем силы, действующие на самолет, на два направления: на касательную к траектории полета и на нормаль к ней.

Сумма проекций сил на касательную к траектории будет

$$m\frac{dV}{dt} = P\cos\alpha - X - G\sin\theta . \qquad (11.1)$$

При определении проекций сил на нормаль к траектории следует иметь в виду, что при искривленной в вертикальной плоскости траектории на самолет действует центробежная сила инерции mv^2/r, где r- радиус кривизны траектории. Так как $r = ds/d\theta$; где s - длина дуги траектории, то $ds = Vdt$, и тогда

$$\frac{mV^2}{r} = \frac{mV^2}{\frac{ds}{d\theta}} = m\frac{V^2}{V\frac{dt}{d\theta}} = mV\frac{d\theta}{dt} . \qquad (11.2)$$

Следовательно, сумма проекций сил на нормаль к траектории запишется следующим образом:

$$mV\frac{d\theta}{dt} = P\sin\alpha + Y - G\cos\theta . \qquad (11.3)$$

Для аэродинамических сил, действующих относительно поперечной оси z, проходящей через центр масс самолета, уравнение их моментов будет иметь вид:

$$J_z\frac{d^2\vartheta}{dt^2} = M_z \qquad (11.4)$$

Величины θ, ν и α связаны очевидным соотношением

$$\nu = \theta + \alpha \qquad (11.5)$$

Полученные уравнения (11.1,11.3,11.4 и 11.5) определяют динамику продольного полета самолета относительно центра масс. Входящие в эти уравнения силы P,X и Y, а также моменты Mz являются функциями параметров режима полета и внешних условий: температуры и плотности воздуха, атмосферного давления и т.д. Входящие в уравнения (11.1),(11.2) и (11.3) силы P, X и У и момент M_z являются функциями параметров режима полета.

Сила тяги P зависит от параметров двигателя и от внешних условий, характеризуемых скоростью полета V, давлением $p_н$ и температурой $T_н$ окружающей среды. Так как в двигателях имеется ручка объединенного

управления, положение которой характеризуется координатой $\delta_р$, то зависимость тяги от параметров можно представить в виде

$$P = P(\delta_р, V, p_н, T_н). \qquad (11.6)$$

Примерная зависимость тяги турбореактивного двигателя P от числа M полета приведена на рис. 11.2. Зависимость тяги от положения ручки управления $\delta_р$ может быть различной, однако нередко бывает необходимо, чтобы приращение тяги было пропорционально приращению координаты $\delta_р$.

Аэродинамические силы X и У зависят от угла атаки α, скорости полета V, плотности воздуха ρ и угла отклонения руля высоты $\delta_в$. Однако ввиду того, что угол отклонения $\delta_в$ оказывает слабое влияние на величины сил X и У, то этим влиянием можно пренебречь.

Воспользовавшись принятым в аэродинамике представлением, что

$$X = c_x S \frac{\rho V^2}{2}; \qquad (11.7)$$

$$Y = c_y S \frac{\rho V^2}{2} \qquad (11.8)$$

где c_x и c_y — коэффициенты лобового сопротивления и подъемной силы, а S — площадь крыльев, можно сказать, что зависимость сил X и Y от параметров α , V и ρ определяется зависимостью коэффициентов c_x и c_y от этих параметров. На рис. 11.3 приведены типичные графики зависимости этих коэффициентов.

Момент аэродинамических сил M_z можно представить в виде

$$M_z = m_z b_a S \frac{\rho V^2}{2}, \qquad (11.9)$$

где m_z — коэффициент момента, b_a —длина хорды крыла.

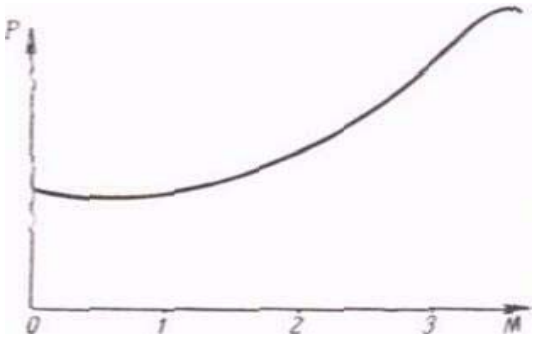

Рис. 11.2 Зависимость тяги турбореактивного двигателя от числа M полёта.

Коэффициент m_z можно представлять состоящим из суммы двух слагаемых, одно из которых зависит от статических параметров α , V, $\delta_в$ и определяет статический момент, а другое от динамических параметров $\dot{\upsilon}, \dot{\alpha}$ и определяет

демпфирующий момент. На рис. 11.4 приведены графики зависимости коэффициента m_z от углов α и $\delta_в$.

Возникновение демпфирующего момента обусловлено угловой скоростью вращения самолета вокруг поперечной оси. При вращении самолета происходит изменение подъемной силы горизонтального оперения, вследствие чего изменяется коэффициент момента m_z. Приращение коэффициента момента m_z, пропорциональное приращению угла атаки вследствие вращения $\alpha_{вр}$, называется коэффициентом момента демпфирования.

$$\Delta m_{z\text{дем}} = k\Delta\alpha_{вр}, \qquad (11.10)$$

где $\Delta\alpha_{вр} = w_z L_1/V = L_1\dot{u}/V$, L_1 — расстояние от оперения до центра масс. Так как с увеличением скорости полета величина $\Delta\alpha_{вр}$ уменьшается, то демпфирование самолетов на больших скоростях меньше, чем на малых; при увеличении высоты полета демпфирование также уменьшается. Если учесть запаздывание скоса потока у горизонтального оперения, то получим уточнённое значение величины

$$\Delta\alpha_{вр} = \frac{L_1}{V}\dot{u} + k'\dot{\alpha} \qquad (11.11).$$

Следовательно, коэффициент момента m_z является следующей функцией:

$$m_z = m_z(\alpha,\ \dot{\alpha}, \dot{u},\ V,\ \delta_в,\ \rho). \qquad (11.12)$$

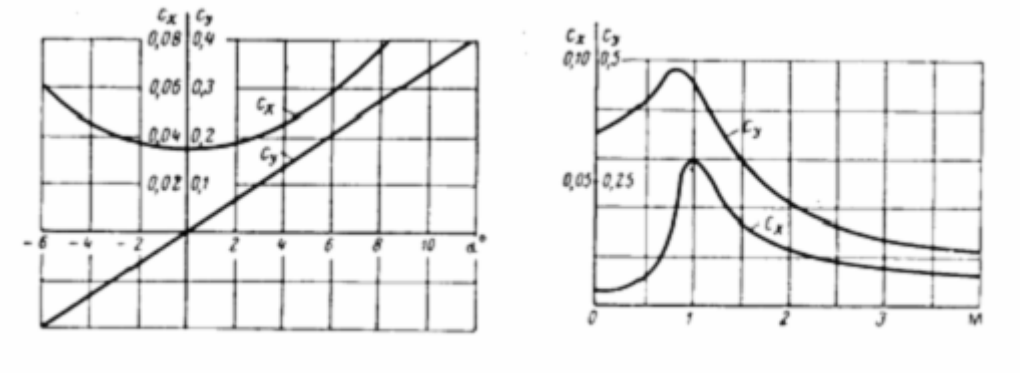

Рис. 11.3 Зависимость коэффициентов c_x и c_y от угла атаки α и числа M полёта.

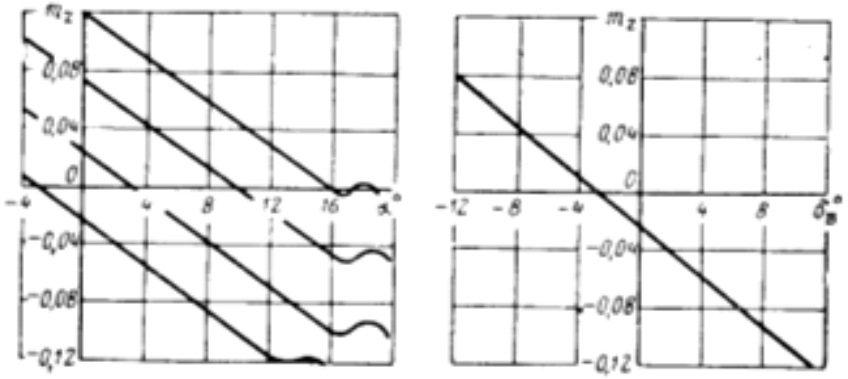

Рис. 11.4 Зависимость коэффициента продольного момента m_z самолёта от угла атаки α и угла отклонения руля высоты $\delta_в$.

К уравнениям (11.1 – 11.5) надо добавить уравнения движения центра масс самолета:

$$\frac{dH}{dt} = V\sin\theta + U_y,$$

$$\frac{dL}{dt} = V\cos\theta + U_x,$$

(11.13)

(11.14)

где Ux и Uy - скорости ветра по соответствующим направлениям. Таким образом, продольное движение самолета описывается уравнениями (11.1,11.3,11.4,11.5,11.13 и 11.14).

11.2.3 Линеаризация уравнений продольного движения самолета

При продольном движении самолета в качестве регулируемых величин можно выбрать углы тангажа υ, атаки α, наклона траектории θ, скорость полета V, вертикальную скорость H', а также высоту полета H и дальность L. В качестве регулирующих факторов используются руль высоты, стабилизатор, тяга двигателя, воздушные тормоза, закрылки и др.

Так как уравнения (11.1,11.2,11.3,11.4,11.5,11.13 и 11.14) нелинейны, то использование их для исследования процессов в системе автоматического управления полетом крайне затруднительно. Обычно эти уравнения линеаризуют в предположении, что параметры υо, θо, V_o, αо и H_o, соответствующие установившемуся режиму, получают малые приращения Δυ, Δθ, ΔV, Δα и ΔH, вызванные действующими на самолет возмущениями. Такое рассмотрение позволяет оценить поведение самолета в установившемся (невозмущенном) движении по его поведению при наличии возмущений.

Разлагая силы P, X, Y и момент M_z в ряды по малым приращениям и ограничиваясь линейными членами приращений, получим

$$\left.\begin{aligned}
\Delta P &= P^V \Delta V + P^{\delta_p}\delta_p + P^\rho \Delta\rho \\
\Delta X &= X^V \Delta V + X^\alpha \Delta\alpha + X^\rho \Delta\rho \\
\Delta Y &= Y^V \Delta V + Y^\alpha \Delta\alpha + Y^\rho \Delta\rho \\
\Delta M_z &= M_z^V \Delta V + M_z^\alpha \Delta\alpha + M_z^{\dot\alpha} \Delta\dot\alpha + M_z^{\dot\upsilon} \Delta\dot\upsilon + M_z^{\delta_в}\delta_в + M_z^\rho \Delta\rho
\end{aligned}\right\} \quad (11.15)$$

где верхние индексы у величин P, X, Y и M_z обозначают частные производные по соответствующим переменным.

Методика дальнейших преобразований данных уравнений рассмотрена в [17].

Линеаризованные дифференциальные уравнения продольного движения самолета как управляемого процесса устанавливают связь между регулируемыми параметрами ν,υ, θ, α ,h и регулирующими факторами $\delta_в$, $\delta_р$ и характеризующие динамические свойства самолётов в их продольном движении, принимают следующий вид:

$$\left.\begin{aligned}
(p+n_{11})\nu + n_{12}\,\alpha + n_{13}\upsilon + n_{14}h &= n_p\,\delta_p + f_1 ;\\
-n_{21}\nu + (p+n_{22})\,\alpha - (p+n_{23})\upsilon + n_{24}h &= f_2 ;\\
n_{31}\nu + (n_0 p+n_{32})\alpha + (p^2+n_{33}p)\upsilon + n_{34}h &= -n_в\delta_в + f_3;\\
-n_{41}\nu + n_{42}\,\alpha - n_{42}\upsilon + ph &= \nu_y ,
\end{aligned}\right\} \qquad (11.16)$$

где f_1 ,f_2 ,f_3 - возмущения, действующие на самолёт, а p – символ дифференцирования.

Эти возмущения складываются из вертикальных и горизонтальных порывов ветра, характеризуемых составляющими скоростей U_x и U_y; изменения веса самолёта ΔG; импульсных сил и моментов, вызванных стрельбой реактивными снарядами, стрельбой из пушек и др.:

$$\left.\begin{aligned}
f_1 &= p\,v_x + f_1' ;\\
f_2 &= p\,v_y + \frac{\Delta G}{S\rho V^2} + f_2' ;\\
f_3 &= \frac{\Delta G}{S\rho V^2}\frac{l_1}{b_a} + f_3' ,
\end{aligned}\right\} \qquad (11.17)$$

где $v_x = \dfrac{\Delta U_x}{V}$; $v_y = \dfrac{\Delta U_y}{V}$;

l_1 - расстояние от местоположения сброшенного груза до центра масс самолёта (предполагается , что после сброса груза появляется момент только вокруг оси z);

f'_1, f'_2, f'_3 - возмущения, вызванные, например, стрельбой. Они будут определяться направлением стрельбы и местоположением оружия на самолёте.

Если стрельба производится вперёд по оси самолёта, то $f_1' = \dfrac{P_1}{\rho S V^2}$,

где P_1 - импульсная сила большой интенсивности и малой продолжительности, импульс которой $P_1 dt$ - конечная величина.

Отличие уравнений (11.16) от исходных уравнений движения самолета состоит в том, что они являются линейными дифференциальными уравнениями с постоянными коэффициентами. Строго говоря, постоянство коэффициентов имеет место только для данного режима полета. При переходе с одного режима полета на другой (например, при изменении высоты полета) характеристики самолета (подъемная сила, сила сопротивления, аэродинамические моменты и

т. д.) будут изменяться, что приведет к изменению коэффициентов уравнений (11.16). Если, однако, время изменения режимов полета значительно больше времени протекания процессов в системах управления, что имеет место в действительности, а характеристики самолета при переходе с одного режима на другой изменяются незначительно, то коэффициенты уравнений можно принять постоянными. При значительном изменении характеристик самолета на различных режимах полета следует вычислять коэффициенты линеаризованных уравнений для каждого из режимов.

Линеаризованные уравнения (11.16) справедливы до тех пор, пока регулируемые величины ν, υ, α и h малы и не превышают 0,1 (углы α и υ измеряются в радианах). При больших отклонениях от установившихся значений вместо линеаризованных уравнений необходимо пользоваться исходными нелинейными уравнениями. Поскольку задача системы управления сводится к поддержанию величин ν, α, υ и h близкими к нулю, то уравнения (11.16) почти всегда оказываются справедливыми.

Если при исследовании динамики систем управления самолетов используются исходные нелинейные уравнения движения, то задача получается настолько сложной, что ее решение возможно только на вычислительных машинах.

В табл.11.1 приведены ориентировочные значения коэффициентов уравнений продольного движения легкого, среднего и тяжелого самолета.

Из этой таблицы видно, что коэффициенты уравнений изменяются по режимам полета. Очевидно, для получения одинаковых переходных процессов в

системе автоматического управления самолета при различных режимах полета необходимо изменять передаточные числа автопилота.

Таблица 11.1

Коэффициенты уравнений продольного движения

	Самолёт					
	лёгкий	средний		тяжёлый		
	H = 11 км M = 0,9 τ_a = 3,8 с	H = 0, посадка	H = 4 км M = 0,65 τ_a = 2,1 с	H = 8 км M = 0,8 τ_a = 2,5 с	H = 12 км M = 0,9 τ_a = 4 с	H = 0, посадка
N_{11}	0,024	0,12	0,019	0,026	0,048	0,12
N_{12}	-0,11	-0,28	0,019	-0,025	-0,079	-0,12
N_{13}	0,2	0,4	0,3	0,1	0,17	0,3
N_{14}	$-4{,}3\cdot10^{-4}$	—	$-4{,}4\cdot10^{-4}$	$-4\cdot10^{-4}$	$-4{,}2\cdot10^{-4}$	—
N_{21}	-0,4	-0,8	-0,6	-0,36	-0,68	-0,65
N_{22}	2,4	2,4	2,66	3	2,4	2,35
N_{23}	0	0,02	0	0	0	0,015
N_{24}	$-1{,}22\cdot10^{-2}$	—	$-1{,}28\cdot10^{-2}$	$-1{,}1\cdot10^{-2}$	$-1{,}2\cdot10^{-2}$	—
N_{31}	0	0	0	0	-1,2	0
n_0	0,4	0,59	0,59	1,17	0,68	0,9
N_{32}	38	6,6	10,63	42	36	8
N_{33}	2,45	1,67	1,69	2,5	2,42	2,35
N_{34}	-0,053	—	-0,055	-0,05	-0,05	—
$n_в$	49	15,2	24,5	28	46	8,4
n_p	0,022	0,019	0,021	0,02	0,02	0,019

Рассмотрим некоторые частные случаи уравнений (11.16).

Если пренебречь влиянием изменения плотности атмосферы на характеристики самолета, то вместо системы (11.16) получим

$$\left.\begin{aligned} (p+n_{11})\nu + n_{12}\alpha + n_{13}\upsilon &= n_p\delta_p + f_1 ; \\ -n_{21}u + (p+n_{22})\alpha - (p+n_{23})\upsilon &= f_2 ; \\ n_{31}\nu + (n_0p+n_{32})\alpha + (p^2+n_{33}p)\upsilon &= -n_в\delta_в + f_3; \\ \alpha + -\upsilon + ph &= \upsilon_y , \end{aligned}\right\} \qquad (11.18)$$

Самолет по отношению к вектору скорости полета обладает значительно большей инерцией, чем по отношению к угловым координатам υ и α. Поэтому в некоторых случаях в уравнениях (11.18) можно приближенно положить v = 0, полагая, что за время изменения величин υ и α скорость полета практически не изменится. Тогда вместо системы (11.18) можно рассматривать систему, справедливую для горизонтального полета ($\theta_0 = 0$) и характеризующую угловые движения самолета:

(11.19)

$$\left.\begin{aligned} (\upsilon - \alpha) &= n_{22}\alpha ; \\ (p^2 + n_{33}p)\upsilon + (n_0 p + n_{32})\alpha &= -n_в\delta_в \end{aligned}\right\}$$

В этих уравнениях члены, характеризующие внешние возмущения, опущены.

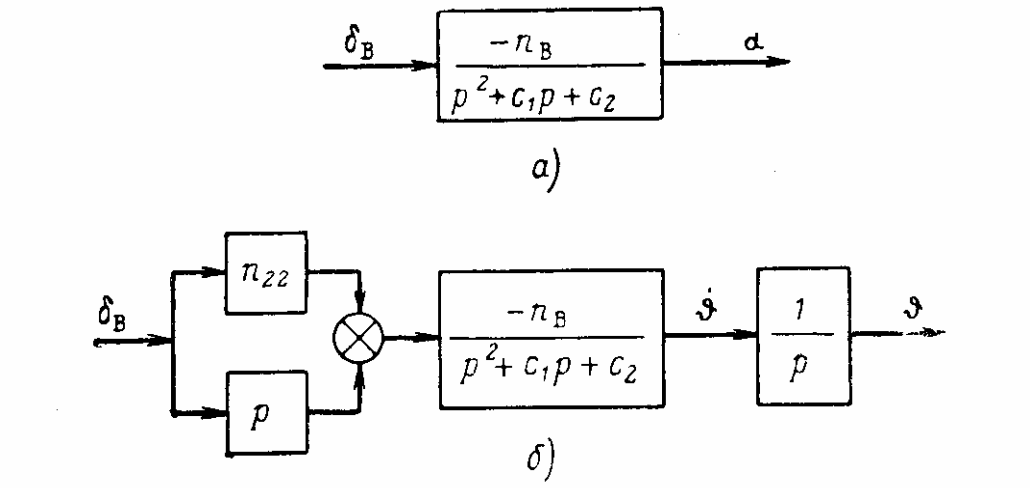

Рис.11.5 – Структурные схемы самолета.

а – структурная схема самолета по углу атаки; б – структурная схема самолета по углу тангажа.

Решая уравнения (11.19) относительно величин υ и α, и разделив эти величины на $б_в$, получим

$$\frac{\upsilon}{\delta_в} = \frac{-n_в(p + n_{22})}{p(p^2 + c_1 p + c_2)},$$

$$\frac{\alpha}{\delta_в} = \frac{-n_в}{p^2 + c_1 p + c_2)}, \qquad (11.20)$$

где $c_1 = n_0 + n_{22} + n_{33}$; $c_2 = n_{32} + n_{22}n_{33}$.

Выражения (11.20) называются передаточными функциями самолета. Они характеризуют реакцию самолета на единичные возмущения, вносимые рулем высоты.

Как следует из выражений (11.20) и рис.11.5, на котором приведены структурные схемы, эквивалентные выражениям (11.20), самолет по отношению к углу атаки (а также к нормальным перегрузкам) при возмущении рулем высоты является колебательным звеном, а по отношению к углу тангажа — сложным звеном, представляющим собой последовательное соединение колебательного, интегрирующего и опережающего звеньев. Последнее звено состоит из параллельно соединенных усилительного и дифференцирующего звеньев.

Если в выражениях (11.20) положить p = jω , где ω — безразмерная круговая частота, то получим частотные характеристики самолета. Например, амплитудно-частотные и фазо-частотные характеристики самолета будут иметь вид

$$\left.\begin{aligned} \left|\frac{\upsilon}{\delta_в}\right| &= \frac{\omega_в\sqrt{w^2+n_{22}^2}}{w\sqrt{c_1^2w^2+\left(c_2-w^2\right)^2}}; \\ \varphi_\upsilon &= arctg\frac{c_1w^2+\omega_{22}(c_2-w^2)}{w(c_1\omega_{22}-(c_2-w^2))}\varphi \end{aligned}\right\} \qquad (11.21)$$

$$\left.\begin{aligned} \left|\frac{\alpha}{\delta_в}\right| &= \frac{\omega_в}{\sqrt{c_1^2w^2+\left(c\quad w\right)^2}}; \\ \varphi_\alpha &= arctg\frac{c_1w}{w^2-c_2}. \end{aligned}\right\} \qquad (11.22)$$

На рис.11.6 а) приведены экспериментальные кривые амплитудно-частотных характеристик самолета с поршневыми двигателями.

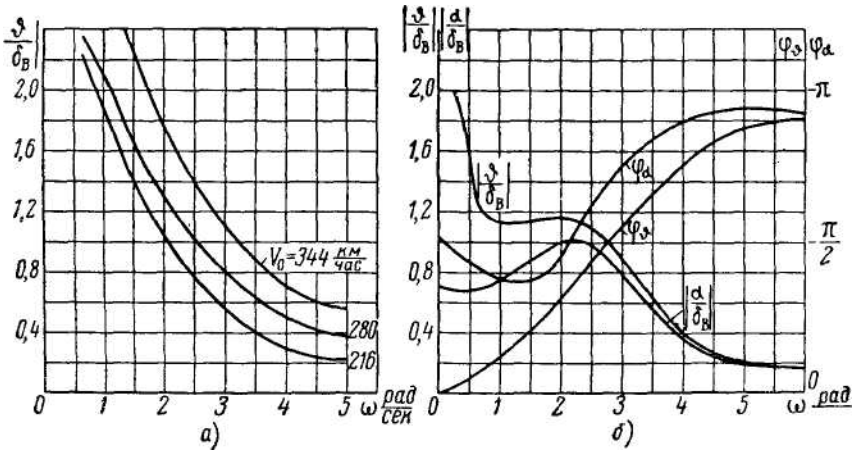

Рис.11.6 Амплитудно-частотные и фазо-частотные характеристики самолета.

Из кривых видно, что с увеличением скорости полета реакция самолета на возмущения рулем высоты возрастает. Область существенных частот самолета не превышает 1 —1,5 Гц.

На рис.11.6 б) приведены амплитудно-частотная и фазо-частотная характеристики реактивного самолета, подсчитанные по формулам (11.21). В амплитудно-частотных характеристиках отчетливо выступают резонансные пики, что свидетельствует о малом естественном демпфировании (малом демпфирующем моменте) реактивных самолетов.

Вместо обычных амплитудно-частотных и фазо-частотных характеристик можно пользоваться логарифмическими характеристиками.

Если положить, что угловые движения υ и α стабилизированы быстродействующим автопилотом и, следовательно, в среднем за время изменения скорости полета V можно считать υ = 0 и α = 0, то вместо системы (11.18) получим

$$(p+n_{11})\nu = n_p\delta_p + f_1 \qquad (11.23)$$

Уравнением (11.23) можно пользоваться при исследовании динамики автоматического регулирования скорости полета. Если стабилизация угловых движений самолета обеспечена автопилотом, то регулирование скорости полета можно осуществить путем изменения тяги.

Если в уравнениях (11.16) пренебречь демпфирующим и инерционным моментами, то получим уравнения движения центра масс самолета:

$$\left.\begin{aligned} (p+n_{11})\nu + (n_{12} + n_{13})\alpha + n_{13}\theta + n_{14}h = n_p\,\delta_p + f_1\ ; \\ -n_{21}\nu + (n_{22} - n_{23})\,\alpha - (p+n_{23})\theta + n_{24}h = f_2\ ; \\ n_{31}\nu + n_{32}\alpha + n_{34}h = -n_в\delta_в + f_3; \\ -n_{41}\nu - n_{42}\theta + ph = v_y\ . \end{aligned}\right\} \qquad (11.24)$$

Из рассмотрения определителя системы (11.24)

$$\Delta = \begin{vmatrix} p+n_{11} & n_{12} + n_{13} & n_{13} & n_{14} \\ -n_{21} & n_{22} - n_{23} & -p-n_{23} & n_{24} \\ n_{31} & n_{32} & 0 & n_{34} \\ -n_{41} & 0 & -n_{42} & p \end{vmatrix} \qquad (11.25)$$

следует, что если не учитывать влияние плотности ($n_{14} = n_{24} = n_{34} = 0$), то самолет по отношению к высоте полета является нейтральным; в противном случае самолет становится статически устойчивым.

Решение уравнений (11.24) относительно величин ν, θ и h, полагая $n_{14} = n_{24} = n_{34} = 0$ и $n_{23} = 0$, представленное в [17], позволяет получить передаточные функции, из структуры которых следует, что в горизонтальном полете изменение тяги на постоянную величину непосредственно не вызывает изменения скорости полета, а приводит только к изменению наклона траектории. Другими словами, при изменении тяги полет из горизонтального становится негоризонтальным. Для изменения скорости полета необходимо одновременно воздействовать на ручку управления двигателем и на руль высоты.

В общем случае каждая из величин ν , υ, α и h зависит от регулирующих факторов $б_в$ и $б_р$. Решение уравнений (11.18) без учёта внешних возмущений

$$\left.\begin{aligned} \nu &= П_{1\nu}(p)\, б_р + П_{2\nu}(p)\, б_в ; \\ \alpha &= П_{1\alpha}(p)\, б_р + П_{2\alpha}(p)\, б_в ; \\ \upsilon &= П_{1\upsilon}(p)\, б_р + П_{2\upsilon}(p)\, б_в ; \\ h &= П_{1h}(p)\, б_р + П_{2h}(p)\, б_в . \end{aligned}\right\} \qquad (11.26)$$

позволяет рассматривать самолет в продольном движении как линейную динамическую систему с входными координатами $б_в$ и $6_Р$ и выходными координатами ν , υ, α и h.

При этом динамические свойства самолета оцениваются передаточной матрицей вида

$$\begin{matrix} \mathbf{П_{1\nu}} & & \\ \mathbf{П_{2\nu}} & & \\ & \mathbf{П_{1\alpha}} & \\ & \mathbf{П_{2\alpha}} & \\ & & \mathbf{П_{1\upsilon}} \\ & & \mathbf{П_{2\upsilon}} \\ & & \mathbf{П_{1h}} \\ & & \mathbf{П_{2h}} \end{matrix} \qquad (11.27)$$

составленной из передаточных функций самолёта (их выражения имеются в [17]).

11.2.4 Устойчивость уравнений продольного движения самолета

Движение самолета как сложной динамической системы в зависимости от режима полета, параметров и характеристик самолета может быть устойчивым или неустойчивым.

Рассмотрим продольное движение самолета с установившейся скоростью полета. Предположим, что в некоторый момент времени изменился угол наклона траектории полета (рис.11.7). Вектор скорости вследствие инерции самолета в первый момент времени будет иметь прежнее значение. При этом возникнет несимметричный обдув, вследствие чего на части самолета, расположенные впереди центра масс и позади него, будут действовать силы, создающие отличный от нуля момент. Если обозначить через M_{z1} момент аэродинамических сил, действующих на части самолета, расположенные впереди от центра масс, а через M_{z2} — момент аэродинамических сил, действующих на части самолета, расположенные позади центра масс, то общий момент будет

$$M_z = M_{z2} - M_{z1} \qquad .(11.28)$$

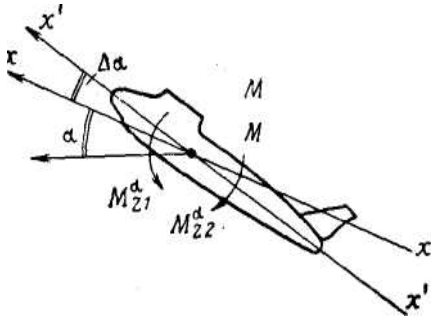

Рис.11.7 Устойчивость продольного движения

Легко видеть, что момент M_{z1} стремится еще более отклонить самолет от первоначального установившегося режима полета, а момент M_{z2} — возвратить самолет в исходное положение. В зависимости от величин моментов M_{zl} и M_{z2} самолеты разделяются на статически устойчивые ($M_{z2}>M_{z1}$), статически неустойчивые ($M_{z2}<M_{z1}$) и нейтральные ($M_{z2} = M_{z1}$),

(здесь M – число полетов). Все современные самолеты, летающие со скоростями до M = 1, являются статически устойчивыми. При сверхзвуковых скоростях полета запас устойчивости самолета снижается, а при M>1,5 самолет может стать неустойчивым.

Для суждения об устойчивости продольного движения самолета рассмотрим характеристическое уравнение системы (11.18)

$$p^4+c_1p^3+c_2p^2+c_3p+c_4 = 0. \qquad (11.29)$$

Устойчивость продольного движения самолета по отношению к координатам v, υ, α определяется видом корней характеристического уравнения. Для устойчивости движения необходимо, чтобы вещественные части всех корней характеристического уравнения (11.29) были отрицательны. Для того чтобы уравнение (11.29) имело корни с отрицательными вещественными частями, необходимо и достаточно, чтобы были выполнены, например, условия Раусса— Гурвица:

$$\left.\begin{array}{l} c_1>0,\quad c_2>0,\quad c_3>0,\quad c_4>0; \\ c_1\,(c_2c_3 - c_1c_4) - c_3^2 > 0. \end{array}\right\} \qquad (11.30)$$

Если нарушится последнее условие (11.30), то в характеристическом уравнении появится пара комплексных сопряженных корней с вещественными частями, вследствие чего движение самолета, соответствующее этим корням, будет колебательным расходящимся.

При нарушении условия $c_4>0$ среди корней появится один вещественный положительный корень, поэтому движение самолета, соответствующее этому корню, будет апериодически расходящимся.

Рассмотрим более детально случай нарушения условия $c_4>0$, для чего обратимся к структуре коэффициента c_4. При горизонтальном полете, когда $c'_x = 0$, получим

$$c_4 = -\frac{1}{2}\mu m_z^{\alpha} c_y\left(c_y+\frac{1}{2}M c_y^{\alpha}\right)+\frac{1}{2}\mu m_z^{v} c_y c_y^{\alpha} \qquad (11.31)$$

При полете на скоростях $M \leq 0{,}8$ момент M_z практически не зависит от скорости полета, поэтому $m^v_z = 0$. Так как c_y и c^α_y положительны, то знак коэффициента c_4 определяется знаком коэффициента m^α_z . Коэффициент m^α_z характеризует статическую устойчивость самолёта и называется коэффициентом статической устойчивости. Если коэффициент $m^\alpha_z<0$, то самолет статически устойчив. Если же $m^\alpha_z = 0$ или $m^\alpha_z>0$, то самолет будет соответственно нейтральным или неустойчивым.

В качестве примера приведем характеристическое уравнение с численными коэффициентами для реактивного самолета:

$$p^4 + 2{,}8p^3+4{,}45p^2+0{,}049p+ 0{,}057 = 0. \qquad (11.32)$$

Применяя критерий Раусса-Гурвица, легко убедиться, что продольное движение рассматриваемого самолета устойчиво. Корни уравнения

$$p_{1,2}= -l{,}4 \pm jl{,}57 ; \quad p_{3,4}= -0{,}0061 \pm j0{,}096 \qquad .(11.33)$$

При других режимах полета малые корни p_3 и p_4 могут стать вещественными, причем один из них может быть положительным. Полученное распределение корней является характерным для продольного движения различных самолетов.

Продольное движение самолета, как правило, состоит из двух движений, одно из которых — короткопериодическое — соответствует большим корням характеристического уравнения, а другое— длиннопериодическое (фугоидное) — малым корням. Обычно для различных самолетов периоды колебаний этих движений изменяются в широких пределах. Так, например, период колебаний короткопериодического движения лежит в пределах 2—6 сек., а длинно-периодического движения (если малые корни комплексные) — в пределах 40—100 сек. При полетах на сверхзвуковых скоростях период длиннопериодических движений может составлять несколько сотен секунд.

Следует заметить, что если не учитывать зависимость аэродинамических характеристик самолета от высоты (плотности воздуха), то движение самолета по отношению к высоте полета всегда неустойчиво. Движение это

можно сделать устойчивым только при введении искусственной стабилизации средствами автоматики.

11.2.5 Боковое движение самолета

Общее движение самолета можно разделить на продольное и боковое. Проекция движения самолета на направление, перпендикулярное плоскости симметрии самолета, называется боковым движением.

Продольное движение самолета можно рассматривать независимо от бокового при любых по величине возмущениях, тогда как боковое движение можно рассматривать независимо от продольного только при малых возмущениях. В дальнейшем боковое движение самолета будет рассматриваться в предположении малых отклонений.

Для описания поведения самолета в пространстве введем связанную систему координат xyz, направив ось x по продольной оси самолета вперед, ось y по вертикальной оси вверх и ось z — по поперечной оси вправо. Введем также неподвижную по отношению,к центру масс самолета координатную систему x0y0z0 Обе системы координат имеют начало в центре масс самолета (рис.11.8).

Положение центра масс самолета по отношению к земным координатам будем определять высотой полета H, боковым отклонением от заданной траектории z и дальностью L. Связь между угловыми скоростями w_x, w_y, w_z и $\dot{\gamma},\dot{\psi},\dot{\upsilon}$ определяется соотношениями

$$\left.\begin{aligned} \omega_x &= \dot{\gamma} + \dot{\psi}\sin\upsilon \\ \omega_y &= \dot{\psi}\cos\upsilon\cos\gamma + \dot{\upsilon}\sin\psi \\ \omega_z &= \dot{\upsilon}\cos\psi - \dot{\psi}\cos\upsilon\sin\gamma \end{aligned}\right\} \qquad (11.34)$$

Скорость полета V можно выразить через ее проекции V_x, V_y, V_z на связанные оси посредством соотношений

$$\left.\begin{aligned} V_x &= V\cos\alpha\cos\beta\ ; \\ V_y &= -V\sin\alpha\cos\beta\ ; \\ V_z &= V\sin\beta\ , \end{aligned}\right\} \qquad (11.35)$$

где α и β - углы атаки и скольжения (рис. 11.6).

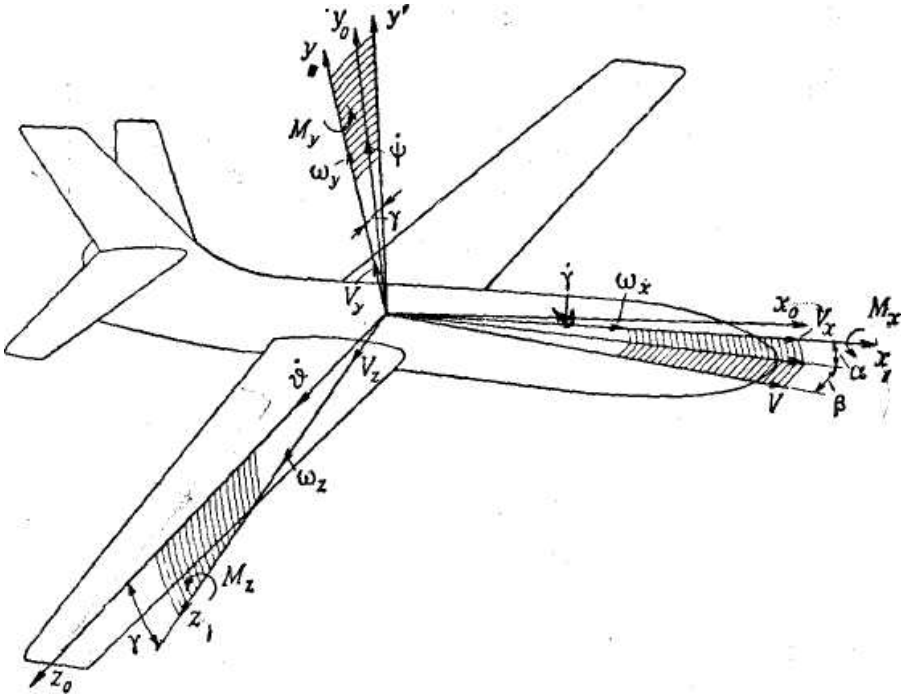

Рис.11.8 Параметры бокового движения самолёта:

ω_x, ω_y, ω_z — проекции угловой скорости ω на связанные оси xyz,

$\dot{\gamma},\dot{\psi},\dot{\upsilon}$ — проекции угловой скорости ω на полусвязанные оси $x_0y_0z_0$,

V_x, V_y, V_z — проекции скорости полета V на связанные оси xyz.

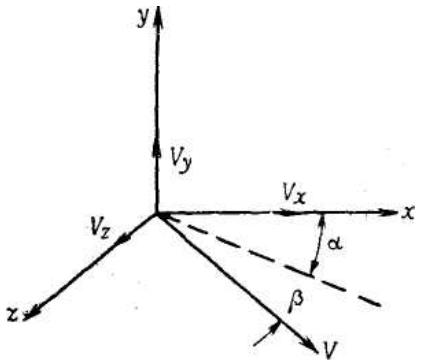

Рис.11.9 К выводу выражений для составляющих вектора скорости.

Для малых углов скольжения β (практически для всех случаев полета) уравнения упрощаются. Будем рассматривать самолет как твердое тело. При таком рассмотрении уравнения движения самолета относительно связанных осей, являющиеся частным случаем уравнений Эйлера, описывают и продольное, боковое движение самолёта [17].

Уравнения бокового движения самолёта по отношению к центру масс:

$$\left.\begin{aligned} m\left(\frac{dV_z}{dt}+\omega_x V_y-\omega_y V_x\right) &= Z+G\sin\gamma\cos\upsilon \\ J_x\frac{d\omega_x}{dt} &= M_x, \\ J_y\frac{d\omega_y}{dt} &= M_y \end{aligned}\right\} \qquad (11.36)$$

где m — масса самолета;

J_x, J_y, J_z — моменты инерции самолета относительно соответствующих осей;

X, Y, Z — проекции действующих на самолет сил (кроме силы веса);

M_x, M_y, M_z — моменты аэродинамических сил.

Для получения уравнений бокового движения центра масс по отношению к земным координатам необходимо составляющие вектора скорости V самолета и вектора скорости U ветра на направление, перпендикулярное

траектории, связать с боковой составляющей вектора скорости центра масс самолета:

$$\frac{dz_1}{dt} = V\sin(\psi - \beta) + U_z, \qquad (11.37)$$

где z_1— боковое отклонение от заданной траектории полета;

U_z — составляющая скорости ветра по оси z.

11.2.6 Линеаризация уравнений бокового движения самолета

Полученные выше уравнения (11.36) и (11.37) бокового движения являются нелинейными, поэтому непосредственное использование их для анализа процессов в системах автоматического управления затруднительно. Для упрощения задачи проведем линеаризацию этих уравнений, предположив, что боковая сила Z и моменты крена M_x и рысканья M_y зависят от параметров режима полета. Опыт показывает, что боковая сила Z зависит только от боковой составляющей скорости полета V_z; зависимостью силы Z от величин и ω_x, ω_y, $б_э$ и $б_п$ можно пренебречь. Моменты M_x и M_y зависят от величин V_z, ω_x, ω_y, $б_э$ и $б_п$. Следует заметить, что момент крена M_x мало зависит от угла отклонения руля поворота $б_п$.

После ряда преобразований [17] получаем дифференциальные уравнения бокового движения самолёта как управляемого процесса, которые устанавливают связь между регулируемыми величинами γ, β, ω_x, ω_y и регулирующими факторами $б_э$ и $б_п$:

$$\left.\begin{aligned}
(p+n_{11})\beta + n_{12}w'_x + n_{13}w'_y + n_{14}\gamma &= f_1 ;\\
n_{21}\beta + (p+n_{22})w'_x + n_{23}w'_y &= -n_{2э}\, б_э + f_2 ;\\
n_{31}\beta + n_{32}w'_x + (p+n_{33})w'_y &= -n_{3э} - n_{3п}\, б_п + f_3 ;\\
n_{42}w'_x + n_{43}w'_y + p\,\gamma &= 0,
\end{aligned}\right\} \qquad (11.38)$$

где $w'_x = \tau w_x$; $w'_y = \tau w_y$; $\tau = \dfrac{m}{\rho S V}$; f_1, f_2, f_3 - возмущения, действующие на самолёт.

Коэффициенты n_{ik} [17] зависят от размаха крыльев l, коэффициента боковой силы cz, момента крена mx, момента рысканья my, а также коэффициентов $m_x^{\beta} = \dfrac{d m_x}{d\beta}$ и $m_y^{\beta} = \dfrac{d m_y}{d\beta}$, характеризующие поперечную статическую устойчивость самолёта и устойчивость пути или флюгерную устойчивость.

Если U_z — боковой порыв ветра, ΔP — разность тяг двигателей, расположенных по разные стороны от оси самолета, ΔG — изменение веса самолета, вызывающее крен, то для возмущений f_1, f_2, f_3 можно написать выражения

$$\left.\begin{aligned} f_1 &= p v_z + \frac{\Delta P}{\rho S V^2} + f'_1 ; \\ f_2 &= \frac{\Delta G}{\rho S V^2}\frac{l_2}{l} + f'_2 ; \\ f_3 &= \frac{\Delta P}{\rho S V^2}\frac{l_3}{l} + f'_3, \end{aligned}\right\} \qquad (11.39)$$

где l_2 — плечо момента крена;

l_3 — расстояние между двигателями, имеющими разные тяги;

f'_1, f'_2, f'_3 — возмущения, вызванные стрельбой из бортового оружия, или возмущения от ударных волн, создаваемых взрывами снарядов или пролетающими соседними самолетами;

$v_z = U_z/V$ — относительная боковая составляющая ветра.

Если полет самолета горизонтален ($\upsilon = 0$), то в безразмерной форме

$$\omega'_x = p\gamma \; ; \quad \omega'_y = p\psi, \qquad (11.40)$$

следовательно, система (*2.49*) примет следующий вид:

$$\left.\begin{aligned} (p+n_{11})\beta + (n_{12}p+n_{14})\gamma + n_{13}\,p\psi &= f_1 ; \\ n_{21}\beta + (p+n_{22})p\gamma \;+ n_{23}\,p\psi &= -n_{2э}\,б_э + f_2 ; \\ n_{31}\beta + n_{32}\,p\gamma + (p+n_{33})\,p\psi &= -n_{3э}\,б_э - n_{3п}\,б_п + f_3 ; \end{aligned}\right\} \qquad (11.41)$$

В таблице приведены ориентировочные значения коэффициентов n_{ik} для бокового движения легкого, среднего и тяжелого самолетов при двух режимах полета.

Таблица 11.2

Коэффициенты уравнений бокового движения

	Самолёт					
	лёгкий		средний		тяжёлый	
	H = 6 км M = 0,77 τ_a = 2,5 с	H = 10 км M = 0,8 τ_a = 3,8 с	H = 6 км M = 0,87 τ_a = 2,1 с	H = 10 км M = 0,9 τ_a = 2,7 с	H = 8 км M = 0,8 τ_a = 2,5 с	H = 12 км M = 0,9 τ_a = 4 с
n_{11}	0,156	0,097	0,26	0,26	0,3	0,3
n_{12}	0	0	-0,055	-0,093	-0,03	-0,04
n_{13}	-1	-1	-1	-1	-1	-1
n_{14}	-0,039	-0,039	-0,13	-0,22	-0,1	-0,14
n_{21}	15,8	9,5	74,1	94	47	98
n_{22}	6,7	4,82	7,03	6,9	7,5	7,55
n_{23}	0,43	0,41	2,09	2,8	1,8	2,1
n_{31}	5,76	4,3	54	69	30	50
n_{32}	0,037	0,0058	0,064	-0,06	0,13	0,068
n_{33}	0,22	0,16	0,79	0,89	0,8	0,9
$n_{2э}$	30,7	19	107,1	136	90	150
$n_{3п}$	3,18	2,26	22,5	29,6	11	19
$n_{3э}$	0	0	-1,5	-3,2	0	0

Можно выделить частные случаи бокового движения самолета:

1) Простейшим боковым движением самолета является движение чистого рысканья без крена, когда в силу большой инерции самолета можно пренебречь движением центра масс под действием боковых сил.

2) Плоское движение (движение рысканья) при неизменном угле крена.

Нейтральность самолета по отношению к углу рысканья (или, что все равно, к курсовому углу) означает, что устойчивость самолета не изменяется при полете по любому курсу.

3) Движение крена без изменения курса ($\psi = 0$).

4) Боковое движение без скольжения ($\beta = 0$). Расчеты показывают, что коэффициент $-n_{23}n_{14}/n_{13}$ отрицателен, поэтому такое движение неустойчиво не только по отношению к углу рысканья ψ, но и по отношению к углу крена γ, – ся спиральное. При этом движении самолет имеет тенденцию перейти в штопор.

Самолет в боковом движении можно рассматривать как динамическую систему, входными координатами которой являются отклонения руля поворота $б_п$ и элеронов $б_э$, а выходными координатами — углы скольжения β и крена γ и угловая скорость рысканья ω_y.

$$\left.\begin{aligned} \beta &= \Pi_{1\beta}(p)\delta_э + \Pi_{2\beta}(p)\delta_п ; \\ \omega'_y &= \Pi_{1w}(p)\delta_э + \Pi_{2w}(p)\delta_п ; \\ \gamma &= \Pi_{1\gamma}(p)\delta_э + \Pi_{2\gamma}(p)\delta_п , \end{aligned}\right\} \qquad (11.42)$$

где $\Pi_{1\beta}(p)$, $\Pi_{2\beta}(p)$, … - передаточные функции самолёта, полностью определяемые коэффициентами n_{ik}.

Динамические свойства самолёта оцениваются передаточной матрицей вида

$$\begin{vmatrix} \mathbf{\Pi_{1\beta}} & \mathbf{\Pi_{2\beta}} \\ \mathbf{\Pi_{1w}} & \mathbf{\Pi_{2w}} \\ \mathbf{\Pi_{1\gamma}} & \mathbf{\Pi_{2\gamma}} \end{vmatrix} \qquad (11.43)$$

11.2.7 Устойчивость бокового движения самолета

Движение самолета в зависимости от его характеристик и режима полета может быть как устойчивым, так и неустойчивым. Для суждения об устойчивости бокового движения самолета следует рассмотреть характеристическое уравнение системы (11.38), которое можно получить, если приравнять нулю определитель матрицы коэффициентов дифференциальных уравнений:

$$p^4 + c_1p^3 + c_2p^2 + c_3p + c_4 = 0. \qquad (11.44)$$

Устойчивость самолета по отношению к параметрам γ, β, ω'_x, ω'_y определяется корнями уравнения (11.44). Для оценки устойчивости движения без нахождения корней уравнения можно воспользоваться критериями устойчивости, например критерием Гурвица-Раусса.

Приведем в качестве примера характеристическое уравнение с численными коэффициентами для бокового движения реактивного самолета:

$$p^4 + 4p^3 + 4{,}08\,p^2 + 10{,}72p - 1{,}44 = 0. \qquad (11.45)$$

Заметим, что для рассматриваемого режима полета последний член уравнения (11.45) является отрицательным, что указывает на неустойчивость самолета. Приближенные значения корней для этого уравнения

$$p_1 = -3{,}68; \quad p_2 = 0{,}134; \quad p_{3,4} = -0{,}225 \pm j1{,}72. \qquad (11.46)$$

В качестве второго примера рассмотрим характеристическое уравнение бокового движения самолета, коэффициенты которого приведены в таб. 2.2 :

$$p^4 + 8{,}6p^3 + 38{,}5p^2 + 24p - 0{,}164 = 0. \qquad (11.47)$$

Приближенные значения корней этого уравнения

$$p1 = -0{,}74; \quad p_2 = 0{,}00684; \quad p_{3,4} = -3{,}93 \pm j4{,}17. \qquad (11.48)$$

Следовательно, характеристическое уравнение бокового движения имеет два вещественных и два комплексных сопряженных корня. Один из

вещественных корней отрицательный и большой, а второй положительный и малый. Комплексные корни занимают промежуточное положение. Такое распределение корней для бокового движения является характерным. При этом малый корень для некоторых режимов полета может оказаться отрицательным.

Движение самолета, соответствующее большому отрицательному корню, называется движением крена. Оно быстро затухает и не оказывает большого влияния на боковое движение самолета.

Движение, соответствующее малому вещественному корню, который в рассматриваемых примерах положителен, называется спиральным движением. При положительном малом корне движение самолета апериодически неустойчиво и самолет имеет тенденцию перейти в спираль.

Движение самолета, соответствующее паре комплексных корней, является колебательным. При этом движении самолет совершает рысканья по курсу с кренами вправо и влево. Движение этого типа иногда называют «голландским шагом». Период колебаний для разных самолетов и на разных режимах может составить от 2 до 10 сек.

11.2.8 Возмущения, действующие на самолет

Самолет в полете подвергается различным возмущениям, среди которых наиболее важными являются воздушные течения (порывы ветра, периодические возмущения в атмосфере и т. д.), нарушение центровки самолета вследствие выработки топлива из баков, сбрасывание бомб и торпед, стрельба и др. Все эти возмущения различаются по длительности и характеру действия, природе и причинам их возникновения.

При произвольной выработке топлива из баков самолета на систему накладываются возмущения, являющиеся медленно изменяющимися функциями времени. Эти возмущения обычно сводятся к нарушению

центровки самолета и к изменению его динамических параметров вследствие изменения (уменьшения) веса.

Возмущения, вызванные стрельбой из орудий или ракетных установок, следует отнести к разряду единичных импульсов. Такие возмущения вследствие кратковременности не оказывают существенного влияния на движение самолета.

Возмущения, вызванные сбрасыванием бомб или грузов, можно представить единичными функциями времени. Интенсивность этих возмущений зависит от величины и расположения в самолете сбрасываемых грузов.

Указанные типы возмущений являются определенными функциями времени, и их воздействие на самолет можно учесть заранее.

Возмущениями, которые невозможно заранее определить и точно учесть, являются возмущения, вызванные воздушными течениями. При этом под воздушными течениями будем подразумевать всякое перемещение воздуха (в том числе и колебание) по отношению к земле. Остановимся более подробно на возмущениях этого типа.

Существуют различные виды воздушных течений, например постоянные ветры, восходящие и нисходящие потоки, порывы ветра, завихрения, периодические возмущения и пр. Если самолет совершает полет при наличии ветра, то его центр масс, помимо движения по отношению к частицам воздуха, перемещается также вместе с частицами воздуха со скоростью ветра. Следовательно, скорость самолета по отношению к земле, (путевая скорость) состоит из геометрической суммы скорости самолета по отношению к воздуху и скорости ветра.

При воздействии переменных возмущений, вызванных воздушными течениями (восходящие и нисходящие потоки, порывы ветра, завихрения), самолет испытывает удары, броски вверх и вниз и из стороны в сторону. Следовательно, самолет испытывает перегрузки и его движение в

возмущенной атмосфере будет отличаться от движения в спокойной атмосфере.

Причиной возникновения воздушных течений является неравномерное распределение давления воздуха в атмосфере. В зависимости от времени суток и года, места над земной поверхностью, рельефа местности, наличия паров в атмосфере воздух по-разному нагревается за счет солнечной энергии. При этом между различными слоями воздуха в горизонтальном и вертикальном направлениях возникают значительные разности температур, приводящие к изменению плотности и давления. Следовательно, передаваемая воздуху тепловая энергия солнца преобразовывается в конечном счете в энергию движения частиц воздуха. Частицы воздуха движутся с разными скоростями и в самых различных направлениях. Если скорость движения частиц превышает определенный предел, то движение становится турбулентным. Такое движение имеет неупорядоченный характер и сопровождается значительным вихреобразованием.

Для более детального изучения воздушных течений проведем их классификацию, положив в основу такой классификации внешние проявления течений. Все воздушные течения можно разбить на два вида.

Вертикальные течения, включающие восходящие и нисходящие потоки, облачные течения, буруны и вихри. Влияние этих течений на самолет зависит от величины вертикальной составляющей скорости движения воздуха, пространственного протяжения течения, скорости полета и др. При полете с постоянным углом атаки самолет на нисходящих потоках будет опускаться, а на восходящих — подниматься.

Горизонтальные течения, включающие постоянные по величине и направлению течения (ветры), ветровые слои, порывы, волны, вихри, буруны.

Восходящие потоки возникают от подъема вверх теплых слоев воздуха, нагреваемого при соприкосновении с земной поверхностью. Эти потоки часто наблюдаются в летние тихие дни после полудня. Скорость подъема

теплых слоев воздуха иногда достигает 10—12 м/сек. При воздействии потока на одно из крыльев самолета подъемная сила крыльев становится различной, вследствие чего самолет получает крен. В момент входа в такой поток или выхода из него самолет получает внезапный толчок.

Если в восходящих потоках подъемная сила самолета возрастает, то в нисходящих, наоборот, уменьшается. При этом, попадая в такой поток, самолет имеет тенденцию проваливаться. Направления и величины скоростей в восходящих и нисходящих потоках самые различные.

Восходящие и нисходящие потоки воздуха наблюдаются также в облаках, особенно кучевых. При этом в пределах облака воздух поднимается вверх, а в промежутках между облаками — опускается вниз.

Часто смежные слои воздуха различаются по температуре, влажности и плотности, при этом они начинают перемещаться по отношению друг к другу, в результате чего образуются ветровые слои. При переходе самолета из одного ветрового слоя в другой подъемная сила изменяется и самолет начинает опускаться или подниматься.

Известно, что на границе раздела двух сред, например на границе двух воздушных слоев, движущихся с разными скоростями, образуются волны, которые могут оказаться опасными для самолета. Попав в область воздушных волн, самолет начинает испытывать тряску.

Во всех видах воздушных течений всегда имеют место порывы ветра, характеризуемые резким изменением скорости и направления ветра. Порывы ветра бывают самыми различными по величине и направлению; при этом чем больше средняя скорость воздушною течения, тем больше интенсивность порывов. В зависимости от величины и направления порывов ветра, самолет начинает испытывать резкие подъемы и опускания, а также боковое скольжение.

Скорость вертикальных порывов ветра в тропосфере и нижних слоях стратосферы может достигать 10—12 м/сек при мощности турбулентного слоя до 1000 м и горизонтальной протяженности до 200 км. Наиболее

сильная турбулентность наблюдается на высоте 7—10 км. Вертикальные порывы ветра вызывают значительные перегрузки самолета.

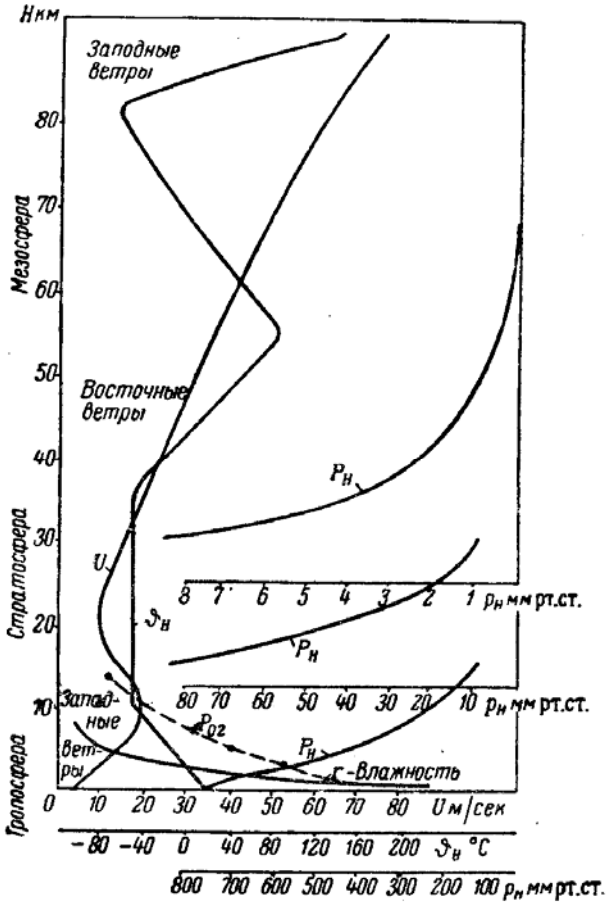

Рис.11.10 Распределение давления p_H, температуры υ_H и скоростей ветра U на высоте.

Горизонтальные порывы ветра, а также горизонтальные воздушные течения, которые в момент попадания в них самолета подобны порывам ветра, имеют значительно большие скорости, возрастающие с увеличением высоты полета. Скорость таких порывов ветра на высоте 7—10 км может достигать 20 м/сек., а на больших высотах — еще больших величин.

Всякое течение воздуха сопровождается вихреобразованием; при этом чем больше скорость течения воздуха, тем больше скорость вращения вихрей. Образование вихрей является результатом турбулентного движения воздуха. Турбулентность вызывается многими причинами, например трением воздуха

о поверхность земли, тепловой конвекцией, трением слоев воздуха друг о друга и др.

Вихри, сопутствующие турбулентному движению, могут быть с вертикальной и горизонтальной осью вращения, причем с последней встречаются чаще. Диаметры вихрей различны и определяются в основном скоростью воздушного течения.

При попадании самолета в область интенсивного вихреобразования наблюдается болтанка. Интенсивность и частота болтанки зависят от интенсивности и размеров вихрей и скорости полета.

Средние значения скорости ветра изменяются в зависимости от широты и долготы места, времени года, высоты и т. д. На рис.11.10 приведен график наиболее вероятного распределения горизонтальных скоростей ветра по высоте для средних широт. На том же рисунке показана картина распределения температуры и давления воздуха. Эти графики получены на основании осреднения иностранных данных.

11.3 Программная реализация аналитических моделей

Программная реализация для системы линейных обыкновенных дифференциальных уравнений была рассмотрена в лабораторной работе № 3.

Программные реализации для лианиризованных систем 11.16, 11.18 продольного движения самолета и для лианиризованных систем 11.38 бокового движения самолета должны быть разработаны студентами самостоятельно.

11.4 Построение имитационных моделей

Имитационная модель для линеаризованной системы дифференциальных уравнений продольного движения самолёта представлена на рис. 11.11. Так как в редакторе модель симулинк не поддерживается греческий шрифт то в уравнениях (11.16) сделаны следующие замены: $\delta_р$-> dr, $\delta_в$->db, α-> a и υ->u.

Осциллограммы выходных величин v, a, u и h для различных начальных данных и при различных внешних и управляющих воздействий приведены на рис. 11.12 – 11.26.

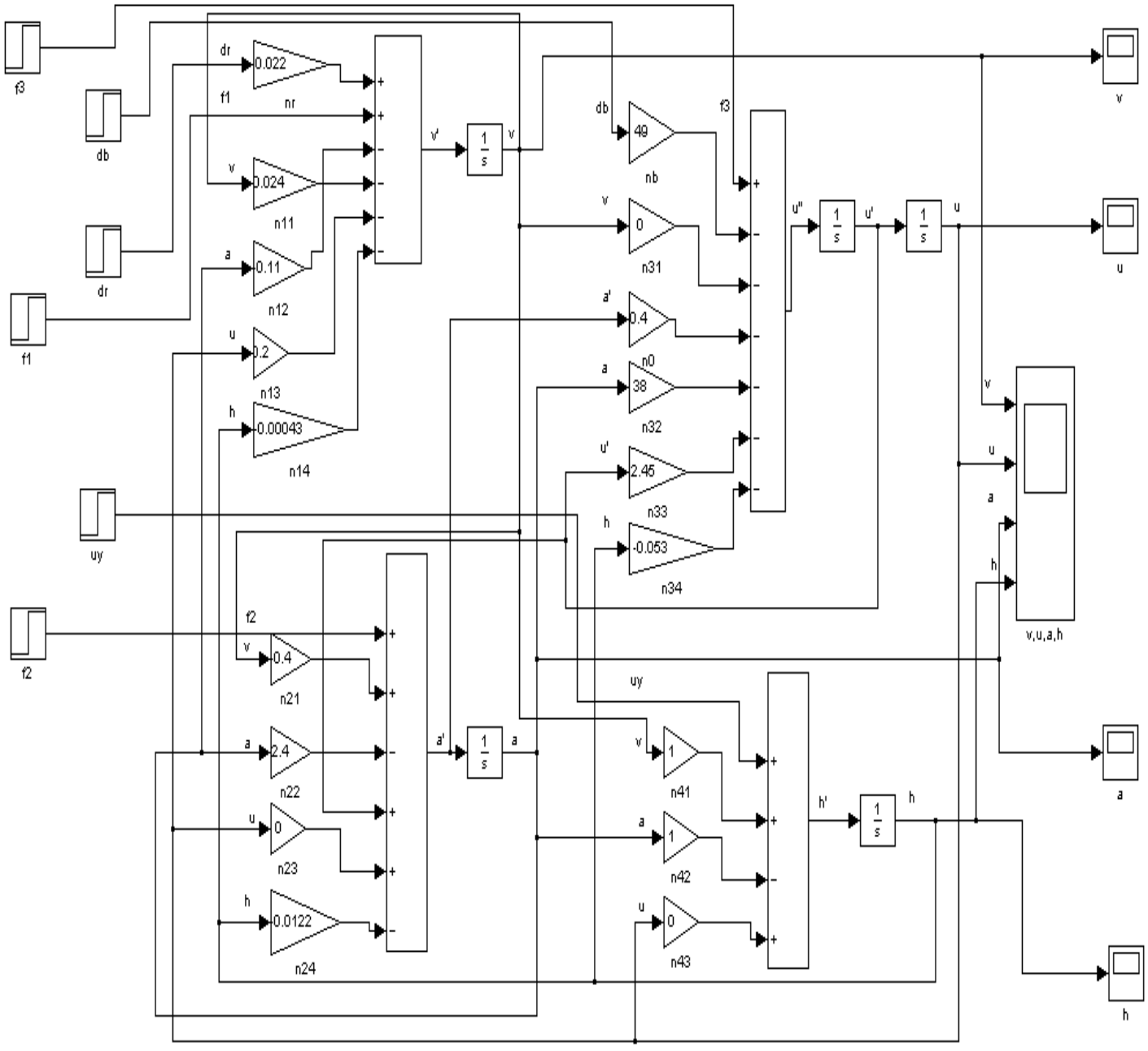

Рис. 11.11 Линеаризованная имитационная модель продольного движения самолёта

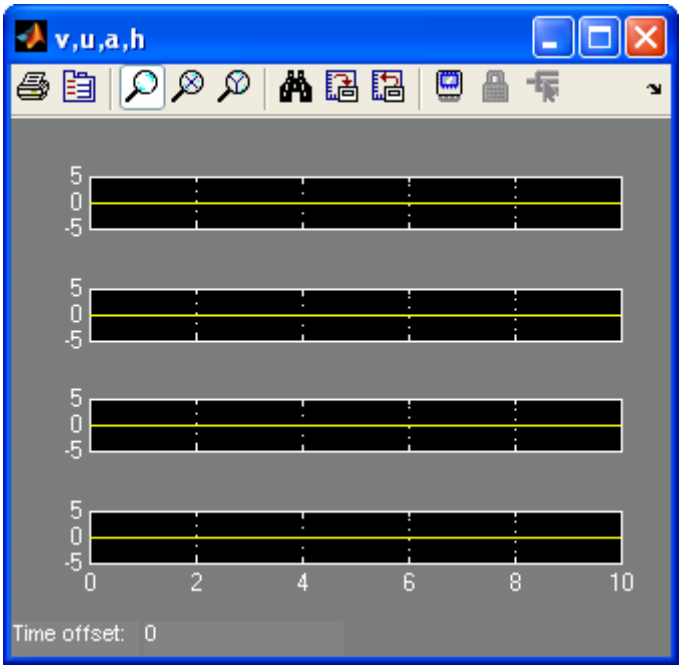

Рис. 11.12. Выходные величины для нулевых начальных условий и нулевых воздействий

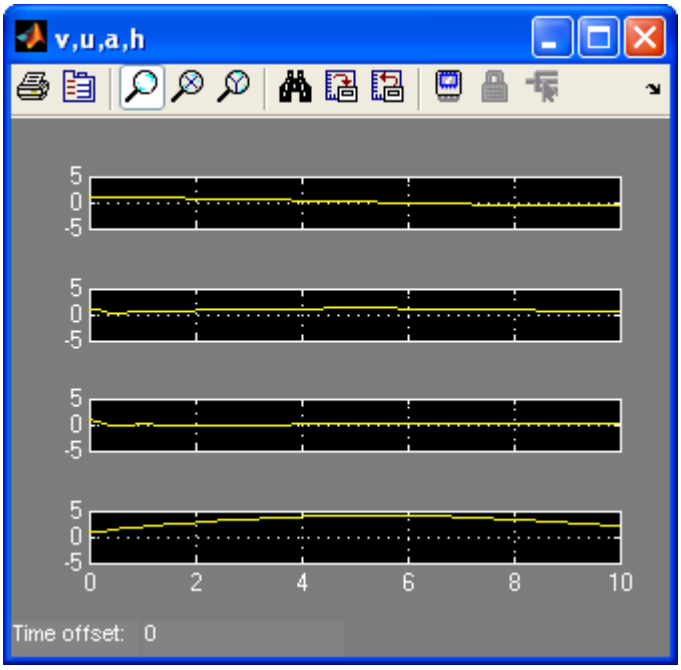

Рис. 11.13. Выходные величины для единичных начальных условий и нулевых воздействий

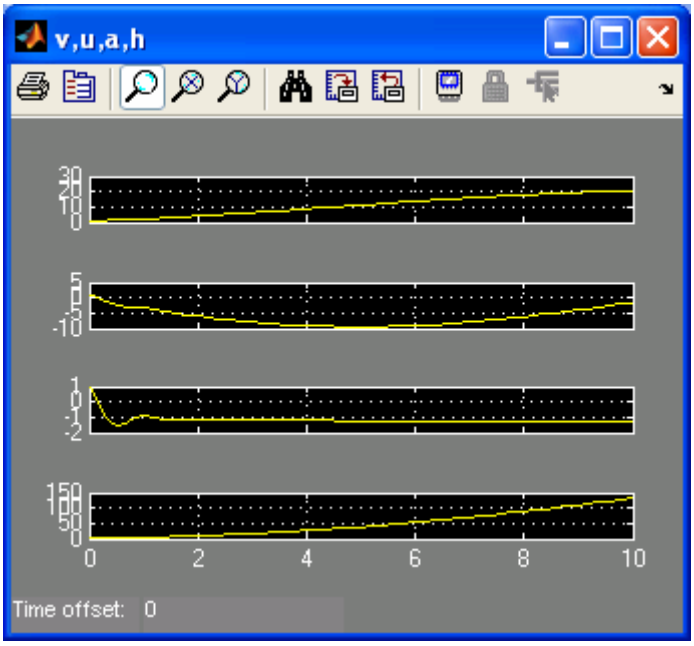

Рис. 11.14. Выходные величины для единичных начальных условий и единичных воздействий

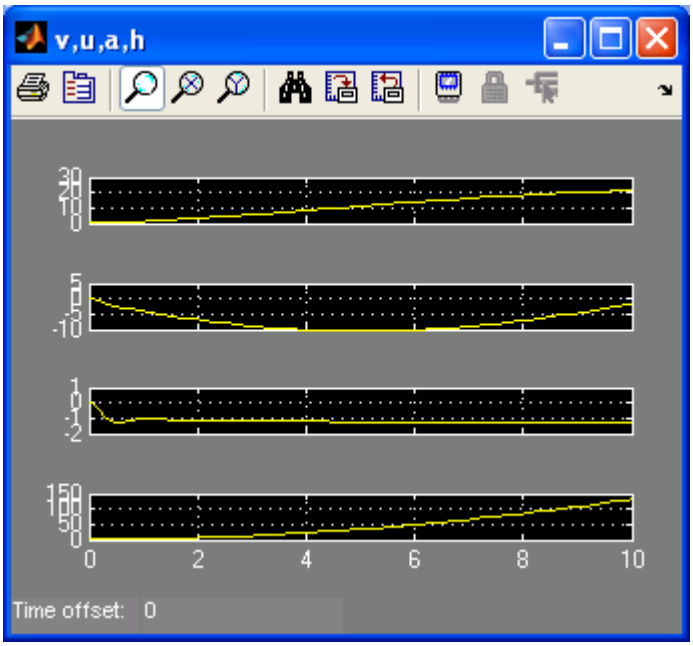

Рис. 11.15. Выходные величины для нулевых начальных условий и единичных воздействий

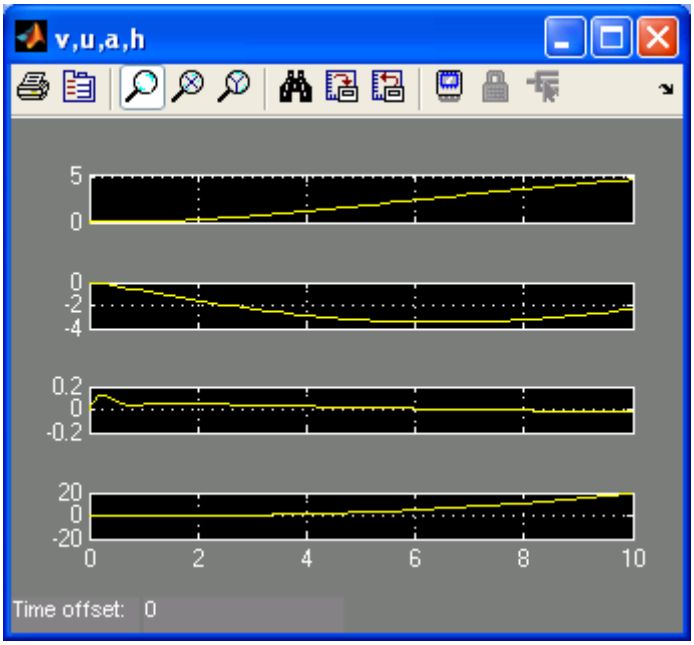

Рис. 11.16. Выходные величины для толчка $f_1(1)$

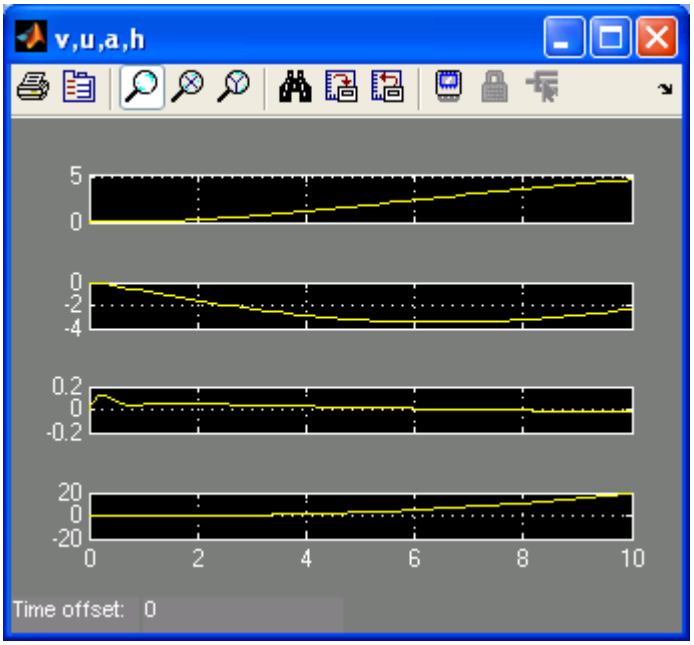

Рис. 11.17. Выходные величины для толчка $f_2(1)$

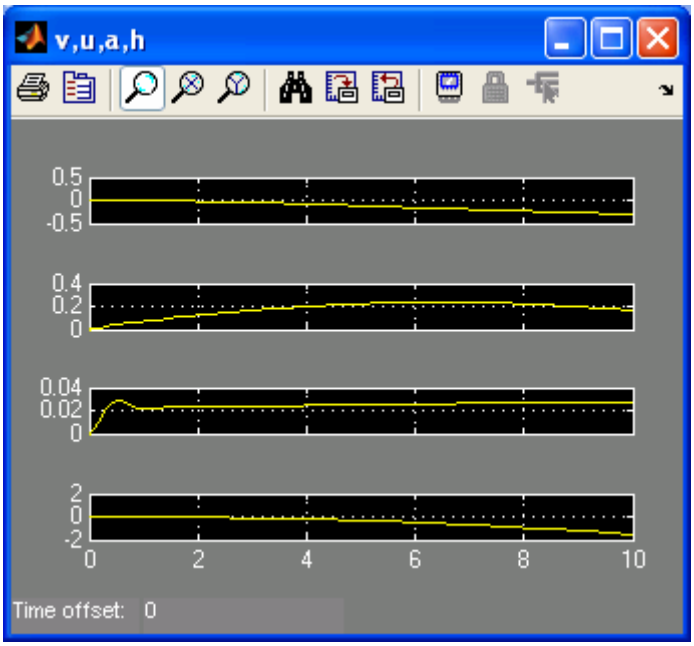

Рис. 11.18. Выходные величины для толчка $f_3(1)$

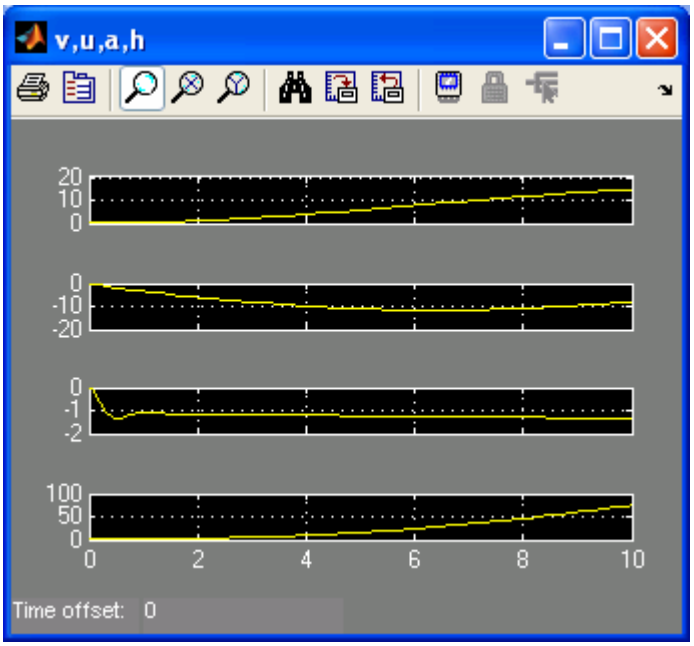

Рис. 11.19. Выходные величины для толчка db(1)

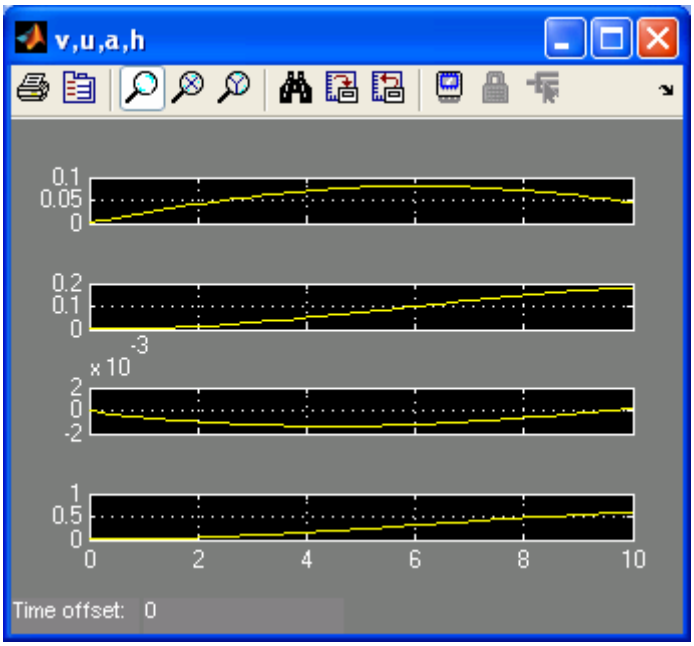

Рис. 11.20. Выходные величины для толчка dr(1)

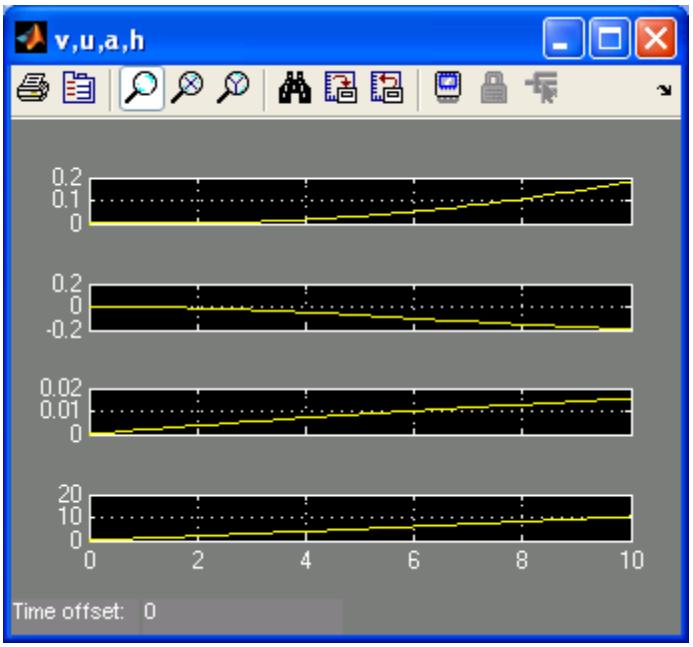

Рис. 11.21. Выходные величины для толчка uy(1)

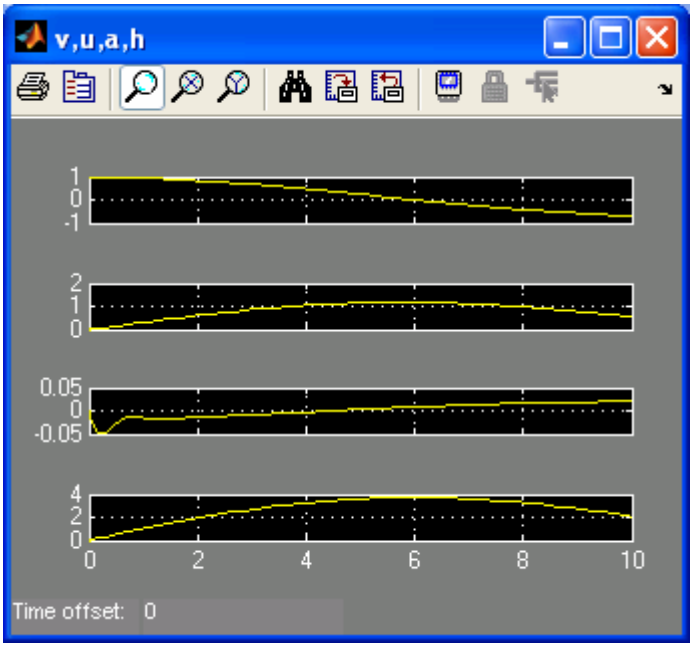

Рис. 11.22. Выходные величины для v(0)=1

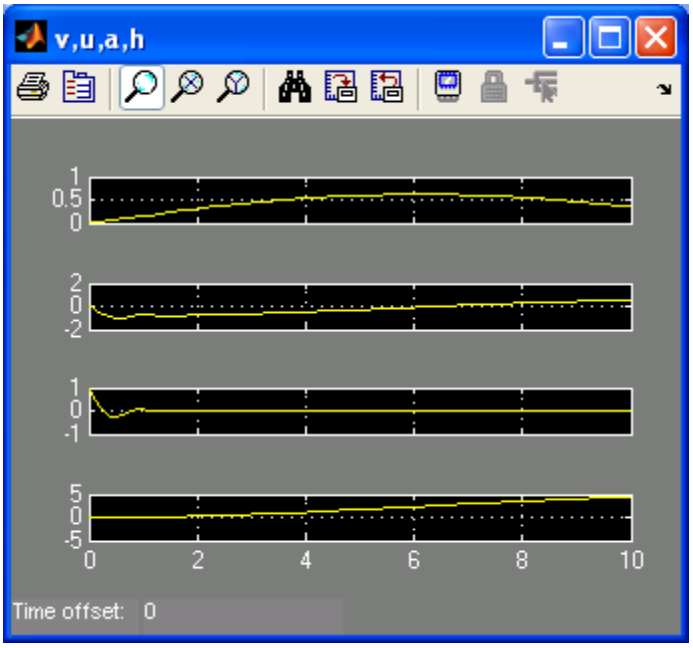

Рис. 11.23. Выходные величины для a(0)=1

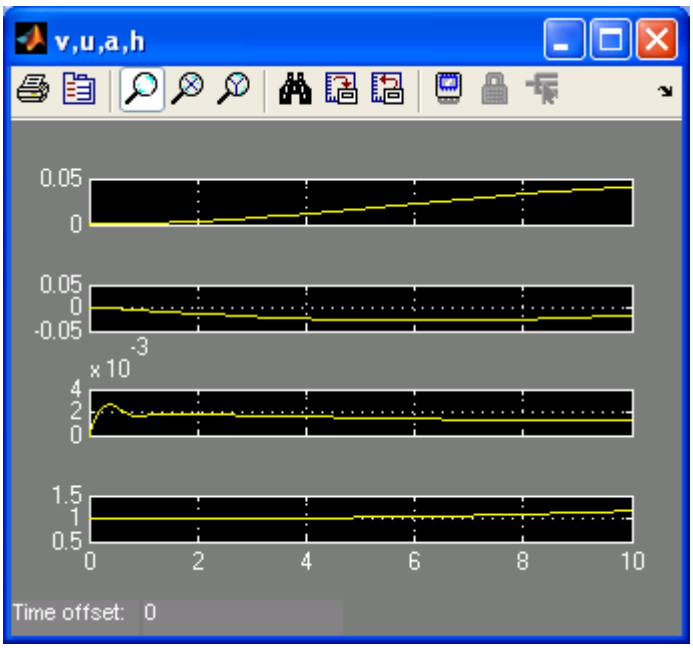

Рис. 11.24. Выходные величины для h(0)=1

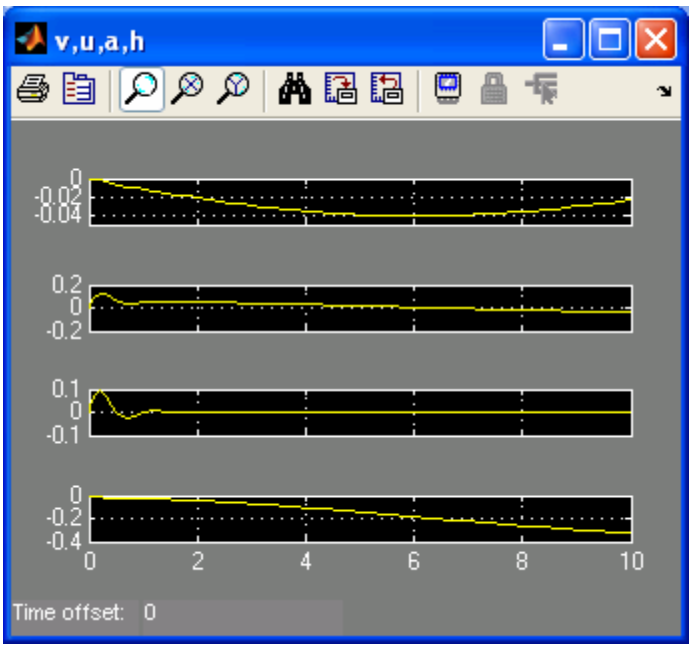

Рис. 11.25. Выходные величины для $u'(0)=1$

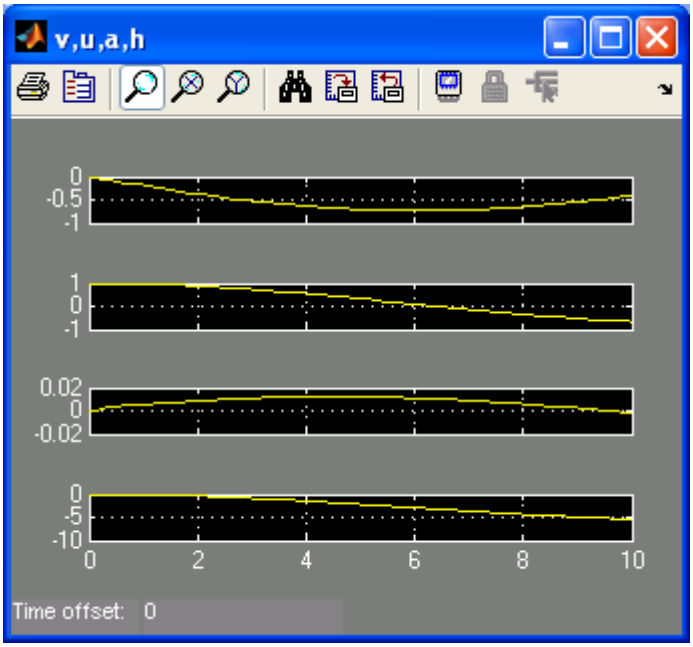

Рис. 11.26. Выходные величины для u(0)=1

11.5 Верификация математических моделей

1. Используя функцию dsolve и символьные вычисления, найти аналитические решения линеаризованных уравнений для продольного и бокового движения самолетов различных типов и сравнить их с результатами моделирования.

11.6 Варианты заданий и порядок их выполнения

1.Варианты данных для исследования продольного полета самолетов различных типов представлены в табл. 11.1, а для бокового — в табл. 11.2 .

2. Необходимо также исследовать общие уравнения движения самолета при различных режимах его полета: взлет, горизонтальный полет, маневр в горизонтальной плоскости, маневр в вертикальной плоскости, пикирование, кобрирование, бомбометание, запуск ракеты и штопор.

11.7 Оформление отчета по результатам исследований

Для завершения лабораторной работы необходимо сгенерировать отчет в формате HTML, затем преобразовать его в формат RTF с помощью текстового редактора, включит в него теоретические результаты, отформатировать текст и графические объекты, записать на дискету и в электронном виде предъявить преподавателю. Обосновать достоверность полученных результатов.

Заключение

Овладение предложенными в лабораторных работах методами и приемами построения и исследования оптимальных систем управления с применением математической системы MATLAB и пакета имитационного моделирования Simulink позволит обучаемым быстро, эффективно и надежно решать сложные инженерные задачи при проектировании систем автоматического управления для реальных динамических процессов из самых разнообразных предметных областей. При этом в зависимости от целей проектирования и характеристик объекта управления можно использовать непрерывные, нелинейные или дискретные блоки, состав которых постоянно расширяется. Предусмотрена возможность создания собственных библиотек. В случае необходимости можно использовать другие пакеты системы MATLAB такие, например, как пакет для решения задач в аналитическом виде Symbolic Math Toolbox, мощную библиотеку математических функций NAG Foundation Toolbox, пакет для решения оптимизационных задач и систем нелинейных уравнений Optimization Toolbox, пакет для решения систем дифференциальных уравнений в частных производных Partial Differential Equations Toolbox, пакет для построения и исследования искусственных нейронных сетей Neural Networks Toolbox и многие другие.

Глубокие знания по математическому анализу, тонкая интуиция и отличные навыки применения современных средств компьютерной математики гарантируют успех в деле решения сложных проблем управления, возникающих в современную эпоху.

Список литературы

1. Шляндин В.М. Основы автоматики. – М.: – Л.: Государственное энергетическое издательство, 1958.– 592с.

2. Фельдбаум А.А. Вычислительные устройства в автоматических системах. – М.: Государственное издательство физико-математической литературы, 1959. – 800с.

3. Иващенко И.Н. Автоматическое регулирование. Теория и элементы системы. – М.: Государственное научно-техническое издательство машиностроительной литературы, 1962. – 628с.

4. Канторович Л.В., Крылов В.И. Приближенные методы высшего анализа. – М.: – Л.: Государственное издательство физико-математической литературы. – 1962. – 708с.

5. Градштейн И.С., Рыжик И.М. Таблицы интегралов, сумм, рядов и произведений. – М.: – Государственное издательство физико-математической литературы, 1962. – 1098с.

6. Андре Анго. Математика для электро- и радиоинженеров. – М.: Издательство «Наука», 1964. – 772с.

7. Алексеев В.М., Тихомиров В.М., Фомин С.В. Оптимальное управление. – М.: Наука. Главная редакция физико-математической литературы, 1979. – 224с.

8. Колмогоров А.Н., Фомин С.В. Элементы теории функций и функционального анализа. – М.: Наука. Главная редакция физико-математической литературы, 1981. – 544с.

9. Смирнов В.И. Курс высшей математики. Том IV. Издание третье. – М.: Государственное издательство технико-теоретической литературы, 1957. – 812с.

10. Этерман И. И. Математические машины непрерывного действия. – М.: Государственное издательство научно-технической машиностроительной литературы, 1957. – 236с.

11. Кобринский Н. Е. Математические машины непрерывного действия. – М.: Государственное издательство технико-теоритической литературы, 1954. – 448с.

12. Дьяконов В. MATLAB 6: Учебный курс. – СПб.: Питер, 2001.– 592с.: ил.

13. Дьяконов В. П. MATLAB 6/6.1/6.5+Simulink 4/5. Основы применения. Полное руководство пользователей/Дьяконов В. П. М.: СОЛОН – Пресс. – 2002. – 768с.

14. Медведев В. С., Потемкин В. Г. Нейронные сети. MATLAB 6/ Под общ. ред. к. т. н. В. Г. Потемкина. – М.: ДИАЛОГ– МИФИ, 2002. – 496с. – (Пакеты прикладных программ; Кн.4).

15. Ануфриев И. Е. Самоучитель MatLab 5.3/6.x. – СПб.: БХВ – Петербург, 2004. – 736с.: ил.

16. Кетков Ю. Л., Кетков А. Ю., Шульц М. М. MATLAB 6.x.: Программирование численных методов. – СПб.: БХВ – Петербург, 2004. – 672с.: ил.

17. Боднер В.А., Козлов М.С. Стабилизация летательных аппаратов и автопилоты/ Под ред. Докт. Техн. Наук проф. В.А. Боднера. М.: Государственное научно-техническое издательство ОБОРОНГИЗ, 1961, 508 с.

Содержание