

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД
УКРАЇНСЬКИЙ ДЕРЖАВНИЙ ХІМІКО-ТЕХНОЛОГІЧНИЙ УНІВЕРСИТЕТ

ІНСТРУКЦІЯ ДО ЛАБОРАТОРНОЇ РОБОТИ №4
з дисципліни
моделювання систем
за темою «Моделювання динамічних систем»

Дніпропетровськ, 2015

I. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

Решение задачи Коши

Анализ поведения многих систем и устройств в динамике базируются на решении систем дифференциальных или интегро-дифференциальных уравнений. Распространенным способом описания поведения динамической системы являются системы обыкновенных дифференциальных уравнений (ОДУ).

Их, как правило, представляют в виде системы из дифференциальных уравнений первого порядка в форме Коши:

$$\begin{aligned}\frac{dy}{dt} &= y'; \\ y' &= f(x, y), \quad y(t_0, t_{end}) = b.\end{aligned}$$

где t_0, t_{end} – начальные и конечные точки интервалов.

Параметр t не обязательно означает время, хотя чаще всего решение дифференциальных уравнений ищется во временной области. Вектор b задает начальные и конечные условия.

Задачей Коши называется задача о решении обыкновенного дифференциального уравнения с известными начальными условиями. Для решения систем ОДУ в MATLAB реализованы различные методы. Их реализации названы решателями ОДУ. В MATLAB имеются три возможности для решения задачи Коши, не считая моделирования в SIMULINK.

Первая из них касается численного решения линейных дифференциальных уравнений с известной правой частью или систем таких уравнений. Оно может быть выполнено с помощью команды *lsim* (для решения однородных уравнений достаточно команды *initial*).

Вторая возможность – аналитическое решение линейных и простых нелинейных дифференциальных уравнений с помощью решателя *dsolve* тулбокса SYMBOLIC.

Пример. Пусть требуется решить линейное уравнение первого порядка:

$$y' + a^2 y = 0.$$

В командной строке набираем:

```
>>y=dsolve('Dy+a^2*y=0')
```

MATLAB выдает ответ:

$$y = C / \exp(a^2 * t)$$

Получено общее решение дифференциального уравнения в аналитическом виде.

Задав начальные условия $y_0 = 1$, получаем задачу Коши. Для ее решения набираем:

```
>> y=dsolve('Dy+a^2*y=0, y(0)=1')
```

Получаем ответ:

$$y = 1/\exp(a^2*t).$$

Получено частное (с учетом начальных условий) решение дифференциального уравнения в аналитическом виде.

Аналогично решатель *dsolve* применяют для систем дифференциальных уравнений.

Третья возможность – численное решение нелинейных дифференциальных уравнений с помощью команд типа *ode23* и *ode45*. Буквенная часть названия этих команд – сокращение от *Ordinary Differential Equation*, цифры указывают порядок используемой версии метода Рунге-Кутты.

Отметим, что решатели реализуют следующие методы решения систем дифференциальных уравнений, причем для решения жестких систем уравнений рекомендуется использовать, только специальные решатели **ode45**, **ode23**:

– **ode45** – одношаговые явные методы Рунге-Кутты 4-го и 5-го порядка. Это классический метод, рекомендуемый для получения начального решения.

– **ode23** – одношаговые явные методы Рунге-Кутта 2-го и 4-го порядка.

Рассмотрим основную модификацию команды *ode23*, которая имеет вид:

$$[t, X] = \text{ode23}('fun', [T_0 \ T_1], X_0)$$

Она обеспечивает решение системы дифференциальных уравнений, записанных в форме Коши:

$$\dot{X} = F(X, t), \quad X(T_0) = X_0$$

на интервале времени $T_0 \leq t \leq T_1$. Результатом её выполнения является массив отсчетов времени t и соответствующий им массив значений X . Для того чтобы команда была выполнена, надо предварительно составить программу для вычисления вектор-функции $F(X, t)$, стоящей в правой части дифференциального уравнения. Эта программа должна быть оформлена в виде m-файла, которому присваивается любое имя, например, '*fun*'.

Пример. Воспользуемся командой *ode23* для моделирования нелинейного уравнения:

$$\ddot{y} + \sin y = 0,8y^3, \quad y(0)=1, \dot{y}(0)=0$$

Перепишем его в виде системы двух уравнений, обозначив $x_1 = y$, $x_2 = \dot{y}$:

$$\begin{aligned} \dot{x}_1 &= x_2, \\ \dot{x}_2 &= 0,8x_1^3 - \sin x_1 \end{aligned} \quad X_0 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

Функцию для вычисления правых частей оформляем в виде m-файла с именем *fun*:

```
function F = fun (t, X)
F = [X(2); .8*X(1)^3-sin(X(1))];
```

Далее задаем численные значения параметров

```
>>T0 = 0, T1 = 20, X0 = [1; 0];
```

после чего уже может быть выполнена основная команда

```
>> [t, X] = ode23('fun', [T0 T1], X0)
```

Ее результатом будут одномерный массив времени t на интервале от 0 до 20 секунд и двумерный массив X , содержащий значения $x_1(t)$, $x_2(t)$. Как правило, шаг времени – переменный. График результата может быть получен командой *plot(t, X)*. Для последующего перехода к равномерной сетке времени (если это необходимо), можно использовать команду *interp1*:

```
>>tt = 0:0.1:20;
XX = interp1(t, X, tt, 'spline');
```

результатом которой будет массив переменных XX для равноотстоящих моментов времени tt . Другая возможность получения равномерного шага связана с использованием команды *deval*. Для этого слегка изменим синтаксис команды *ode23*:

```
>> SOL = ode23(@fun,[T0 T1], X0)
```

Теперь результатом будет структура SOL , о чем свидетельствует сообщение:

```
SOL = solver: 'ode23'    extdata: [1x1 struct]    x: [1x90 double]    y: [2x90 double]
```

Структура SOL имеет поля x и y . Поле x содержит набор отсчетов времени, а поле y – массив значений вектора $X(t)$. Доступ к полям производится командами $SOL.x$ и $SOL.y$. Первый аргумент команды *deval* – структура SOL , а второй – вектор точек, в которых нужно вычислить аппроксимацию решения. Задаем равномерную сетку по времени и строим график:

```
>>t=0:.1:20;
y = deval(SOL,t);
figure(2), plot(t,y,'o')
```

Результат приведен на рис. 1.

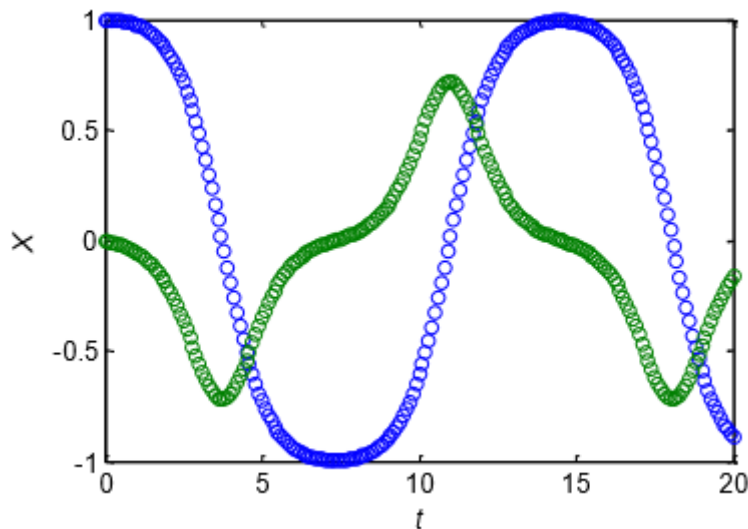


Рис. 1 – Результаты моделирования с постоянным и переменным шагом

В качестве дополнительного аргумента в команде `ode23` можно указать требуемую точность решения `eps` (по умолчанию `eps = 0.001`). Команда `ode45` выполняется аналогично и имеет такие же модификации. Обе команды можно использовать для моделирования нелинейных систем автоматического управления, если в правой части дифференциальных уравнений учесть управляющее воздействие, задавая его как явную функцию времени.

В MATLAB существуют и другие решатели дифференциальных уравнений, например, `ode15s`, `ode23s`, `ode23t`. Команды с буквой *S* предназначены для моделирования жестких (*Stiff*) дифференциальных уравнений, буква *T* указывает на использование метода трапеций. Большинство из них способны решать и дифференциально-алгебраические системы уравнения вида $M(X,t)\dot{X} = F(X,t)$ где M – матрица, F – вектор-функция.

Пример использования солвера `ode45`, записанного в скриптовый файл:

```
function solvdem
Y0 = [1; 2];
options = odeset('OutputFcn', @odeplot)
[T, Y] = ode45(@soldy, [0 10], Y0, options);...
```

Результаты моделирования представлены на графике:

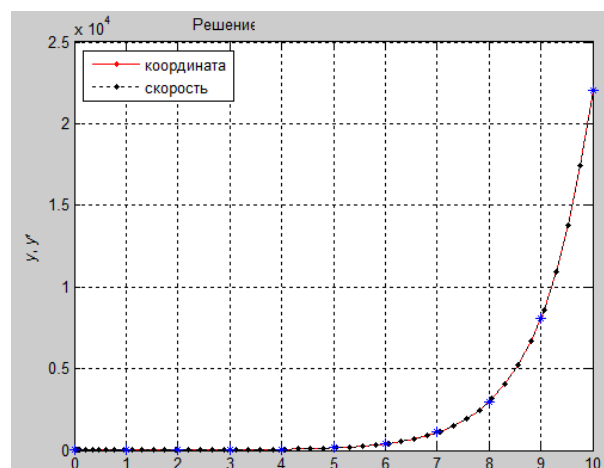


Рис. 2 – Графическое представление результатов моделирования

В описанных далее функциях для решения систем дифференциальных уравнений приняты следующие обозначения и правила:

- *options* – аргумент, создаваемый функцией *odeset* (еще одна функция позволяет вывести параметры, установленные по умолчанию);

- *tspan* – вектор, определяющий интервал интегрирования $[t_0 \ t_{final}]$. Для получения решений в конкретные моменты времени $t_0, t_1, \dots, t_{final}$ (расположенные в порядке уменьшения или увеличения) нужно использовать $tspan = [t_0 \ t_1 \ \dots \ t_{final}]$;

- $y0$ – вектор начальных условий;

- p_1, p_2, \dots – произвольные параметры, передаваемые в функцию F ;

- T, Y – матрица решений Y , где каждая строка соответствует времени, возвращенном в векторе-столбце T .

Перейдем к описанию функций для решения систем дифференциальных уравнений:

- $[T, Y] = \text{solver}(@F, tspan, y0)$ – где вместо *solver* подставляем имя конкретного решателя – интегрирует систему дифференциальных уравнений вида $y' = F(t, y)$ на интервале $tspan$ с начальными условиями $y0$, $@F$ – дескриптор ODE-функции. Каждая строка в массиве решений Y соответствует значению времени, возвращаемому в векторе-столбце T ;

- $[T, Y] = \text{solver}(@F, tspan, y0, options)$ – дает решение, подобное описанному выше, но с параметрами, определяемыми значениями аргумента *options*, созданного функцией *odeset*.

Пример. Приведем технологию решения дифференциальных уравнений в системе MATLAB:

1. Шаг: Создание m-файла. Независимо от вида системы он имеет вид:

```
function dy = solverDE(t, y)
dy = zeros(n, 1);
dy(1) = f1 (t, y(1), y(2), ..., y(n));
dy(2) = f2 (t, y(1), y(2), ..., y(n));
.....
dy(n) = fn (t, y(1), y(2), ..., y(n));
```

2. Шаг: Получение решения и сопровождающий его график:

```
>> [T, Y] = solver('solverDE', [t0 tfinal], [y10 y20 ... yn0]);
>> plot(T, Y)
```

Пусть, к примеру, требуется решить дифференциальное уравнение:

$$y''' - 2y'' - y' + 2y = 0$$

с единичными начальными условиями.

Данное дифференциальное уравнение второго порядка приведем к системе дифференциальных уравнений первого порядка:

$$\begin{cases} \frac{dy_1}{dt} = y_2; \\ \frac{dy_2}{dt} = y_3; \\ \frac{dy_3}{dt} = 2y_3 + y_2 - 2y_1, \end{cases}$$

с начальными условиями $y_1(0)=1, y_2(0)=1, y_3(0)=1$.

Вектор dy/dt правых частей системы уравнений, вычисляем с помощью собственной функции ex21 (рис. 3):

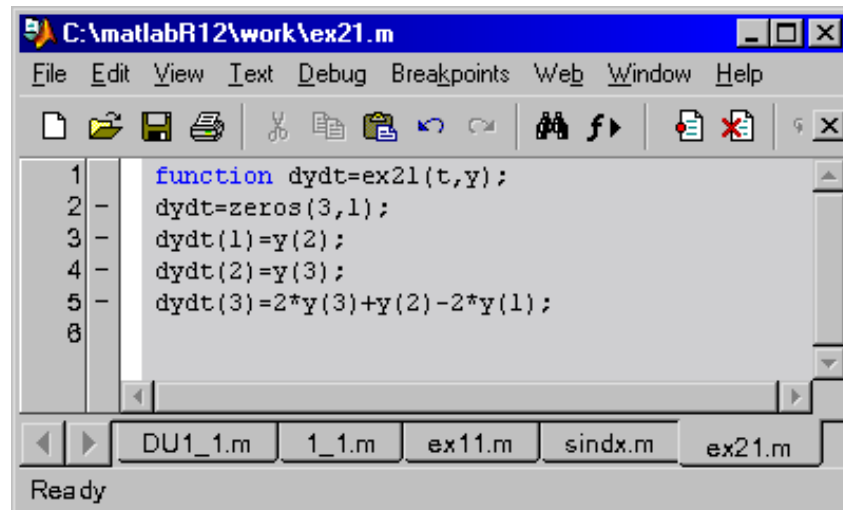


Рис. 3 – Пример создания функции для решения системы ОДУ

Теперь можно вызывать функцию **ode45**, находящую решение нашей системы дифференциальных уравнений с начальными условиями $[1,1,1]$ на отрезке $[0,20]$ (рис. 4):

```
>> y0=[1 1 1];
>> tspan=[0 20];
>> [T,Y]=ode45('ex21',tspan,y0);
>> plot(T,Y)
```

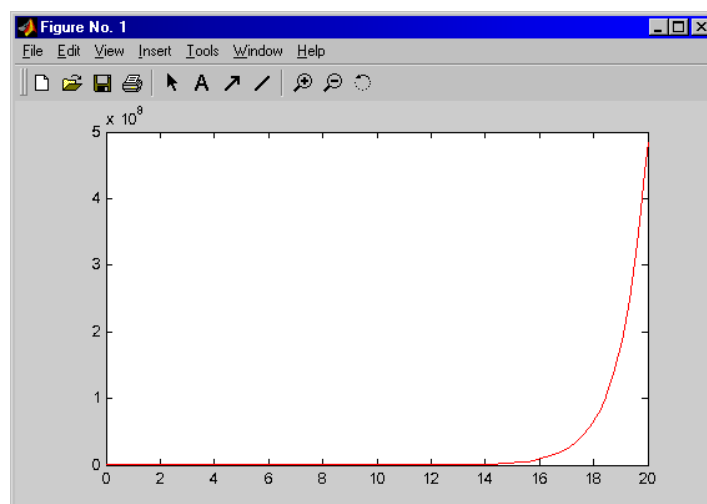


Рис. 4 – Результат работы программы

Для решения дифференциальных уравнений в MATLAB зарезервирована функция *dsolve*, которая имеет следующие форматы обращения и возвращает аналитическое решение системы дифференциальных уравнений с начальными условиями:

– $y = \text{dsolve}('Dy(x)'),$ где $Dy(x)$ – уравнение, y – возвращаемые функцией *dsolve* решения.

– $y = \text{dsolve}('Dy(x)', 'HY'),$ где $Dy(x)$ – уравнение, HY – начальные условия.

Первая производная функции обозначается Dy , вторая производная – $D2y$ и так далее. Функция *dsolve* предназначена также для решения системы дифференциальных уравнений в символьном виде (как отмечалось уже ранее). В этом случае она имеет следующий формат обращения:

– $[f, g] = \text{dsolve}('Df(x), Dg(x)', 'HY'),$ где $Df(x), Dg(x)$ – система уравнений, HY – начальные условия.

Решить заданное дифференциальное уравнение $y''' - 2y'' - y' + 2y = 0$ и использованием функции *dsolve* (рис. 5).

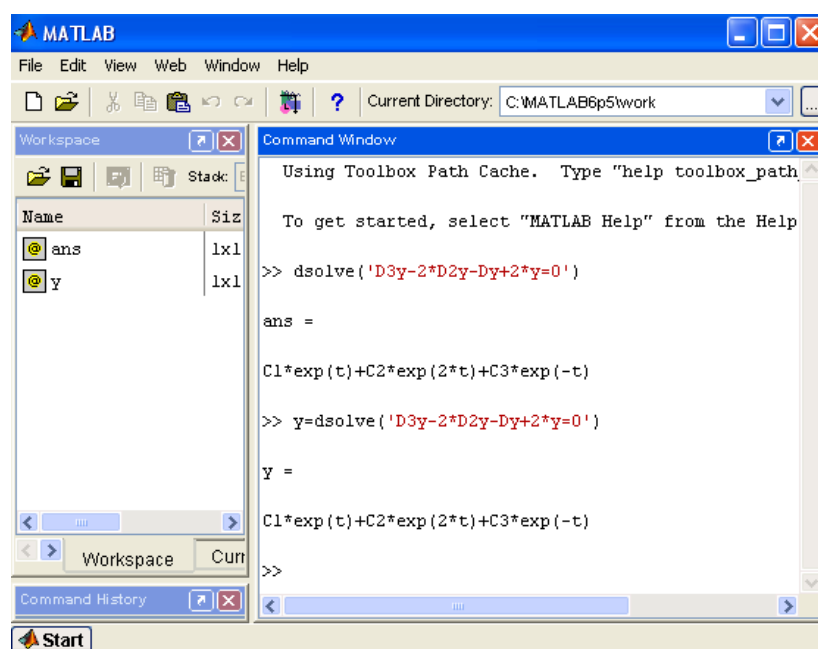


Рис. 5 – Пример использования команды **dsolve**

Моделирование в SIMULINK

Далее описываются некоторые дополнительные возможности SIMULINK по составлению и анализу схем моделирования, а также его взаимодействию с MATLAB. Рассмотрим решение задачи Коши с помощью средств SIMULINK.

Реализацию таких математических моделей в Simulink рассмотрим на ряде примеров.

Пример. Модель физического маятника, находящегося под воздействием экспоненциально-затухающего косинусоидального возмущения. Уравнение движения такого маятника имеет вид:

$$\frac{d^2 y(t)}{dt^2} + a_1 \frac{dy(t)}{dt} + a_2 y(t) = a_3 e^{-a_4 t} \cdot \cos a_5 t,$$

$$y(t)|_{t=0} = y_0,$$

$$\frac{dy(t)}{dt}.$$

Выбрав числовые значения параметров, например: $a_1 = a_2 = 0,1$, $a_3 = -5$, $a_4 = 1$, $a_5 = 0,1$ получим следующее уравнение:

$$y'' + 0,1y' + 0,1y = -5e^{-t} \cdot \cos(0,1t),$$

$$y(0) = -1,5,$$

$$y'(0) = 2.$$

Структурная схема модели будет иметь вид, показанный на рис. 6. Результаты работы модели показан на экране виртуального осциллографа (рис. 7), а параметрический график зависимости производной сигнала от сигнала (фазовый портрет маятника) изображен на рис. 8.

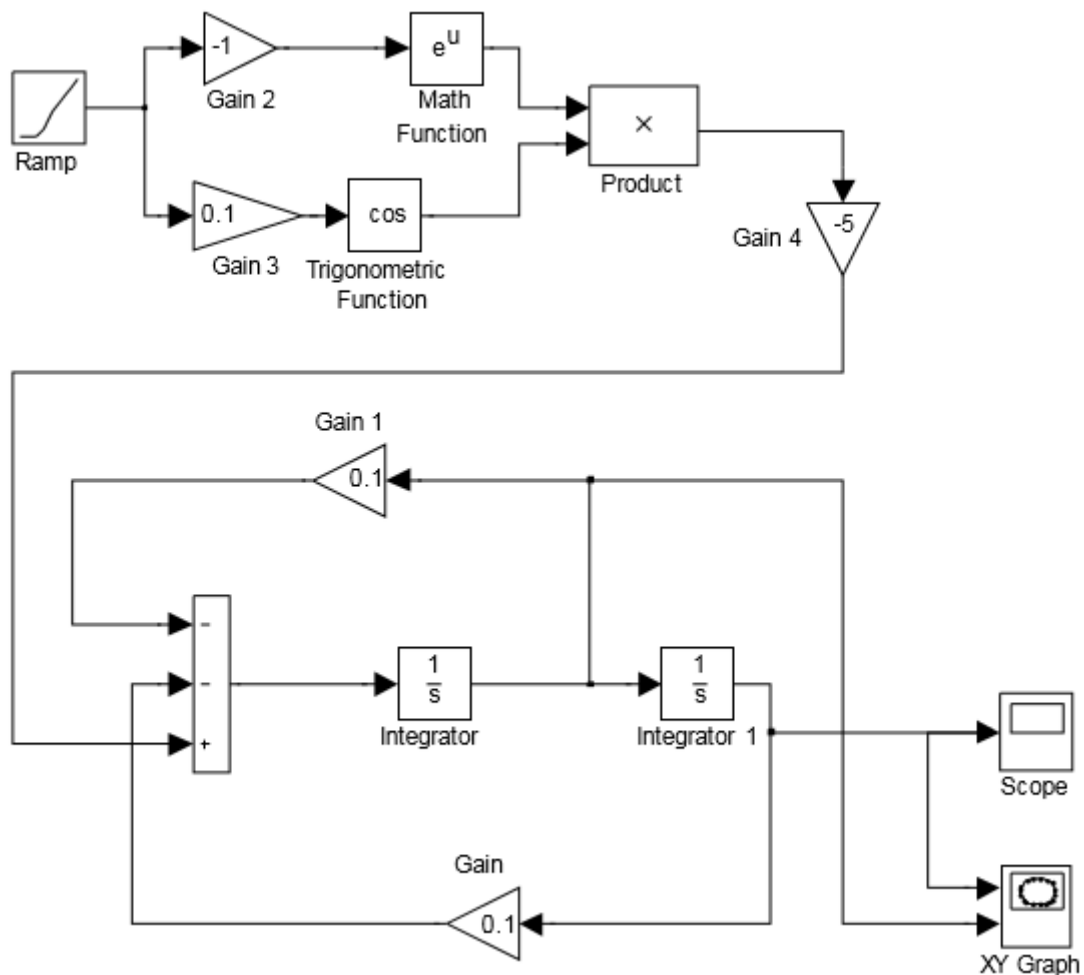


Рис. 6 – Модель физического маятника

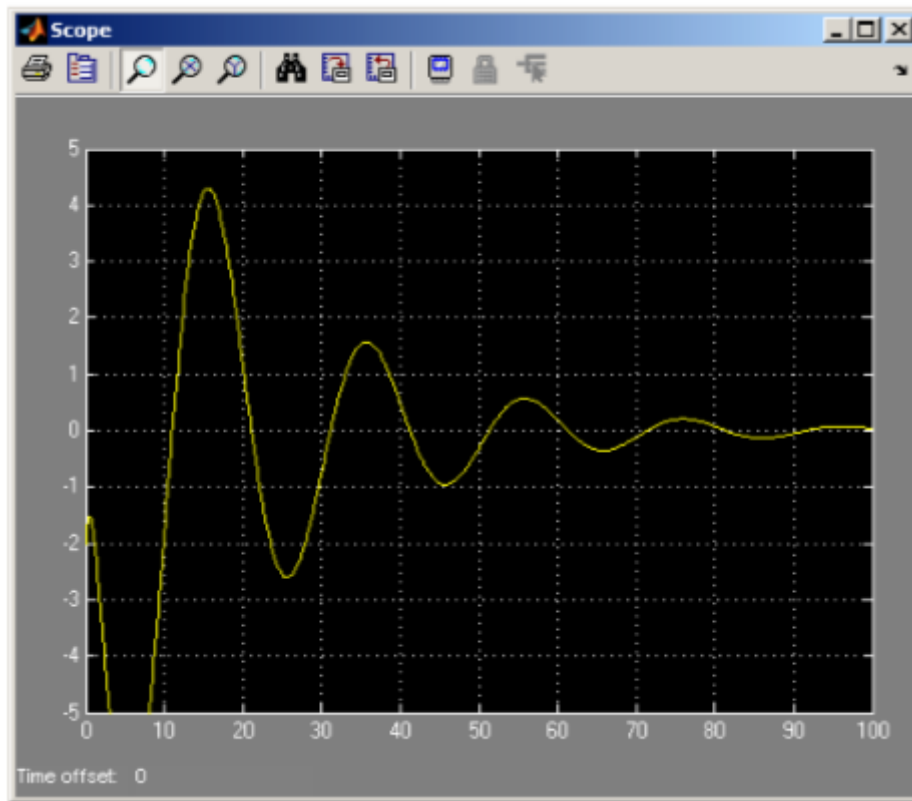


Рис. 7 – Движение физического маятника

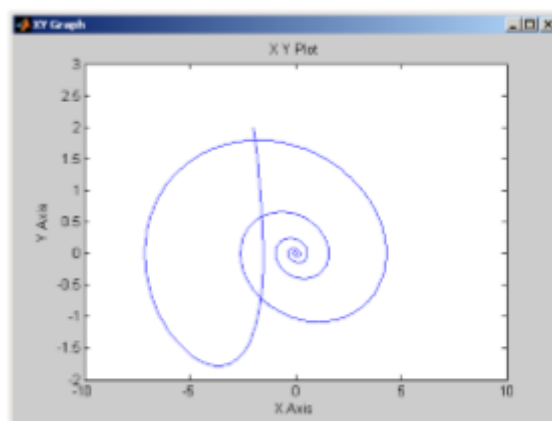


Рис. 8 – Фазовый портрет физического маятника

Редактор дифференциальных уравнений DEE

Эффективным средством для решения нелинейных дифференциальных уравнений с известными начальными условиями является моделирование в SIMULINK. Оно требует построения эквивалентной структурной схемы и ее реализации на стандартных линейных и нелинейных блоках. Определенную помощь в этом процессе может оказать редактор дифференциальных уравнений DEE (Differential Equation Editor). Этот редактор сам строит схему моделирования для SIMULINK по системе дифференциальных уравнений, записанной в форме Коши. Это немного эффективней, чем "ручное" построение схемы на интеграторах, и несколько проще, чем написание блока «с нуля». Редактор вызывается командой `dee`. Появляется окно с несколькими блоками, один из них называется Differential Equation Editor, его надо скопировать (перетащить мышкой) на рабочую страницу SIMULINK, войти в него

двойным щелчком мыши и ввести параметры дифференциального уравнения (левые части, начальные условия и др.). Приведем пример заполнения параметров для моделирования дифференциального уравнения математического маятника:


$$\ddot{y} + \sin y = 0, \quad y(0) = \dot{y}(0) = 0$$

записанного в форме Коши:

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -\sin x_1 \end{aligned}$$

| | | |
|--------------------------------|--------------------|--------|
| Name: | pendulum | |
| # of inputs | 0 | |
| First order equations, f(x,u): | x0 | |
| dx/dt= | x(2) -sin(x(1)) | 1 0 |
| Output equations, f(x,u) | | |
| y= | x(1) x(2) | |

В дальнейшем входить в блок для изменения параметров можно с помощью команды *diffeqed*. Схема моделирования приведена слева на рис. 9. Чтобы одновременно наблюдать графики двух сигналов в блоке *Scope* в параметрах осциллографа устанавливаем поле "Number of axes" равным двум. Параметры

осциллографа можно редактировать, нажав кнопку . Если есть желание взглянуть непосредственно на схему моделирования, соответствующую заданному уравнению, надо, выделив блок, нажать правую кнопку мыши и выбрать пункт меню look under mask, при этом появится схема, показанная справа на рис. 9.

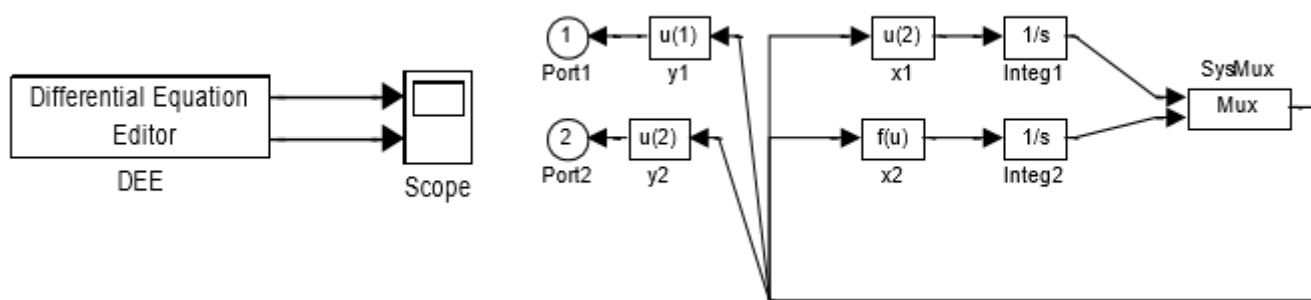


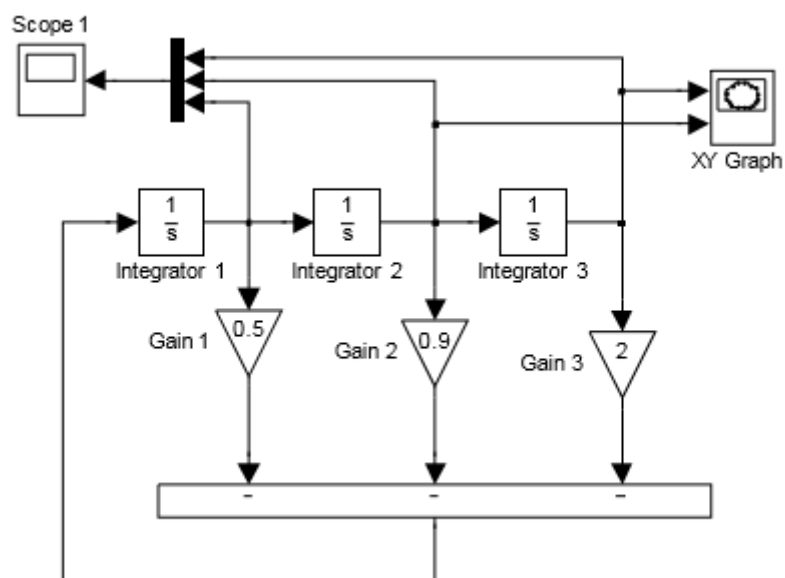
Рис. 9 – Структурная схема модели

Дополнительные примеры

Синтезировать структурную схему модели дифференциального уравнения 3-го порядка:

$$\frac{d^3 y}{dx^3} + 0.5 \frac{d^2 y}{dx^2} + 0.9 \frac{dy}{dx} + 2y = 0$$

Решение:



12

Пример синтезирование структурной схемы модели системы линейных алгебраических уравнений в Simulink:

$$\begin{aligned} 10x_1 + 4x_2 + x_3 &= 10, \\ 4x_1 + 10x_2 + x_3 &= -29, \\ x_1 + 4x_2 + 10x_3 &= -3.5. \end{aligned}$$

На цифровом регистраторе (рис. 13) показано решение системы линейных уравнений ($x_1 = 2.5$, $x_2 = -4$, $x_3 = 1$).

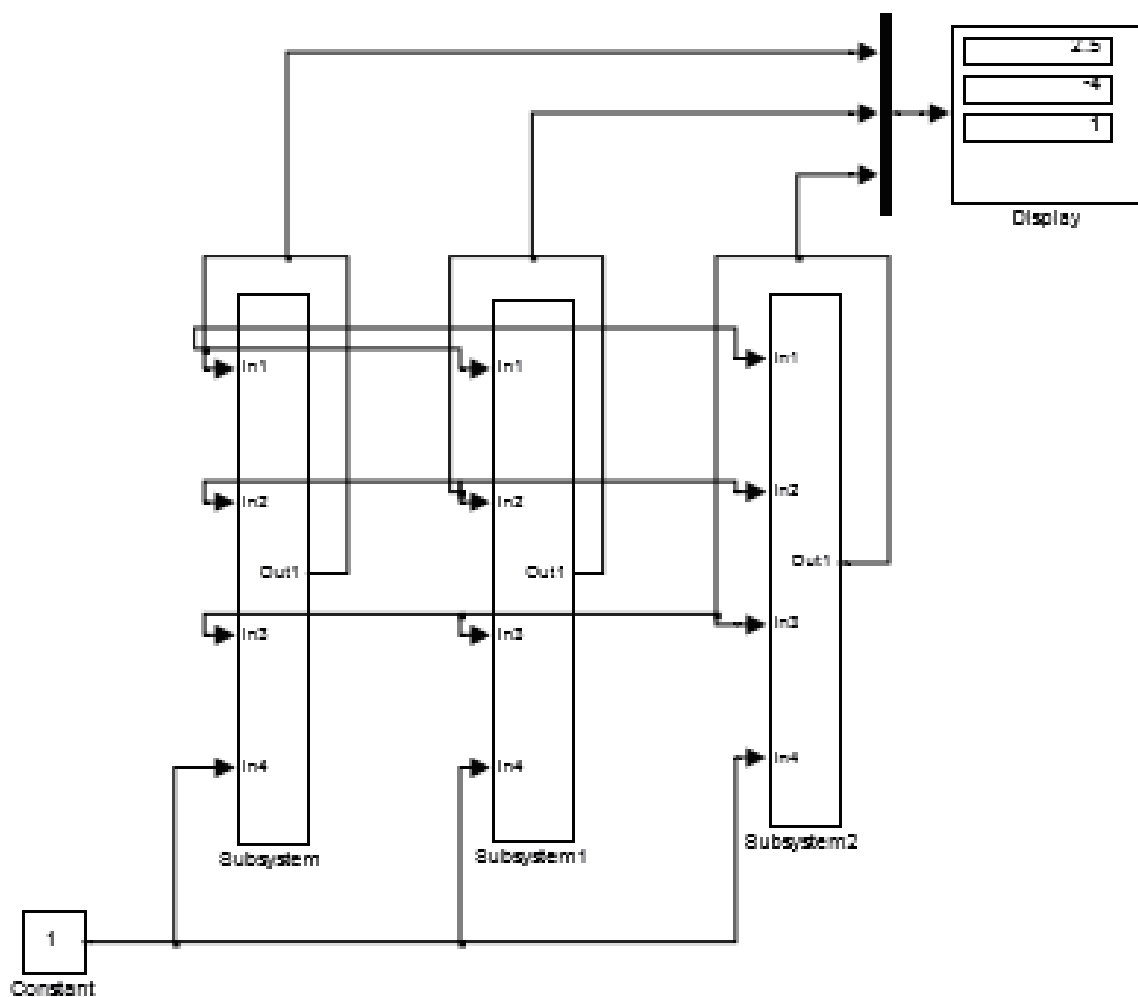


Рис. 10 – Структурная схема модели СЛАУ

II. ПРАКТИЧЕСКАЯ ЧАСТЬ

Требуется для индивидуального задания:

1. Найти аналитическое решение в общем виде заданной динамической системы, которая представлена в виде дифференциального уравнения.
2. Найти частное решение заданного дифференциального уравнения с учетом начальных условий (т.е. решить задачу Коши).
3. Получить численное решение с применением одного из известных солверов Matlab, записанного в виде m-функции.
4. Использовать для построения структурной схемы динамической системы средства Simulink.
5. *Применить DEE-редактор для заданной системы.
6. Построить графики полученных аналитического и численного решений. Выполнить сопоставление графиков решений.
7. Провести сравнительный анализ полученных результатов.
8. Сделать выводы.

Індивідуальні завдання з дисципліни «Моделювання систем»

| № варіанта | Диференційне рівняння | Початкові умови |
|---------------|--|---|
| 1 | $y'' + y' - 2y = \cos x - 3 \sin x$ | $y(0) = 1$ $y'(0) = 2$ |
| 2 | $y'' + 6y' + 10y = 80e^x \cos x$ | $y(0) = 4$ $y'(0) = 10$ |
| 3 | $y'' - 4y' + 3y = e^{5x}$ | $y(0) = 3$ $y'(0) = 9$ |
| 4 | $y'' - 8y' + 16y = e^{4x}$ | $y(0) = 0$ $y'(0) = 1$ |
| 5 | $y'' + y = \cos 3x$ | $y(\frac{\pi}{2}) = 4$ $y'(\frac{\pi}{2}) = 1$ |
| 6 | $2y'' - y' = 1$ | $y(0) = 0$ $y'(0) = 1$ |
| 7 | $y'' + 4y = \sin 2x + 1$ | $y(0) = \frac{1}{4}$ $y'(0) = 0$ |
| 8 | $4y'' + 16y' + 15y = 4e^{-\frac{3}{2}x}$ | $y(0) = 3$ $y'(0) = -5.5$ |
| 9 | $y'' - 2y' + 10y = 10x^2 + 18x + 6$ | $y(0) = 1$ $y'(0) = 3.2$ |
| 10 | $y'' - y' = 2(1 - x)$ | $y(0) = 1$ $y'(0) = 1$ |
| 11 | $y'' - 2y' = e^x(x^2 + x - 3)$ | $y(0) = 2$ $y'(0) = 2$ |
| 12 | $y'' + y = -2 \sin 2x$ | $y(\pi) = 1$ $y'(\pi) = 1$ |
| 13 | $y'' + y' = 4x + 8e^x$ | $y(0) = 0$ $y'(0) = 0$ |
| 14 | $y'' - y' = 4e^x + 2e^{2x}$ | $y(0) = 0$ $y'(0) = 0$ |
| 15 | $y'' - 4y' = 96x^2$ | $y(0) = 0$ $y'(0) = 0$ |

| | | |
|----|--------------------------------|-------------------------------|
| 16 | $y'' - y' - 2y = -3e^x$ | $y(0) = 0$ $y'(0) = 2$ |
| 17 | $y'' + y' = 9x^2$ | $y(0) = 0$ $y'(0) = 0$ |
| 18 | $y'' + 9y = -6 \cos 3x$ | $y(0) = 0$ $y'(0) = 4$ |
| 19 | $y'' + 4y = \sin x$ | $y(0) = 1$ $y'(0) = 1$ |
| 20 | $y'' - 2y' = e^x(x^2 + x - 3)$ | $y(0) = 2$ $y'(0) = 2$ |
| 21 | $y'' + y = -\sin 2x$ | $y(\pi) = 1$ $y'(\pi) = 1$ |
| 22 | $y'' - y' = 2(1 - x)$ | $y(0) = 1$ $y'(0) = 1$ |
| 23 | $y'' - y = x^2$ | $y(0) = -2$ $y'(0) = 1$ |
| 24 | $y'' - 2y' + 2y = 2x$ | $y(0) = 0$ $y'(0) = 0$ |
| 25 | $y'' + 9y = 15 \sin 2x$ | $y(0) = -7$ $y'(0) = 0$ |
| 26 | $y'' - 2y' = x$ | $y(0) = 0$ $y'(0) = 0$ |

Литература

1. Ануфриев И. Е. Самоучитель MatLab 5.3/6.x. -СПб.: БХВ-Петербург, 2003. 736 с.
2. Ануфриев И.Е., Смирнов А.Б., Смирнова Е.Н. MATLAB 7. – СПб.: БХВ – Петербург, 2005. – 1104 с.
3. Васильев В.В., Симак Л.А., Рыбникова А.М. Математическое и компьютерное моделирование процессов и систем в среде MATLAB/SIMULINK. Учебное пособие для студентов и аспирантов / В.В. Васильев, Л.А. Симак, А.М. Рыбникова. – К.: НАН Украины, 2008. – 91 с.
4. Дьяконов В., Круглов В. Математические пакеты расширения MATLAB. -СПб.: Питер, 2001. 560 с.
5. Дьяконов В.П. MATLAB 6/6.1/6.5 + SIMULINK 4/5 в математике и моделировании. -М.: Солон-Пресс, 2003. 576 с.
6. Дзэбни Дж., Хароган Т. SIMULINK 4. Секреты мастерства. -М.: Бином, Лаборатория знаний. 2003. 403 с.
7. Кетков Ю., Кетков А., Шульц М. MATLAB 6.x: программирование численных методов. - СПб.: БХВ-Петербург, 2004. 742 с.
8. Мироновский Л. А. Моделирование линейных систем. Учеб. пособие с грифом УМО. СПб. ГУАП, 2009. – 244 с.
9. Лазарев Ю.Ф. MatLAB 5.x. – К.: Издательская группа BHV, 2000. – 384 с.
10. Мироновский Л.А., Петрова К.Ю. Введение в MATLAB: Учеб. пособие. СПб., ГУАП. 2006. 163с.
11. Конев В.Ю., Мироновский Л.А. Основные функции пакета MATLAB. Учеб. пособие. СПб. ГААП.1992,75с.; 1994. 79с.
12. Мироновский Л.А. Моделирование динамических систем. Учеб. пособие. СПб., ГААП. 1992. 92 с.
13. Наместников А.М. Разработка имитационных моделей в среде MATLAB: Методические указания для студентов специальностей 01719, 351400 / Ульяновск, УлГТУ, 2004. – 72 с.
14. Половко А.М., Бутусов П.Н. Matlab для студентов. - СПб: БХВ-Петербург, 2005. 320 с.
15. Потемкин В. Г. MATLAB 6: среда проектирования инженерных приложений. М.: Диалог-МИФИ, 2003. 448 с.
16. Сергиенко А. Цифровая обработка сигналов. -СПб.: Питер, 2002. 606 с.
17. Терёхин В.В. Моделирование в системе MATLAB: Учебное пособие Часть 1. Основы работы в MATLAB /Кемеровский государственный университет. – Новокузнецк: Кузбассвуиздат, 2005. -376с.
18. Терёхин В.В. Моделирование в системе MATLAB: Учебное пособие Часть 2. Simulink /Кемеровский государственный университет. – Новокузнецк: Кузбассвуиздат, 2004. -376 с.