

Отчёт по лабораторной работе 4

Модель гармонических колебаний

Аристова Арина Олеговна

Содержание

1	Цель работы	5
2	Задание	6
2.1	Вариант 4	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	10
5	Анализ полученных результатов	23
6	Выводы	24
	Список литературы. Библиография	25

Список иллюстраций

4.1	Определение варианта.	10
4.2	Случай 1. Решение, полученное на Julia	12
4.3	Случай 1. Фазовый портрет, полученный на Julia	12
4.4	Случай 2. Решение, полученное на Julia	14
4.5	Случай 2. Фазовый портрет, полученный на Julia	15
4.6	Случай 3. Решение, полученное на Julia	17
4.7	Случай 3. Фазовый портрет, полученный на Julia	17
4.8	Случай 1. Решение, полученное на Modelica	19
4.9	Случай 1. Фазовый портрет, полученный на Modelica	19
4.10	Случай 2. Решение, полученное на Modelica	20
4.11	Случай 2. Фазовый портрет, полученный на Modelica	21
4.12	Случай 3. Решение, полученное на Modelica	22
4.13	Случай 3. Фазовый портрет, полученный на Modelica	22

Список таблиц

1 Цель работы

- Изучить понятие гармонического осциллятора,
- Построить фазовый портрет гармонического осциллятора,
- Найти решение уравнения гармонического осциллятора.

2 Задание

2.1 Вариант 4

Постройте фазовый портрет гармонического осциллятора и решение уравнения гармонического осциллятора для следующих случаев

1. Колебания гармонического осциллятора без затуханий и без действий внешней силы $\ddot{x} + 15x = 0$
2. Колебания гармонического осциллятора с затуханием и без действий внешней силы $\ddot{x} + 10\dot{x} + x = 0$
3. Колебания гармонического осциллятора с затуханием и под действием внешней силы $\ddot{x} + 3\dot{x} + x = \sin(3t)$. На интервале $t \in [0; 55]$ (шаг 0.05) с начальными условиями $x_0 = 0, y_0 = 2$

3 Теоретическое введение

Julia – высокоуровневый язык, который разработан для научного программирования. Язык поддерживает широкий функционал для математических вычислений и работы с большими массивами данных[1].

OpenModelica — свободное открытое программное обеспечение для моделирования, симуляции, оптимизации и анализа сложных динамических систем. Основано на языке Modelica. Активно развивается Open Source Modelica Consortium, некоммерческой неправительственной организацией. Open Source Modelica Consortium является совместным проектом RISE SICS East AB и Линчёпингского университета. По своим возможностям приближается к таким вычислительным средам как Matlab Simulink, Scilab xCos, имея при этом значительно более удобное представление системы уравнений исследуемого блока [2].

- Осциллятор (лат. *oscillo* — качаюсь) — система, совершающая колебания, то есть показатели которой периодически повторяются во времени.
- Гармонический осциллятор [3] — система, которая при смещении из положения равновесия испытывает действие возвращающей силы F , пропорциональной смещению x .
- Гармоническое колебание [4] - колебание, в процессе которого величины, характеризующие движение (смещение, скорость, ускорение и др.), изменяются по закону синуса или косинуса (гармоническому закону).

Движение груза на пружинке, маятника, заряда в электрическом контуре, а также эволюция во времени многих систем в физике, химии, биологии и других науках при определенных предположениях можно описать одним и тем же дифференциальным уравнением, которое в теории колебаний выступает в качестве основной модели. Эта модель называется линейным гармоническим осциллятором.

Уравнение свободных колебаний гармонического осциллятора имеет следующий вид:

(1):

$$\ddot{x} + 2\gamma\dot{x} + \omega_0^2 = 0$$

где x - переменная, описывающая состояние системы (смещение груза, заряд конденсатора и т.д.), γ - параметр, характеризующий потери энергии (трение в механической системе, сопротивление в контуре), ω_0 - собственная частота колебаний, t - время. (Обозначения

$$\ddot{x} = \frac{\delta^2 x}{\delta t^2}, \dot{x} = \frac{\delta x}{\delta t}$$

)

Уравнение (1) есть линейное однородное дифференциальное уравнение второго порядка и оно является примером линейной динамической системы. При отсутствии потерь в системе ($\gamma = 0$) вместо уравнения (1.1) получаем уравнение консервативного осциллятора энергия колебания которого сохраняется во времени.

(2):

$$\ddot{x} + \omega_0^2 x = 0$$

Для однозначной разрешимости уравнения второго порядка (2) необходимо задать два начальных условия вида:

(3):

$$\begin{cases} x(t_0) = x_0 \\ \dot{x}(t_0) = y_0 \end{cases}$$

Уравнение второго порядка (2) можно представить в виде системы двух уравнений первого порядка:

(4):

$$\begin{cases} \dot{x} = y \\ \dot{y} = -\omega_0^2 x \end{cases}$$

Начальные условия (3) для системы (4) примут вид:

(5):

$$\begin{cases} x(t_0) = x_0 \\ y(t_0) = y_0 \end{cases}$$

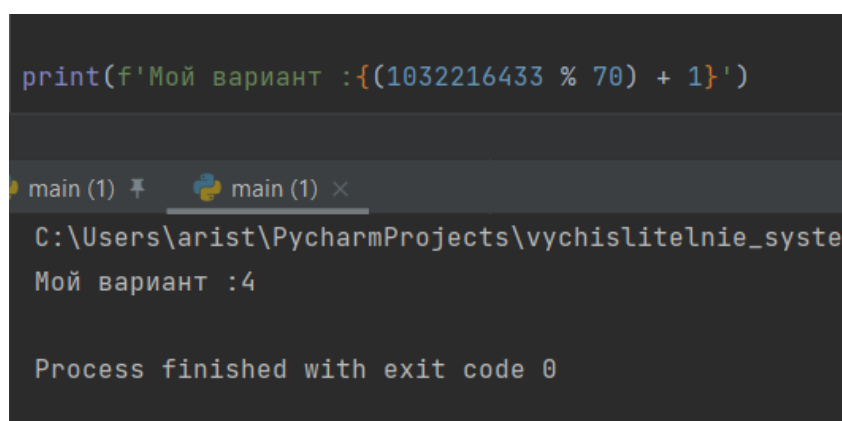
Независимые переменные x, y определяют пространство, в котором «движется» решение. Это фазовое пространство системы, поскольку оно двумерно будем называть его фазовой плоскостью.

Значение фазовых координат x, y в любой момент времени полностью определяет состояние системы. Решению уравнения движения как функции времени отвечает гладкая кривая в фазовой плоскости. Она называется фазовой траекторией. Если множество различных решений (соответствующих различным начальным условиям) изобразить на одной фазовой плоскости, возникает общая картина поведения системы.

Такую картину, образованную набором фазовых траекторий, называют фазовым портретом.

4 Выполнение лабораторной работы

Мой вариант лабораторной работы: 4. Я получила его по заданной формуле:



```
print(f'Мой вариант :{(1032216433 % 70) + 1}')
```

main (1) ×

C:\Users\arist\PycharmProjects\vychislitelnie_syste

Мой вариант :4

Process finished with exit code 0

Рис. 4.1: Определение варианта.

Затем я написала 3 программы для каждого из случаев на языке Julia:

Вот листинг первой программы для случая **без затуханий и без действий внешней силы**:

$$\ddot{x} + 15x = 0$$

```
using Plots; gr()
using DifferentialEquations;

# Случай 1:  $x'' + 15x = 0$ 

function lorenz!(du, u, p, t)
    a = p;
```

```

    du[1] = u[2];
    du[2] = -a*u[1]
end

# Задаем начальные условия
const x = 0;
const y = 2;
u0 = [x, y]

p = 15
tspan = [0, 55]

# Задаем задачу
problem = ODEProblem(lorenz!, u0, tspan, p)

# Решение данной задачи

solution = solve(problem, dtmax = 0.05)

# Создаю холст 1
plot(solution)

# Сохраняю результат в файл
savefig("lab04_1_julia.png")

# Создаю холст 2
plot(solution, vars=(2,1))

# Сохраняю результат в файл

```

```
savefig("lab04_1_pp_julia.png")
```

Полученный результат:

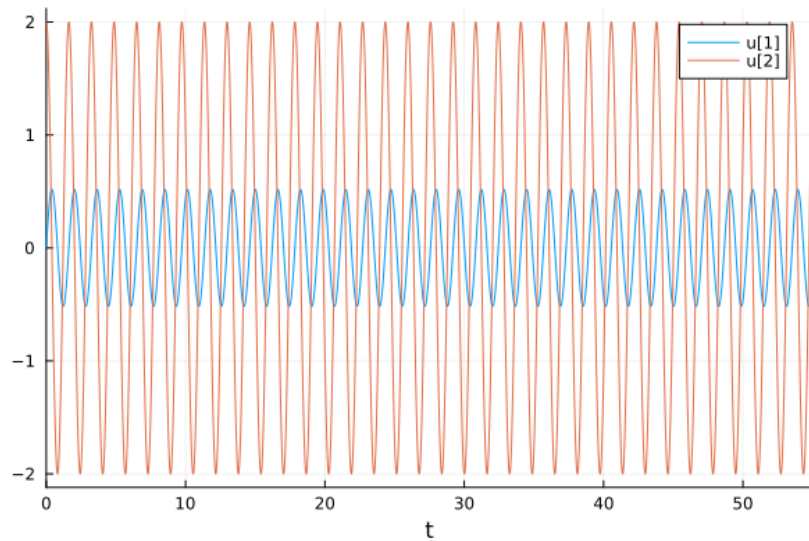


Рис. 4.2: Случай 1. Решение, полученное на Julia

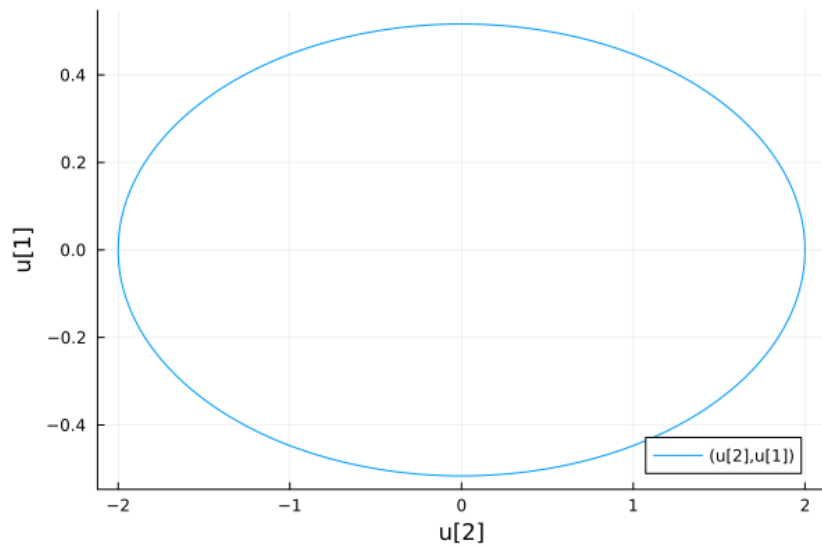


Рис. 4.3: Случай 1. Фазовый портрет, полученный на Julia

Вот листинг второй программы для случая **с затуханием и без действий внешней силы**:

$$\ddot{x} + 10\dot{x} + x = 0$$

```

using Plots; gr()
using DifferentialEquations;

# Случай 2:  $x'' + 10x' + x = 0$ 

function lorenz!(du, u, p, t)
    a, b = p;
    du[1] = u[2];
    du[2] = -a*du[1] -b*u[1]
end

# Задаем начальные условия
const x = 0;
const y = 2;
u0 = [x, y]

p = (sqrt(10), 1)
tspan = (0.0, 55.0)

# Задаем задачу
problem = ODEProblem(lorenz!, u0, tspan, p)

# Решение данной задачи

solution = solve(problem, dtmax = 0.05)

# Создаю холст 1
plot(solution)

```

```
# Сохраняю результат в файл
savefig("lab04_2_julia.png")

# Создаю холст 2
plot(solution, vars=(2,1))

# Сохраняю результат в файл
savefig("lab04_2_pp_julia.png")
```

Полученный результат:

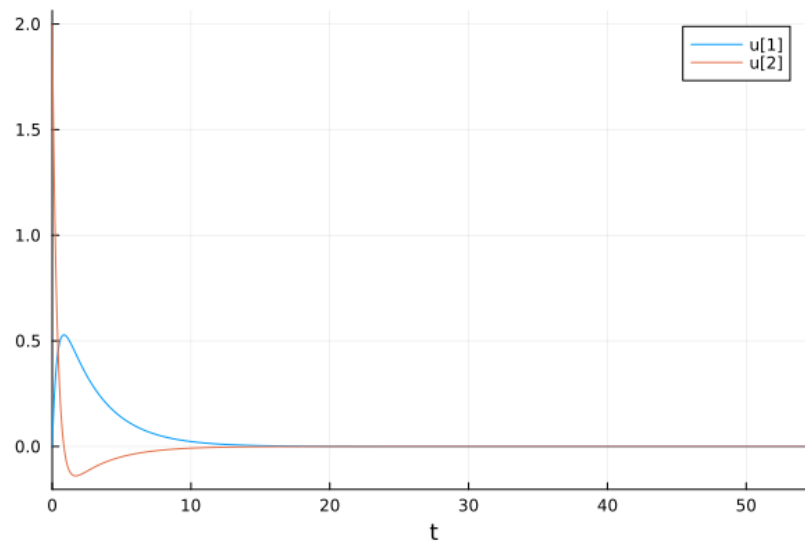


Рис. 4.4: Случай 2. Решение, полученное на Julia

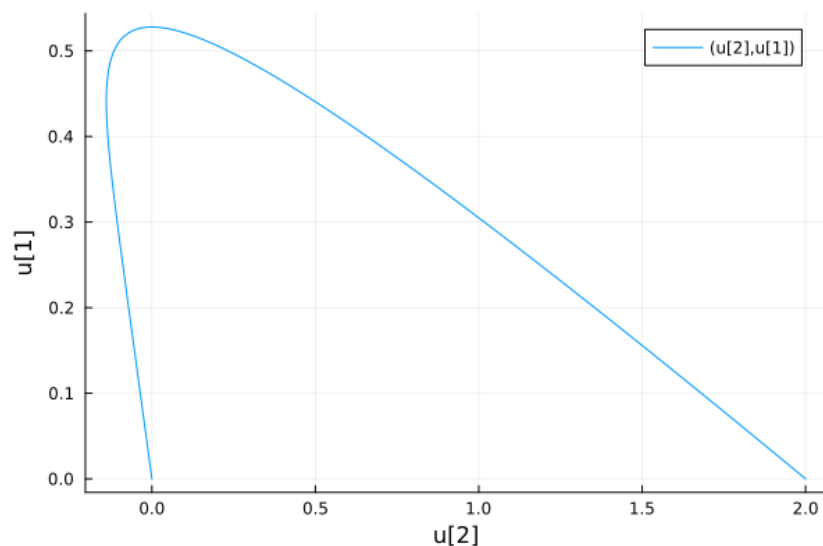


Рис. 4.5: Случай 2. Фазовый портрет, полученный на Julia

Вот листинг третьей программы для случая **с затуханием и под действием внешней силы**:

$$\ddot{x} + 3\dot{x} + x = \sin(3t)$$

```
using Plots; gr()
using DifferentialEquations;

# Случай 3:  $x'' + 3x' + x = \sin(3t)$ 

function lorenz!(du, u, p, t)
    a, b = p;
    du[1] = u[2];
    du[2] = -a*du[1] - b*u[1] + sin(3*t)
end

# Задаем начальные условия
const x = 0;
const y = 2;
```

```
u0 = [x, y]

p = (sqrt(3), 1)
tspan = (0.0, 55.0)

# Задаем задачу
problem = ODEProblem(lorenz!, u0, tspan, p)

# Решение данной задачи

solution = solve(problem, dtmax = 0.05)

# Создаю холст 1
plot(solution)

# Сохраняю результат в файл
savefig("lab04_3_julia.png")

# Создаю холст 2
plot(solution, vars=(2,1))

# Сохраняю результат в файл
savefig("lab04_3_pp_julia.png")
```

Полученный результат:

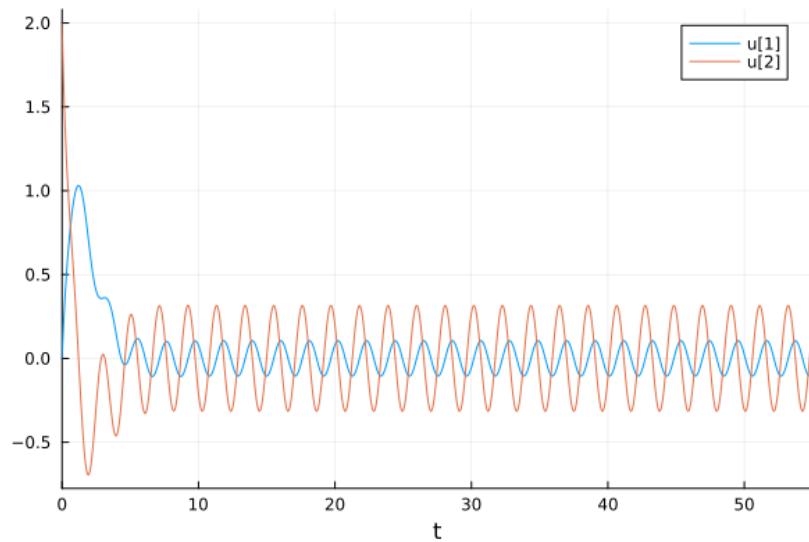


Рис. 4.6: Случай 3. Решение, полученное на Julia

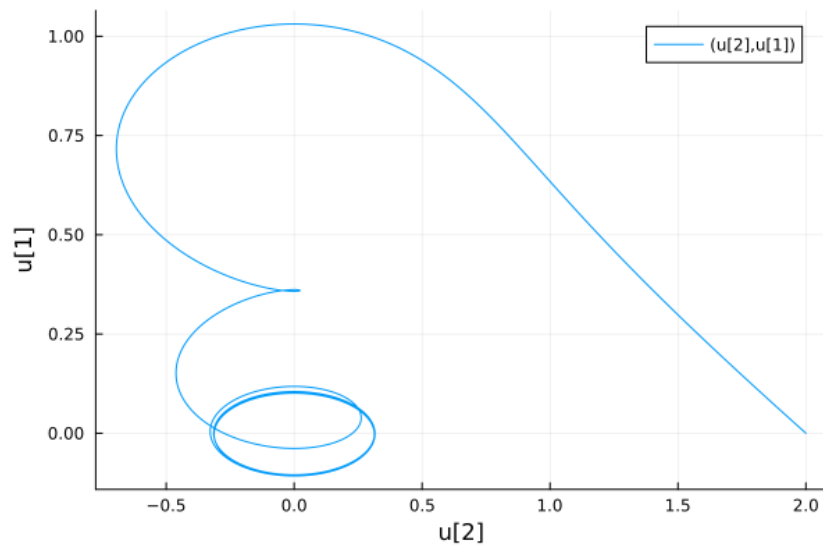


Рис. 4.7: Случай 3. Фазовый портрет, полученный на Julia

Затем я написала 3 программы для каждого из случаев для получения решений на языке Modelica в OpenModelica:

Вот листинг первой программы для случая **без затуханий и без действий внешней силы**:

$$\ddot{x} + 15x = 0$$

//Задание 1: $x'' + 15x = 0$

```

model lab4_1
//x'' + g* x' + w^2* x = f(t)
//w - частота
//g - затухание
parameter Real w = sqrt(15.0);
parameter Real g =0;

// Начальные условия
parameter Real x0 = 0;
parameter Real y0 = 2;

Real x(start=x0);
Real y(start=y0);

// Внешнее воздействие
function f
input Real t ;
output Real res;
algorithm
res := 0;
end f;

equation
der(x) = y;
der(y) = -w*w*x - g*y + f(time);
end lab4_1;

```

Полученный результат:

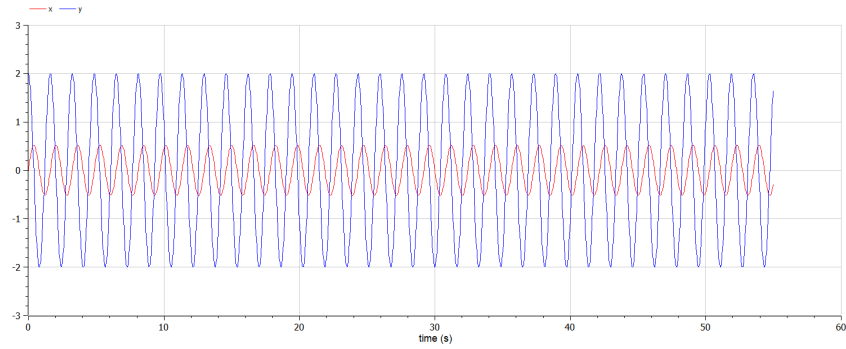


Рис. 4.8: Случай 1. Решение, полученное на Modelica

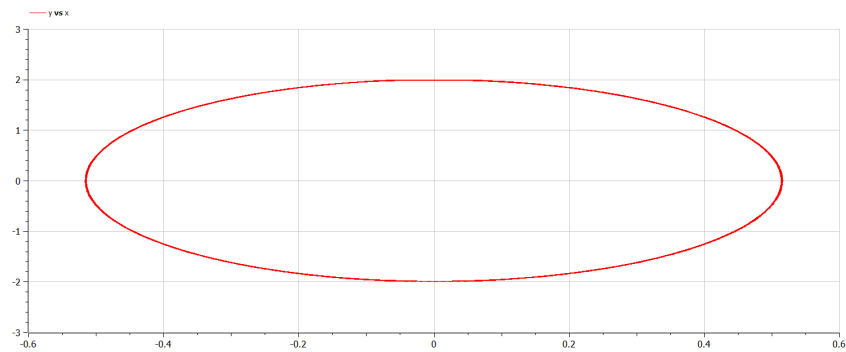


Рис. 4.9: Случай 1. Фазовый портрет, полученный на Modelica

Вот листинг второй программы для случая **с затуханием и без действий внешней силы**:

$$\ddot{x} + 10\dot{x} + x = 0$$

```
//Задание 2:  $x'' + 10x' + x = 0$   
model lab4_2
```

```
//Параметры осциллятора  
// $x'' + g \cdot x' + w^2 \cdot x = f(t)$   
//w - частота  
//g - затухание  
parameter Real w = sqrt(1);  
parameter Real g = 10;
```

```

parameter Real x0 = 0;
parameter Real y0 = 2;

Real x(start=x0);
Real y(start=y0);

// f(t)
function f
input Real t ;
output Real res;
algorithm
res := 0;
end f;

equation
der(x) = y;
der(y) = -w*w*x - g*y + f(time);

end lab4_2;

```

Полученный результат:

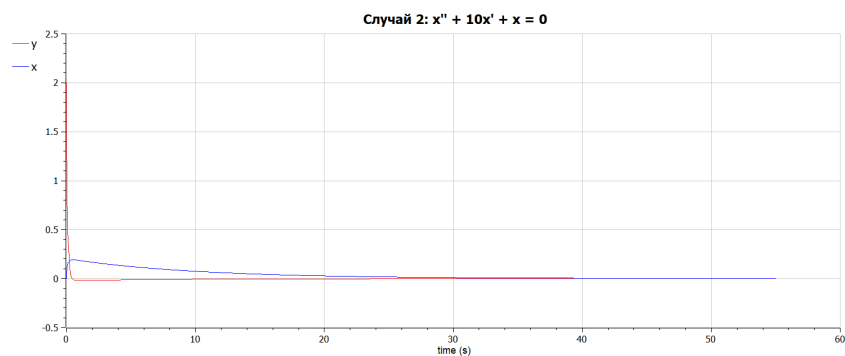


Рис. 4.10: Случай 2. Решение, полученное на Modelica

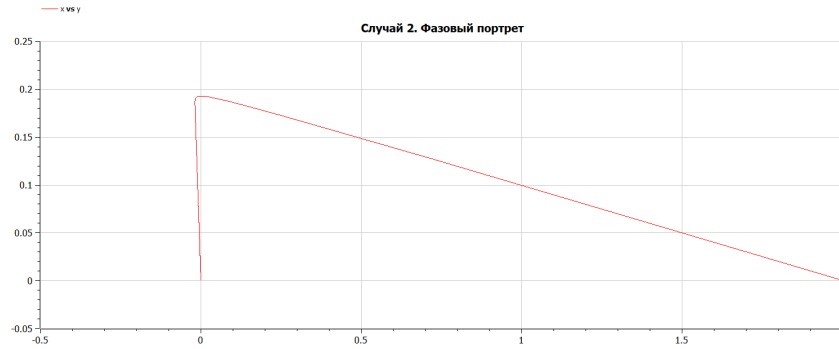


Рис. 4.11: Случай 2. Фазовый портрет, полученный на Modelica

Вот листинг третьей программы для случая с затуханием и под действием внешней силы:

$$\ddot{x} + 3\dot{x} + x = \sin(3t)$$

```
//Задание3:  $x'' + 3x' + x = \sin(3t)$ 
```

```
model lab4_3
```

```
//Параметры осциллятора
```

```
// $x'' + g \cdot x' + w^2 \cdot x = f(t)$ 
```

```
//w - частота
```

```
//g - затухание
```

```
parameter Real w = sqrt(1);
```

```
parameter Real g = 3;
```

```
// Начальные условия
```

```
parameter Real x0 = 0;
```

```
parameter Real y0 = 2;
```

```
Real x(start=x0);
```

```
Real y(start=y0);
```

```
// Внешнее воздействие
```

```

function f
input Real t ;
output Real res;
algorithm
res := sin(3*t);
end f;

equation
der(x) = y;
der(y) = -w*w*x - g*y - f(time);
end lab4_3;

```

Полученный результат:

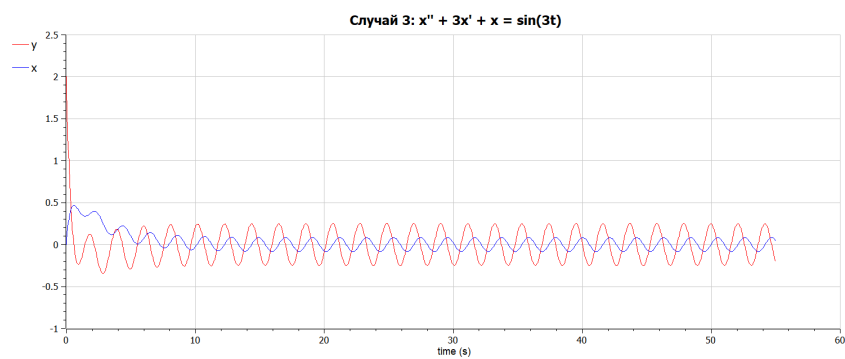


Рис. 4.12: Случай 3. Решение, полученное на Modelica

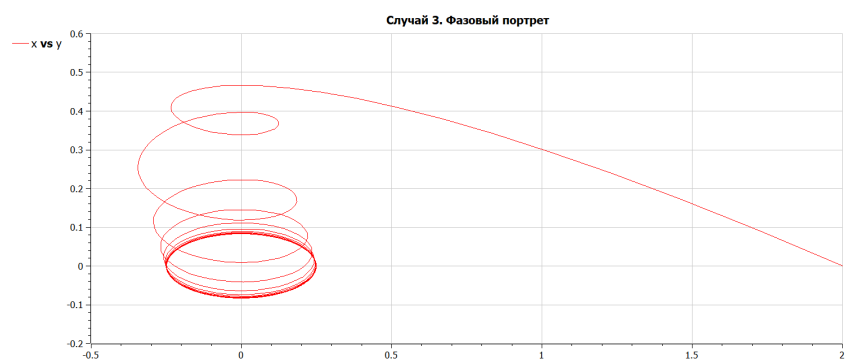


Рис. 4.13: Случай 3. Фазовый портрет, полученный на Modelica

5 Анализ полученных результатов

В результате выполнения работы я построила по три модели (включающих в себя графики решения и фазовой модели системы) на языках Julia и OpenModelica. По итогам выполнения задания, можно сделать вывод о том, что построение моделей колебания на языке Modelica занимает меньше строк, чем аналогичное построение на Julia. Поэтому удобнее делать это, скорее, на первом, нежели на втором.

6 Выводы

В процессе и результате выполнения лабораторной работы я построила три графика решения различных уравнений гармонического осциллятора и три фазовых портрета системы при гармонических колебаниях без затухания; с затуханием и без действия внешней силы; с затуханием и при действии внешней силы на языках Julia и Open Modelica.

Список литературы. Библиография

- [1] Документация по Julia: <https://docs.julialang.org/en/v1/>
- [2] Документация по OpenModelica: <https://openmodelica.org/>
- [3] Решение дифференциальных уравнений: http://www.mathprofi.ru/differencialnye_uravnen
- [4] Справка по осциллятору: <https://ru.wikipedia.org/wiki/%D0%9E%D1%81%D1%86%D0%B8%>