

# **Лабораторная работа №4**

**Модель гармонических колебаний**

Аникин Константин Сергеевич

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>7</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>8</b>
<b>5</b>	<b>Выводы</b>	<b>23</b>
	<b>Список литературы</b>	<b>24</b>

## Список иллюстраций

4.1	Код первого случая на Julia . . . . .	8
4.2	Решение первого случая на Julia . . . . .	9
4.3	Портрет первого случая на Julia . . . . .	10
4.4	Код второго случая на Julia . . . . .	11
4.5	Решение второго случая на Julia . . . . .	12
4.6	Портрет второго случая на Julia . . . . .	13
4.7	Код третьего случая на Julia . . . . .	14
4.8	Решение третьего случая на Julia . . . . .	15
4.9	Портрет третьего случая на Julia . . . . .	16
4.10	Код первого случая на OpenModelica . . . . .	17
4.11	Решение первого случая на OpenModelica . . . . .	17
4.12	Портрет первого случая на OpenModelica . . . . .	18
4.13	Код второго случая на OpenModelica . . . . .	18
4.14	Решение второго случая на OpenModelica . . . . .	19
4.15	Портрет второго случая на OpenModelica . . . . .	20
4.16	Код третьего случая на OpenModelica . . . . .	20
4.17	Решение третьего случая на OpenModelica . . . . .	21
4.18	Портрет третьего случая на OpenModelica . . . . .	22

## Список таблиц

# 1 Цель работы

Построить модель гармонических колебаний в трёх случаях в Julia и OpenModelica.

## 2 Задание

Вариант 6

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

1. Колебания гармонического осциллятора без затуханий и без действий внешней силы
2. Колебания гармонического осциллятора с затуханием и без действий внешней силы
3. Колебания гармонического осциллятора с затуханием и под действием внешней силы

### 3 Теоретическое введение

Движение грузика 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.

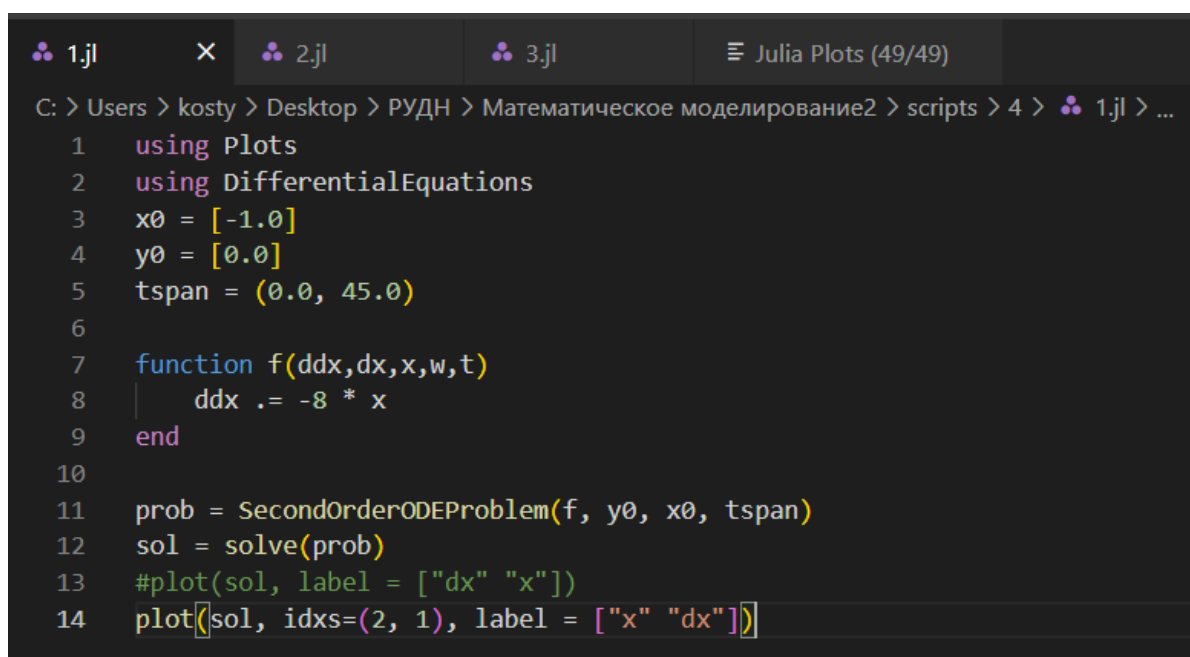
Независимые переменные  $x, y$  определяют пространство, в котором «движется» решение. Это фазовое пространство системы, поскольку оно двумерно будем называть его фазовой плоскостью.

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.

Более подробно о модели гармонического осциллятора см. в [1].

## 4 Выполнение лабораторной работы

На рис. 4.1 представлен код, реализованный на Julia, для первого случая. На рис. 4.2 и рис. 4.3 представлены график решения и фазовый портрет.



```
1.jl × 2.jl 3.jl Julia Plots (49/49)
C: > Users > kosty > Desktop > РУДН > Математическое моделирование2 > scripts > 4 > 1.jl > ...
1 using Plots
2 using DifferentialEquations
3 x0 = [-1.0]
4 y0 = [0.0]
5 tspan = (0.0, 45.0)
6
7 function f(ddx,dx,x,w,t)
8     ddx .= -8 * x
9 end
10
11 prob = SecondOrderODEProblem(f, y0, x0, tspan)
12 sol = solve(prob)
13 #plot(sol, label = ["dx" "x"])
14 plot(sol, idxs=(2, 1), label = ["x" "dx"])
```

Рис. 4.1: Код первого случая на Julia



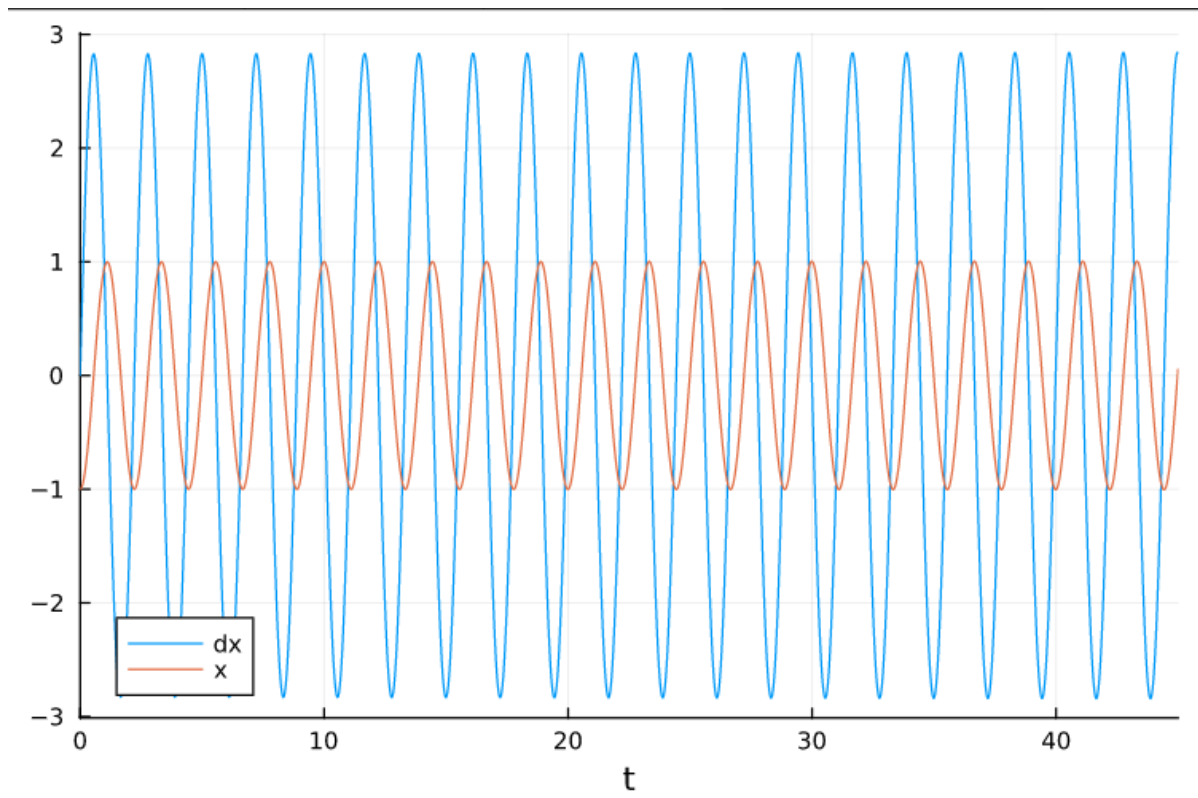


Рис. 4.2: Решение первого случая на Julia

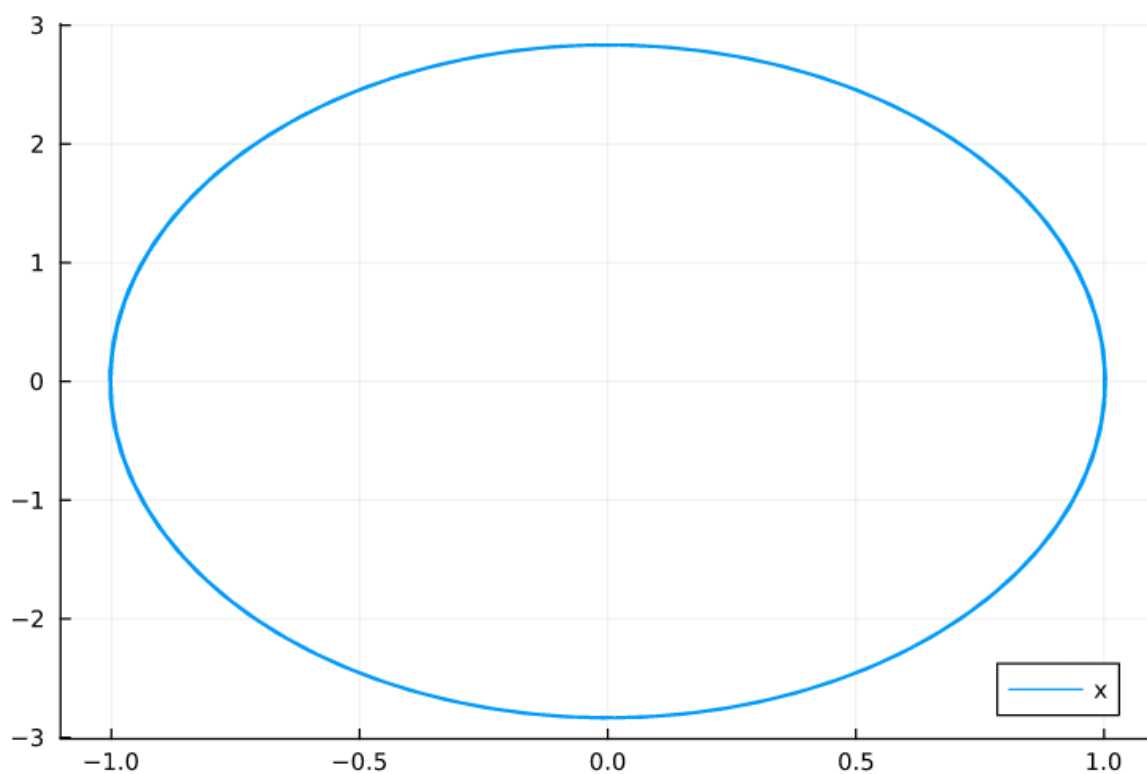
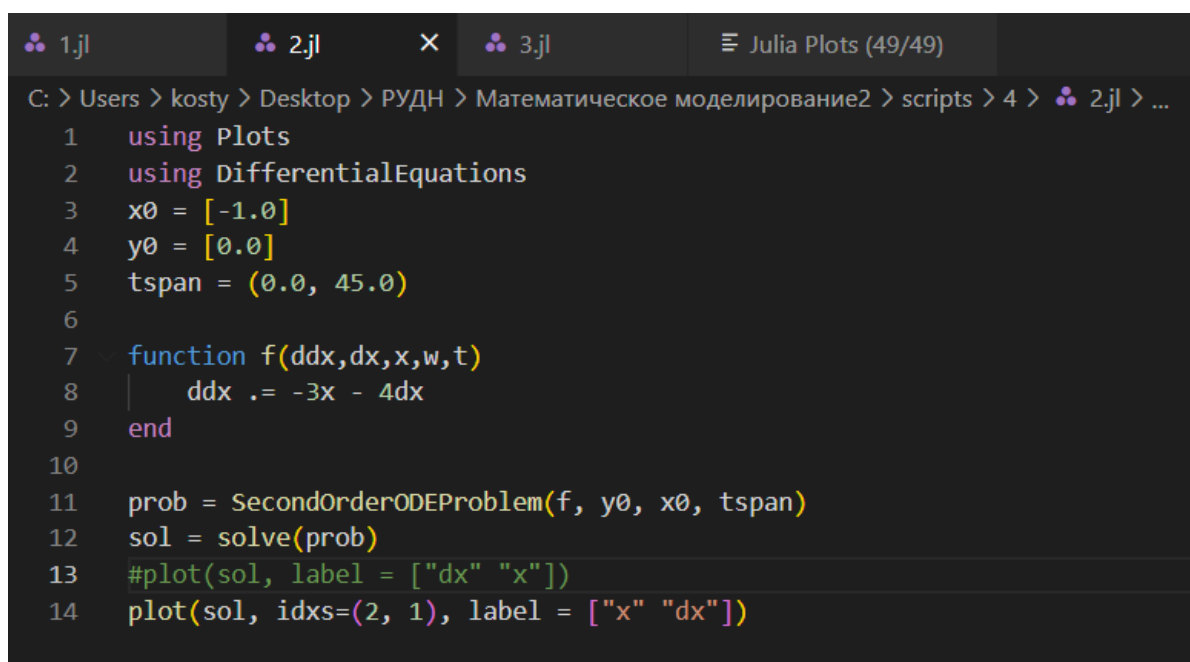


Рис. 4.3: Портрет первого случая на Julia

На рис. 4.4 представлен код, реализованный на Julia, для второго случая. На рис. 4.5 и рис. 4.6 представлены график решения и фазовый портрет.



```
1.jl 2.jl X 3.jl Julia Plots (49/49)
C: > Users > kosty > Desktop > РУДН > Математическое моделирование2 > scripts > 4 > 2.jl > ...
1 using Plots
2 using DifferentialEquations
3 x0 = [-1.0]
4 y0 = [0.0]
5 tspan = (0.0, 45.0)
6
7 function f(ddx,dx,x,w,t)
8     ddx .= -3x - 4dx
9 end
10
11 prob = SecondOrderODEProblem(f, y0, x0, tspan)
12 sol = solve(prob)
13 #plot(sol, label = ["dx" "x"])
14 plot(sol, idxs=(2, 1), label = ["x" "dx"])
```

Рис. 4.4: Код второго случая на Julia

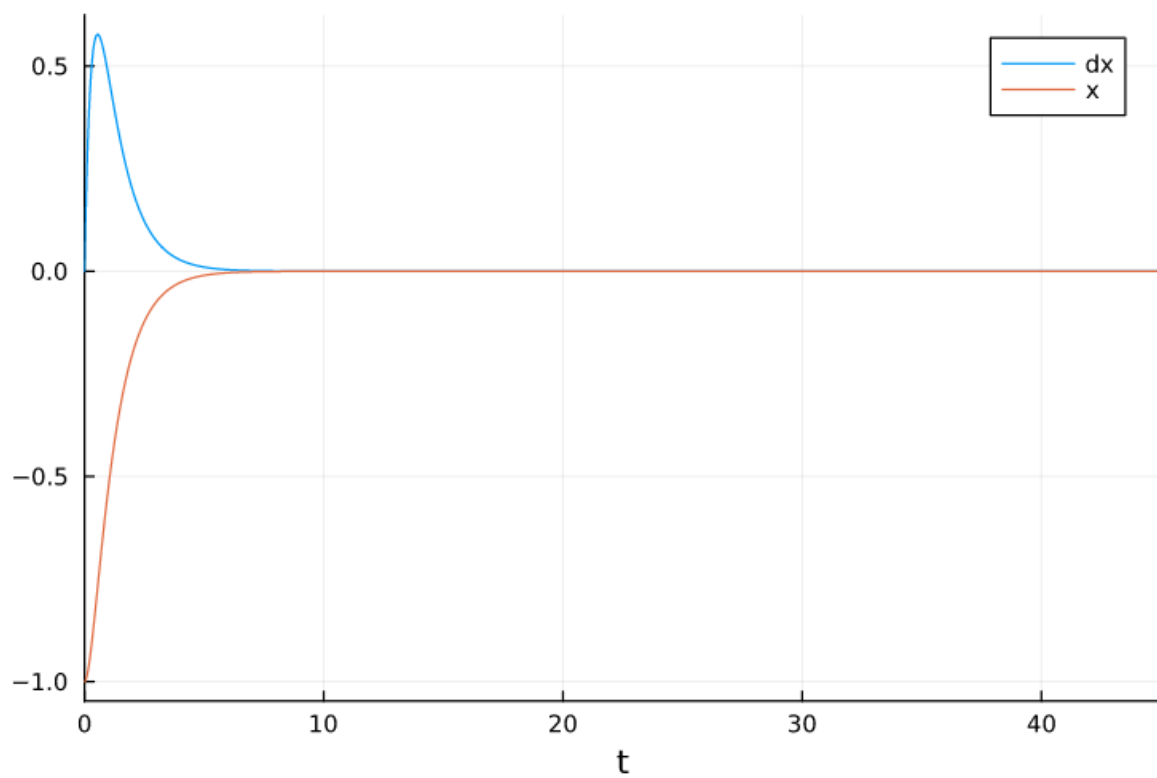


Рис. 4.5: Решение второго случая на Julia

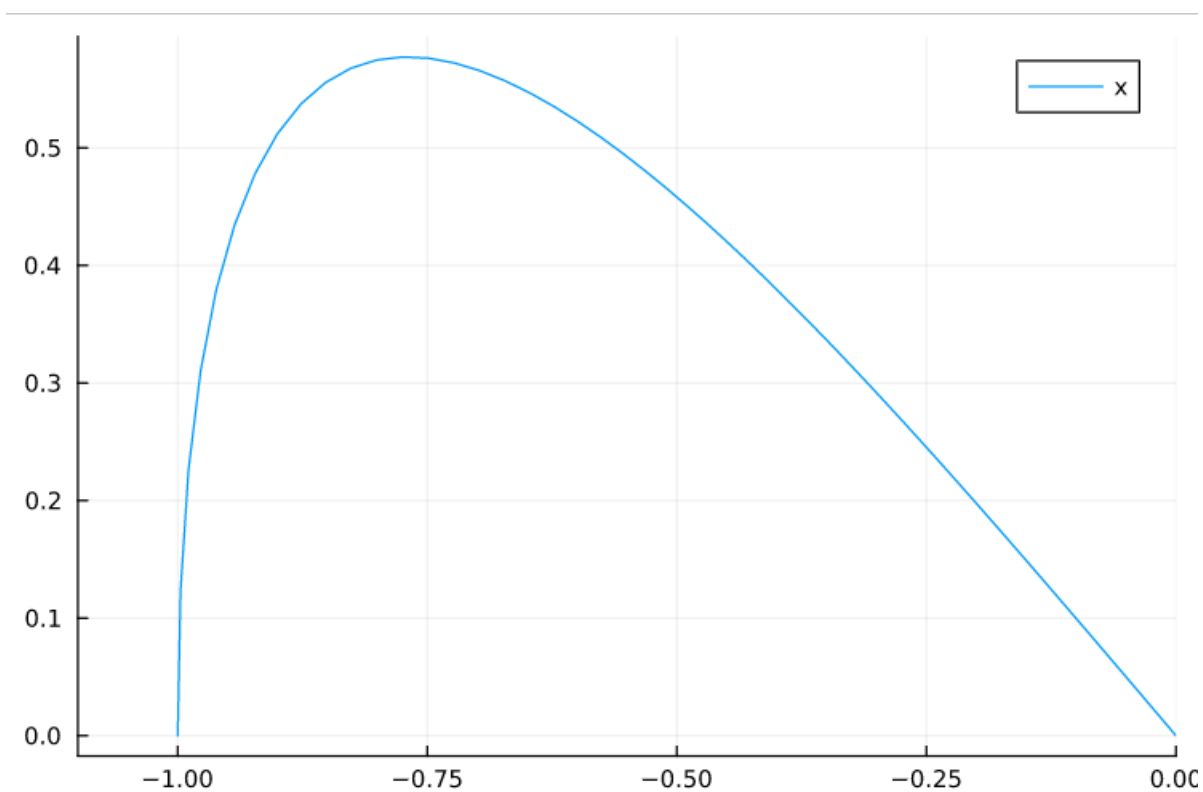
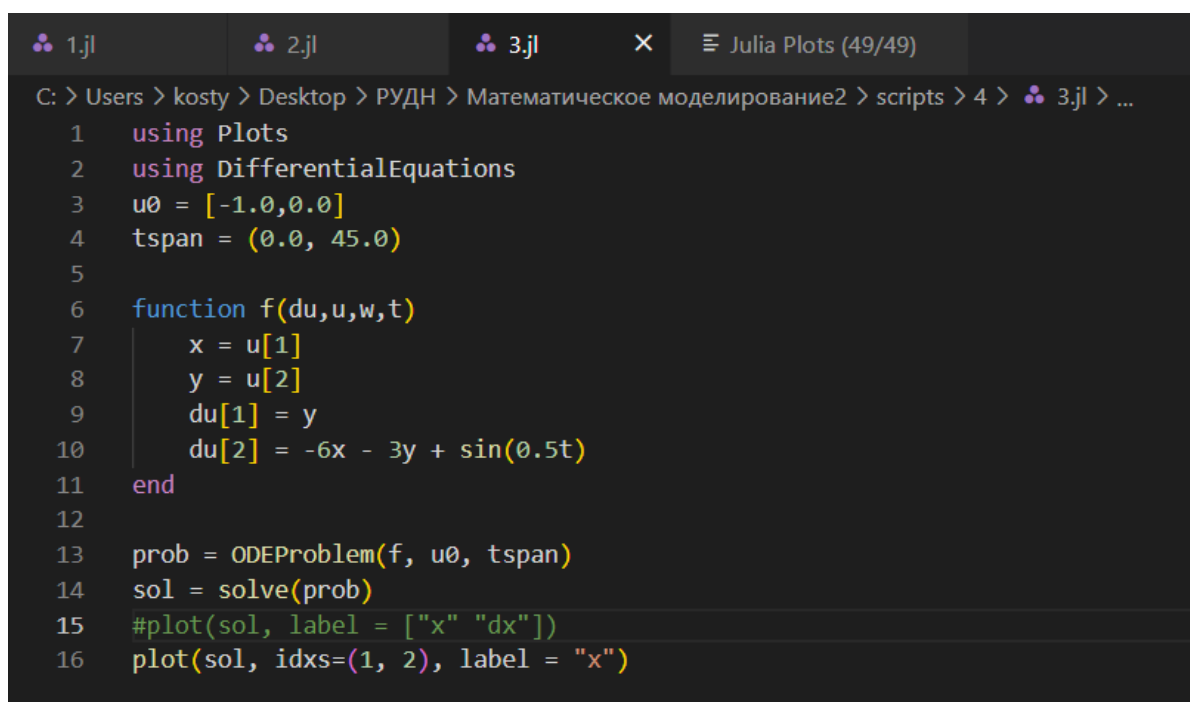


Рис. 4.6: Портрет второго случая на Julia

На рис. 4.7 представлен код, реализованный на Julia, для третьего случая. На рис. 4.8 и рис. 4.9 представлены график решения и фазовый портрет.



```
1.jl 2.jl 3.jl X Julia Plots (49/49)
C: > Users > kosty > Desktop > РУДН > Математическое моделирование2 > scripts > 4 > 3.jl > ...
1 using Plots
2 using DifferentialEquations
3 u0 = [-1.0, 0.0]
4 tspan = (0.0, 45.0)
5
6 function f(du, u, w, t)
7     x = u[1]
8     y = u[2]
9     du[1] = y
10    du[2] = -6x - 3y + sin(0.5t)
11 end
12
13 prob = ODEProblem(f, u0, tspan)
14 sol = solve(prob)
15 #plot(sol, label = ["x" "dx"])
16 plot(sol, idxs=(1, 2), label = "x")
```

Рис. 4.7: Код третьего случая на Julia

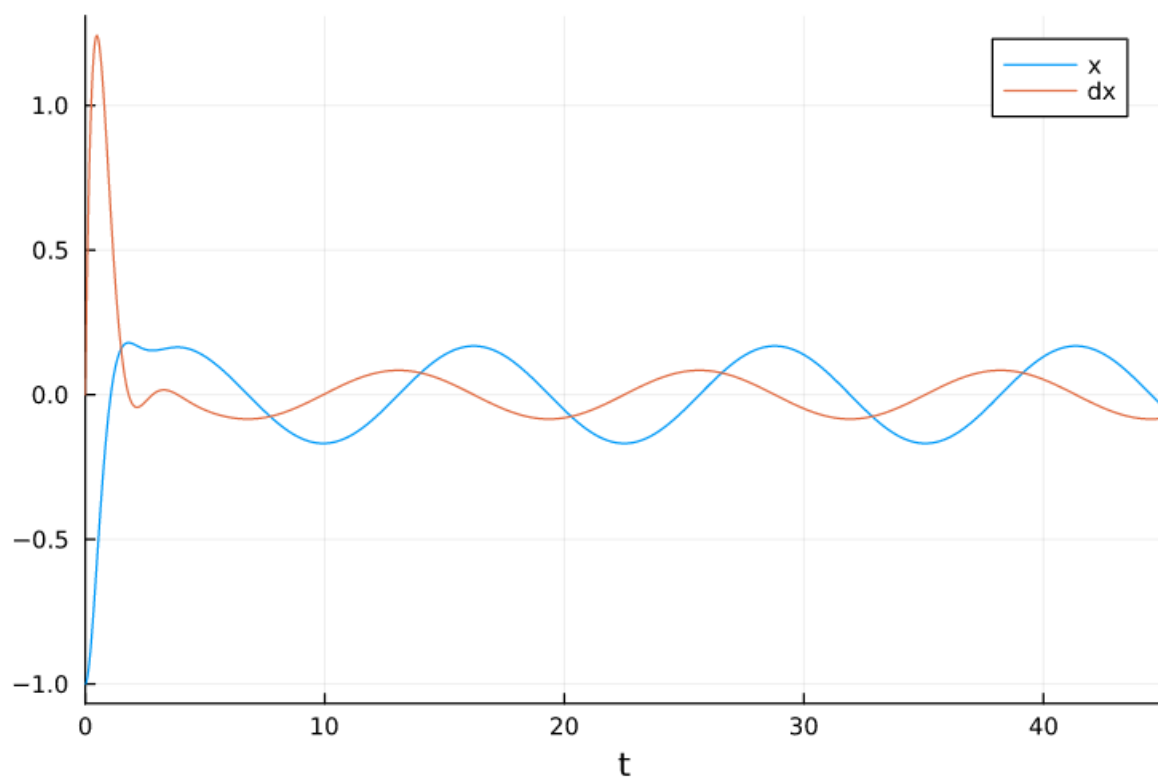


Рис. 4.8: Решение третьего случая на Julia

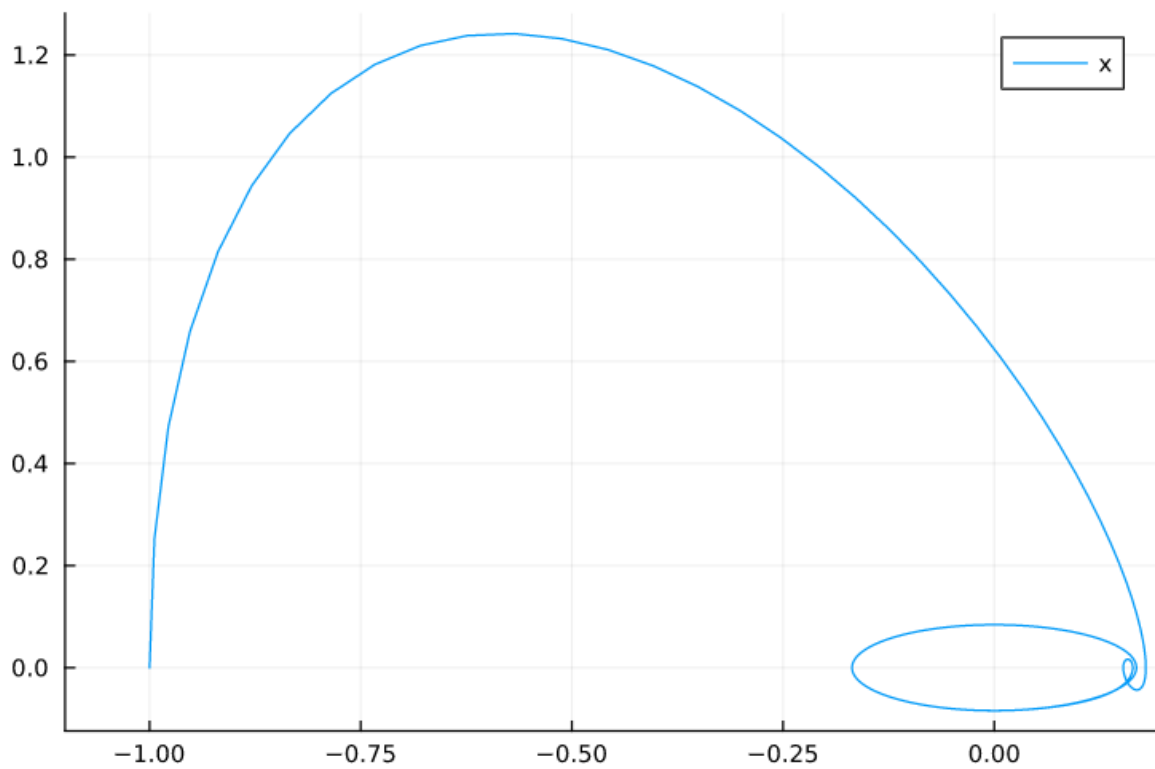


Рис. 4.9: Портрет третьего случая на Julia

На рис. 4.10 представлен код, реализованный на OpenModelica, для первого случая. На рис. 4.11 и рис. 4.12 представлены график решения и фазовый портрет.



```
1 model o1
2   Real x;
3   Real y;
4   initial equation
5     x = -1;
6     y = 0;
7   equation
8     y = der(x);
9     der(y) = -8*x;
10  annotation(
11    experiment(StartTime = 0, StopTime = 45));
12 end o1;
```

Рис. 4.10: Код первого случая на OpenModelica

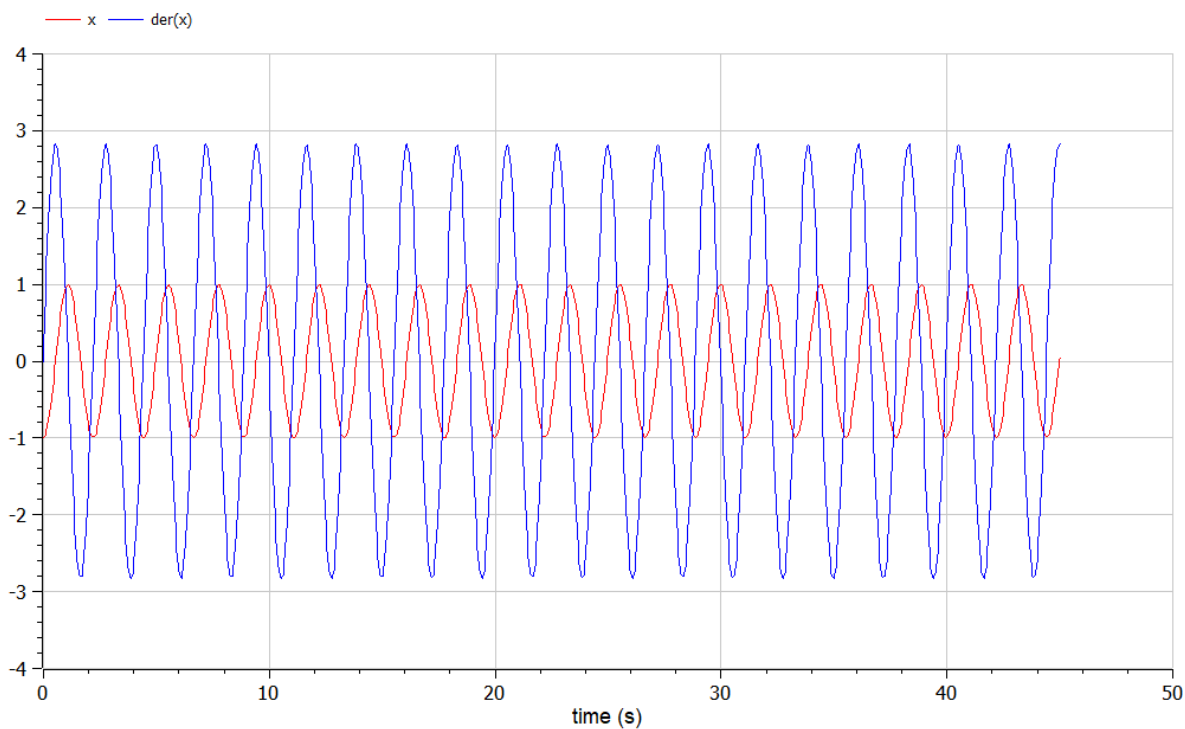


Рис. 4.11: Решение первого случая на OpenModelica

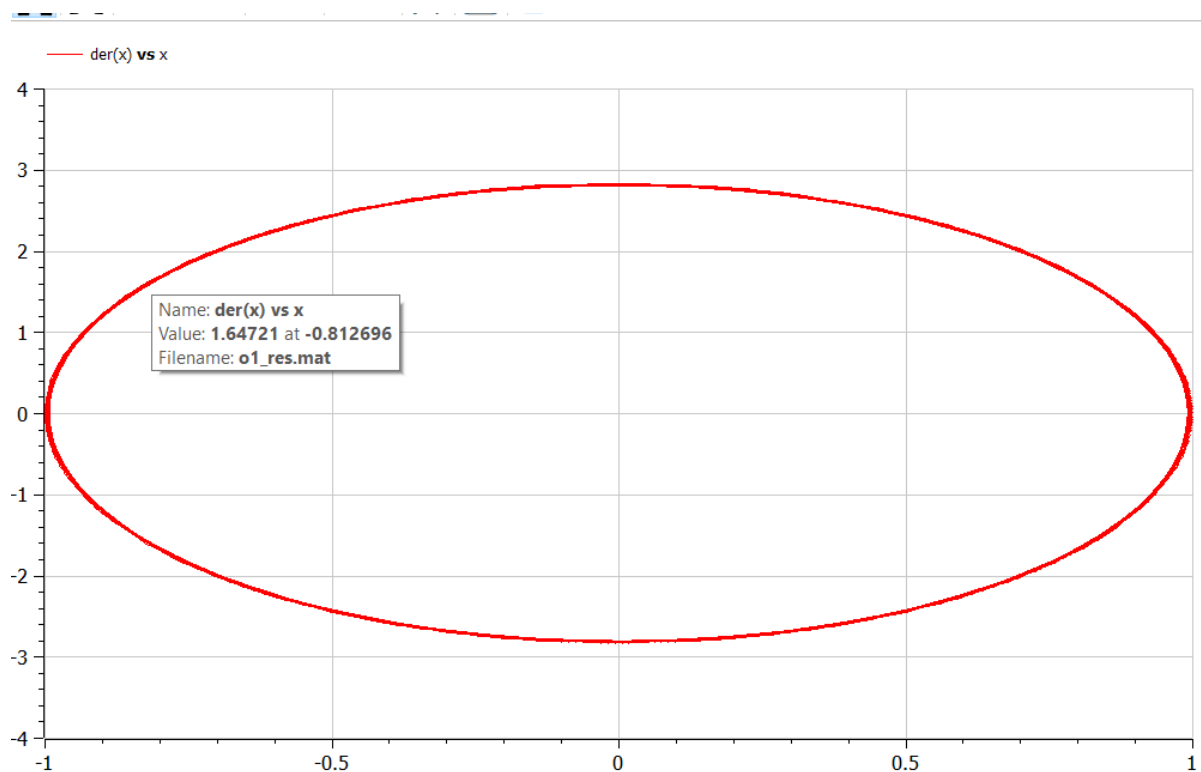


Рис. 4.12: Портрет первого случая на OpenModelica

На рис. 4.13 представлен код, реализованный на OpenModelica, для второго случая. На рис. 4.14 и рис. 4.15 представлены график решения и фазовый портрет.

```

1  model o2
2    Real x;
3    Real y;
4  initial equation
5    x = -1;
6    y = 0;
7  equation
8    y = der(x);
9    der(y) = -3*x - 4*y;
10   annotation(experiment(StartTime=0, StopTime=45));
11 end o2;
12

```

Рис. 4.13: Код второго случая на OpenModelica

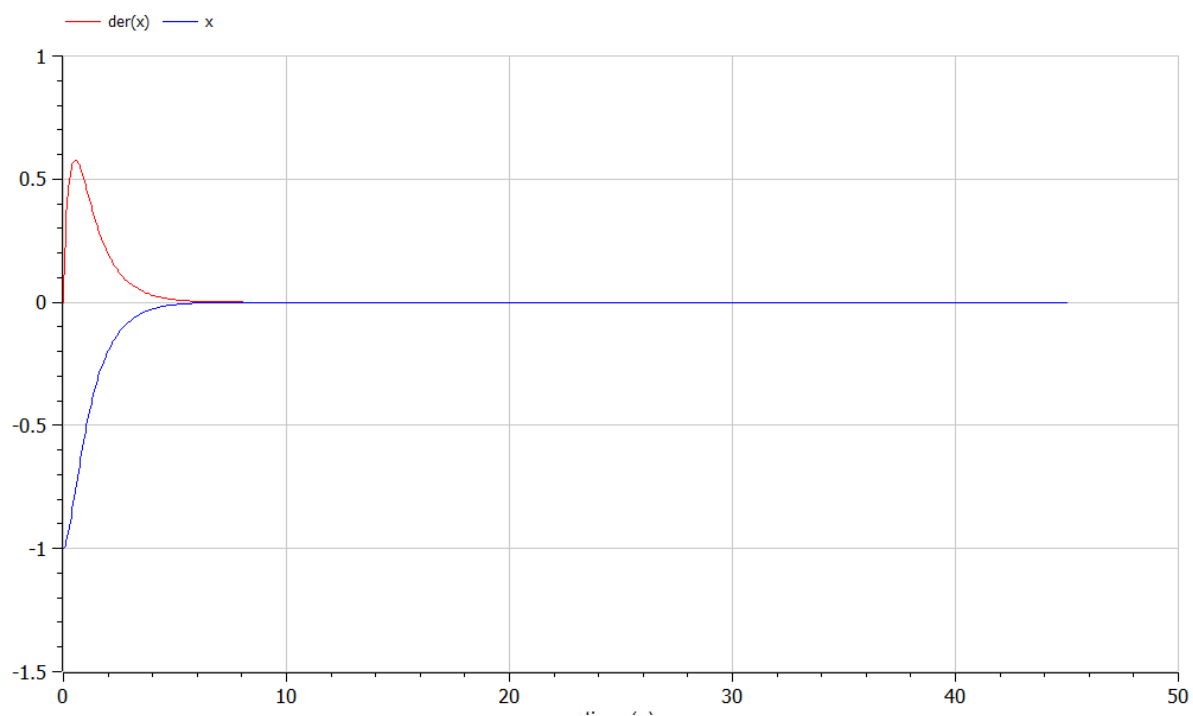


Рис. 4.14: Решение второго случая на OpenModelica

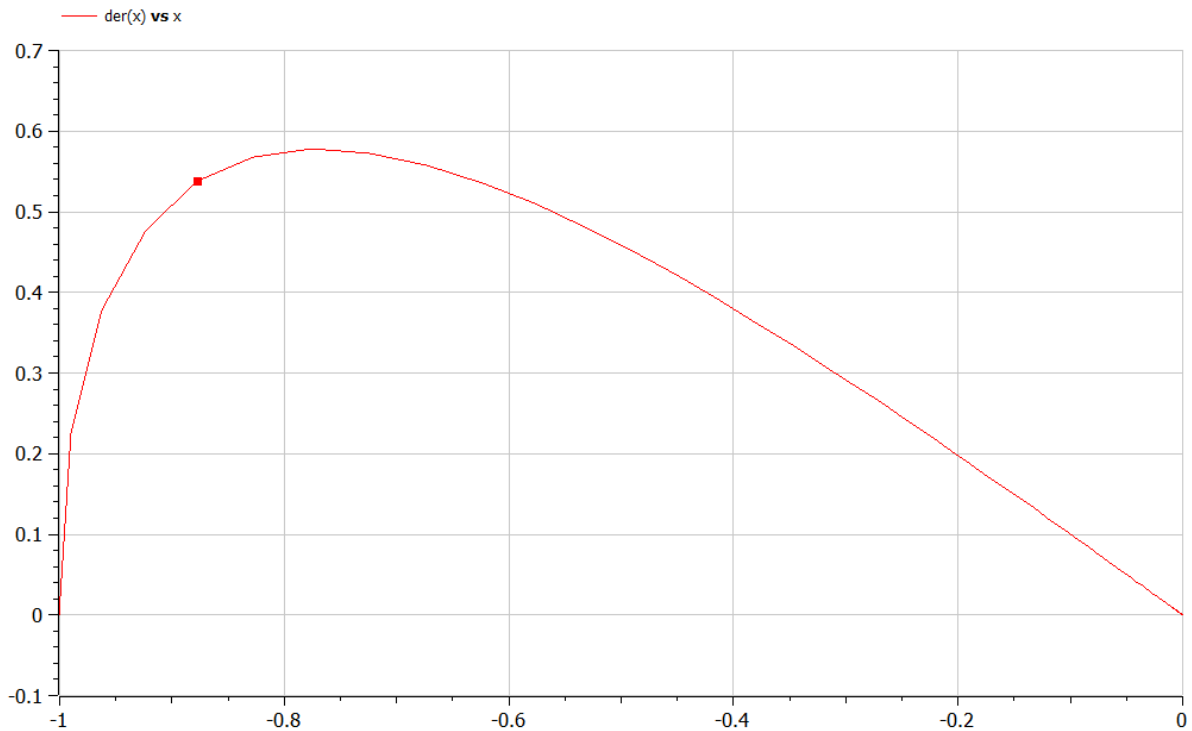


Рис. 4.15: Портрет второго случая на OpenModelica

На рис. 4.16 представлен код, реализованный на Julia, для третьего случая. На рис. 4.17 и рис. 4.18 представлены график решения и фазовый портрет.

```

1  model o3
2    Real x;
3    Real y;
4    initial equation
5      x = -1;
6      y = 0;
7    equation
8      y = der(x);
9      der(y) = -6*x - 3*y + sin(0.5*time);
10     annotation(experiment(StartTime=0,StopTime=45));
11 end o3;
12

```

Рис. 4.16: Код третьего случая на OpenModelica

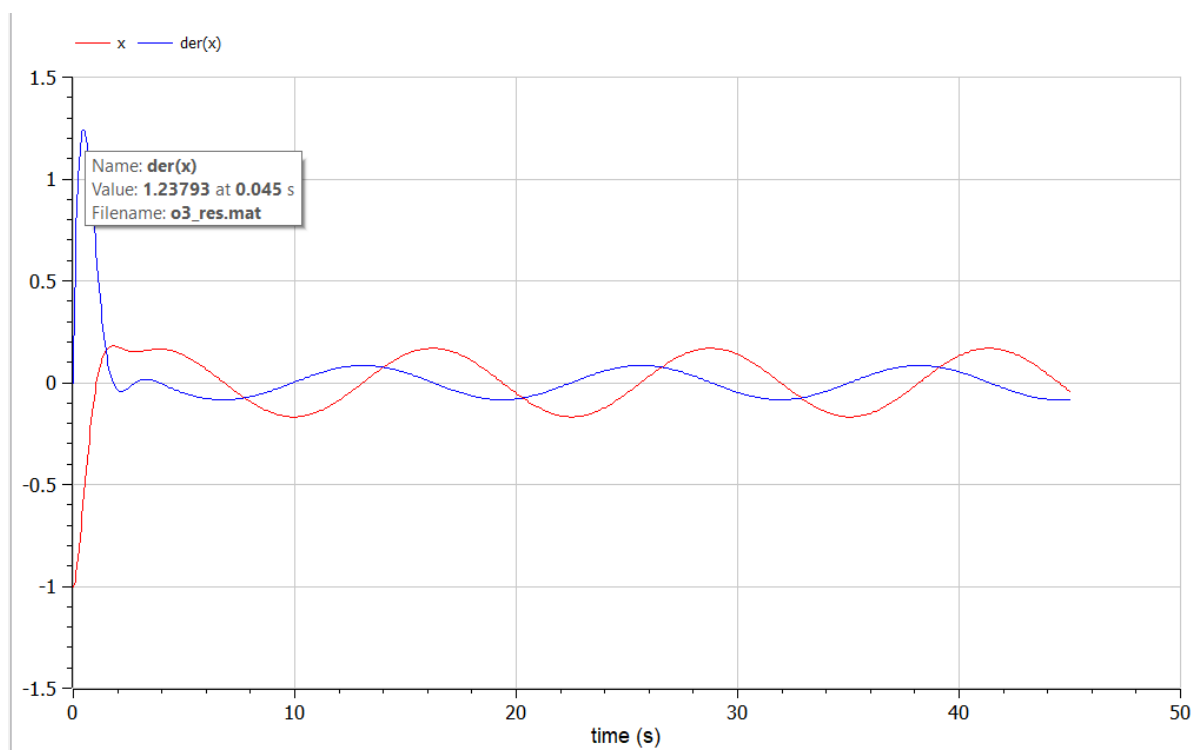


Рис. 4.17: Решение третьего случая на OpenModelica

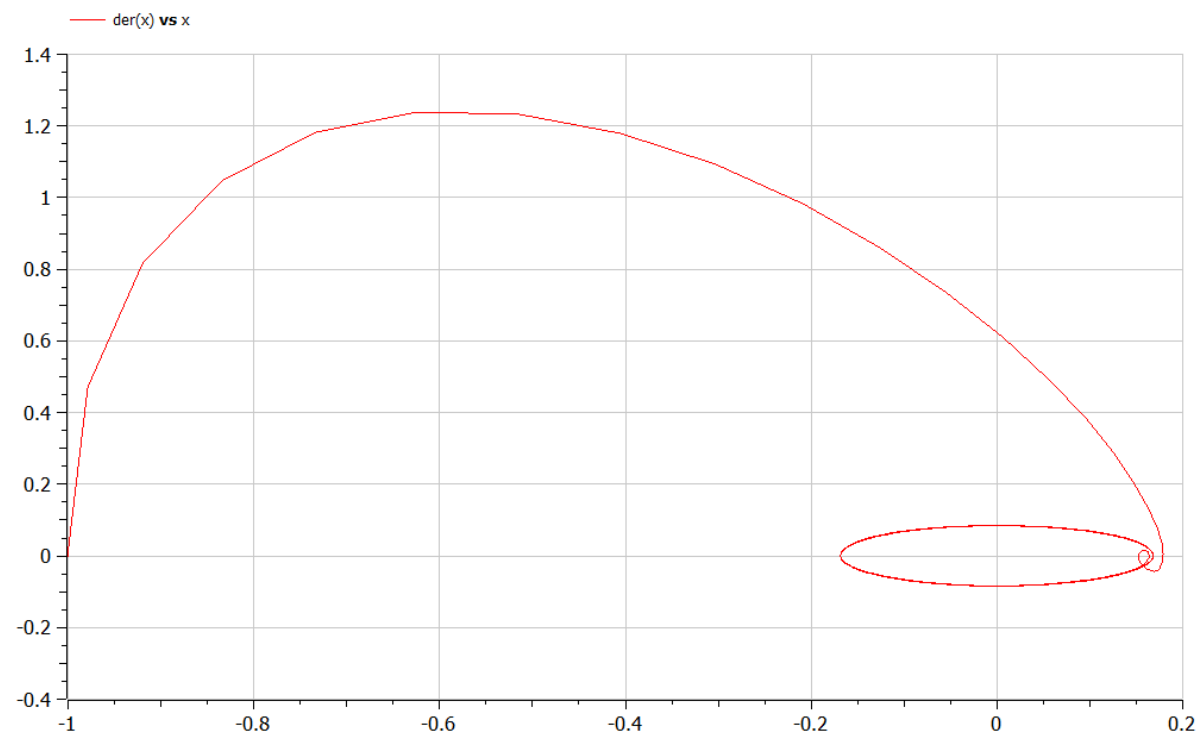


Рис. 4.18: Портрет третьего случая на OpenModelica

## 5 Выводы

Работа выполнена полностью и без ошибок. Код можно прокачать, но жить можно.

## Список литературы

1. Л. К. Мартинсон Е.В.С. Физика в техническом университете. Том 5. МГТУ им. Н.Э.Баумана, 2002.