Отчет по лабораторной работе №4

по дисциплине: Математическое моделирование

Ким Михаил Алексеевич

Содержание

# 1 Цель работы

Продолжить знакомство с функционалом языка программирования Julia, дополнительных библиотек (DifferentialEquations, Plots), интерактивного блокнота Pluto, а также интерактивной командной строкой REPL. Продолжить ознакомление с языком моделирования Modelica и программным обеспечением OpenModelica. Используя эти средства, описать математическую модель гармонических колебаний.

# 2 Задание

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 c затуханием и без действий внешней силы .
3. Колебания гармонического осциллятора c затуханием и под действием внешней силы .

На интервале (шаг ) с начальными условиями .

# 3 Теоретическое введение

## 3.1 Модель гармонических колебаний

Модель гармонических колебаний является одной из фундаментальных моделей в физике. Она описывает поведение объекта, который движется с постоянной частотой и амплитудой.

Представим, что у нас есть объект, который движется вокруг своей равновесной позиции. Это может быть, к примеру, маятник часов или колебательный контур в электрической цепи. Если мы будем наблюдать за движением объекта с течением времени, то заметим, что движение повторяется с постоянной частотой и амплитудой. Эти повторения и называются гармоническими колебаниями [1].

Модель гармонических колебаний находит применение во многих областях, включая физику, математику, инженерию, акустику и другие.

Гармонические колебания широко используются в радиофизике и телекоммуникациях для передачи информации. В этом случае частота колебаний используется для кодирования информации, которая затем может быть передана по радиоволнам или кабельным линиям.

Также гармонические колебания используются при изучении механики и электромагнетизма. Они позволяют предсказывать поведение системы в зависимости от начальных условий и параметров [2].

Система, способная совершать гармонические колебания, называется линейным гармоническим осциллятором.

Линейный гармонический осциллятор может быть использован для моделирования многих процессов, включая электрические контуры, оптические системы и колебания молекул.

Также линейный гармонический осциллятор является базовым элементом в физике твердого тела и квантовой механике [3].

В квантовой механике линейный гармонический осциллятор является важнейшей моделью для изучения колебаний атомов в молекулах и фотонах внутри оптических резонаторов.

Помимо этого, линейный гармонический осциллятор может быть использован для изучения резонанса и демпфирования, а также для моделирования волновых процессов в физике и инженерии [4].

Уравнение свободных колебаний гармонического осциллятора имеет следующий вид:

где — переменная, описывающая состояние системы (смещение грузика, заряд конденсатора и т.д.), — параметр, характеризующий потери энергии (трение в механической системе, сопротивление в контуре), — собственная частота колебаний, — время.

Для однозначной разрешимости ОДУ второго порядка необходимо задать два начальных условия вида:

Уравнение ОДУ второго порядка можно переписать в следующем виде как систему:

Начальные условия для такой системы примут вид:

Также в наше уравнение ОДУ второго порядка можно добавить правую часть. Тогда система примет вид:

Заметим, что ОДУ может иметь нулевой коэффицент — тогда система будет являться консервативной, и слагаемое из системы просто убирается.

При отображении зависимости координаты от координаты мы получим фазовый портрет, который также нужно отрисовать при выполнении лабораторной работы [5].

# 4 Выполнение лабораторной работы

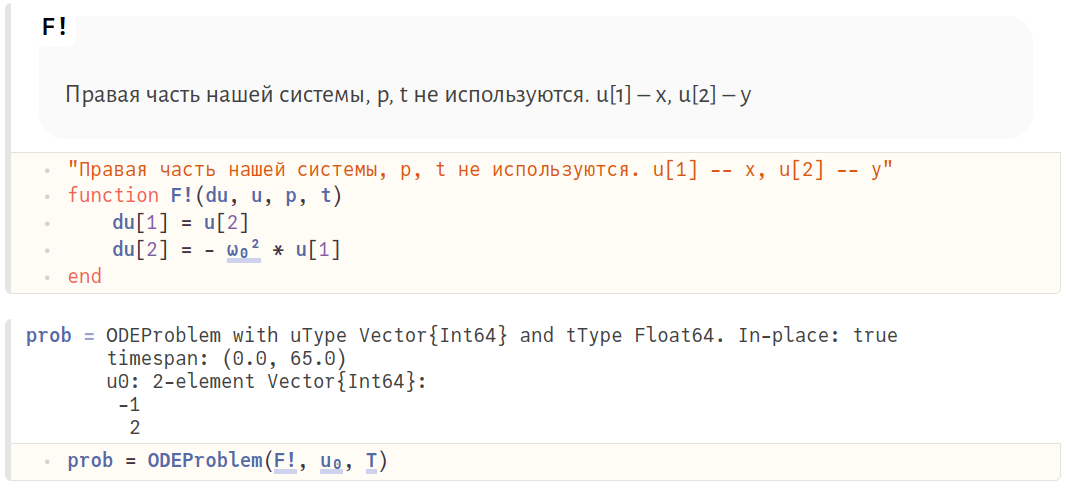
## 4.1 Pluto.jl

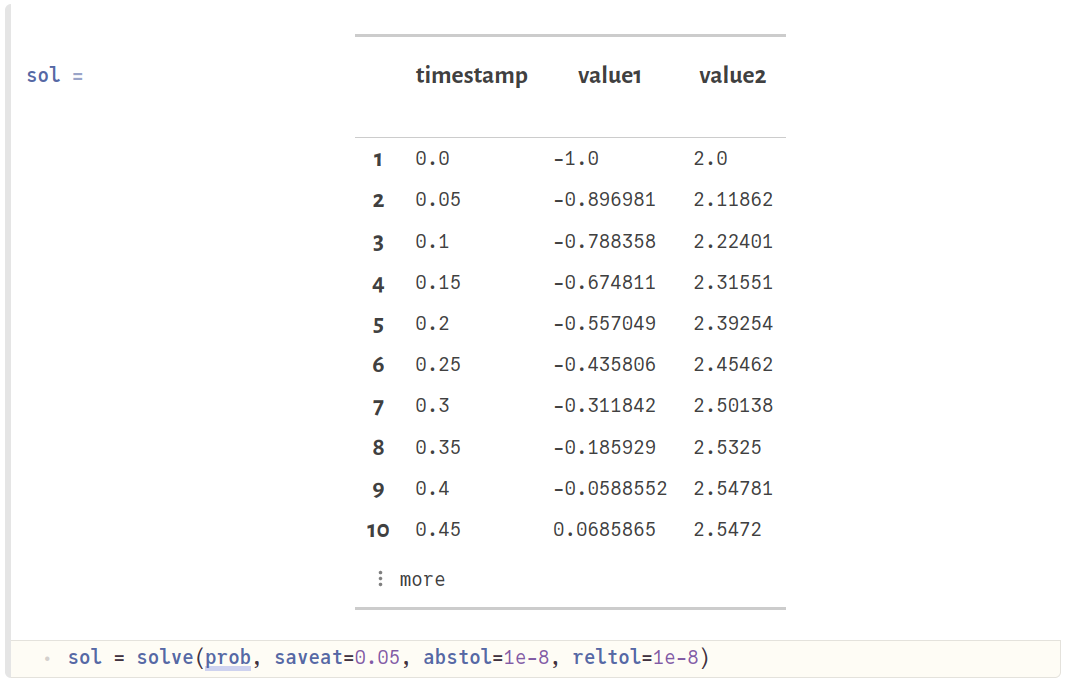
### 4.1.1 Задание №1

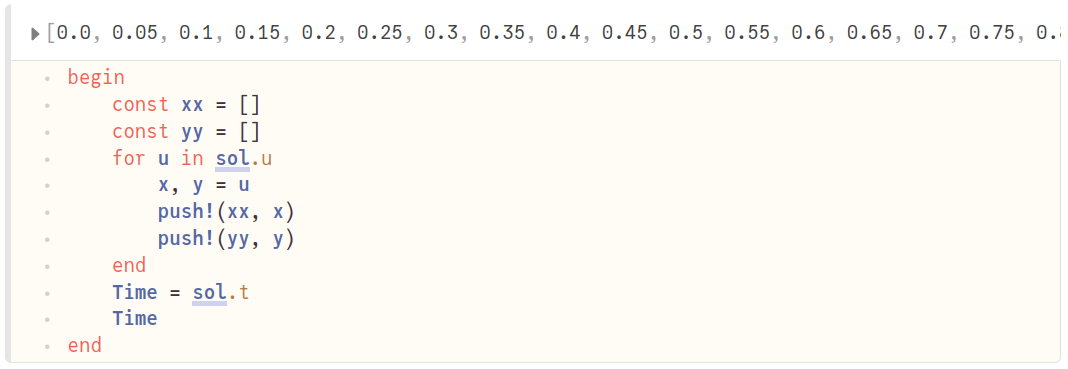
1. Пишем программу, воспроизводящую модель на языке программирования Julia с использованием интерактивного блокнота Pluto (рис. [1](#fig:01), [2](#fig:02), [3](#fig:03), [4](#fig:04), [5](#fig:05)).

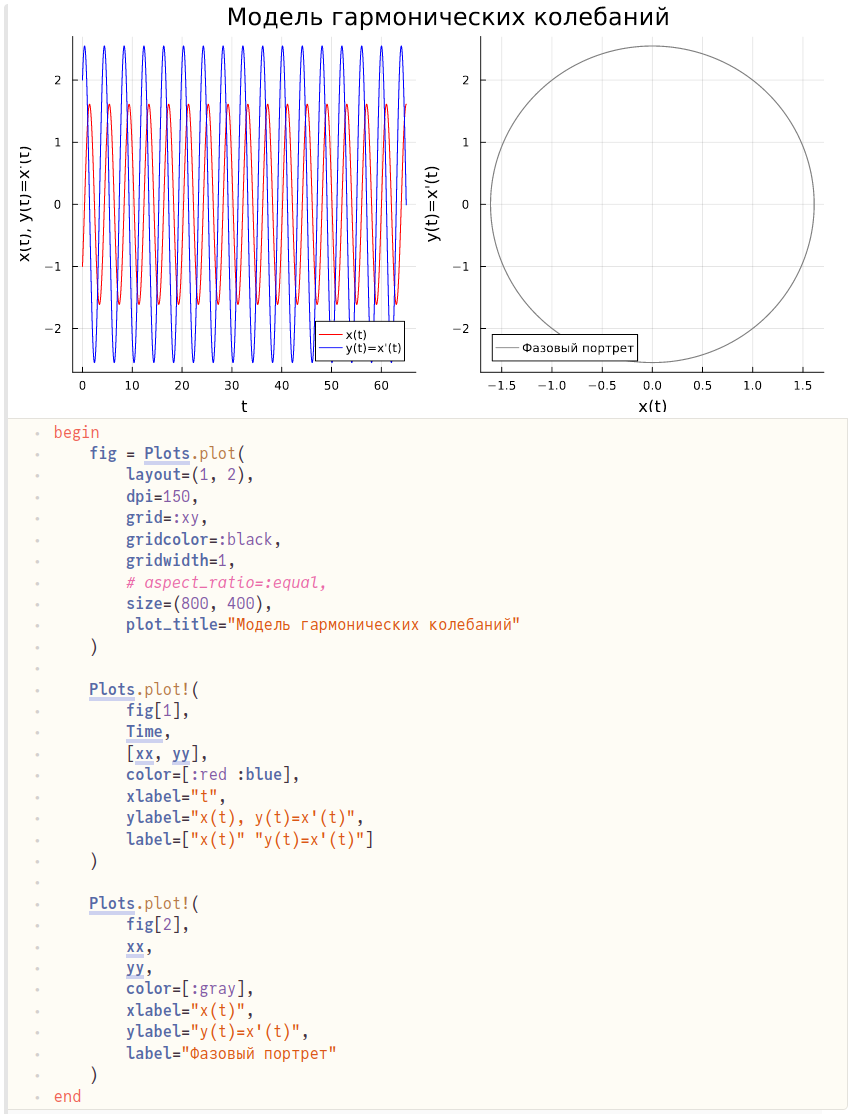
* begin  
   import Pkg  
   Pkg.activate()  
   using DifferentialEquations  
   using LaTeXStrings  
   import Plots  
  end
* begin  
   const ω₀² = 2.5  
    
   "Начальные условия: u₀[1] -- x₀, u₀[2] -- y₀"  
   u₀ = [-1, 2]  
    
   "Период времени"  
   T = (0.0, 65.0)  
  end
* "Правая часть нашей системы, p, t не используются. u[1] -- x, u[2] -- y"  
  function F!(du, u, p, t)  
   du[1] = u[2]  
   du[2] = - ω₀² \* u[1]  
  end
* prob = ODEProblem(F!, u₀, T)
* sol = solve(prob, saveat=0.05, abstol=1e-8, reltol=1e-8)
* begin  
   const xx = []  
   const yy = []  
   for u in sol.u  
   x, y = u  
   push!(xx, x)  
   push!(yy, y)  
   end  
   Time = sol.t  
   Time  
  end
* begin  
   fig = Plots.plot(  
   layout=(1, 2),  
   dpi=150,  
   grid=:xy,  
   gridcolor=:black,  
   gridwidth=1,  
   # aspect\_ratio=:equal,  
   size=(800, 400),  
   plot\_title="Модель гармонических колебаний"  
   )  
    
   Plots.plot!(  
   fig[1],  
   Time,  
   [xx, yy],  
   color=[:red :blue],  
   xlabel="t",  
   ylabel="x(t), y(t)=x'(t)",  
   label=["x(t)" "y(t)=x'(t)"]  
   )  
    
   Plots.plot!(  
   fig[2],  
   xx,  
   yy,  
   color=[:gray],  
   xlabel="x(t)",  
   ylabel="y(t)=x'(t)",  
   label="Фазовый портрет"  
   )  
  end

* 
* Figure 1: Импорт библиотек. Задание коэффициентов, начальных условий, периода времени

* 
* Figure 2: Запись систему уравнений в виде функции. Постановка задачи

* 
* Figure 3: Решение задачи

* 
* Figure 4: Формирование трех массивов, содержащих значения x, y, t

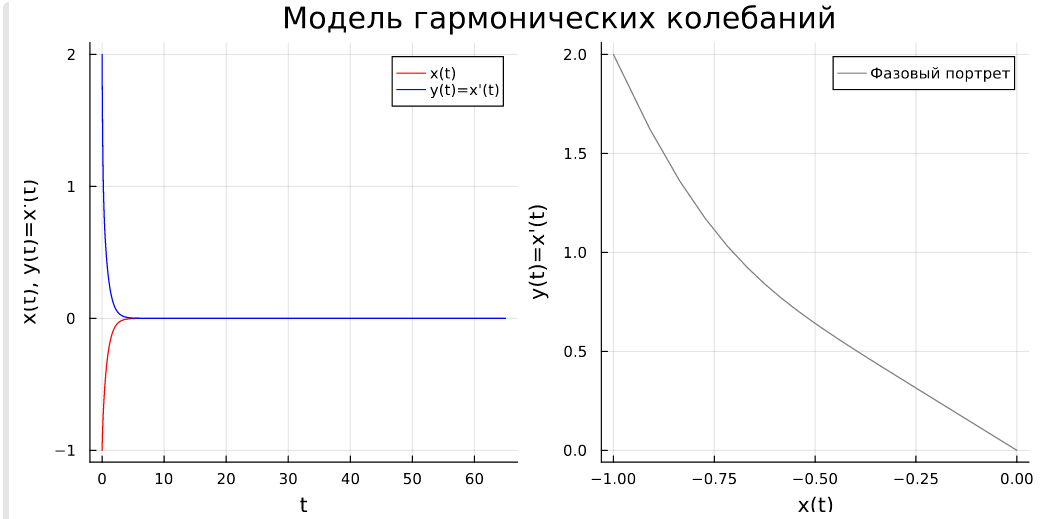
* 
* Figure 5: Отрисовка графиков

### 4.1.2 Задание №2

1. Помимо коэффициента, представляющего собой квадрат собственной частоты колебаний (), добавляем коэффициент, представляющий собой потерю энергии в системе, умноженную на два (). Остальные блоки кода оставляем без изменений. Любуемся результатом (рис. [6](#fig:06), [7](#fig:07)).

* begin  
   const ω₀² = 11.0  
   const 𝛄⬝2 = 10.0  
    
   "Начальные условия: u₀[1] -- x₀, u₀[2] -- y₀"  
   u₀ = [-1, 2]  
    
   "Период времени"  
   T = (0.0, 65.0)  
  end
* "Правая часть нашей системы, p, t не используются. u[1] -- x, u[2] -- y"  
  function F!(du, u, p, t)  
   du[1] = u[2]  
   du[2] = - 𝛄⬝2 \* u[2] - ω₀² \* u[1]  
  end

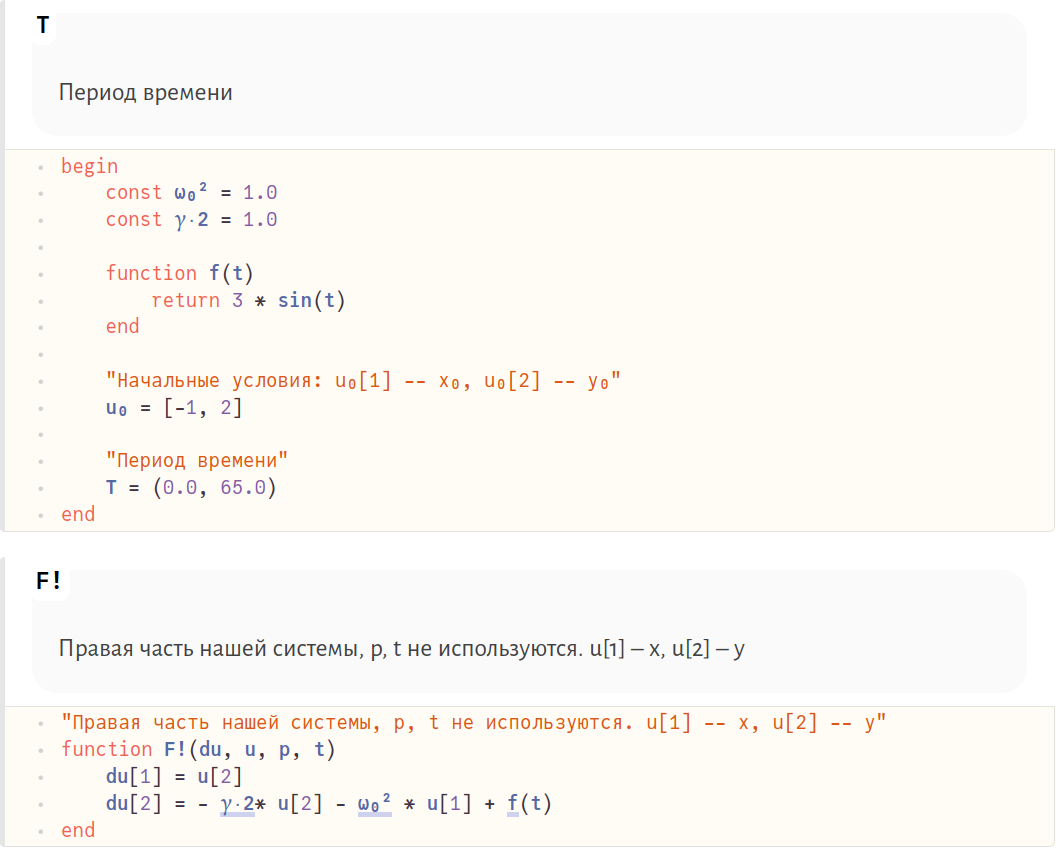
* 
* Figure 6: Добавление необходимых коэффициентов и изменение системы уравнений

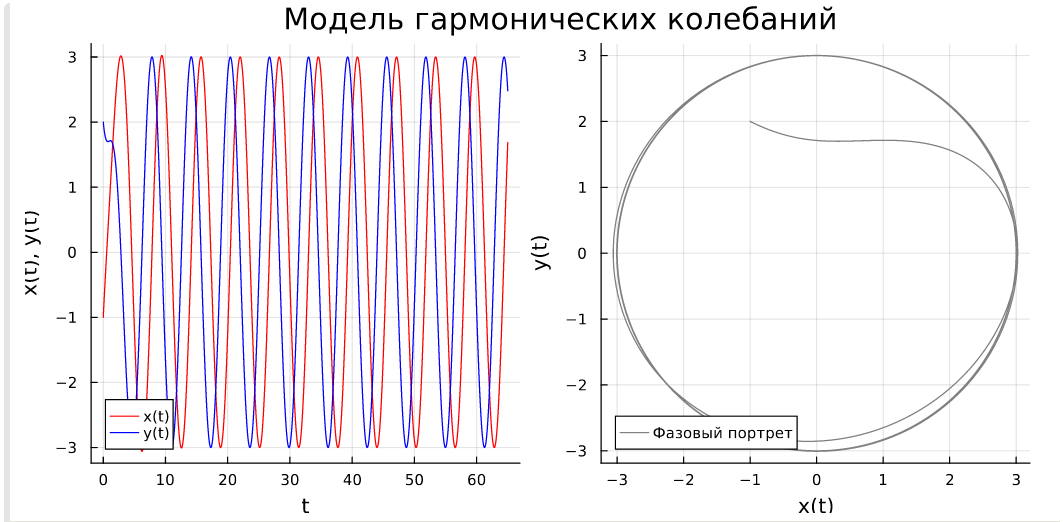
* 
* Figure 7: Результат в виде графиков

### 4.1.3 Задание №3

1. Добавляем правую часть ОДУ () в систему. Остальные блоки кода оставляем без изменений. Любуемся результатом (рис. [8](#fig:08), [9](#fig:09)).

* begin  
   const ω₀² = 1.0  
   const 𝛄⬝2 = 1.0  
    
   function f(t)  
   return 3 \* sin(t)  
   end  
    
   "Начальные условия: u₀[1] -- x₀, u₀[2] -- y₀"  
   u₀ = [-1, 2]  
    
   "Период времени"  
   T = (0.0, 65.0)  
  end
* "Правая часть нашей системы, p, t не используются. u[1] -- x, u[2] -- y"  
  function F!(du, u, p, t)  
   du[1] = u[2]  
   du[2] = - 𝛄⬝2\* u[2] - ω₀² \* u[1] + f(t)  
  end

* 
* Figure 8: Добавление правой части ОДУ и изменение системы уравнений

* 
* Figure 9: Результат в виде графиков

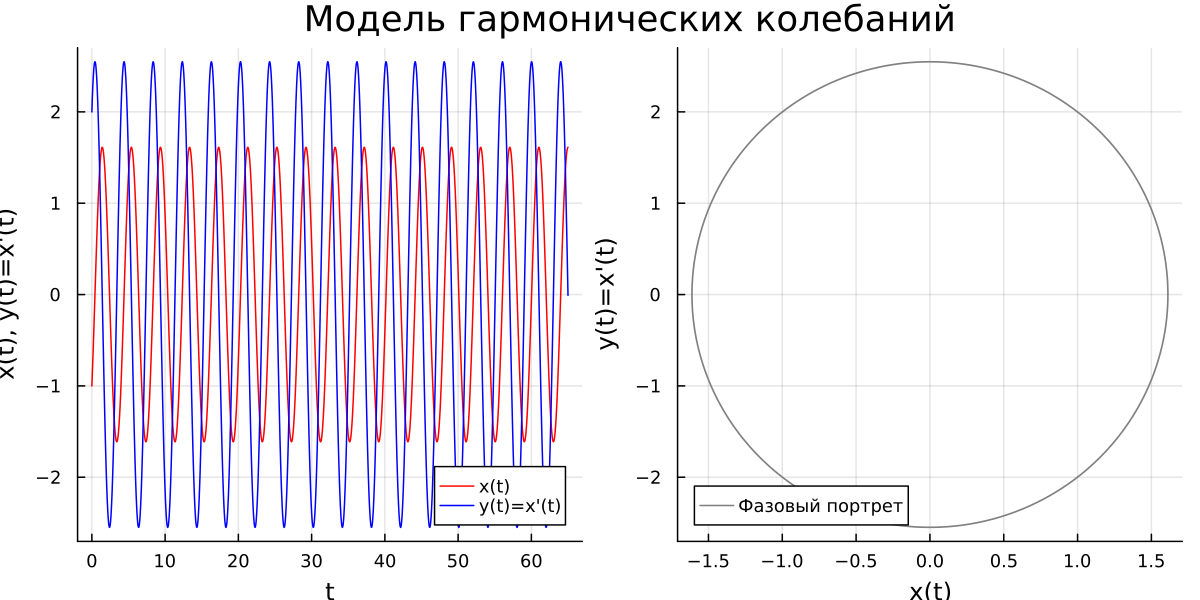
## 4.2 Julia

### 4.2.1 Задание №1

1. Код на Julia в файле аналогичен тому же, написанному с использованием Pluto. Единственные различия: блоки перенесены в файл в виде построчного алгоритма без повторяющихся ‘begin’ и ‘end’, отличающийся синтаксис подключения библиотек, выгрузка графиков в виде изображений при помощи метода в последней строчке кода (рис. [10](#fig:10), [11](#fig:001)).

* using DifferentialEquations  
  using Plots  
    
  const ω₀² = 2.5  
    
  "Начальные условия: u₀[1] -- x₀, u₀[2] -- y₀"  
  u₀ = [-1, 2]  
    
  "Период времени"  
  T = (0.0, 65.0)  
    
  "Правая часть нашей системы, p, t не используются. u[1] -- x, u[2] -- y"  
  function F!(du, u, p, t)  
   du[1] = u[2]  
   du[2] = - ω₀² \* u[1]  
  end  
    
    
  prob = ODEProblem(F!, u₀, T)  
  sol = solve(prob, saveat=0.05, abstol=1e-8, reltol=1e-8)  
    
  const xx = []  
  const yy = []  
  for u in sol.u  
   x, y = u  
   push!(xx, x)  
   push!(yy, y)  
  end  
  Time = sol.t  
    
  fig = Plots.plot(  
   layout=(1, 2),  
   dpi=150,  
   grid=:xy,  
   gridcolor=:black,  
   gridwidth=1,  
   # aspect\_ratio=:equal,  
   size=(800, 400),  
   plot\_title="Модель гармонических колебаний"  
   )  
    
  Plots.plot!(  
   fig[1],  
   Time,  
   [xx, yy],  
   color=[:red :blue],  
   xlabel="t",  
   ylabel="x(t), y(t)=x'(t)",  
   label=["x(t)" "y(t)=x'(t)"]  
  )  
    
  Plots.plot!(  
   fig[2],  
   xx,  
   yy,  
   color=[:gray],  
   xlabel="x(t)",  
   ylabel="y(t)=x'(t)",  
   label="Фазовый портрет"  
  )  
    
    
  savefig(fig, "../lab4\_1")

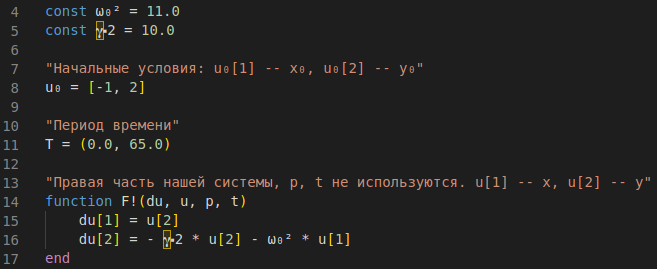
* 
* Figure 10: Код программы на Julia. Аналогичен коду задания для Pluto.jl

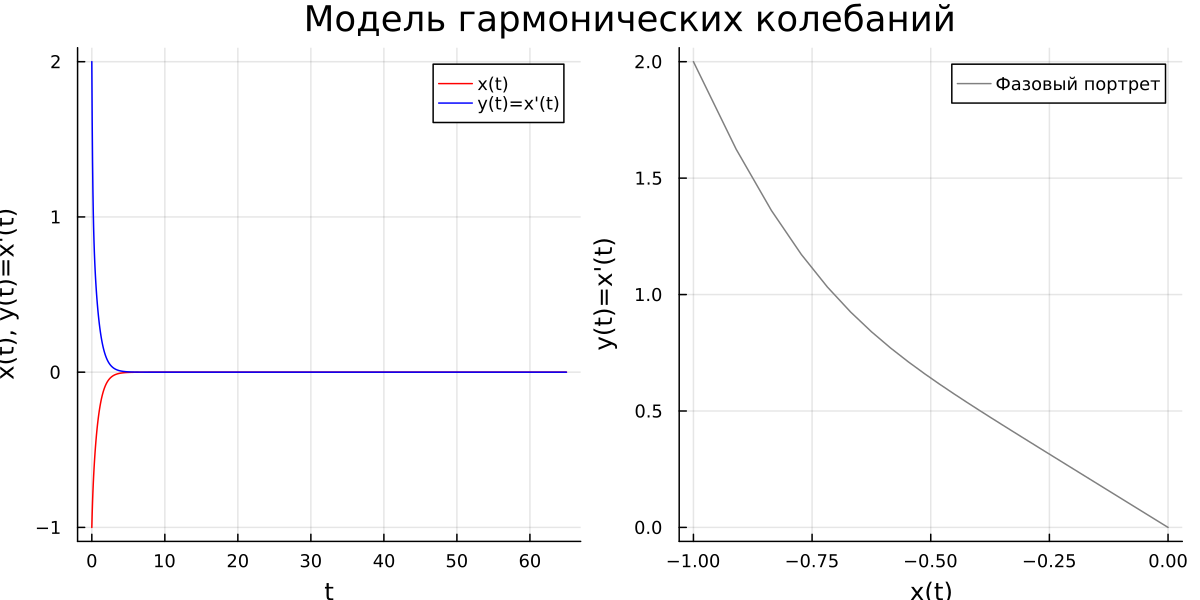
* 
* Figure 11: Результат в виде графиков

### 4.2.2 Задание №2

1. Изменяем необходимые строчки и любуемся результатом (подробное объяснение давалось в предыдущей главе) (рис. [12](#fig:11), [13](#fig:002)).

* const ω₀² = 11.0  
  const 𝛄⬝2 = 10.0  
    
  function F!(du, u, p, t)  
   du[1] = u[2]  
   du[2] = - 𝛄⬝2 \* u[2] - ω₀² \* u[1]  
  end

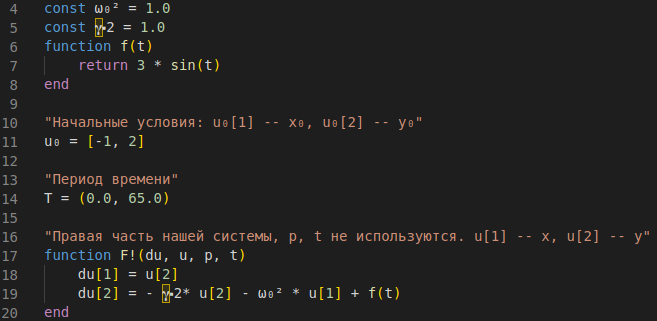
* 
* Figure 12: Измененная часть кода

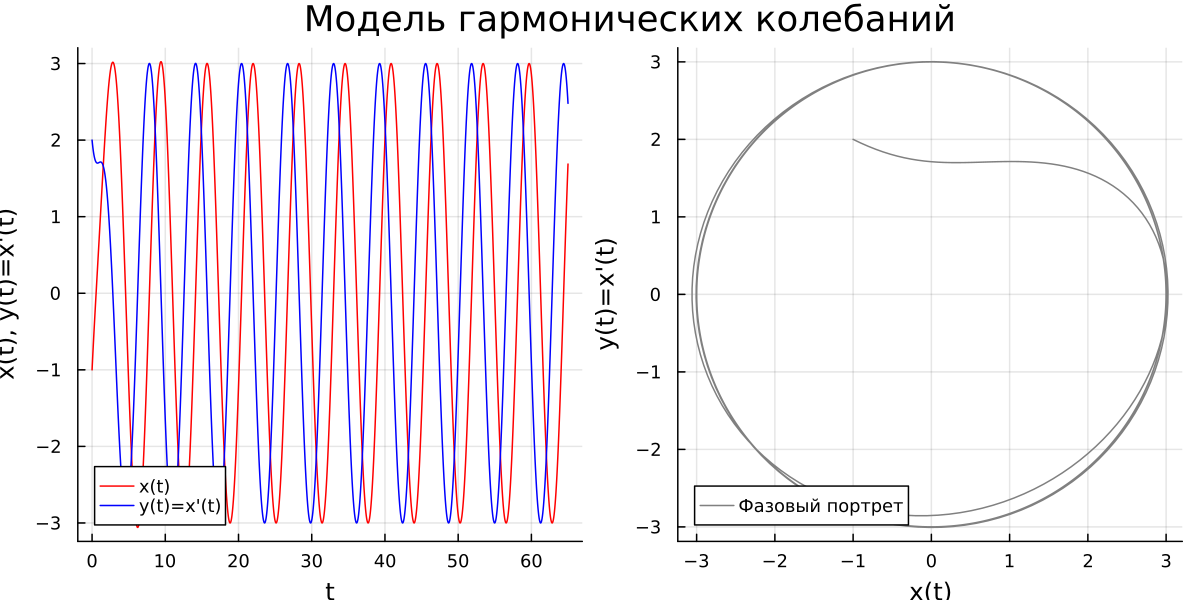
* 
* Figure 13: Результат в виде графиков

### 4.2.3 Задание №3

1. Изменяем необходимые строчки и любуемся результатом (подробное объяснение давалось в предыдущей главе) (рис. [14](#fig:12), [15](#fig:003)).

* const ω₀² = 1.0  
  const 𝛄⬝2 = 1.0  
  function f(t)  
   return 3 \* sin(t)  
  end  
    
  function F!(du, u, p, t)  
   du[1] = u[2]  
   du[2] = - 𝛄⬝2\* u[2] - ω₀² \* u[1] + f(t)  
  end

* 
* Figure 14: Измененная часть кода

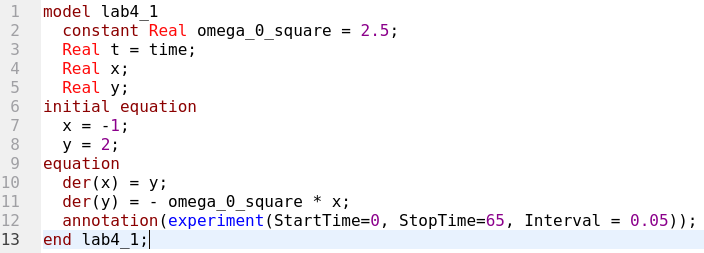
* 
* Figure 15: Результат в виде графиков

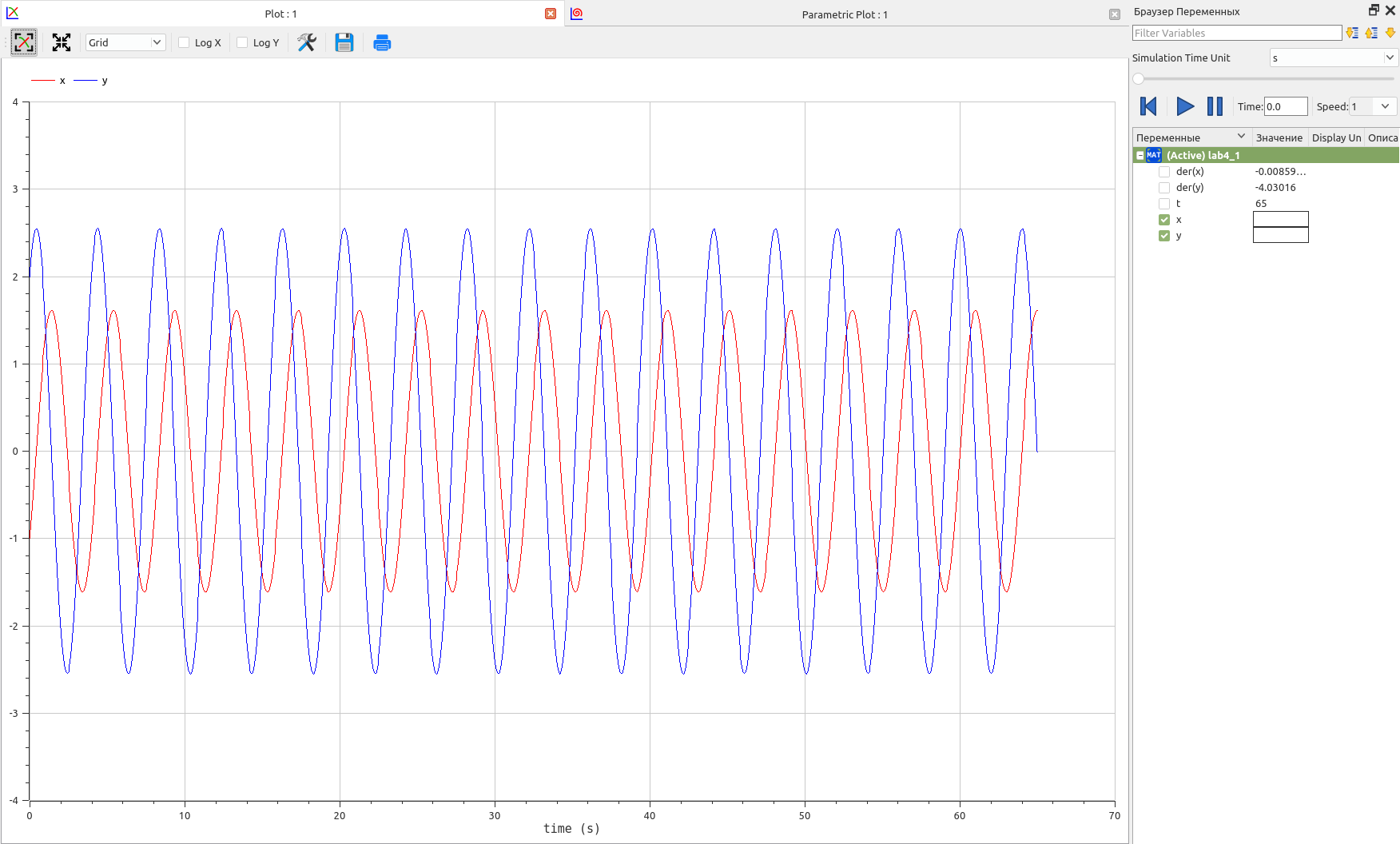
## 4.3 Modelica

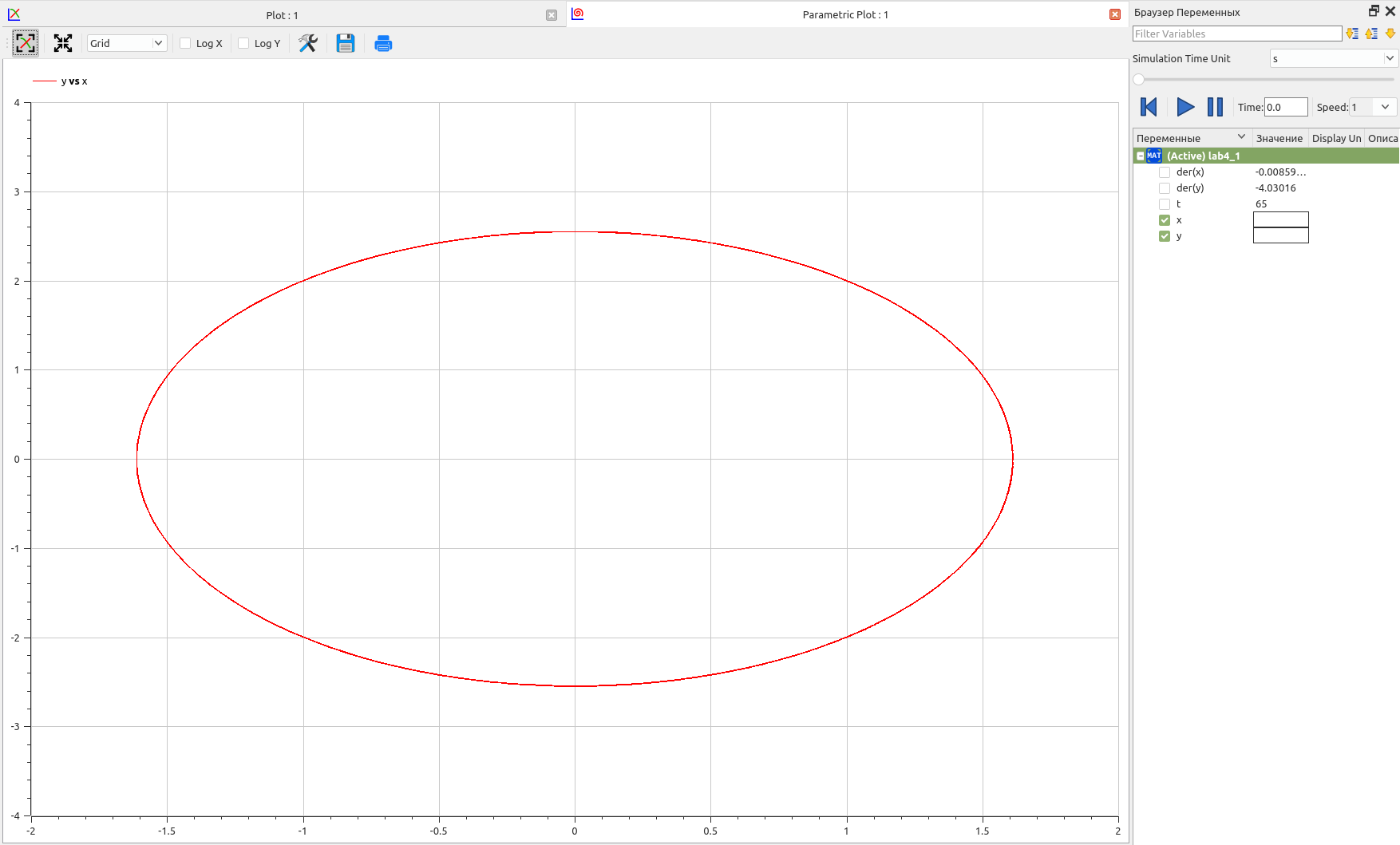
### 4.3.1 Задание №1

1. По аналогии с Julia пишем программу, воспроизводящую модель гармонических колебаний на языке моделирования Modelica с использованием ПО OpenModelica. Любуемся результатами (рис. [16](#fig:13), [17](#fig:14), [18](#fig:15)).

* model lab4\_1  
   constant Real omega\_0\_square = 2.5;  
   Real t = time;  
   Real x;  
   Real y;  
  initial equation  
   x = -1;  
   y = 2;  
  equation  
   der(x) = y;  
   der(y) = - omega\_0\_square \* x;  
   annotation(experiment(StartTime=0, StopTime=65, Interval = 0.05));  
  end lab4\_1;

* 
* Figure 16: Определяем коэффициенты, изменяемые переменные, начальные условия, систему уравнений, а также начальное/конечное время и частоту разбиения при симуляции

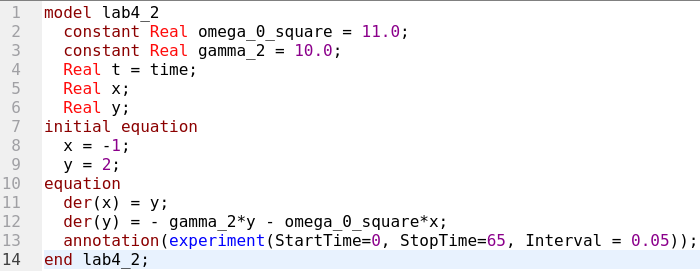
* 
* Figure 17: Результат в виде графика зависимости x и y от t

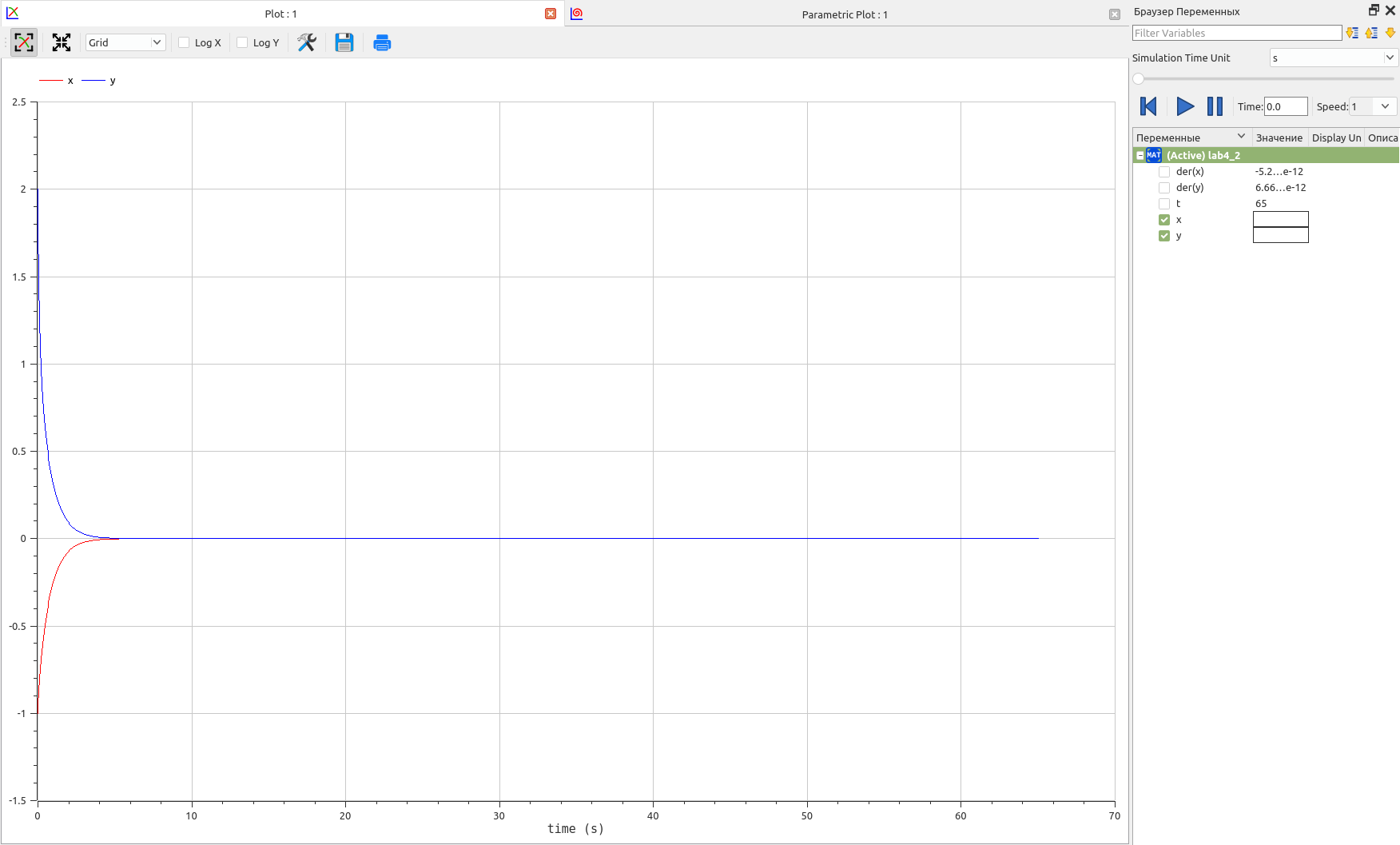
* 
* Figure 18: Результат в виде графика зависимости y от x

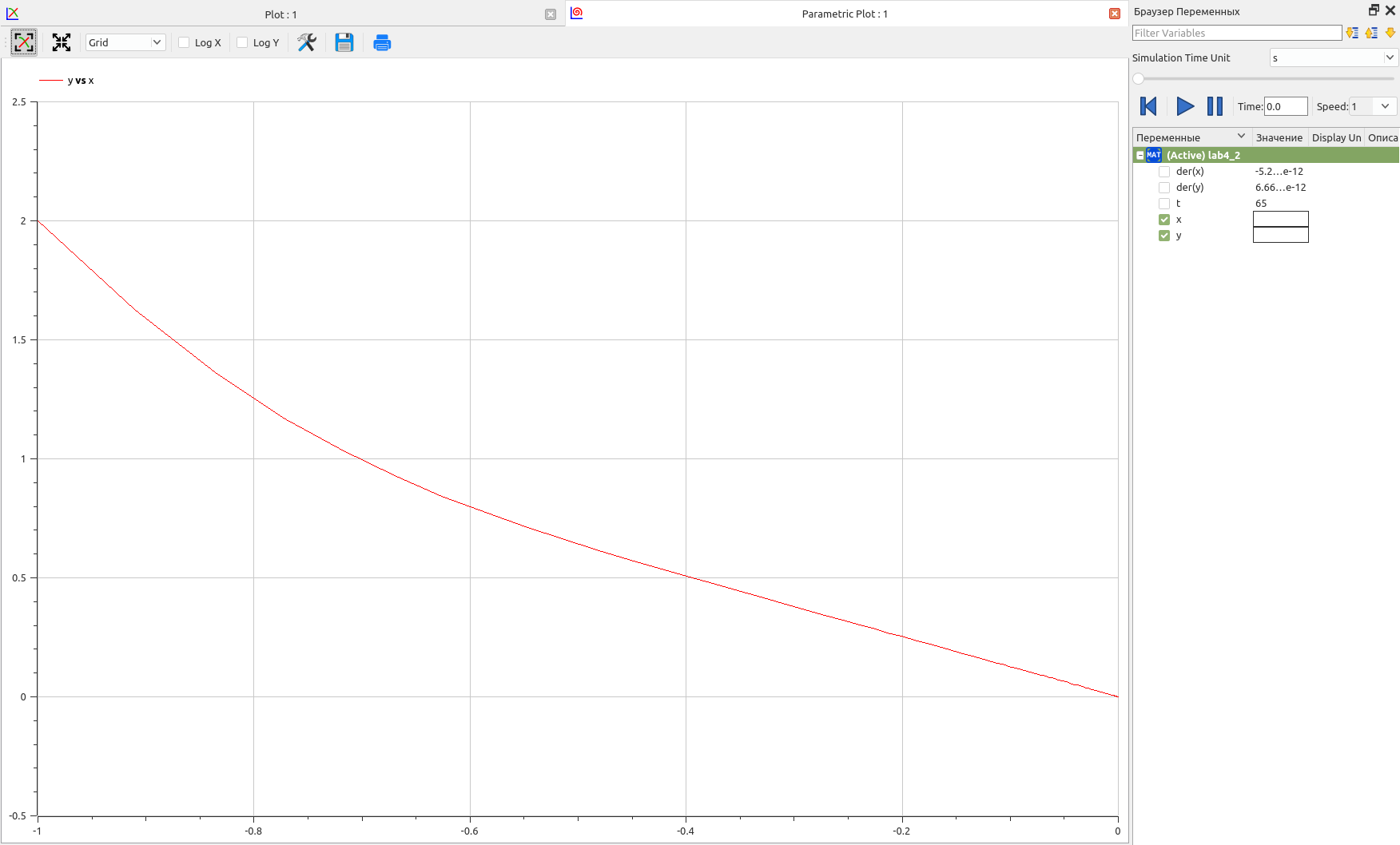
### 4.3.2 Задание №2

1. По аналогии с Julia пишем программу для второго случая. Любуемся результатами (рис. [19](#fig:16), [20](#fig:17), [21](#fig:18)).

* model lab4\_2  
   constant Real omega\_0\_square = 11.0;  
   constant Real gamma\_2 = 10.0;  
   Real t = time;  
   Real x;  
   Real y;  
  initial equation  
   x = -1;  
   y = 2;  
  equation  
   der(x) = y;  
   der(y) = - gamma\_2\*y - omega\_0\_square\*x;  
   annotation(experiment(StartTime=0, StopTime=65, Interval = 0.05));  
  end lab4\_2;

* 
* Figure 19: По сравнению с прошлым случаем добавляется коэффициент и изменятся система

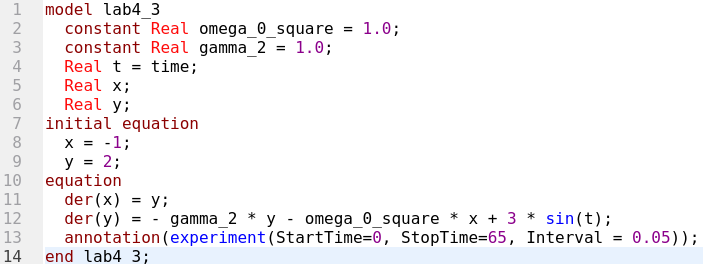
* 
* Figure 20: Результат в виде графика зависимости x и y от t

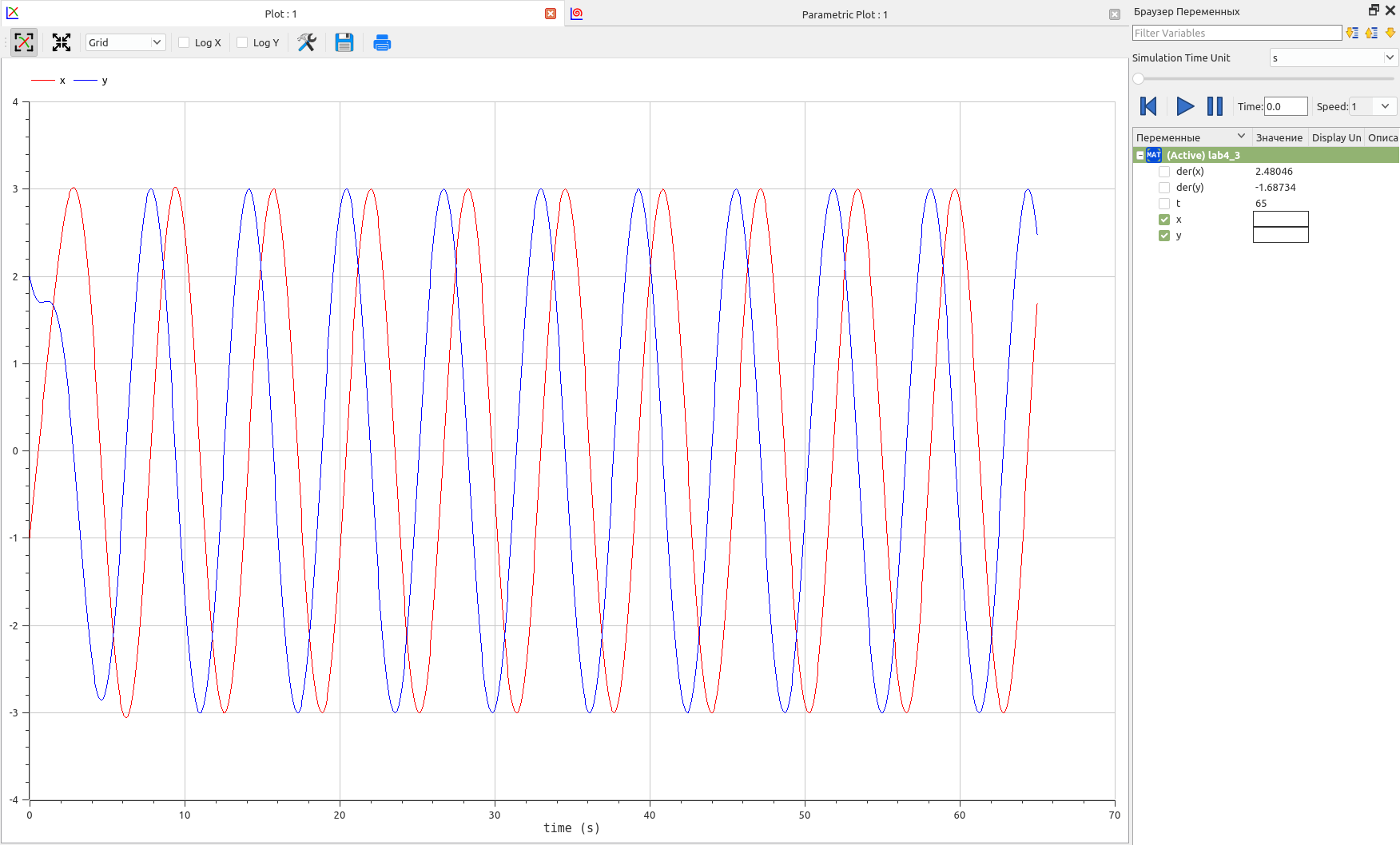
* 
* Figure 21: Результат в виде графика зависимости y от x

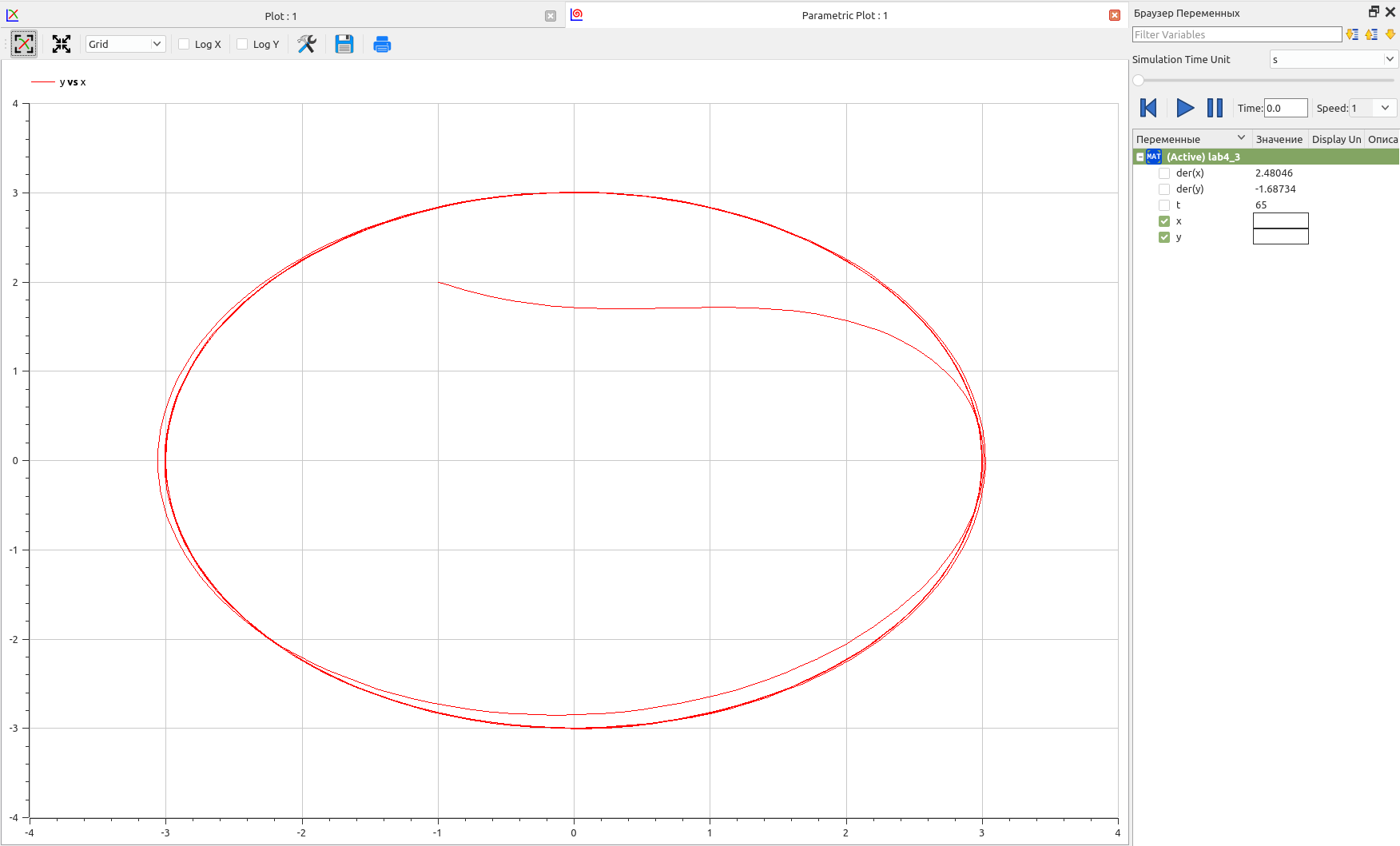
### 4.3.3 Задание №3

1. По аналогии с Julia пишем программу для третьего случая. Любуемся результатами (рис. [22](#fig:19), [23](#fig:20), [24](#fig:21)).

* model lab4\_3  
   constant Real omega\_0\_square = 1.0;  
   constant Real gamma\_2 = 1.0;  
   Real t = time;  
   Real x;  
   Real y;  
  initial equation  
   x = -1;  
   y = 2;  
  equation  
   der(x) = y;  
   der(y) = - gamma\_2 \* y - omega\_0\_square \* x + 3 \* sin(t);  
   annotation(experiment(StartTime=0, StopTime=65, Interval = 0.05));  
  end lab4\_3;

* 
* Figure 22: По сравнению с прошлым случаем добавляется правая часть ОДУ и изменятся система

* 
* Figure 23: Результат в виде графика зависимости x и y от t

* 
* Figure 24: Результат в виде графика зависимости y от x

# 5 Анализ результатов

На текущем примере построения математической модели гармонических колебаний мы можем продолжить сравнивать язык программирования Julia и язык моделирования Modelica. По сравнению с анализом результатов предыдущей лабораторной работы хотелось бы отметить, что определенные недостатки Julia по сравнению с Modelica (медленная скорость выполнения, объем и читабельность кода) лично для меня сглаживаются, т.к. в первую очередь, я теперь использую Pluto. Скорость отрисовки графиков после изменения кода в интерактивном блокноте в разы быстрее по сравнению с со скоростью сохранения графиков в файл при запуске \*.jl. Также частая работа с библиотекой DifferentialEquations ведет к более легкому пониманию кода, пусть и более объемного, нежели на Modelica. Также гибкость настройки точности численного метода решения ОДУ в данный момент на стороне Julia.

Однако, с другой стороны, OpenModelica все еще предоставлят больше возможностей для настройки отображения графиков.

# 6 Выводы

Продолжил знакомство с функционалом языка программирования Julia, дополнительных библиотек (DifferentialEquations, Plots), интерактивного блокнота Pluto, а также интерактивной командной строкой REPL. Продолжил ознакомление с языком моделирования Modelica и программным обеспечением OpenModelica. Используя эти средства, описал математическую модель гармонических колебаний.

# Список литературы

1. Simple harmonic motion [Электронный ресурс]. Wikimedia Foundation, Inc., 2023. URL: <https://en.wikipedia.org/wiki/Simple_harmonic_motion>.

2. Simple Harmonic Motion (SHM) [Электронный ресурс]. BYJU’S. URL: <https://byjus.com/jee/simple-harmonic-motion-shm/>.

3. Harmonic oscillator [Электронный ресурс]. Wikimedia Foundation, Inc., 2022. URL: <https://en.wikipedia.org/wiki/Harmonic_oscillator>.

4. The Harmonic Oscillator [Электронный ресурс]. California Institute of Technology, 2013. URL: <https://www.feynmanlectures.caltech.edu/I_21.html>.

5. Модель гармонических колебаний [Электронный ресурс]. RUDN, 2023. URL: <https://esystem.rudn.ru/mod/resource/view.php?id=967241>.