Лабораторная работа №4

Модель гармонических колебаний

Ильин Андрей Владимирович

Содержание

Список иллюстраций

1 Цель работы

Рассмотреть уравнение гармонических колебаний. Построить модель гармонических колебаний средствами OpenModellica и Julia.

2 Задачи

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

1. Колебания гармонического осциллятора без затуханий и без действий внешней силы $\dot{x} + 3.3x = 0$.
2. Колебания гармонического осциллятора с затуханием и без действий внешней силы $\ddot{x} + 3\dot{x} + 0.3x = 0$.
3. Колебания гармонического осциллятора с затуханием и под действием внешней силы $\ddot{x} + 3.3\dot{x} + 0.3x = 3.3\sin(3t)$.

На интервале $t \in [0; 33]$ (шаг 0.05) с начальными условиями $x_0 = 1.3,\ y_0 = 0.3$.

3 Среда

- Julia – это открытый свободный высокопроизводительный динамический язык высокого уровня, созданный специально для технических (математических) вычислений. Его синтаксис близок к синтаксису других сред технических вычислений, таких как Matlab и Octave. [1]

- OpenModelica — свободное открытое программное обеспечение для моделирования, симуляции, оптимизации и анализа сложных динамических систем. Основано на языке Modelica. [2]

4 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. [3]

Эта модель называется линейным гармоническим осциллятором. Уравнение свободных колебаний гармонического осциллятора имеет следующий вид:

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

где x – переменная, описывающая состояние системы (смещение грузика, 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 $\ddot{x} = \frac{\partial^2 x}{\partial^2 t}, \dot{x} = \frac{\partial x}{\partial t}$)

При отсутствии потерь в системе вместо вышекуказанного уравнения получаем уравнение консервативного осциллятора энергия колебания которого сохраняется во времени

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

Для однозначной разрешимости уравнения второго порядка необходимо задать два начальных условия вида

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

Уравнение консервативного осциллятора энергия колебания которого сохраняется во времени можно представить в виде системы двух уравнений первого порядка:

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

Начальные условия для системы примут вид:

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

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.

5 Выполнение лабораторной работы

1. Начнем выполнения поставленных задач в Julia. Для этого запустим Pluto [4]. (рис. 5.1)

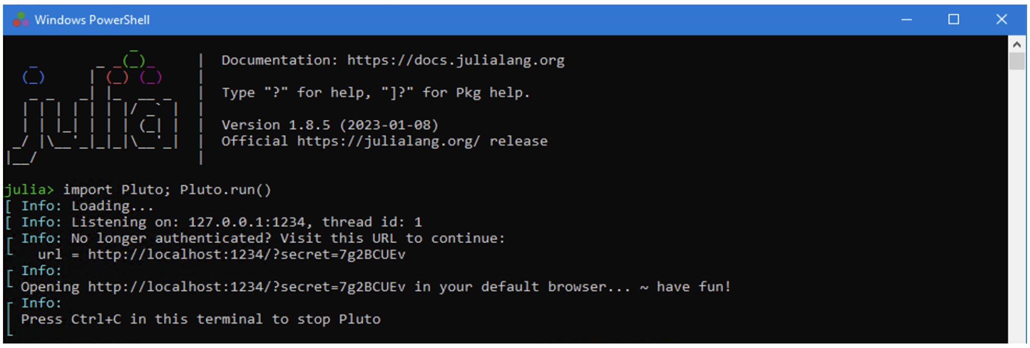


Рис. 5.1: Julia. Запуск Pluto

2. Первым делом подкючим пакеты “Plots” [5] и “DifferentialEquations” [6]. Далее объявим начальные данные при помощи констант. Также объявим начальное условие для системы ДУ и промежуток времени, на котором будет проходить моделирование. (рис. 5.2)

```
# подключение пакетов
using Plots
using DifferentialEquations

# входные данные
const startT = 0
```

```
const endT = 33
const stepT = 0.05
const x0 = 1.3
const y0 = 0.3

# начальные условия
u0 = [x0, y0]
# промежуток времени
spanT = (startT, endT)
```

Рис. 5.2: Julia. Начало написания скрипта для моделирование колебания гармонического осциллятора

3. В следующей ячейке Pluto построим фазовый портрет и решение уравнения гармонического осциллятора. Для этого оъявим параметры осциллятора, а кокретно частоту. Также по аналогии с прошлой лабораторной работой при помощи ‘DifferentialEquations’ зададим и решим систему ДУ, после чего построим график ее решения. Так же создадим два списка, в которых

будут храниться точки уравнений. Воспользуемся данным списком, чтобы построить фазовый портрет. (рис. 5.3, 5.4, 5.5)

```
w = 3.3

# используем DifferentialEquations,
# чтобы описать и решить систему ОДУ
function Fluctuations!(df, u, p, t)
  df[1] = u[2]
  df[2] = -w * u[1]
end

prob = ODEProblem(Fluctuations!, u0, spanT)
sol = solve(prob, dtmax=stepT)

X = [u[1] for u in sol.u]
Y = [u[2] for u in sol.u]

# используем Plots,
# чтобы построить график решения уравнения
plt01 = plot(sol,
      dpi=500,
      xlabel="Время (s)",
      ylabel="x, y",
      legend=false)
savefig(plt01, "artifacts/JL.lab04-010.png")

# используем Plots,
# чтобы построить фазовый портрет
plt02 = plot(X, Y,
```

```
        dpi=500,
        xlabel="x",
        ylabel="y",
        legend=false)
savefig(plt02, "artifacts/JL.lab04-011.png")

println("Success!")
```

Рис. 5.3: Julia. Скрипт. Колебания гармонического осциллятора без затуханий и без действий внешней силы

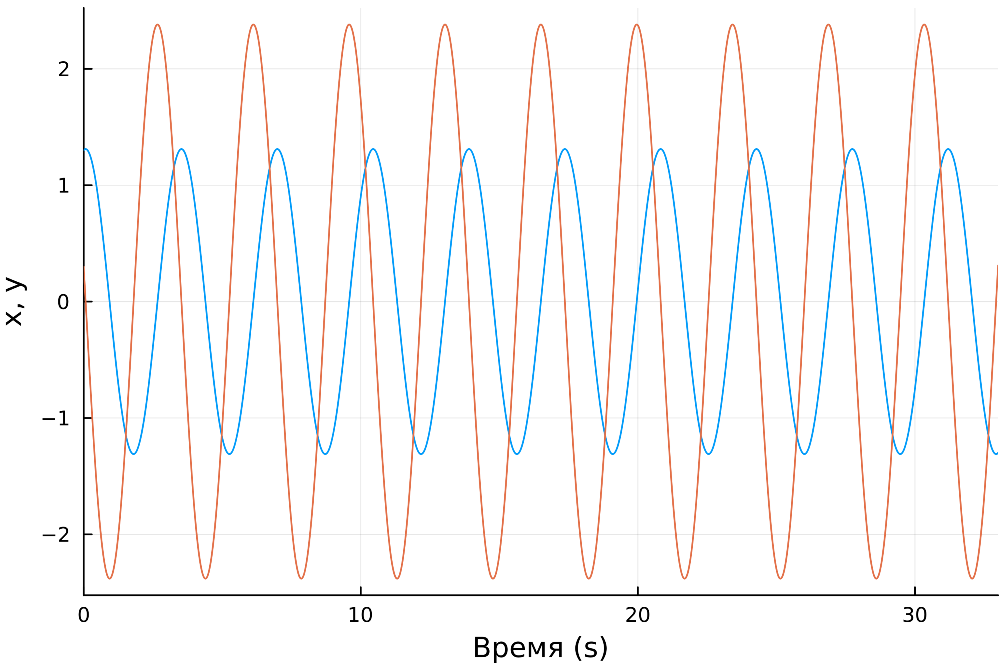


Рис. 5.4: Julia. Модель. Решение уравнения гармонического осциллятора без затуханий и без действий внешней силы

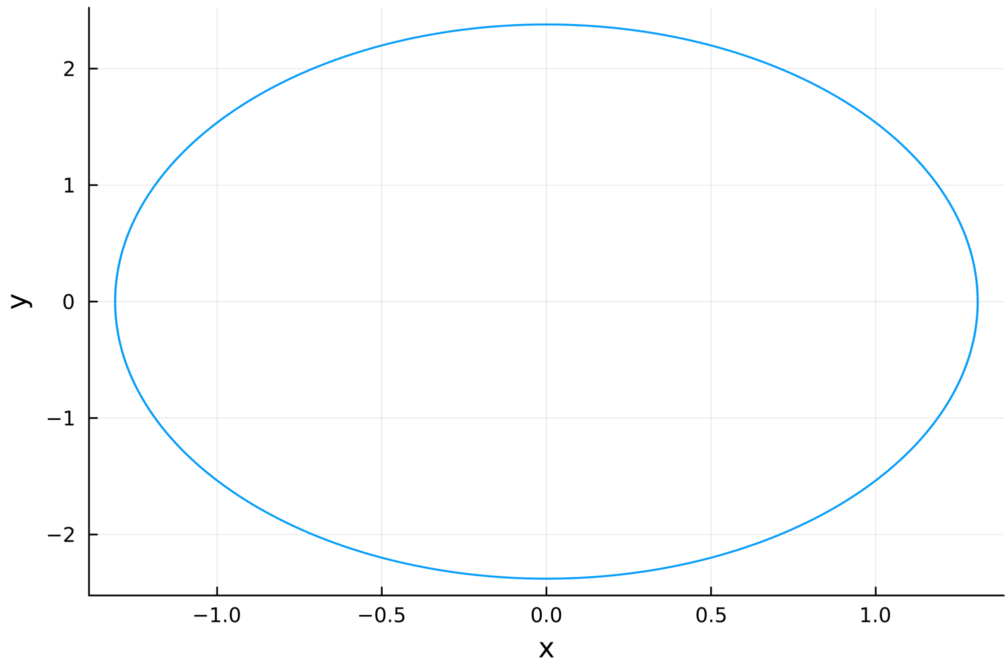


Рис. 5.5: Julia. Модель. Фазовый портрет осциллятора без затуханий и без действий внешней силы

4. Доработаем данный скрипт, чтобы построить решение уравнения и фазовый портрет гармонического осциллятора с затуханием и без действий внешней силы. Для этого нам необходимо добавить новый параметр - затухание. Также необходимо изменить функцию системы ДУ. (рис. 5.6, 5.7, 5.8)

```
w = 0.3 #!
g = 3 #!

function Fluctuations!(df, u, p, t)
  df[1] = u[2]
  df[2] = -w * u[1] - g * u[2] #!
end

prob = ODEProblem(Fluctuations!, u0, spanT)
```

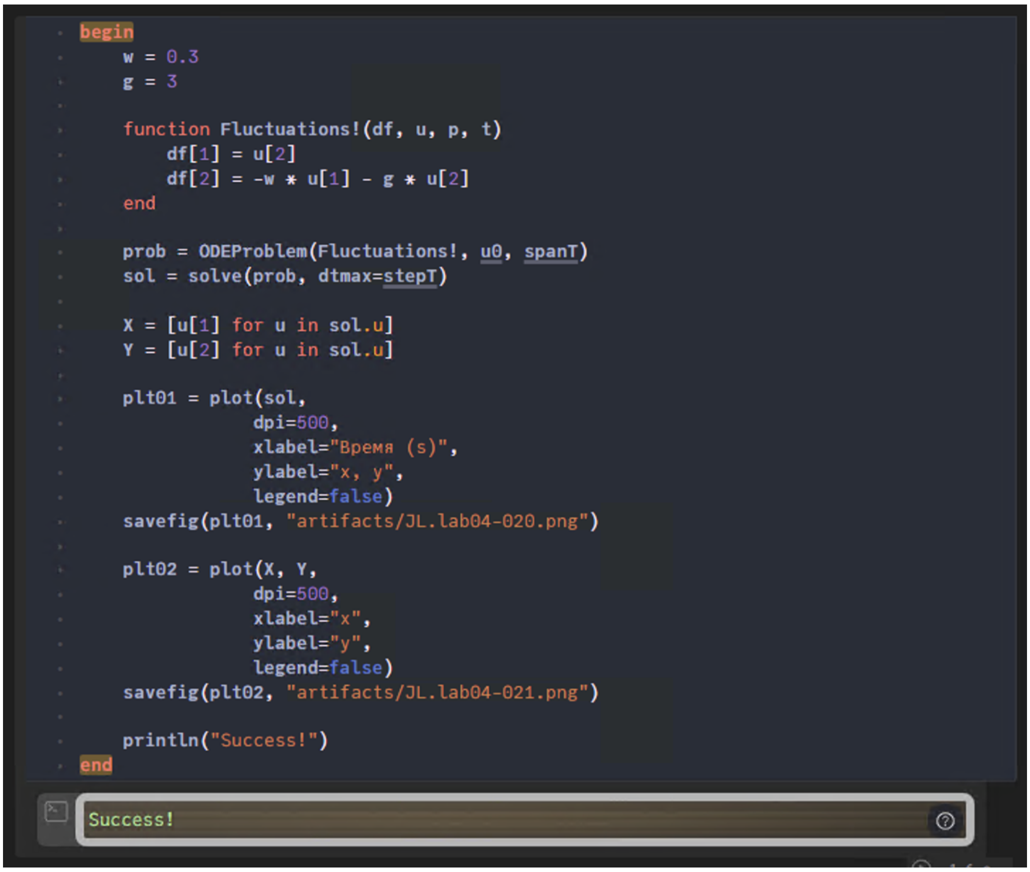
```
sol = solve(prob, dtmax=stepT)

X = [u[1] for u in sol.u]
Y = [u[2] for u in sol.u]

plt01 = plot(sol,
      dpi=500,
      xlabel="Время (s)",
      ylabel="x, y",
      legend=false)
savefig(plt01, "artifacts/JL.lab04-020.png")

plt02 = plot(X, Y,
      dpi=500,
      xlabel="x",
      ylabel="y",
      legend=false)
savefig(plt02, "artifacts/JL.lab04-021.png")

println("Success!")
```



```
begin
    w = 0.3
    g = 3

    function Fluctuations!(df, u, p, t)
        df[1] = u[2]
        df[2] = -w * u[1] - g * u[2]
    end

    prob = ODEProblem(Fluctuations!, u0, spanT)
    sol = solve(prob, dtmax=stepT)

    X = [u[1] for u in sol.u]
    Y = [u[2] for u in sol.u]

    plt01 = plot(sol,
                dpi=500,
                xlabel="Время (s)",
                ylabel="x, y",
                legend=false)
    savefig(plt01, "artifacts/JL.lab04-020.png")

    plt02 = plot(X, Y,
                dpi=500,
                xlabel="x",
                ylabel="y",
                legend=false)
    savefig(plt02, "artifacts/JL.lab04-021.png")

    println("Success!")
end
```

```
Success!
```

Рис. 5.6: Julia. Скрипт. Колебания гармонического осциллятора с затуханием и без действий внешней силы

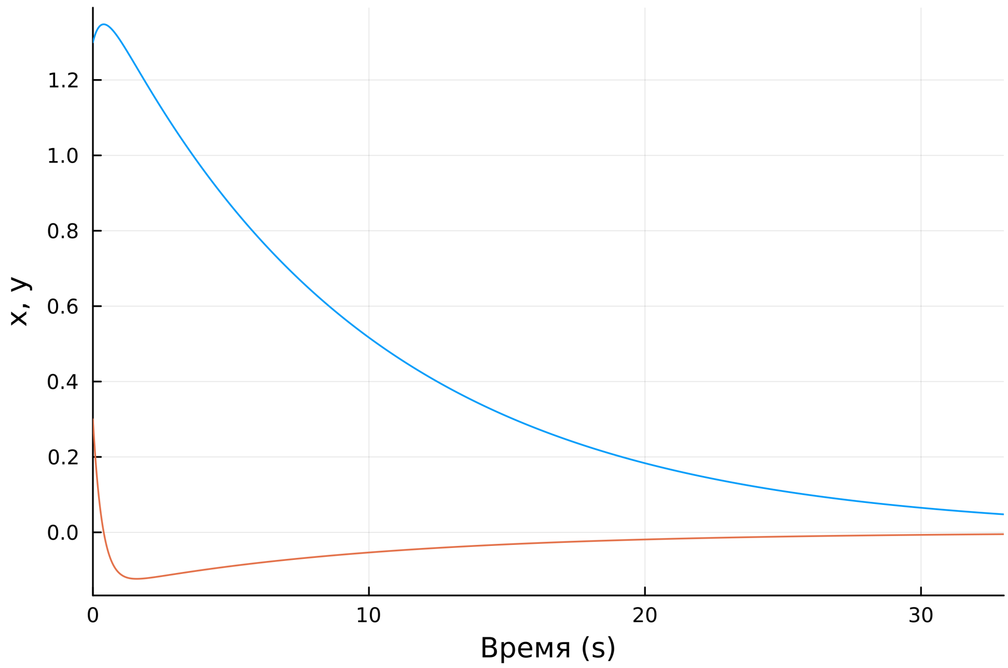


Рис. 5.7: Julia. Модель. Решение уравнения гармонического осциллятора с затуханием и без действий внешней силы

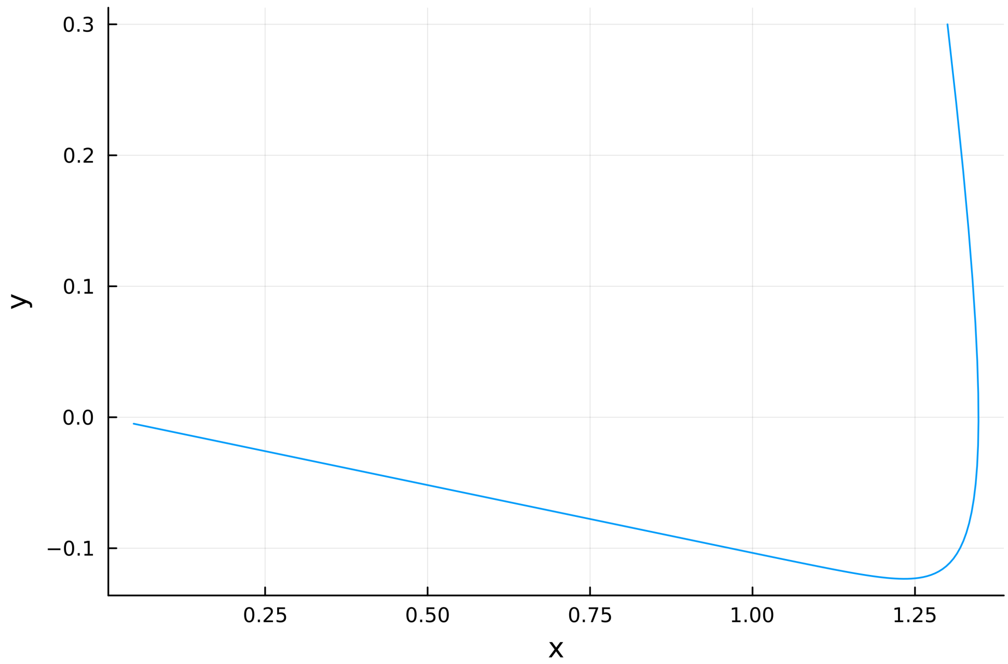


Рис. 5.8: Julia. Модель. Фазовый портрет осциллятора с затуханием и без действий внешней силы

5. Еще раз доработаем скрипт, чтобы построить решение уравнения и фазовый портрет гармонического осциллятора с затуханием и под действием внешней силы. Для этого нам необходимо добавить новый параметр - функция внешней силы. Также необходимо изменить функцию системы ДУ. (рис. 5.9, 5.10, 5.11)

```
w = 3 #!
g = 3.3 #!
f(t) = 3.3 * sin.(3 * t) #!

function Fluctuations!(df, u, p, t)
  df[1] = u[2]
  df[2] = -w * u[1] - g * u[2] - f(t) #!
end
```

```
prob = ODEProblem(Fluctuations!, u0, spanT)
sol = solve(prob, dtmax=stepT)

X = [u[1] for u in sol.u]
Y = [u[2] for u in sol.u]

plt01 = plot(sol,
      dpi=500,
      xlabel="Время (s)",
      ylabel="x, y",
      legend=false)
savefig(plt01, "artifacts/JL.lab04-030.png")

plt02 = plot(X, Y,
      dpi=500,
      xlabel="x",
      ylabel="y",
      legend=false)
savefig(plt02, "artifacts/JL.lab04-031.png")

println("Success!")
```

```julia
begin
    w = 3
    g = 3.3
    f(t) = 3.3 * sin.(3 * t)

    function Fluctuations!(df, u, p, t)
        df[1] = u[2]
        df[2] = -w * u[1] - g * u[2] - f(t)
    end

    prob = ODEProblem(Fluctuations!, u0, spanT)
    sol = solve(prob, dtmax=stepT)

    X = [u[1] for u in sol.u]
    Y = [u[2] for u in sol.u]

    plt01 = plot(sol,
                 dpi=500,
                 xlabel="Время (s)",
                 ylabel="x, y",
                 legend=false)
    savefig(plt01, "artifacts/JL.lab04-030.png")

    plt02 = plot(X, Y,
                 dpi=500,
                 xlabel="x",
                 ylabel="y",
                 legend=false)
    savefig(plt02, "artifacts/JL.lab04-031.png")

    println("Success!")
end
```

```
Success!
```

Рис. 5.9: Julia. Скрипт. Колебания гармонического осциллятора с затуханием и под действием внешней силы

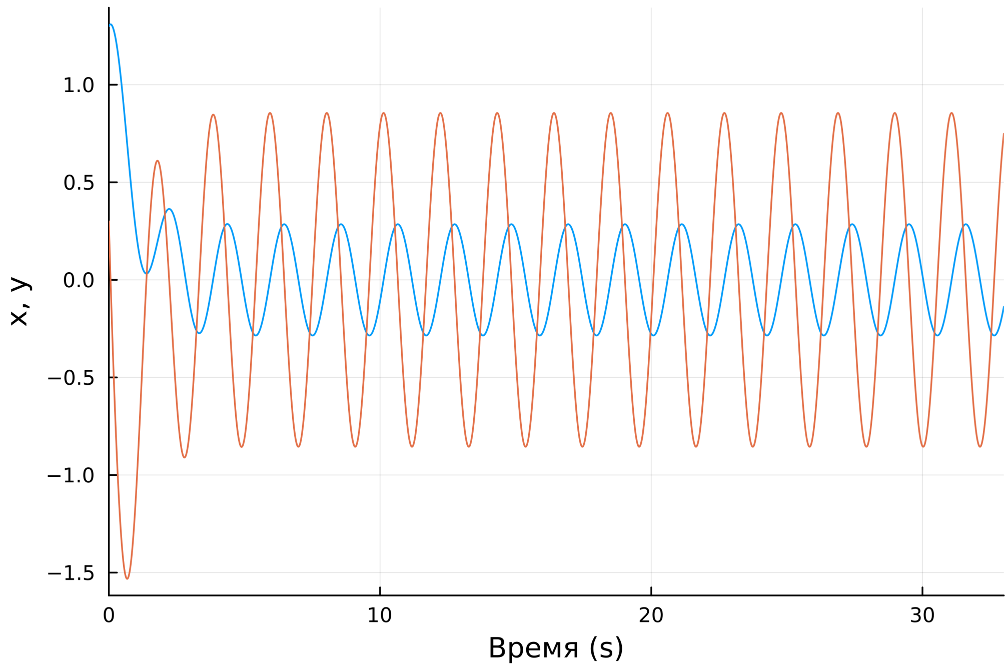


Рис. 5.10: Julia. Модель. Решение уравнения гармонического осциллятора с затуханием и под действием внешней силы

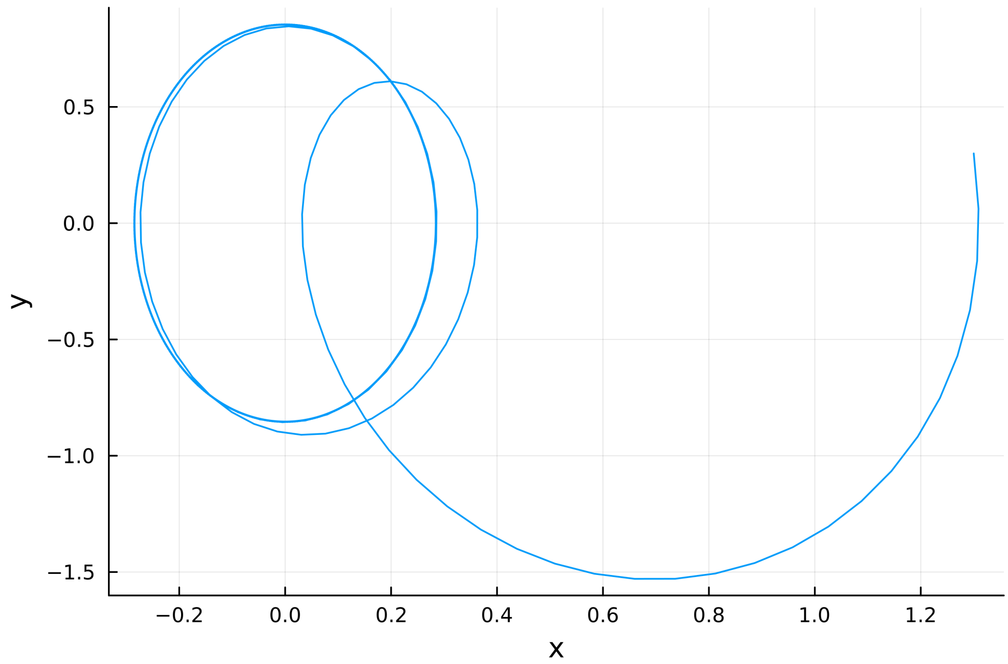
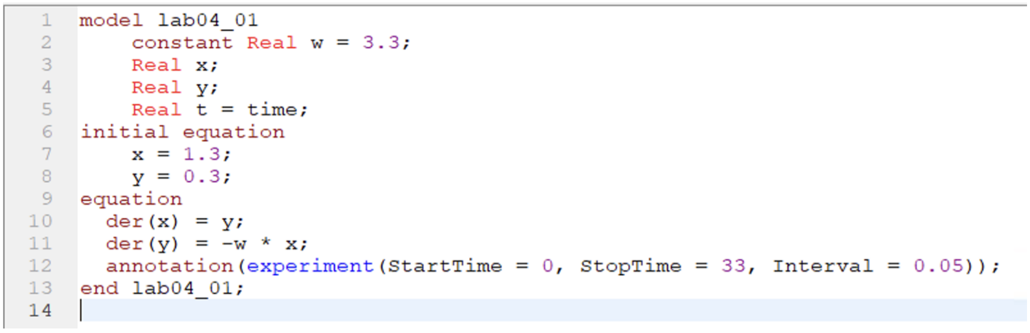


Рис. 5.11: Julia. Модель. Фазовый портрет осциллятора с затуханием и под действием внешней силы

6. Построим модель колебания гармонического осциллятора без затуханий и без действий внешней силы на Modelica. (рис. 5.12, 5.13, 5.14)

```
model lab04_01
    constant Real w = 3.3;
    Real x;
    Real y;
    Real t = time;
initial equation
    x = 1.3;
    y = 0.3;
equation
  der(x) = y;
  der(y) = -w * x;
  annotation(experiment(StartTime = 0, StopTime = 33, Interval = 0.05));
```

```
end lab04_01;
```



```
model lab04_01
    constant Real w = 3.3;
    Real x;
    Real y;
    Real t = time;
initial equation
    x = 1.3;
    y = 0.3;
equation
  der(x) = y;
  der(y) = -w * x;
  annotation(experiment(StartTime = 0, StopTime = 33, Interval = 0.05));
end lab04_01;
```

Рис. 5.12: Modelica. Скрипт. Колебания гармонического осциллятора без затуханий и без действий внешней силы

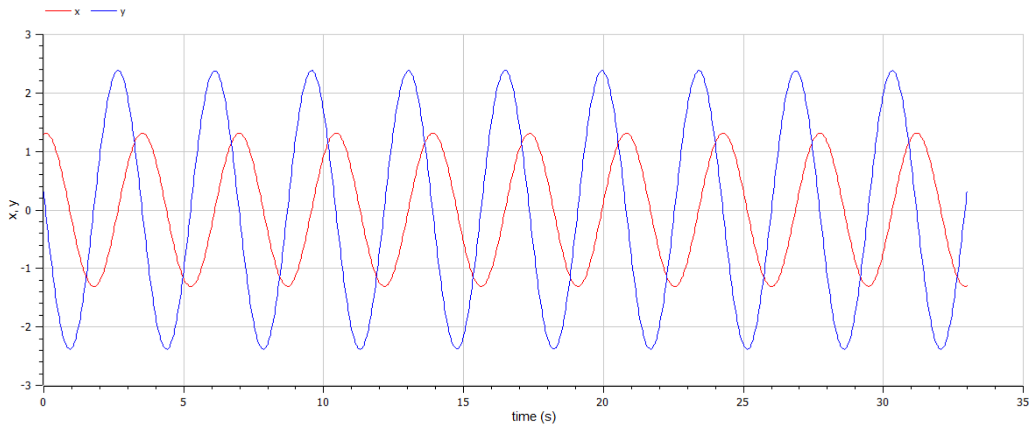


Рис. 5.13: Modelica. Модель. Решение уравнения гармонического осциллятора без затуханий и без действий внешней силы

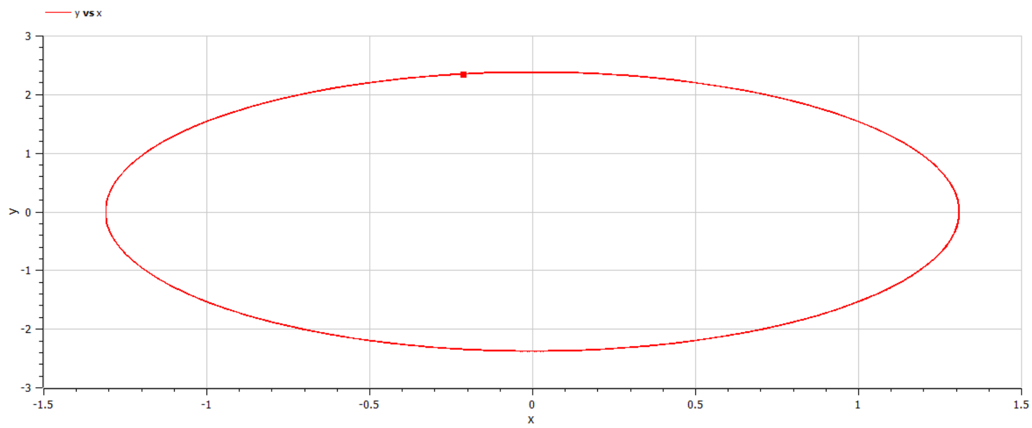


Рис. 5.14: Modelica. Модель. Фазовый портрет осциллятора без затуханий и без действий внешней силы

7. Построим модель колебания гармонического осциллятора с затуханием и без действий внешней силы на Modelica. (рис. 5.15, 5.16, 5.17)

```
model lab04_02
    constant Real w = 0.3;
    constant Real g = 3;
    Real x;
    Real y;
    Real t = time;
initial equation
    x = 1.3;
    y = 0.3;
equation
  der(x) = y;
  der(y) = -w * x - g * y;
  annotation(experiment(StartTime = 0, StopTime = 33, Interval = 0.05));
end lab04_02;
```

```
model lab04_02
    constant Real w = 0.3;
    constant Real g = 3;
    Real x;
    Real y;
    Real t = time;
initial equation
    x = 1.3;
    y = 0.3;
equation
  der(x) = y;
  der(y) = -w * x - g * y;
  annotation(experiment(StartTime = 0, StopTime = 33, Interval = 0.05));
end lab04_02;
```

Рис. 5.15: Modelica. Скрипт. Колебания гармонического осциллятора с затуханием и без действий внешней силы

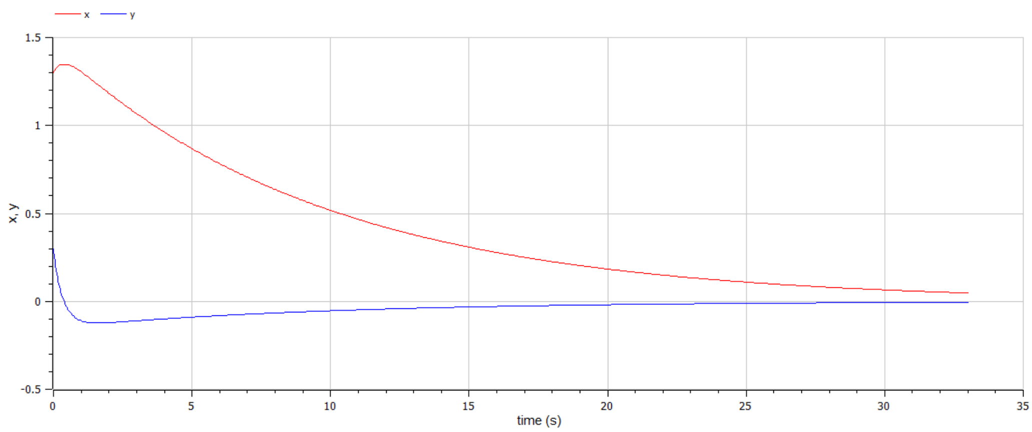


Рис. 5.16: Modelica. Модель. Решение уравнения гармонического осциллятора с затуханием и без действий внешней силы

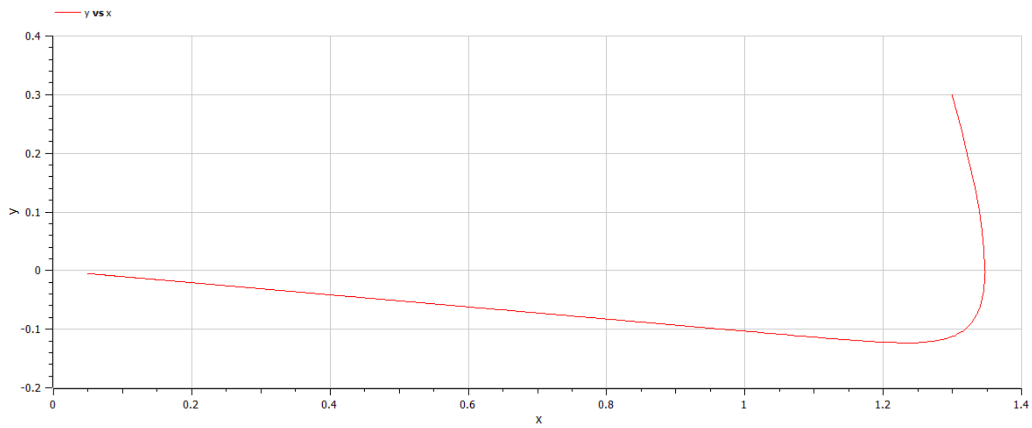


Рис. 5.17: Modelica. Модель. Фазовый портрет осциллятора с затуханием и без действий внешней силы

8. Построим модель колебания гармонического осциллятора с затуханием и под действием внешней силы на Modelica. (рис. 5.18, 5.19, 5.20)

```
model lab04_02
    constant Real w = 0.3;
    constant Real g = 3;
    Real x;
    Real y;
    Real t = time;
initial equation
    x = 1.3;
    y = 0.3;
equation
  der(x) = y;
  der(y) = -w * x - g * y;
  annotation(experiment(StartTime = 0, StopTime = 33, Interval = 0.05));
end lab04_02;
```

```
model lab04_03
    constant Real w = 3;
    constant Real g = 3.3;
    Real x;
    Real y;
    Real t = time;
initial equation
    x = 1.3;
    y = 0.3;
equation
  der(x) = y;
  der(y) = -w * x - g * y - 3.3 * sin(3 * t);
  annotation(experiment(StartTime = 0, StopTime = 33, Interval = 0.05));
end lab04_03;
```

Рис. 5.18: Modelica. Скрипт. Колебания гармонического осциллятора с затуханием и под действием внешней силы

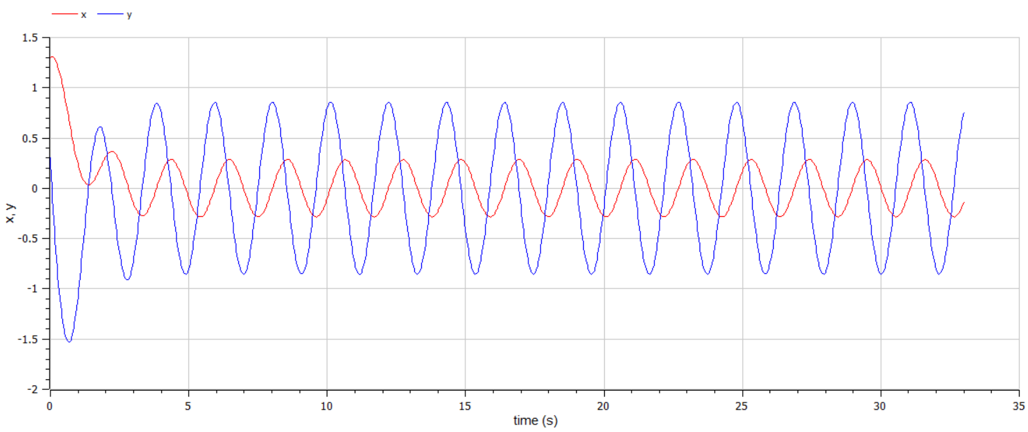


Рис. 5.19: Modelica. Модель. Решение уравнения гармонического осциллятора с затуханием и под действием внешней силы

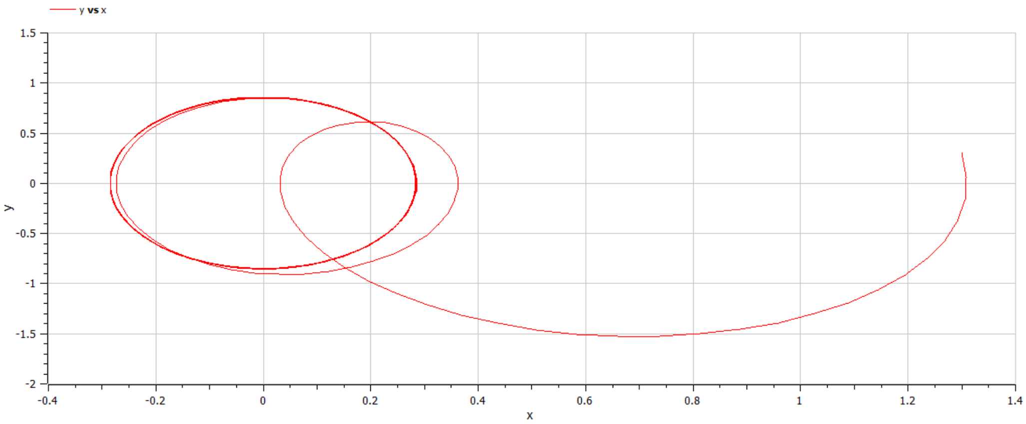


Рис. 5.20: Modelica. Модель. Фазовый портрет осциллятора с затуханием и без под действием внешней силы

6 Анализ результатов

Работа выполненна без непредвиденных проблем в соответствии с руководством. Ошибок и сбоев не произошло.

Моделирование на OMEdit было проще и быстрее, чем при использовании средств Julia. Скрипт на Modelica вышел более лакончиным, понятным и коротким. Более того OpenModelica быстрее обрабатывала скрипт и симмулировала модель. Стоит отметить, что OpenModelica имеет множество разлиных полезных инструментов для настройки с симмуляцией и работой с ней. К плюсам Julia можно отнести, что она является языком программирования, который хорошо подходит для математических и технических задач.

7 Выводы

Мы улучшили практические навыки в области дифференциальных уравнений, улучшили навыки моделирования на Julia, также приобрели навыки моделирования на OpenModelica. Изучили модель голебания гармонического осциллятора. Научились строить фазовые портреты.

Список литературы

1. Julia [Электронный ресурс]. URL: http://www.unn.ru/books/met_files/JULIA_tutorial.pdf.
2. OpenModelica [Электронный ресурс]. URL: https://ru.wikipedia.org/wiki/OpenModelica.
3. Модель гармонических колебаний [Электронный ресурс]. RUDN. URL: https://esystem.rudn.ru/mod/resource/view.php?id=967241.
4. Pluto [Электронный ресурс]. URL: https://plutojl.org/.
5. Plots in Julia [Электронный ресурс]. URL: https://docs.juliaplots.org/latest/tutorial/.
6. Differential Equations in Julia [Электронный ресурс]. URL: https://docs.sciml.ai/DiffEqDocs/stable/getting_started/.