Лабораторная работа №4

Модель гармонических колебаний

Абу Сувейлим Мухаммед Мунифович

Содержание

Список иллюстраций

Цель работы

- Целью работы является познокомится с моделью гармонических колебаний.

Задание

1. Постройте фазовый портрет гармонического осциллятора и решение уравнения гармонического осциллятора для следующих случаев
    - Постройте фазовый портрет гармонического осциллятора и решение уравнения гармонического осциллятора для следующих случаев $\ddot{x} + 6x = 0$;
    - Колебания гармонического осциллятора с затуханием и без действий внешней силы $\ddot{x} + 6\dot{x} + 6x = 0$;
    - Колебания гармонического осциллятора с затуханием и без действий внешней силы $\ddot{x} + 6\dot{x} + 12x = \sin 6t$.

На интервале $t \in [0, 60]$ (шаг 0.05) с начальными условиями $x_0 = 0.6$ и $y_0 = 1.6$

Теоретическое введение

Гармонические колебания – это одно из основных понятий в физике, которое широко применяется для описания различных явлений и систем. Они характеризуются регулярным и повторяющимся движением вокруг равновесного положения. Гармонические колебания имеют важное значение в различных областях, включая механику, электродинамику, акустику и оптику.

Выполнение лабораторной работы

Модлеирование на языке программеровании Julia

Колебания гармонического осциллятора без затуханий и без действий внешней силы Julia

1. Во-первых, я использвал пакеты Plots и DifferentialEquations.

   ```
   using Plots
   using DifferentialEquations
   ```

2. Инициализировал нужны нам константи и функции в моделии. w - частота; g - затухание; f(t) - внешняя сила, действующая на осциллятор. Вместо урванения второго порядка я написал систему из двух уравнений первого порядка caseOne.

   ```
   #x'' + g * x' + w^2 * x = f(t)
   #для колебании гармонического осциллятора без затуханий и без действий внешн
   #g - затухание
   #w - частота
   #f(t) - внешняя сила, действующая на осциллятор


   w = 6
   ```

```
g = 0
#f(t) = 0

function caseOne(du, u, p, t)
   x, y = u
   du[1] = u[2]
   du[2] = -g*u[2] - w*u[1]
end
```

3. Далее я обозначал наши начальные параметры, которые были нам данни в задании.

```
x0 = 0.6
y0 = 1.6
u0 = [x0, y0]
tspan = (0, 60) #интервал временни
```

4. Теперь я начал решать диффернцалбное уравнение.

```
probOne = ODEProblem(caseOne, u0, tspan)
```

5. Осталось только решить ОДУ.

```
solOne = solve(probOne, dtmax = 0.05)
```

6. Здесь я переименавал названия переменных.

```
X = [u[1] for u in solOne.u]
Y = [u[2] for u in solOne.u]
Time = [t for t in solOne.t]
```

7. Далее я поготовил место для графиков.

```
plt = plot(layout = (1, 2), dpi = 300, legend = false)
```

8. Наконец, построил два графика.

```
plot!(
 plt[1],
 Time,
 X,
 title = "Решение Уравнения",
 color=:red)
plot!(
 plt[2],
 X,
 Y,
 title = "Фазовый Портрет",
 color=:blue)
```

9. Получуные график.

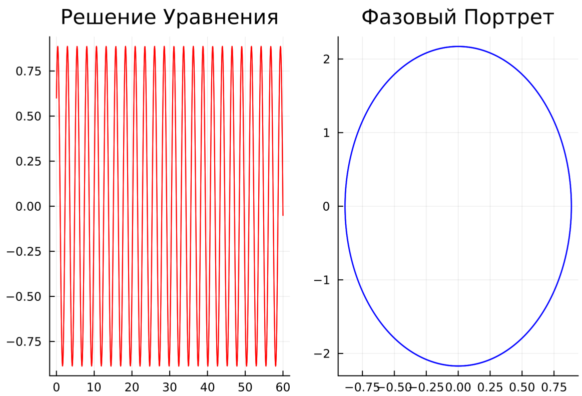


Рис. 1: Case One Julia

Колебания гармонического осциллятора с затуханием и без действий внешней силы Julia

1. Во-первых, я использвал пакеты Plots и DifferentialEquations.

```
using Plots
using DifferentialEquations
```

2. Инициализировал нужны нам константи и функции в моделии. w - частота; g - затухание; f(t) - внешняя сила, действующая на осциллятор. Вместо урванения второго порядка я написал систему из двух уравнений первого порядка caseOne.

```
#x'' + g * x' + w^2 * x = f(t)
#для колебании гармонического осциллятора с затуханием и без действий внешне
#g - затухание
#w - частота
#f(t) - внешняя сила, действующая на осциллятор

w = 6
g = 6
#f(t) = 0

function caseTwo(du, u, p, t)
   x, y = u
   du[1] = u[2]
   du[2] = -g*u[2] - w*u[1]
end
```

3. Далее я обозначал наши начальные параметры, которые были нам данни в задании.

```
x0 = 0.6
y0 = 1.6
u0 = [x0, y0]
tspan = (0, 60) #интервал временни
```

4. Теперь я начал решать диффернцалбное уравнение.

```
probTwo = ODEProblem(caseTwo, u0, tspan)
```

5. Осталось только решить ОДУ.

```
solTwo = solve(probTwo, dtmax = 0.05)
```

6. Здесь я переименавал названия переменных.

```
X = [u[1] for u in solTwo.u]
Y = [u[2] for u in solTwo.u]
Time = [t for t in solTwo.t]
```

7. Далее я поготовил место для графиков.

```
plt = plot(layout = (1, 2), dpi = 300, legend = false)
```

8. Наконец, построил два графика.

```
plot!(
 plt[1],
 Time,
 X,
 title = "Решение Уравнения",
 color=:red)
plot!(
 plt[2],
 X,
 Y,
 title = "Фазовый Портрет",
 color=:blue)
```

9. Получуные график.

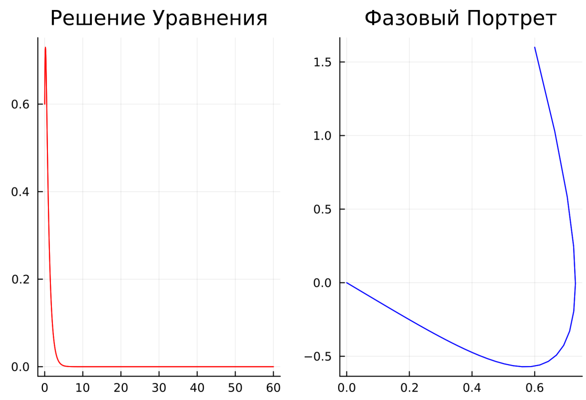


Рис. 2: Case Two Julia

Колебания гармонического осциллятора с затуханием и под действием внешней силы Julia

1. Во-первых, я использвал пакеты Plots и DifferentialEquations.

```
using Plots
using DifferentialEquations
```

2. Инициализировал нужны нам константи и функции в моделии. w - частота; g - затухание; f(t) - внешняя сила, действующая на осциллятор. Вместо урванения второго порядка я написал систему из двух уравнений первого порядка caseOne.

```
#x'' + g * x' + w^2 * x = f(t)
##x'' = - g * x' - w^2 * x + f(t)
#для колебании гармонического осциллятора с затуханием и без действий внешне
#g - затухание
#w - частота
#f(t) - внешняя сила, действующая на осциллятор

w = 12
g = 6
```

3. Далее я обозначал наши начальные параметры, которые были нам данни в задании.

```
x0 = 0.6
y0 = 1.6
u0 = [x0, y0]
tspan = (0, 60) #интервал временни
```

4. Теперь я начал решать диффернцалбное уравнение.

```
probThree = ODEProblem(caseThree, u0, tspan)
```

5. Осталось только решить ОДУ.

```
solThree = solve(probThree, dtmax = 0.05)
```

6. Здесь я переименавал названия переменных.

```
X = [u[1] for u in solThree.u]
Y = [u[2] for u in solThree.u]
Time = [t for t in solThree.t]
```

7. Далее я поготовил место для графиков.

```
plt = plot(layout = (1, 2), dpi = 300, legend = false)
```

8. Наконец, построил два графика.

```
plot!(
 plt[1],
 Time,
 X,
 title = "Решение Уравнения",
 color=:red)
plot!(
 plt[2],
```

```
X,
Y,
title = "Фазовый Портрет",
color=:blue)
```

9. Получуные график.

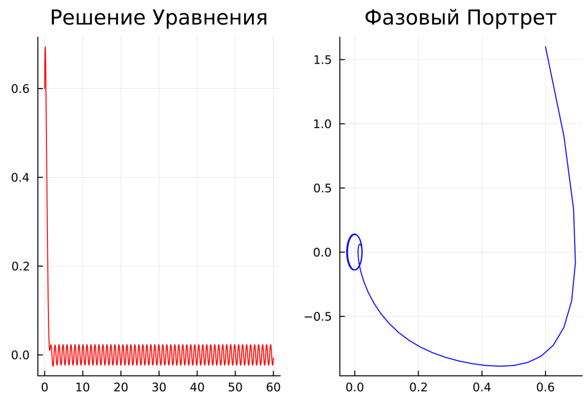


Рис. 3: Case Two Julia

Модлеирование на языке программеровании OpenModelica

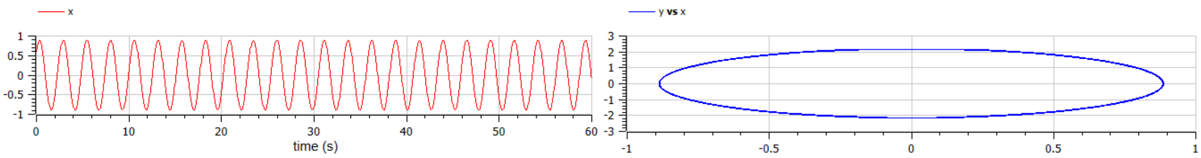
Колебания гармонического осциллятора без затуханий и без действий внешней силы OpenModelica

1. В OpenModelica все прощее. Я просто переписал код из Julia. В этой прошраиие все величины имею тот же смысл, что и в Julia.

```
model lab4_1
Real x;
Real y;
Real w = 6;
Real g = 0;
Real t = time;
```

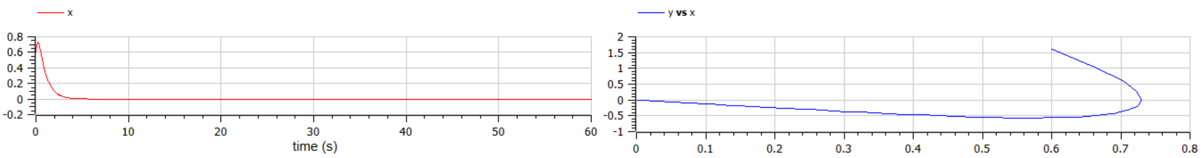
```
initial equation

x = 0.6;
y = 1.6;

equation

der(x) = y;
der(y) = - g*y - w*x;

end lab4_1;
```

2. График в OpenModelica указывает на решению уравнений и фазовый портрет.



Колебания гармонического осциллятора с затуханием и без действий внешней силы OpenModelica

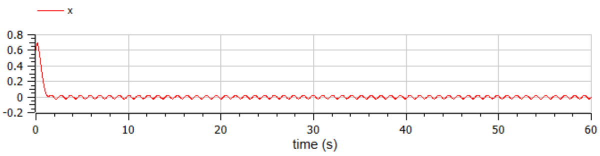
1. График в OpenModelica указывает на решению уравнений и фазовый портрет.

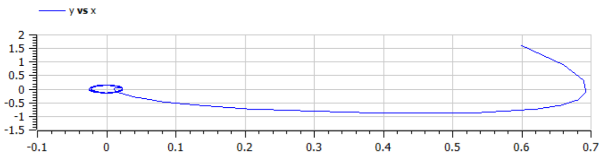


Колебания гармонического осциллятора с затуханием и под действием внешней силы

OpenModelica

1. График в OpenModelica указывает на решению уравнений и фазовый портрет.





Исходный код

Julia

1. Колебания гармонического осциллятора без затуханий и без действий внешней силы

```
using Plots
using DifferentialEquations

#x'' + g * x' + w^2 * x = f(t)
#для колебании гармонического осциллятора без затуханий и без действий внешн
#g - затухание
#w - частота
#f(t) - внешняя сила, действующая на осциллятор

w = 6
g = 0
#f(t) = 0

function caseOne(du, u, p, t)
    x, y = u
```

```
    du[1] = u[2]
    du[2] = -g*u[2] - w*u[1]
end
x0 = 0.6
y0 = 1.6
u0 = [x0, y0]
tspan = (0, 60) #интервал временни
probOne = ODEProblem(caseOne, u0, tspan)
solOne = solve(probOne, dtmax = 0.05)
X = [u[1] for u in solOne.u]
Y = [u[2] for u in solOne.u]
Time = [t for t in solOne.t]
plt = plot(layout = (1, 2), dpi = 300, legend = false)
plot!(
    plt[1],
    Time,
    X,
    title = "Решение Уравнения",
    color=:red)
plot!(
    plt[2],
    X,
    Y,
    title = "Фазовый Портрет",
    color=:blue)
```

2. Колебания гармонического осциллятора с затуханием и без действий внешней силы

```
using Plots
using DifferentialEquations
```

```
#x'' + g * x' + w^2 * x = f(t)
#для колебании гармонического осциллятора без затуханий и без действий внешн
#g - затухание
#w - частота
#f(t) - внешняя сила, действующая на осциллятор

w = 6
g = 6
#f(t) = 0

function caseTwo(du, u, p, t)
   x, y = u
   du[1] = u[2]
   du[2] = -g*u[2] - w*u[1]
end
x0 = 0.6
y0 = 1.6
u0 = [x0, y0]
tspan = (0, 60) #интервал временни
probTwo = ODEProblem(caseTwo, u0, tspan)
solTwo = solve(probTwo, dtmax = 0.05)
X = [u[1] for u in solTwo.u]
Y = [u[2] for u in solTwo.u]
Time = [t for t in solTwo.t]
plt = plot(layout = (1, 2), dpi = 300, legend = false)
plot!(
   plt[1],
   Time,
   X,
```

```
    title = "Решение Уравнения",
    color=:red)
plot!(
    plt[2],
    X,
    Y,
    title = "Фазовый Портрет",
    color=:blue)
```

3. Колебания гармонического осциллятора с затуханием и под действием внешней силы.

```
using Plots
using DifferentialEquations
#x'' + g * x' + w^2 * x = f(t)
#для колебании гармонического осциллятора без затуханий и без действий внешн
#g - затухание
#w - частота
#f(t) - внешняя сила, действующая на осциллятор

w = 12
g = 6

function caseThree(du, u, p, t)
    x, y = u
    du[1] = u[2]
    du[2] = sin(6*t) -g*u[2] - w*u[1]
end
x0 = 0.6
y0 = 1.6
u0 = [x0, y0]
```

```
tspan = (0, 60) #интервал временни
probThree = ODEProblem(caseThree, u0, tspan)
solThree = solve(probThree, dtmax = 0.05)
X = [u[1] for u in solThree.u]
Y = [u[2] for u in solThree.u]
Time = [t for t in solThree.t]
plt = plot(layout = (1, 2), dpi = 300, legend = false)
plot!(
   plt[1],
   Time,
   X,
   title = "Решение Уравнения",
   color=:red)
plot!(
   plt[2],
   X,
   Y,
   title = "Фазовый Портрет",
   color=:blue)
```

OpenModelica

1. Колебания гармонического осциллятора без затуханий и без действий внешней силы

```
model lab4_1
Real x;
Real y;
Real w = 6;
Real g = 0;
```

```
Real t = time;

initial equation

x = 0.6;
y = 1.6;

equation

der(x) = y;
der(y) = - g*y - w*x;

end lab4_1;
```

2. Колебания гармонического осциллятора с затуханием и без действий внешней силы

```
model lab4_2

Real x;
Real y;
Real w = 6;
Real g = 6;
Real t = time;

initial equation

x = 0.6;
y = 1.6;

equation
```

```
der(x) = y;
der(y) = - g*y - w*x;

end lab4_2;
```

3. Колебания гармонического осциллятора с затуханием и под действием внешней силы

```
model lab4_3

Real x;
Real y;
Real w = 12;
Real g = 6;
Real t = time;

initial equation

x = 0.6;
y = 1.6;

equation

der(x) = y;
der(y) = sin(6*t) - g*y - w*x;

end lab4_3;
```

Вывод

- Движение грузика на пружинке, а также эволюция во времени многих систем можно описать одним и тем же дифференциальным уравнением, которое в теории колебаний выступает в качестве основной модели
- Можно построить моделт гармонического колебаний осциллятора без затуханий / с затуханием и без действий / под действием внешней силы

Библиография

1. Julia 1.10 Documentation // Julia URL: https://docs.julialang.org/en/v1/ (дата обращения: 24.02.2024).

2. Медведев Д. А., Куперштох А. Л., Прууэл Э. Р., Сатонкина Н. П., Карпов Д. И. Моделирование физических процессов и явлений на ПК: Учеб. пособие / Новосибирск: Новосиб. гос. ун-т., 2010. — 101 с