

Отчёт по лабораторной работе №2

Задача о погоне

Желдакова Виктория Алексеевна

Содержание

1	Цель работы	5
2	Задание	6
2.1	Вариант 16	6
3	Теоретическое введение	7
3.1	Справка о языках программирования	7
3.2	Математическая справка	7
4	Выполнение лабораторной работы	9
4.1	Математическая модель	9
4.2	Решение с помощью языков программирования	11
4.2.1	OpenModelica	11
4.2.2	Julia	11
5	Выводы	16
	Список литературы	17

Список иллюстраций

4.1	Установка Julia	11
4.2	Проверка установки пакетов	12
4.3	Запуск программы и её вывод	14
4.4	График для первого случая	14
4.5	График для второго случая	15

Список таблиц

1 Цель работы

Изучить основы языков Julia и OpenModelica. Познакомиться с библиотеками Plots и DifferentialEquations для построения графиков и решения дифференциальных уравнений. Решить задачу о погоне с использованием обоих языков.

2 Задание

2.1 Вариант 16

На море в тумане катер береговой охраны преследует лодку браконьеров. Через определенный промежуток времени туман рассеивается, и лодка обнаруживается на расстоянии 9,5 км от катера. Затем лодка снова скрывается в тумане и уходит прямолинейно в неизвестном направлении. Известно, что скорость катера в 3,3 раза больше скорости браконьерской лодки.

1. Запишите уравнение, описывающее движение катера, с начальными условиями для двух случаев (в зависимости от расположения катера относительно лодки в начальный момент времени).
2. Постройте траекторию движения катера и лодки для двух случаев.
3. Найдите точку пересечения траектории катера и лодки

3 Теоретическое введение

3.1 Справка о языках программирования

Julia — высокоуровневый свободный язык программирования с динамической типизацией, созданный для математических вычислений. Эффективен также и для написания программ общего назначения. Синтаксис языка схож с синтаксисом других математических языков (например, MATLAB и Octave), однако имеет некоторые существенные отличия. Julia написан на Си, C++ и Scheme. Имеет встроенную поддержку многопоточности и распределённых вычислений, реализованные в том числе в стандартных конструкциях.

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. OpenModelica используется в академической среде и на производстве. В промышленности используется в области оптимизации энергоснабжения, автомобилестроении и водоочистке.

3.2 Математическая справка

Дифференциальное уравнение — уравнение, которое помимо функции содержит её производные. Порядок входящих в уравнение производных может быть

различен (формально он ничем не ограничен). Производные, функции, независимые переменные и параметры могут входить в уравнение в различных комбинациях или отсутствовать вовсе, кроме хотя бы одной производной. Не любое уравнение, содержащее производные неизвестной функции, является дифференциальным.

Дифференциальные уравнения являются частным случаем функциональных уравнений. В отличие от алгебраических уравнений, в результате решения которых ищется число (несколько чисел), при решении дифференциальных уравнений ищется функция (семейство функций).

4 Выполнение лабораторной работы

4.1 Математическая модель

Согласно варианту расстояние между лодкой и катером равно 9,5 км, а отношение скорости катера в 3,3 раза больше скорости браконьерской лодки.

Введём полярные координаты с центром в точке нахождения браконьеров и осью, проходящей через катер береговой охраны. Траектория катера должна быть такой, чтобы и катер, и лодка все время были на одном расстоянии от полюса, только в этом случае траектория катера пересечется с траекторией лодки.

Поэтому для начала катер береговой охраны должен двигаться некоторое время прямолинейно, пока не окажется на том же расстоянии от полюса, что и лодка браконьеров. После этого катер береговой охраны должен двигаться вокруг полюса удаляясь от него с той же скоростью, что и лодка браконьеров.

Чтобы найти расстояние x (расстояние после которого катер начнет двигаться вокруг полюса), необходимо составить простое уравнение. Пусть через время t катер и лодка окажутся на одном расстоянии x от полюса. За это время лодка пройдет x , а катер $9,5 + x$ (или $9,5 - x$, в зависимости от начального положения катера относительно полюса). Время, за которое они пройдут это расстояние, вычисляется как $\frac{x}{v}$ или $\frac{9,5 - x}{3,3v}$ (во втором случае $\frac{9,5 + x}{3,3v}$). Так как время одно и то же, то эти величины одинаковы. Мы получаем объединение из двух уравнений для двух различных начальных положений катера:

$$\begin{cases} \frac{x}{v} = \frac{9,5-x}{3,3v} \\ \frac{x}{v} = \frac{9,5+x}{3,3v} \end{cases}$$

Из данных уравнений можно найти расстояние, после которого катер начнёт раскручиваться по спирали. Для данных уравнений решения будут следующими: $x_1 = \frac{95}{43}$, $x_2 = \frac{95}{23}$. Задачу будем решать для двух случаев. После того, как катер береговой охраны окажется на одном расстоянии от полюса, что и лодка, он должен сменить прямолинейную траекторию и начать двигаться вокруг полюса удаляясь от него со скоростью лодки v . Для этого скорость катера раскладываем на две составляющие: $v_r = \frac{dr}{dt} = v$ - радиальная скорость и $v_\tau = r \frac{d\theta}{dt}$ - тангенциальная скорость.

4. Решение исходной задачи сводится к решению системы из двух дифференциальных уравнений:

$$\begin{cases} \frac{dr}{dt} = v \\ r \frac{d\theta}{dt} = \sqrt{8925}v \end{cases}$$

с начальными условиями

$$\begin{cases} \theta_0 = 0 \\ r_0 = x_1 = \frac{95}{43} \end{cases}$$

или

$$\begin{cases} \theta_0 = -\pi \\ r_0 = x_2 = \frac{95}{23} \end{cases}$$

Исключая из полученной системы производную по t , можно перейти к следующему уравнению (с неизменными начальными условиями):

$$\frac{dr}{d\theta} = \frac{r}{\sqrt{8925}}$$

Решением этого уравнения с заданными начальными условиями и будет являться траектория движения катера в полярных координатах.

4.2 Решение с помощью языков программирования

4.2.1 OpenModelica

Реализация решения данной задачи невозможна с помощью OpenModelica, т.к. в ней не поддерживаются полярные координаты. [1]

4.2.2 Julia

Установим Julia (рис. 4.1).



Рис. 4.1: Установка Julia

Установим пакеты Plots и DifferentialEquations для создания графиков и решения дифференциальных уравнений соответственно и проверим их установку

(рис. 4.2). [2]

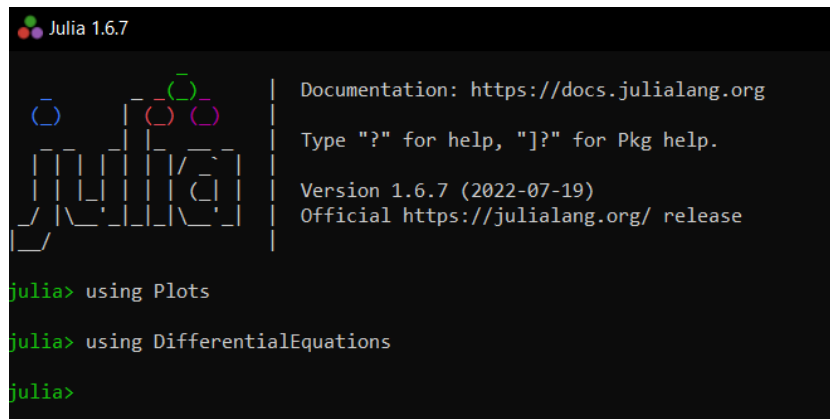
A screenshot of the Julia REPL interface. The title bar shows 'Julia 1.6.7'. The left pane displays a stylized ASCII art logo of the letter 'J' composed of various colored characters. The right pane shows the following text: 'Documentation: https://docs.julialang.org', 'Type "?" for help, "]"? for Pkg help.', 'Version 1.6.7 (2022-07-19)', and 'Official https://julialang.org/ release'. The command prompt shows the following commands and their outputs: 'julia> using Plots' (no output), 'julia> using DifferentialEquations' (no output), and 'julia>' (no output).

Рис. 4.2: Проверка установки пакетов

Напишем программу для решения нашей задачи. Код программы:

```
using Plots
using DifferentialEquations

const k = 12.2
const n = 4.1

const r0 = k/(n + 1)
const r02 = k/(n - 1)

const T = (0, 2*pi)
const T2 = (-pi, pi)

function F(u, p, t)
    return u / sqrt(n*n - 1)
end

problem = ODEProblem(F, r0, T)
```

```

result = solve(problem, abstol=1e-8, reltol=1e-8)
@show result.u
@show result.t

dxR = rand(1:size(result.t)[1])
rAngles = [result.t[dxR] for i in 1:size(result.t)[1]]

plt = plot(proj=:polar, aspect_ratio=:equal, dpi = 1000, legend=true, bg=:white)

plot!(plt, xlabel="theta", ylabel="r(t)", title="Задача о погоне -
случай 1", legend=:outerbottom)
plot!(plt, [rAngles[1], rAngles[2]], [0.0, result.u[size(result.u)[1]]], label="Путь
scatter!(plt, rAngles, result.u, label="", mc=:blue, ms=0.0005)
plot!(plt, result.t, result.u, xlabel="theta", ylabel="r(t)", label="Путь катера",
scatter!(plt, result.t, result.u, label="", mc=:green, ms=0.0005)

savefig(plt, "lab02_01.png")

problem = ODEProblem(F, r02 , T2)
result = solve(problem, abstol=1e-8, reltol=1e-8)
dxR = rand(1:size(result.t)[1])
rAngles = [result.t[dxR] for i in 1:size(result.t)[1]]

plt1 = plot(proj=:polar, aspect_ratio=:equal, dpi = 1000, legend=true, bg=:white)

plot!(plt1, xlabel="theta", ylabel="r(t)", title="Задача о погоне -
случай 2", legend=:outerbottom)
plot!(plt1, [rAngles[1], rAngles[2]], [0.0, result.u[size(result.u)[1]]], label="Путь
scatter!(plt1, rAngles, result.u, label="", mc=:blue, ms=0.0005)

```

```
plot!(plt1, result.t, result.u, xlabel="theta", ylabel="r(t)", label="Путь катера",
scatter!(plt1, result.t, result.u, label="", mc=:green, ms=0.0005)

savefig(plt1, "lab02_02.png")
```

Запустим программу (рис. 4.3).

```
PS C:\Users\User\Documents\work\study\2023-2024\Математическое моделирование\m
athmod\labs\lab02> julia lab02.jl
result.u = [2.392156862745098, 2.4210416156098287, 2.6004733828958853, 2.92803
16393866026, 3.361113841947996, 3.927901652954028, 4.662711002542758, 5.610905
703769135, 6.830054841787026, 8.394584389620817, 10.399764856118955, 11.616218
145962364]
result.t = [0.0, 0.047723911275813456, 0.3320033451151524, 0.8037249159730531,
1.3522070799947596, 1.9718267271582297, 2.653708381528198, 3.389759214853756,
4.171557738799711, 4.9916608921545995, 5.843344188440021, 6.283185307179586]
```

Рис. 4.3: Запуск программы и её вывод

В результате работы программы получаем графики для обоих случаев началь-
ного положения катера относительно полюса (рис. 4.4) (рис. 4.5).

Задача о погоне - случай 1

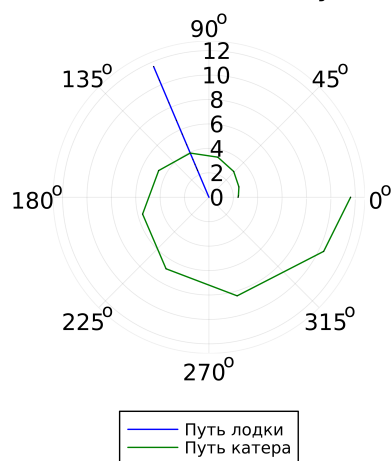


Рис. 4.4: График для первого случая

Задача о погоне - случай 2

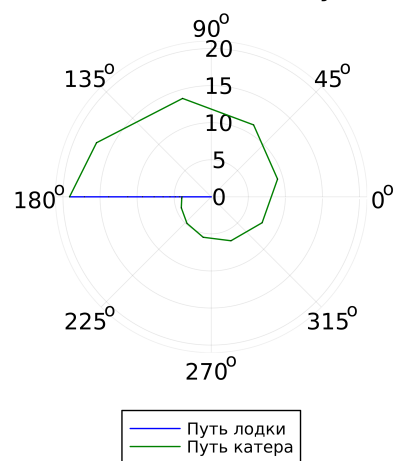


Рис. 4.5: График для второго случая

5 Выводы

Изучила основы языков Julia и OpenModelica. Познакомилась с библиотеками Plots и DifferentialEquations для построения графиков и решения дифференциальных уравнений. Решила задачу о погоне только с использованием языка Julia, т.к. OpenModelica не поддерживает работу с полярными координатами.

Список литературы

[1] Документация по OpenModelica: <https://openmodelica.org/>

[2] Документация по Julia: <https://docs.julialang.org/en/v1/>