

Презентация к лабораторной работе 2

Задача о погоне

Аристова А. О.

17 февраля 2024

Российский университет дружбы народов, Москва, Россия

Информация

- Аристова Арина Олеговна
 - студентка группы НФбд-01-21
 - Российский университет дружбы народов
 - 1032216433@rudn.ru
 - <https://github.com/aoaristova>



Вводная часть

- Ознакомиться с основами языков программирования Julia и OpenModelica.
- Освоить библиотеки этих языков, которые необходимы для построения графиков и решения дифференциальных уравнений.
- Решить задачу «о погоне».

Вариант 4:

На море в тумане катер береговой охраны преследует лодку браконьеров. Через определенный промежуток времени туман рассеивается, и лодка обнаруживается на расстоянии 8,5 км от катера. Затем лодка снова скрывается в тумане и уходит прямолинейно в неизвестном направлении. Известно, что скорость катера в 3,5 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

О языках программирования

Julia – высокоуровневый язык, который разработан для научного программирования. Язык поддерживает широкий функционал для математических вычислений и работы с большими массивами данных.

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. По своим возможностям приближается к таким вычислительным средам как Matlab Simulink, Scilab xCos, имея при этом значительно более удобное представление системы уравнений исследуемого блока.

Математическая составляющая

Дифференциальное уравнение содержит помимо функции, содержит ее производные. Порядок производных в уравнении может быть разным (не ограничен формально). В уравнении могут присутствовать производные, функции, независимые переменные и параметры в различных комбинациях или даже отсутствовать, за исключением хотя бы одной производной. Не каждое уравнение с производными неизвестной функции является дифференциальным.

В отличие от алгебраических уравнений, которые решаются для нахождения числа (или нескольких чисел), решение дифференциальных уравнений направлено на поиск функции (или семейства функций).

Дифференциальное уравнение высшего порядка можно преобразовать в систему уравнений первого порядка, где количество уравнений равно порядку исходного дифференциального уравнения.

Физическая составляющая

- Тангенциальная скорость - компонента вектора скорости, перпендикулярная линии, соединяющей источник и наблюдателя. Измеряется через собственное движение - угловое перемещение источника.
- Радиальная скорость - проекция скорости точки на прямую, соединяющую ее с выбранным началом координат.
- Полярная система координат - двумерная система координат, в которой каждая точка на плоскости определяется двумя числами: полярным углом и полярным радиусом.

Выполнение лабораторной работы

1. Начнем отсчет времени с первого момента исчезновения тумана. Центром введенных полярных координат будем считать точку нахождения браконьеров, и осью, проходящей через катер береговой охраны. Тогда начальные координаты катера $(8,5; 0)$. Обозначим скорость лодки v .
2. Для того чтобы траектория катера пересеклась с траекторией лодки, необходимо, чтобы оба судна всегда находились на одинаковом расстоянии от полюса. Поэтому в начале катер береговой охраны должен двигаться прямолинейно, пока не достигнет того же расстояния от полюса, что и лодка браконьеров. Затем катер должен двигаться вокруг полюса, удаляясь от него с такой же скоростью, как и лодка.

3. Для определения расстояния x , после которого катер начнет двигаться по круговой траектории вокруг полюса, необходимо составить следующие уравнения.

Предположим, что через время t катер и лодка окажутся на одинаковом расстоянии от полюса, равном x . Получается, что за t лодка пройдет x , а катер $8,5 + x$ (или $8,5 - x$, два случая, так как начальное положение катера относительно полюса может быть разным). Чтобы вычислить время, время, за которое они пройдут это расстояние, составим уравнения: как $\frac{x}{v}$ или $\frac{8,5-x}{3,5v}$ (во втором случае $\frac{8,5+x}{3,5v}$). Эти величины равны, так как очевидно, что встретятся они через одно время. Получаем два разных уравнения (два случая, так как начальное положение катера относительно полюса может быть разным).

$$\begin{cases} \frac{x}{v} = \frac{8,5-x}{3,5v} \\ \frac{x}{v} = \frac{8,5+x}{3,5v} \end{cases}$$

Из данных уравнений можно найти расстояние, после которого катер начнёт раскручиваться по спирали. Для данных уравнений решения будут следующими: $x_1 = \frac{17}{9}$, $x_2 = \frac{17}{5}$. 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 задачи сводится к системе из двух дифференциальных уравнений, описывающих движение катера вокруг полюса.:

$$\begin{cases} \frac{dr}{dt} = v \\ r \frac{d\theta}{dt} = \sqrt{1125}v \end{cases}$$

Начальные условия для этих уравнений зависят от выбранной начальной позиции катера относительно полюса:

для одного случая:

$$\begin{cases} \theta_0 = 0 \\ r_0 = x_1 = \frac{17}{9} \end{cases}$$

для другого:

$$\begin{cases} \theta_0 = -\pi \\ r_0 = x_2 = \frac{17}{5} \end{cases}$$

Исключив из системы, которую мы получили, производную по t , можно перейти к следующему уравнению (с неизменными начальными условиями):

$$\frac{dr}{d\theta} = \frac{r}{\sqrt{1125}}$$

Решив систему уравнений, получим траекторию движения катера в полярных координатах.

Решением задачи будем считать точку пересечения траекторий катера и лодки.

Решение задачи в программной среде

В начале по заданной формуле определяю номер своего варианта:

```
print(f'Мой вариант :{(1032216433 % 70) + 1}')
```

main (1) ↗

main (1) ×

C:\Users\arist\PycharmProjects\vychislitelnie_syste

Мой вариант :4

Process finished with exit code 0

Рис. 1: Рисунок 1. Определение варианта.

Для работы мне необходимо установить программную среду julia, делаю это:

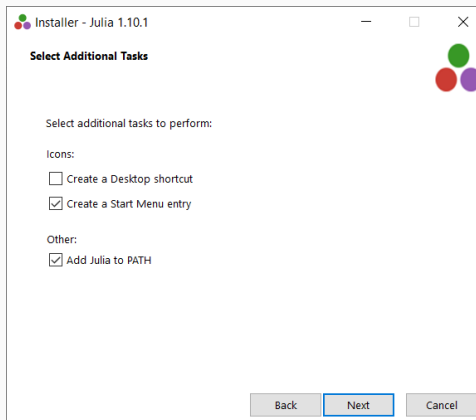


Рис. 2: Рисунок 2. Установка julia.

Решение задачи в программной среде

Также для выполнения лабораторной работы мне необходим установить следующие пакеты: Plots, DifferentialEquations. Устанавливаю их:

```
[80085c20] + Distributed
[9fa8497b] + Future
[4af54fe1] + LazyArtifacts
[9abbd945] + Profile
[1a1011a3] + SharedArrays
[4607b0f0] + SuiteSparse
Info Packages marked with ⚙ and ⚙ have new versions available. Those with ⚙ may be upgradable, but those with ⚙ are restricted by compatibility constraints from upgrading. To see why use `status --outdated -m`
Precompiling project...
Progress [=====] 93/163
⚙ MKL_jll
⚙ ArrayLayouts
⚙ ArnoldiMethod
⚙ MathOptInterface
⚙ Distributions
⚙ ArrayInterface → ArrayInterfaceGPUArraysCoreExt
⚙ Graphs
```

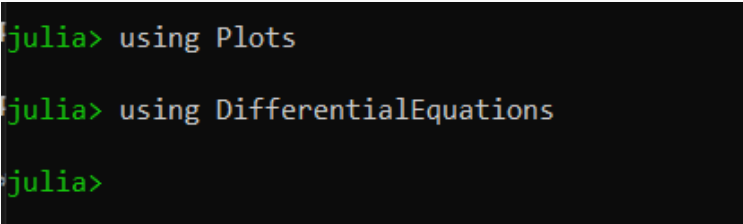
Рис. 3: Рисунок 3. Установка необходимых пакетов.

cmd Командная строка - julia

```
Installed OpenSSL_jll _____ v3.0.13+0
Installed XML2_jll _____ v2.12.2+0
Installed Libiconv_jll _____ v1.17.0+0
Installed Qt6Base_jll _____ v6.5.3+1
Installed PrecompileTools _____ v1.2.0
Installed Glib_jll _____ v2.76.5+0
Installed LaTeXStrings _____ v1.3.1
Installed URIs _____ v1.5.1
Installed libvorbis_jll _____ v1.3.7+1
Installed Libglvnd_jll _____ v1.6.0+0
Installed Xorg_libX11_jll _____ v1.8.6+0
Installed Xorg_libXdmcp_jll _____ v1.1.4+0
Installed Requires _____ v1.3.0
Installed Unzip _____ v0.2.0
Installed UnitfulLatexify _____ v1.6.3
Installed SortingAlgorithms _____ v1.2.1
Downloaded artifact: JpegTurbo
Downloaded artifact: x265
Downloaded artifact: libfdk_aac
Downloaded artifact: GR
Downloaded artifact: LERC
Downloaded artifact: Opus
Downloading artifact: Cairo
```

Рис. 4: Рисунок 4. Установка необходимых пакетов.

Затем я проверяю корректность установки пакетов:

A screenshot of a Julia REPL session. The background is black, and the text is in a monospaced font. The prompt 'julia>' is shown in green. The first line shows 'using Plots' in white. The second line shows 'using DifferentialEquations' in white. The third line shows the prompt 'julia>' in green, indicating the session is still active.

```
julia> using Plots  
julia> using DifferentialEquations  
julia>
```

Рис. 5: Рисунок 5. Проверка корректности установки пакетов.

Затем я пишу программу на языке julia для получения графиков траекторий катера и лодки.

Вот её листинг:

```
using Plots
```

```
using DifferentialEquations
```

```
# Объявляем значения
```

```
const k = 8.5
```

```
const n = 3.5
```

```
# Начальные расстояния для двух разных случаев погони
```

```
r0 = k/(n+1)
```

```
r0_2 = k/(n-1)
```

```
# Задаем интервалы  
const T = (0, 2*pi)  
const T2 = (-pi, pi)  
  
# Задаем функцию, представляющую наше ДУ  
function F(u, p, t)  
    return u / sqrt(n*n - 1)  
end
```

```
# Задаем проблему(задачу) для случая 1
```

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

```
# Решение для случая 1
```

```
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]]
```


Решение задачи в программной среде

```
# График траекторий для случая 1
```

```
plt1 = plot(proj=:polar, aspect_ratio=:equal, dpi = 1000, legend=true, bg=:li
```

```
# Настрою холст
```

```
plot!(plt1, xlabel="theta", ylabel="r(t)", title="Задача о погоне. Случай 1."
```

```
plot!(plt1, [rAngles[1], rAngles[2]], [0.0, result.u[size(result.u)[1]]], lab
```

```
scatter!(plt1, rAngles, result.u, label="", mc=:red, ms=0.0005)
```

```
plot!(plt1, result.t, result.u, xlabel="theta", ylabel="r(t)", label="Траекто
```

```
scatter!(plt1, result.t, result.u, label="", mc=:green, ms=0.0005)
```

```
savefig(plt1, "lab02_img1.png")
```

```
# Задаем проблему(задачу) для случая 2
```

```
problem = ODEProblem(F, r0_2 , T2)
```

```
# Решение для случая 2
```

```
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]]
```

```
# График траекторий для случая 2
```

```
plt2 = plot(proj=:polar, aspect_ratio=:equal, dpi = 1000, legend=true, bg=:li
```

```
# Настрою холст
```

```
plot!(plt2, xlabel="theta", ylabel="r(t)", title="Задача о погоне. Случай 2",  
plot!(plt2, [rAngles[1], rAngles[2]], [0.0, result.u[size(result.u)[1]]], lab  
scatter!(plt2, rAngles, result.u, label="", mc=:red, ms=0.0005)  
plot!(plt2, result.t, result.u, xlabel="theta", ylabel="r(t)", label="Траекто  
scatter!(plt2, result.t, result.u, label="", mc=:green, ms=0.0005)  
  
savefig(plt2, "lab02_img2.png")
```

Выполняю эту программу:

```
PS C:\Users\arist\OneDrive\Документы\work\study\2023-2024\Математическое моделирование\mathmod\labs\lab02> julia lab02.jl
result.u = [1.8888888888888888, 1.9155914849669538, 2.0598922024420765, 2.3226267326090997, 2.6698729402482004, 3.1249132984975163, 3.7151215007562866, 4.4773577769769
336864495, 12.295983877098815]
result.t = [0.0, 0.047083802229947715, 0.2906828638688455, 0.6933263866552275, 1.1606605438198243, 1.6885145375769615, 2.268790427572364, 2.8947386633892966, 3.5590028
6.283185307179586]
PS C:\Users\arist\OneDrive\Документы\work\study\2023-2024\Математическое моделирование\mathmod\labs\lab02>
```

Рис. 6: Рисунок 6. Выполнение написанной программы.

В итоге получаю следующие графики:



Рис. 7: Рисунок 7. График движения. Случай 1.

В итоге получаю следующие графики:

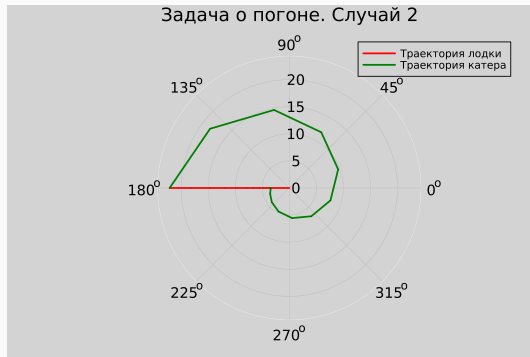


Рис. 8: Рисунок 8. График движения. Случай 2.

В результате выполнения данной лабораторной работы мною были получены графики для обоих случаев. На них изображены траектории катера и лодки, что позволило наглядно определить точки их пересечения. Задача о погоне была успешно решена.

В процессе и результате выполнения лабораторной работы я ознакомилась с основами программирования на языках Julia и OpenModelica. Также я освоила библиотеки этих языков, которые используются для создания графиков и решения дифференциальных уравнений. В данной лабораторной работе я использовала язык Julia для работы с полярными координатами.

1. Документация по Julia: <https://docs.julialang.org/en/v1/>
2. Документация по OpenModelica: <https://openmodelica.org/>
3. Документация по работе с пакетом Plots: <https://docs.juliaplots.org/latest/tutorial/>
4. Решение дифференциальных уравнений: <https://www.wolframalpha.com/>
5. Решение дифференциальных уравнений:
http://www.mathprofi.ru/differencialnye_uravnenija_primery_reshenii.html