Презентация по лабораторной работе 2

Елизавета Александровна Гайдамака

Цель работы

Целью данной работы является введение в работу с Julia и Modelica.

Задание

1. Провести аналогичные рассуждения и вывод дифференциальных уравнений, если скорость катера больше скорости лодки в n раз (значение n задайте самостоятельно)
2. Построить траекторию движения катера и лодки для двух случаев. (Задайте самостоятельно начальные значения) Определить по графику точку пересечения катера и лодки

Теоретическое введение

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.

Modelica — объектно-ориентированный, декларативный, мультидоменный язык моделирования для компонентно-ориентированного моделирования сложных систем, в частности, систем, содержащих механические, электрические, электронные, гидравлические, тепловые, энергетические компоненты, а также компоненты управления и компоненты, ориентированные на отдельные процессы. Modelica разработана некоммерческой организацией Modelica Association. Эта компания также разрабатывает свободно распространяемую стандартную библиотеку Modelica Standard Library, в версии 3.2.1 содержащую порядка 1360 типичных элементов моделей и 1280 функций из различных областей.

1. Провести аналогичные рассуждения и вывод дифференциальных уравнений, если скорость катера больше скорости лодки в n раз (значение n задайте самостоятельно) 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 k-x (или k+x , в зависимости от начального положения катера относительно полюса). Время, за которое они пройдут это расстояние, вычисляется как x/v или k-x/3.5v (во втором случае x+k/3.5v). Так как время одно и то же, то эти величины одинаковы. Тогда неизвестное расстояние x можно найти из следующего уравнения:

$$\frac{x}{v} = \frac{k - x}{3.5v}$$

Отсюда мы найдем два значения x1=k/4.5, x2=k/2.5 , 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: vr радиальная скорость и v_τ - тангенциальная скорость.

Радиальная скорость - это скорость, с которой катер удаляется от полюса, vr=dr/dt . Нам нужно, чтобы эта скорость была равна скорости лодки, поэтому полагаем dr/dt=v . Тангенциальная скорость – это линейная скорость вращения катера относительно полюса. Она равна произведению угловой скорости $\frac{d\theta}{dt}$ на радиус r, $v_\tau = r\frac{d\theta}{dt}$

Получается: $v_\tau = \sqrt{12.25v^2 - v^2} = \sqrt{11.25}v$. Тогда $r\frac{d\theta}{dt} = \sqrt{11.25}v$

Решение исходной задачи сводится к решению системы из двух дифференциальных уравнений

$$\frac{dr}{dt} = v, r\frac{d\theta}{dt} = \sqrt{11.25}v$$

с начальными условиями

$$\theta_0 = 0, r_0 = x_1$$

или

$$\theta_0 = -\pi, r_0 = x_2$$

Начальные условия остаются прежними. Решив это уравнение, вы получите траекторию движения катера в полярных координатах.

2. Построить траекторию движения катера и лодки для двух случаев. (Задайте самостоятельно начальные значения)

Мой номер варианта - 5.

Скачиваем Julia и пишем программу

Код моей программы:

```
using Plots using DifferentialEquations

const a = 8.5 const n = 3.5

const r0 = a/(n + 1) const r0_2 = a/(n - 1)

const T = (0, 2*pi) const T_2 = (-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="Случай 1") plot!(plt, [rAngles[1], rAngles[2]], [0.0, result.u[size(result.u)[1]]], label="Лодка", color=:blue, lw=1) scatter!(plt, rAngles, result.u, label="", mc=:blue, ms=0.0005) plot!(plt, result.t, result.u, xlabel="theta", ylabel="r(t)", label="Катер", color=:green, lw=1) scatter!(plt, result.t, result.u, label="", mc=:green, ms=0.0005)

savefig(plt, "lab2_1.png")

#—–

problem = ODEProblem(F, r0_2, T_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]]

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

plot!(plt1, xlabel="theta", ylabel="r(t)", title="Случай 2") plot!(plt1, [rAngles[1], rAngles[2]], [0.0, result.u[size(result.u)[1]]], label="Лодка", color=:blue, lw=1) scatter!(plt1, rAngles, result.u, label="", mc=:blue, ms=0.0005) plot!(plt1, result.t, result.u, xlabel="theta", ylabel="r(t)", label="Катер", color=:green, lw=1) scatter!(plt1, result.t, result.u, label="", mc=:green, ms=0.0005)

savefig(plt1, "lab2_2.png")

В результате работы программы получаем следующие графики для двух случаев.

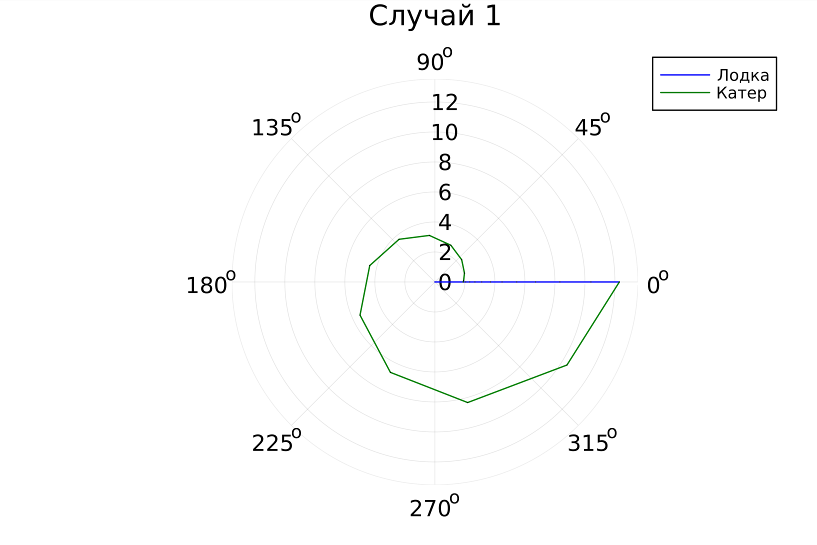

Рис. 1: Рис.1

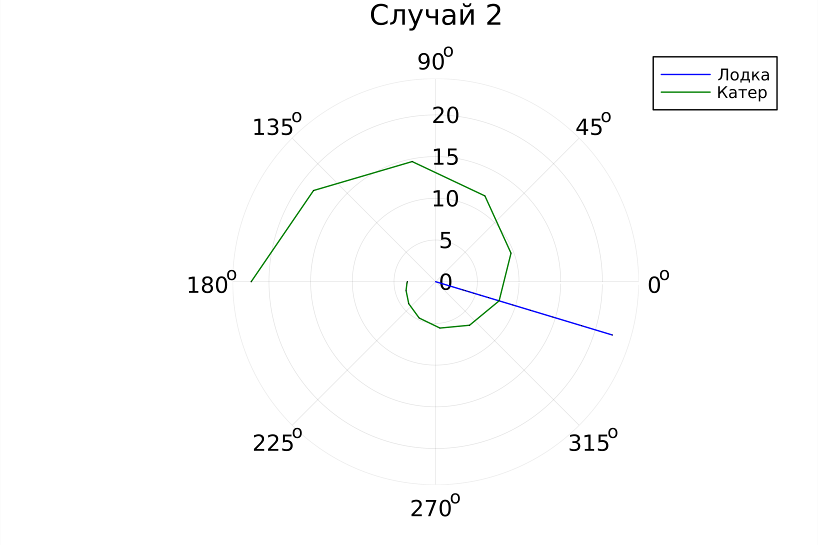

Рис. 2: Рис.2

Язык Modelica не предназначен для работы с полярными координатами

Выводы

Благодаря данной работе я ознакомилась с основами Julia и Modelica.