

Лабораторная работа №2

Задача о погоне

Аникин Константин Сергеевич

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
5	Выводы	16
	Список литературы	17

Список иллюстраций

4.1	Программа	9
4.2	График $s +$ на 0 радиан	10
4.3	График $s -$ на 0 радиан	11
4.4	График $s +$ на 1 радиан	12
4.5	График $s -$ на 1 радиан	13
4.6	График $s +$ на 6 радиан	14
4.7	График $s -$ на 6 радиан	15

Список таблиц

1 Цель работы

Решить задачу о погоне охраны за браконьерами на Julia.

2 Задание

На море в тумане катер береговой охраны преследует лодку браконьеров. Через определенный промежуток времени туман рассеивается, и лодка обнаруживается на расстоянии 6,3 км от катера. Затем лодка снова скрывается в тумане и уходит прямолинейно в неизвестном направлении. Известно, что скорость катера в 2,3 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

Julia — высокоуровневый высокопроизводительный свободный язык программирования с динамической типизацией, созданный для математических вычислений. Эффективен также и для написания программ общего назначения. Имеет встроенную поддержку многопоточности и распределённых вычислений, реализованные в том числе в стандартных конструкциях.

Язык является динамическим, при этом поддерживает JIT-компиляцию (JIT-компилятор на основе LLVM входит в стандартный комплект), благодаря чему, по утверждению авторов языка, приложения, полностью написанные на языке (без использование низкоуровневых библиотек и векторных операций) практически не уступают в производительности приложениям, написанным на статически компилируемых языках, таких как Си или С++. Большая часть стандартной библиотеки языка написана на нём же.

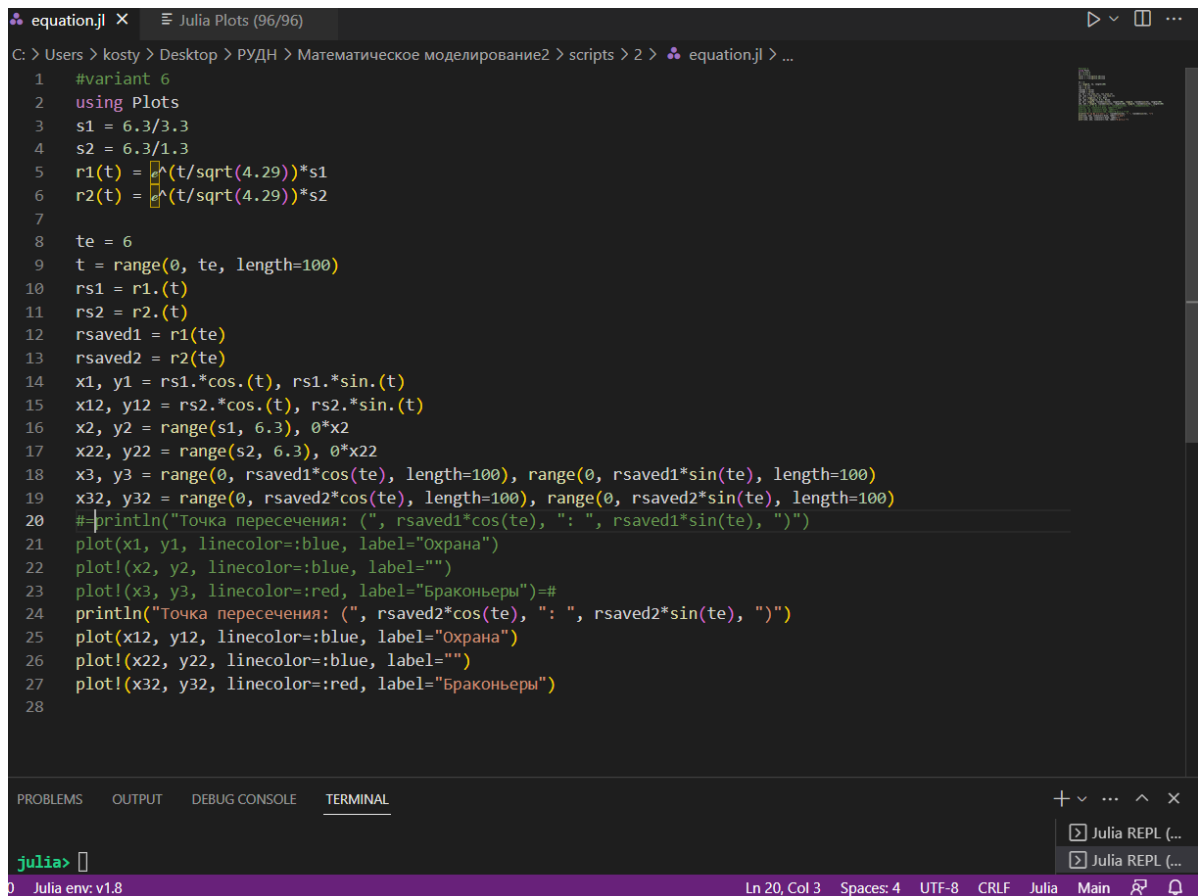
Поддерживается перегрузка функций и операторов (которые фактически также являются функциями), при этом опционально можно указывать тип для аргументов функции, чего обычно нет в динамически типизируемых языках. Это позволяет создавать специализированные варианты функций и операторов для ускорения вычислений. Наиболее подходящий вариант функции выбирается автоматически в процессе выполнения. Также благодаря перегрузке операторов можно создавать новые типы данных, которые ведут себя подобно встроенным типам.

Более подробно о работе с Julia см. в [1,2].

4 Выполнение лабораторной работы

В ходе выполнения работы я написал программу, высчитывающую траекторию кораблей и строящую графики их движения до точки пересечения (рис. 4.1).

Программа высчитывает траекторию движения корабля, находит длину отрезка от начала координат до точки пересечения, строит прямую движения браконьеров, а после рисует это на графике и выводит точку пересечения в терминал. Вычисления проводятся для обоих вариантов, перекомментировав нужное можно увидеть вывод второго варианта.



```
1 #variant 6
2 using Plots
3 s1 = 6.3/3.3
4 s2 = 6.3/1.3
5 r1(t) = e^(t/sqrt(4.29))*s1
6 r2(t) = e^(t/sqrt(4.29))*s2
7
8 te = 6
9 t = range(0, te, length=100)
10 rs1 = r1.(t)
11 rs2 = r2.(t)
12 rsaved1 = r1(te)
13 rsaved2 = r2(te)
14 x1, y1 = rs1.*cos.(t), rs1.*sin.(t)
15 x12, y12 = rs2.*cos.(t), rs2.*sin.(t)
16 x2, y2 = range(s1, 6.3), 0*x2
17 x22, y22 = range(s2, 6.3), 0*x22
18 x3, y3 = range(0, rsaved1*cos(te), length=100), range(0, rsaved1*sin(te), length=100)
19 x32, y32 = range(0, rsaved2*cos(te), length=100), range(0, rsaved2*sin(te), length=100)
20 #println("Точка пересечения: (", rsaved1*cos(te), ", ", rsaved1*sin(te), ")")
21 plot(x1, y1, linecolor=:blue, label="Охрана")
22 plot!(x2, y2, linecolor=:blue, label="")
23 plot!(x3, y3, linecolor=:red, label="Браконьеры")=#
24 println("Точка пересечения: (", rsaved2*cos(te), ", ", rsaved2*sin(te), ")")
25 plot(x12, y12, linecolor=:blue, label="Охрана")
26 plot!(x22, y22, linecolor=:blue, label="")
27 plot!(x32, y32, linecolor=:red, label="Браконьеры")
28
```

Рис. 4.1: Программа

Далее представлены графики, посчитанные для разных входных данных. На рис. 4.2 и 4.3 представлены графики при угле в 0 радиан, на рис. 4.4 и 4.5 представлены графики при угле в 1 радиан, и на рис. 4.6 и 4.7 представлены графики при угле в 6 радиан. Первый и второй графики каждой пары рассчитаны на первый и второй вариант задачи соответственно.

Точка пересечения считается в декартовых координатах

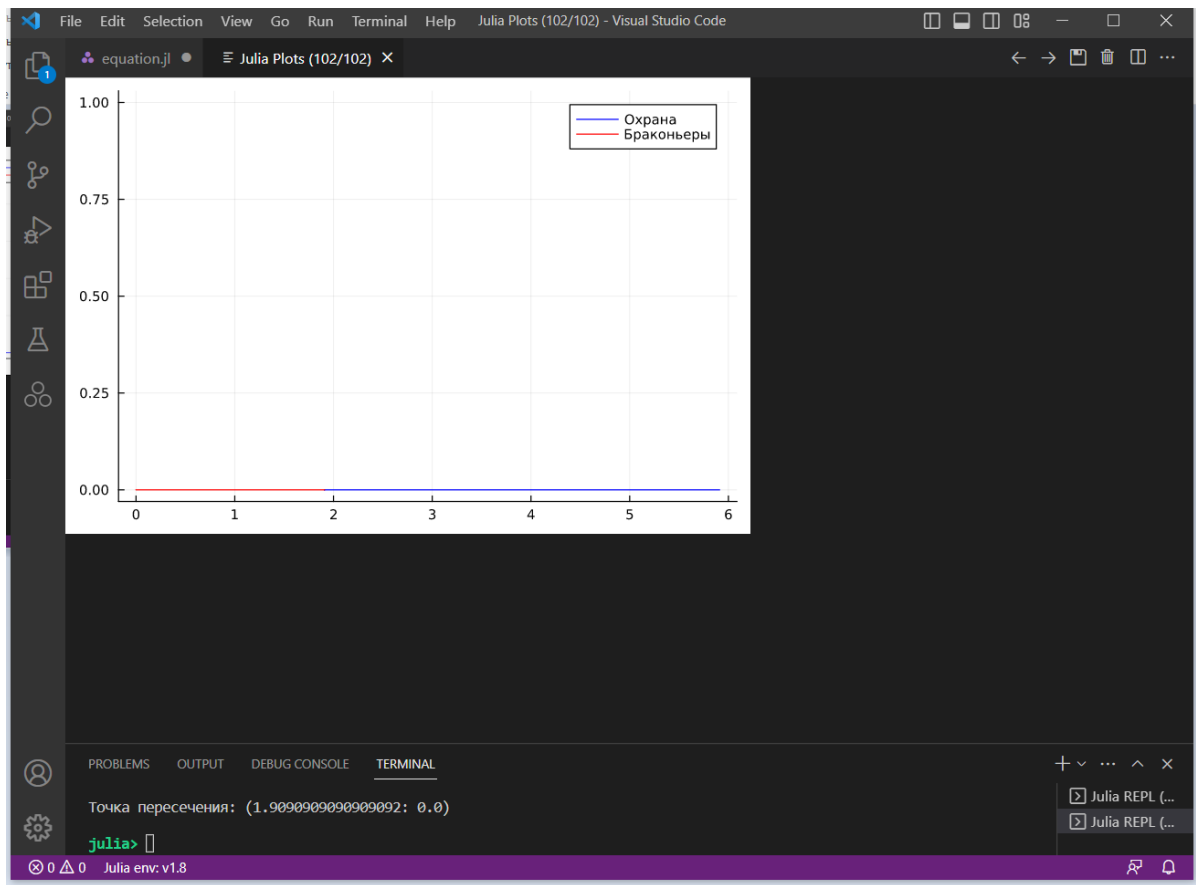


Рис. 4.2: График с + на 0 радиан

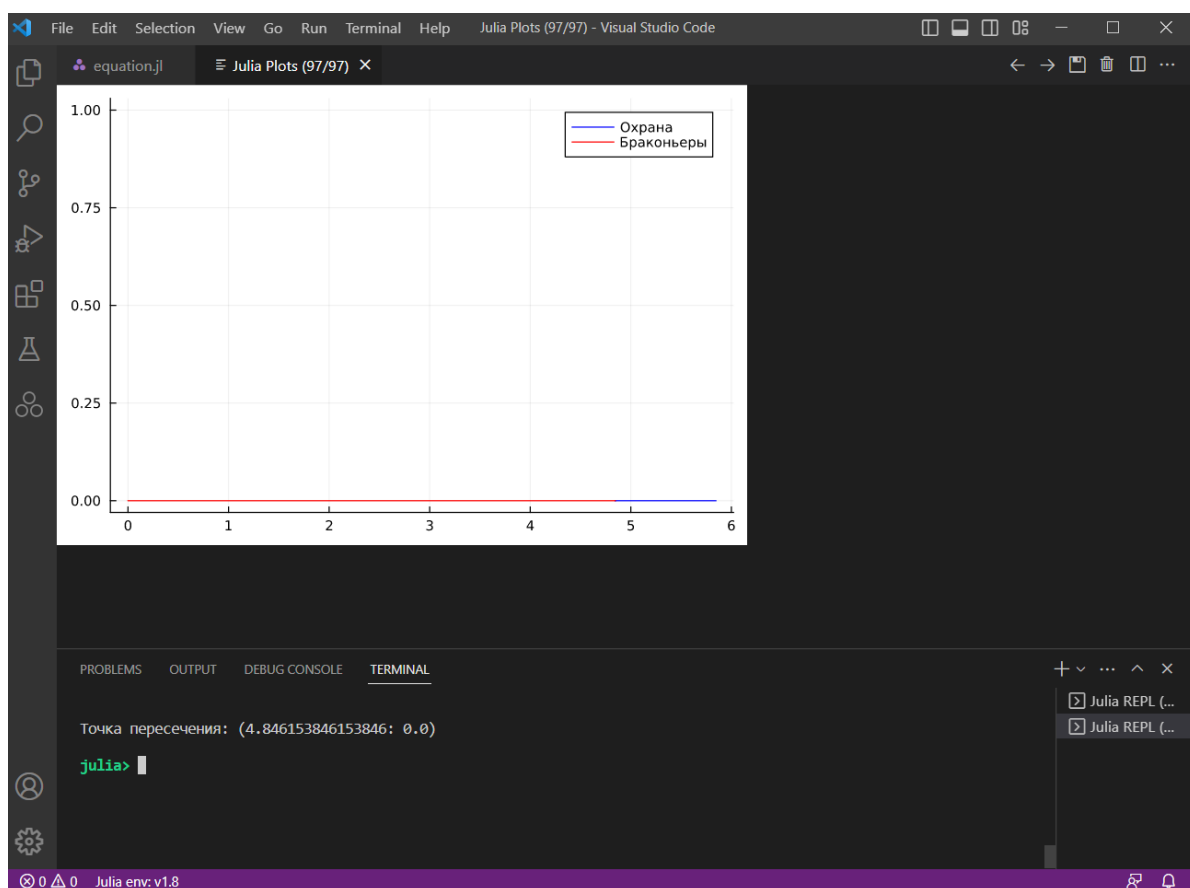


Рис. 4.3: График с - на 0 радиан

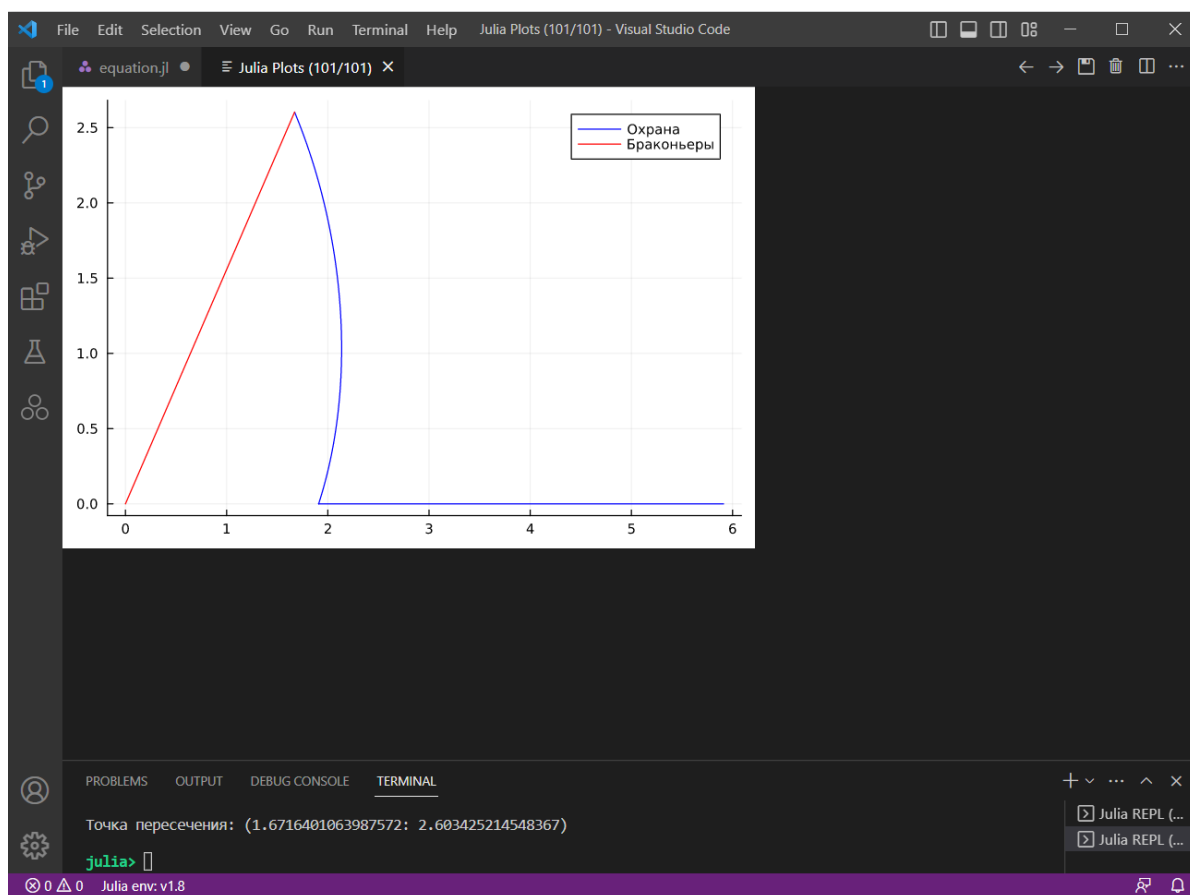


Рис. 4.4: График с + на 1 радиан

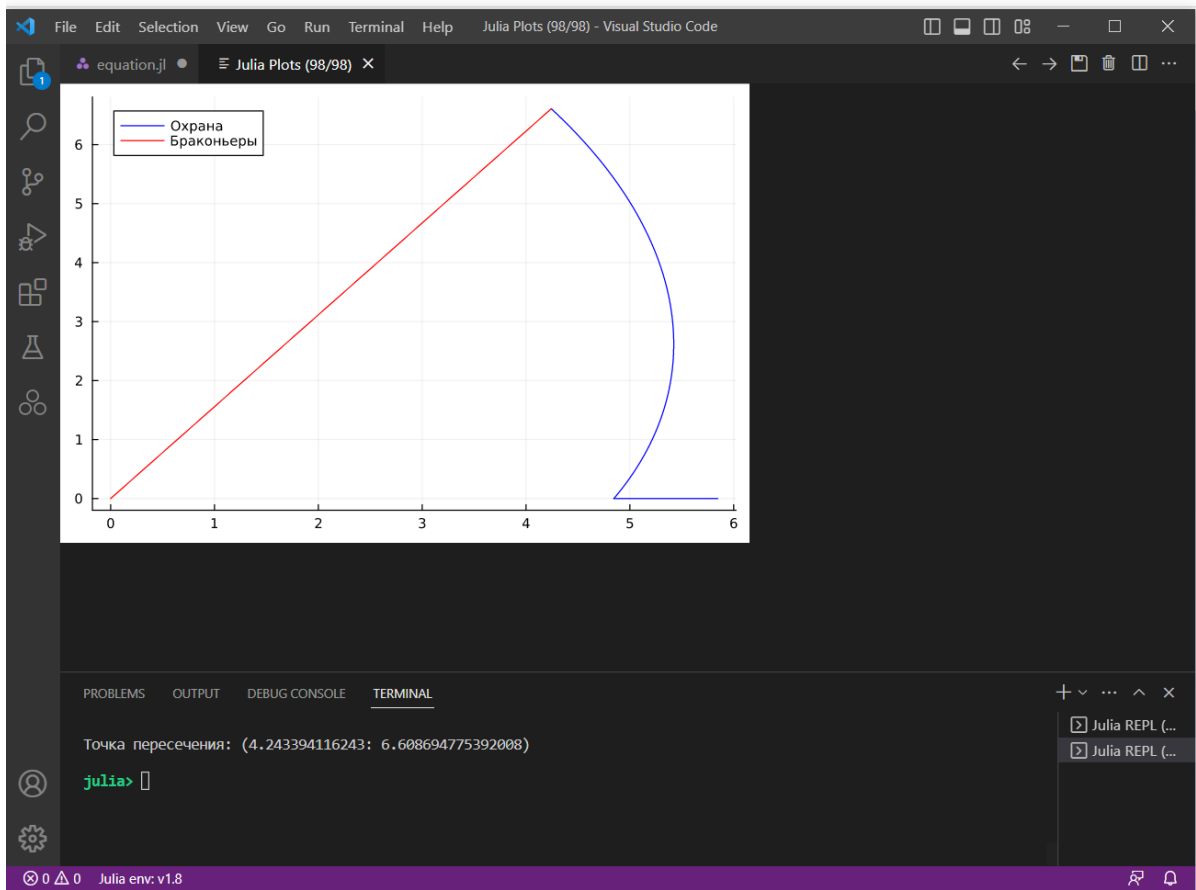


Рис. 4.5: График с - на 1 радиан

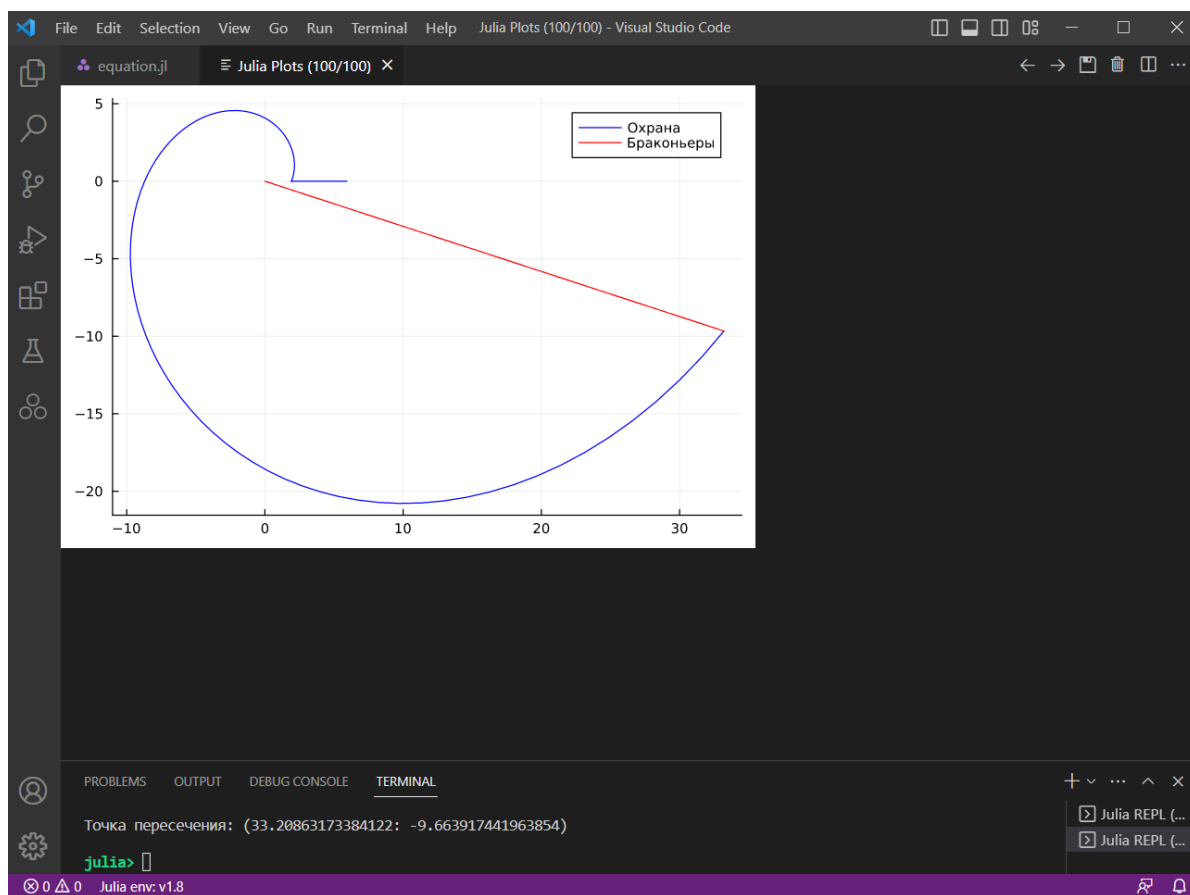


Рис. 4.6: График с + на 6 радиан

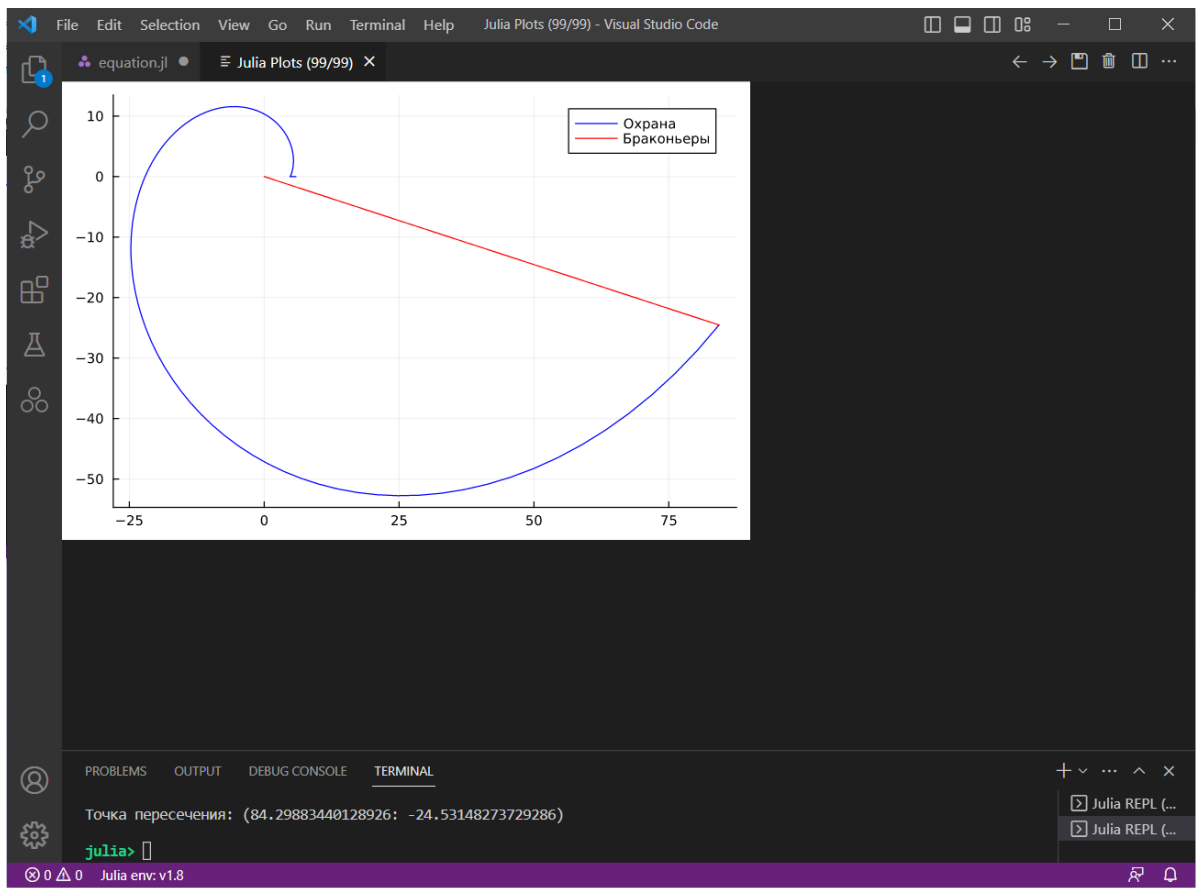


Рис. 4.7: График с - на 6 радиан

5 Выводы

В ходе работы я познакомился с Julia, решил задачу о погоне и решил её с помощью Джулии. Мог ошибиться в математике. Код выглядит отвратительно, хочется взять и... и отрефакторить. Хотя бы работает, и на том спасибо.

Список литературы

1. The Julia Programming Language Tutorial [Электронный ресурс]. Netlify, Franklin.jl; the Julia Programming Language, 2021. URL: <https://julialang.org/learning/tutorials/>.
2. CATAM material in Julia [Электронный ресурс]. Cambridge University Press, 2022. URL: <https://sje30.github.io/catam-julia/>.