

Лабораторная работа № 5

Построение графиков

Беличева Дарья Михайловна

Содержание

1 Цель работы	4
2 Задание	5
3 Теоретическое введение	6
4 Выполнение лабораторной работы	7
4.1 Задания для самостоятельного выполнения	25
5 Выводы	34
Список литературы	35

Список иллюстраций

4.1 Основные пакеты для работы с графиками в Julia	7
4.2 Основные пакеты для работы с графиками в Julia	8
4.3 Опции при построении графика	9
4.4 Опции при построении графика	10
4.5 Точечный график	11
4.6 Точечный график	12
4.7 Аппроксимация данных	13
4.8 Полярные координаты	14
4.9 График поверхности	15
4.10 Линии уровня	16
4.11 Векторные поля	17
4.12 Анимация	18
4.13 Гипоциклоида	19
4.14 Errorbars	20
4.15 Errorbars	21
4.16 Использование пакета Distributions	22
4.17 Подграфики	23
4.18 Подграфики	24
4.19 Подграфики	24
4.20 Задание №1 и №2	25
4.21 Задание №3	26
4.22 Задание №4	26
4.23 Задание №5	27
4.24 Задание №6	27
4.25 Задание №7 и №8	28
4.26 Задание №9	29
4.27 Задание №10	30
4.28 Задание №10	31
4.29 Задание №11	32
4.30 Задание №11	33

1 Цель работы

Основная цель работы – освоить синтаксис языка Julia для построения графиков.

2 Задание

1. Используя JupyterLab, повторите примеры. При этом дополните графики обозначениями осей координат, легендой с названиями траекторий, названиями графиков и т.п.
2. Выполните задания для самостоятельной работы.

3 Теоретическое введение

Julia – высокоуровневый свободный язык программирования с динамической типизацией, созданный для математических вычислений [1]. Эффективен также и для написания программ общего назначения. Синтаксис языка схож с синтаксисом других математических языков, однако имеет некоторые существенные отличия.

Для выполнения заданий была использована официальная документация Julia [2].

4 Выполнение лабораторной работы

Выполним примеры из лабораторной работы для знакомства с пакетами по отрисовки графиков и их функциями (рис. 4.1-4.19).

```
[2]: using Plots
[ Info: Precompiling IJuliaExt [2f4121a4-3b3a-5ce6-9c5e-1f2673ce168a]
[3]: # задание функции:
f(x) = (3x.^2 + 6x .- 9).*exp.(-0.3x)
[3]: f (generic function with 1 method)

[180]: # генерирование массива значений x в диапазоне от -5 до 10 с шагом 0.1
# ( шаг задан через указание длины массива):
x = collect(range(-5,10,length=151));

[21]: # указывается, что для построения графика используется gr():
gr()

[21]: Plots.GRBackend()

[7]: # задание опций при построении графика
# (название кривой, подписи по осям, цвет графика):
plot(x,y,
      title="A simple curve",
      xlabel="Variable x",
      ylabel="Variable y",
      color="blue")
[7]: A simple curve

[8]: # задание опций при построении графика
# (название кривой, подписи по осям, цвет графика):
```

Рис. 4.1: Основные пакеты для работы с графиками в Julia

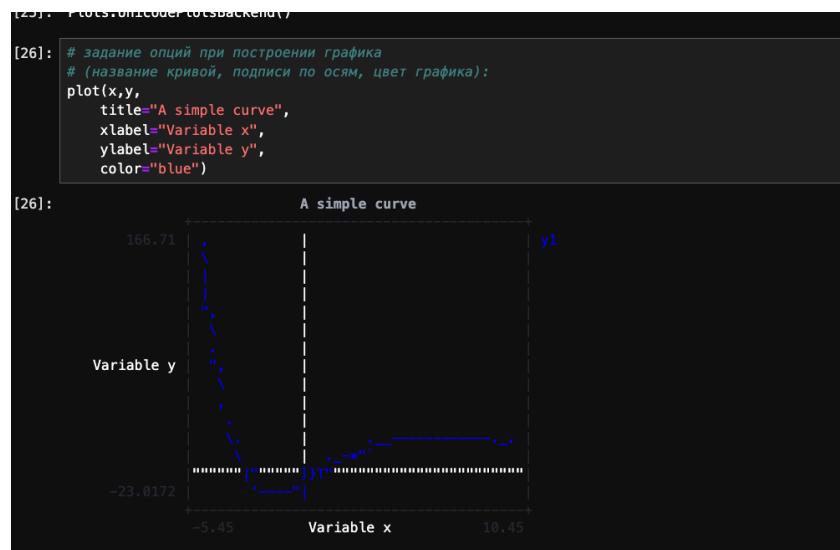


Рис. 4.2: Основные пакеты для работы с графиками в Julia



Рис. 4.3: Опции при построении графика

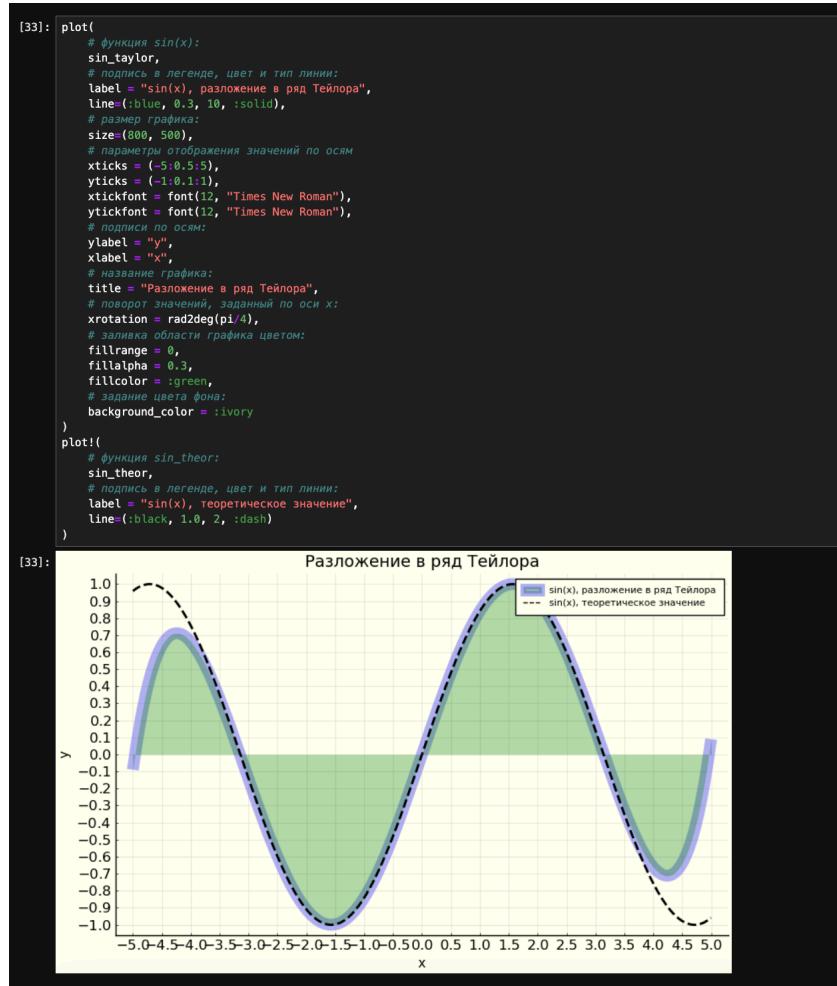


Рис. 4.4: Опции при построении графика

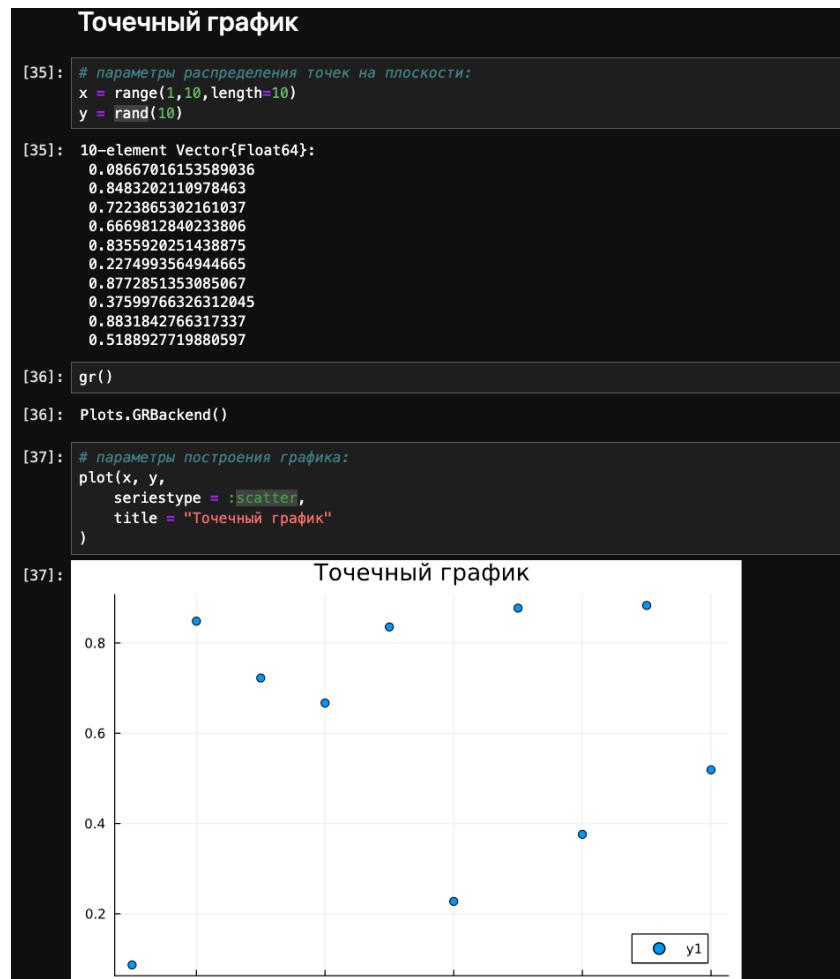


Рис. 4.5: Точечный график

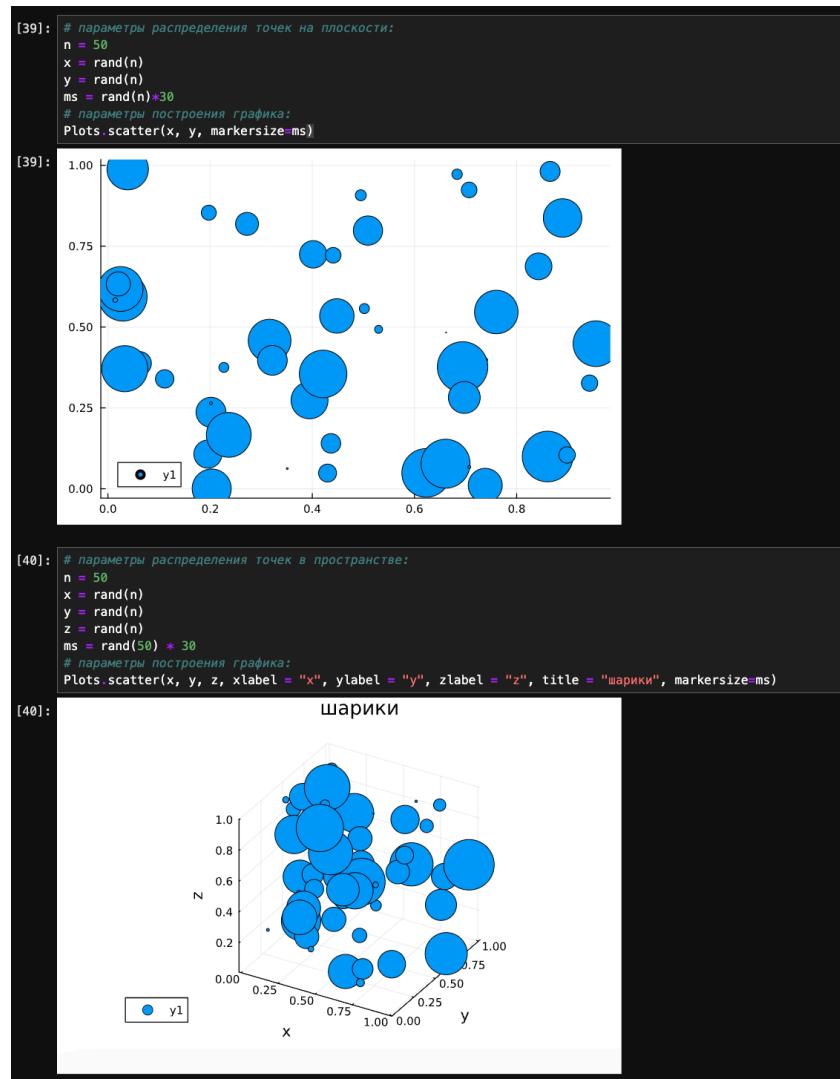


Рис. 4.6: Точечный график

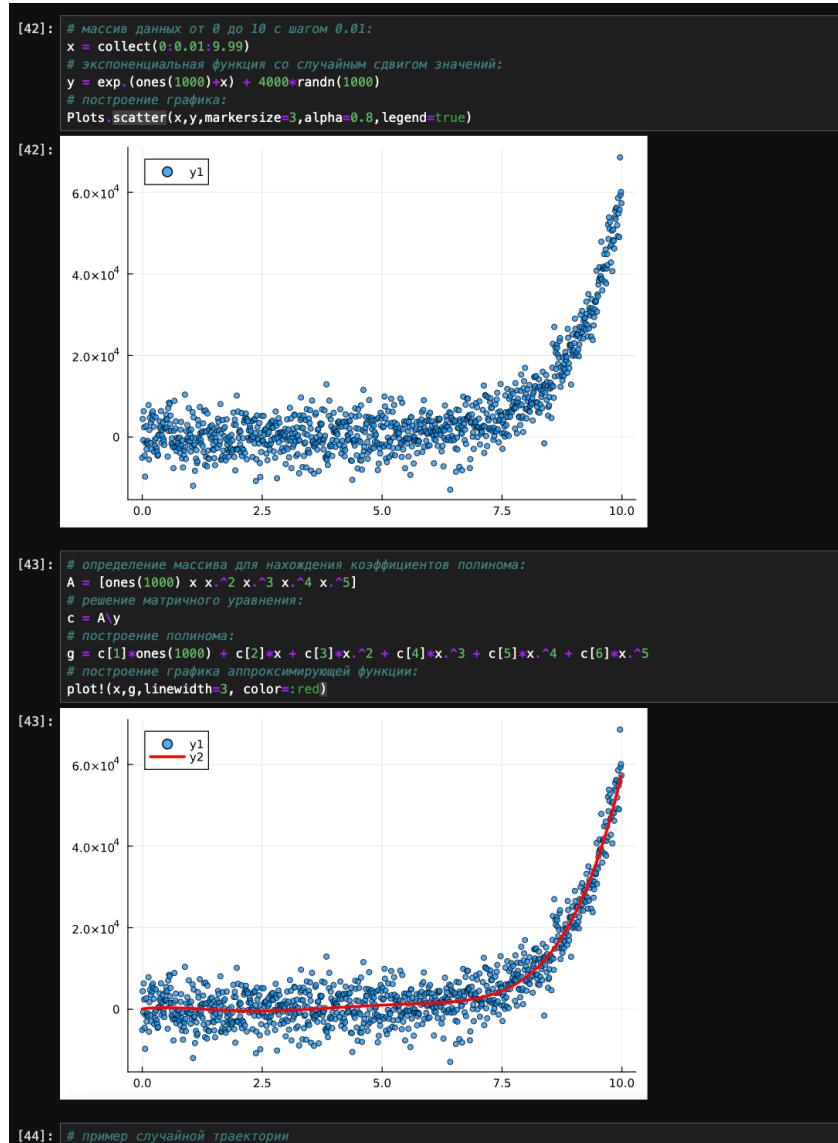


Рис. 4.7: Аппроксимация данных

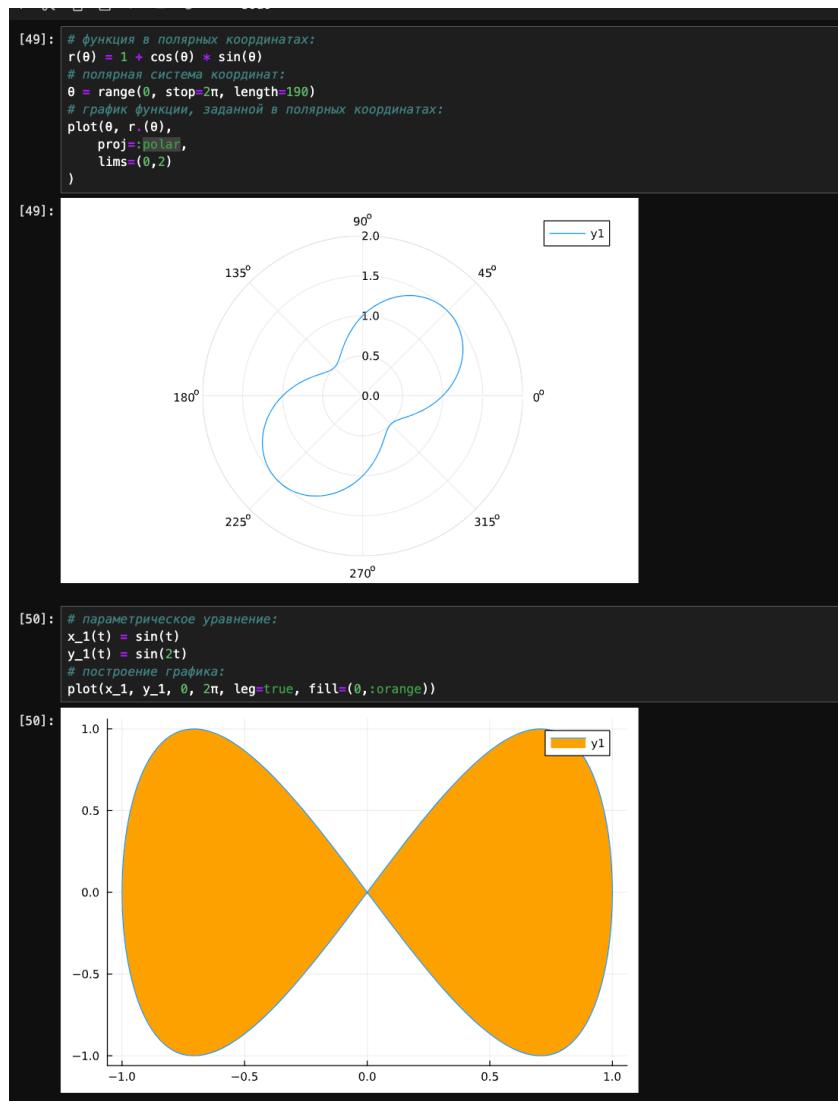


Рис. 4.8: Полярные координаты

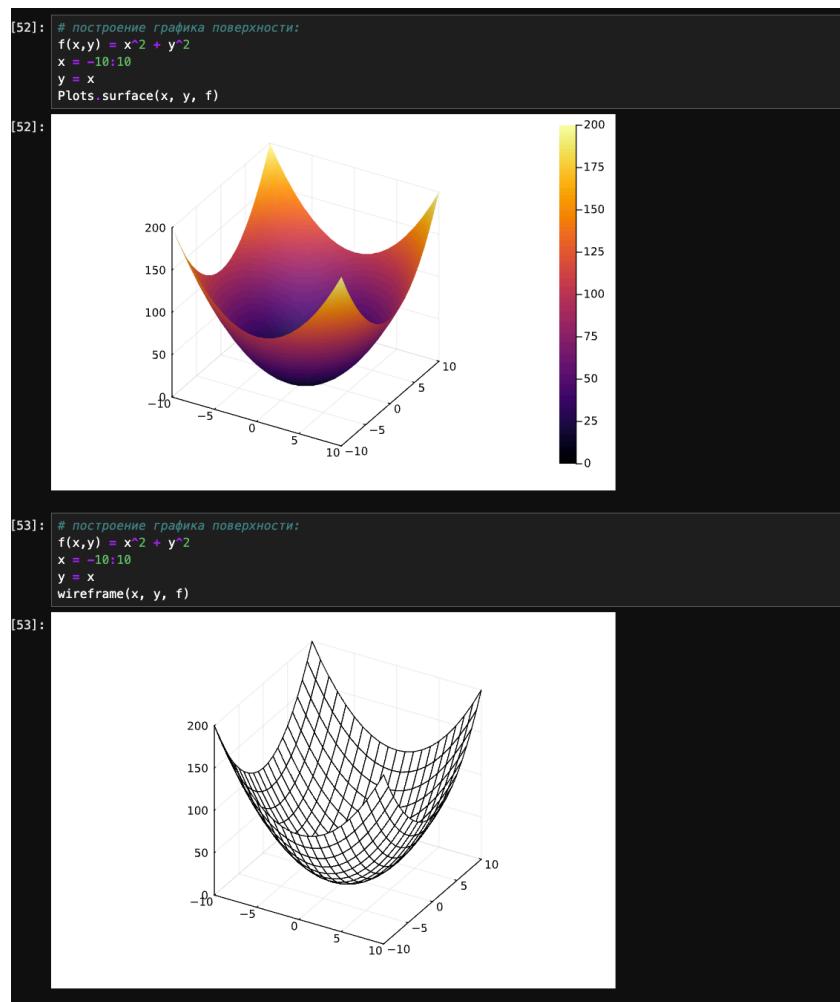


Рис. 4.9: График поверхности

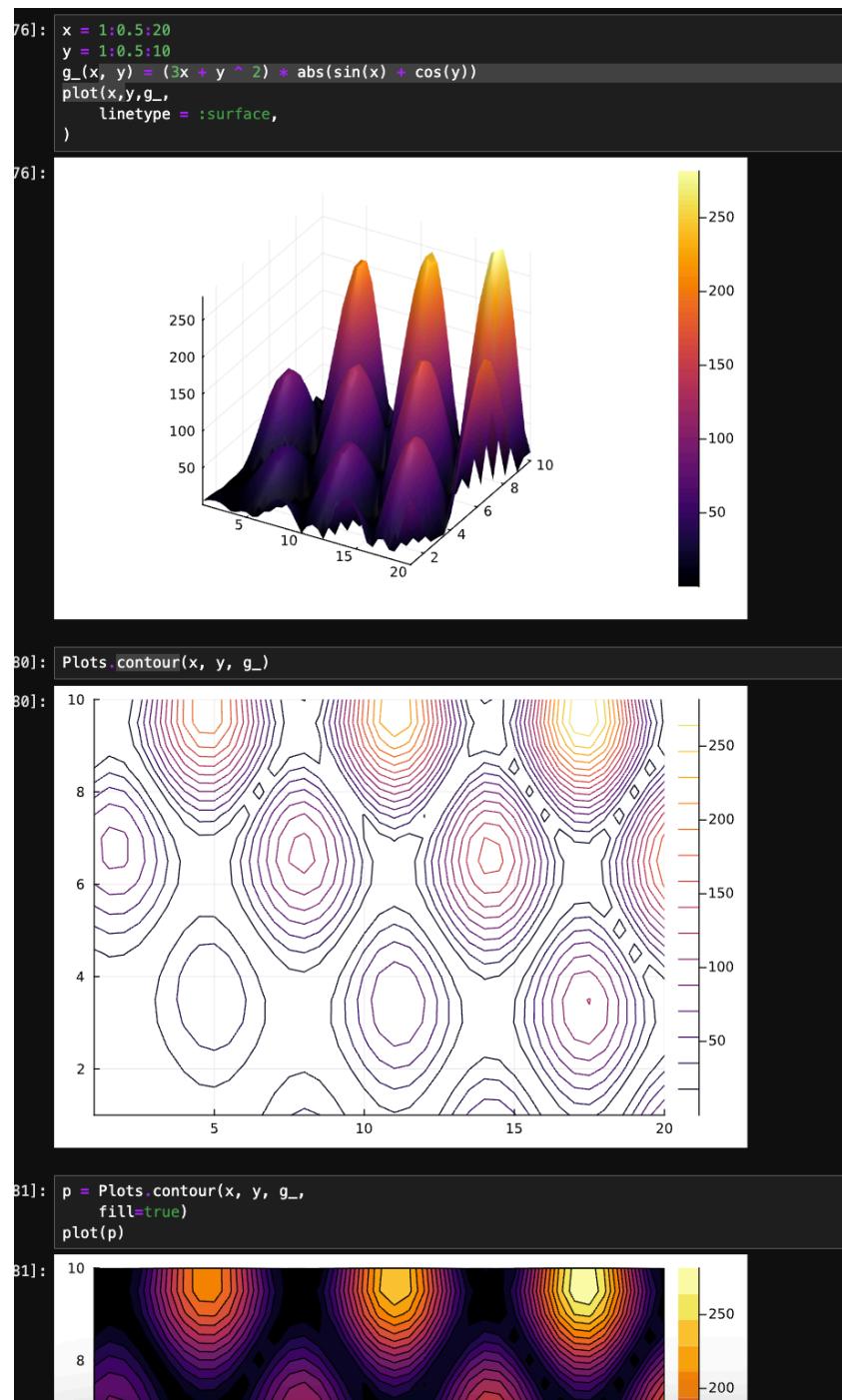


Рис. 4.10: Линии уровня

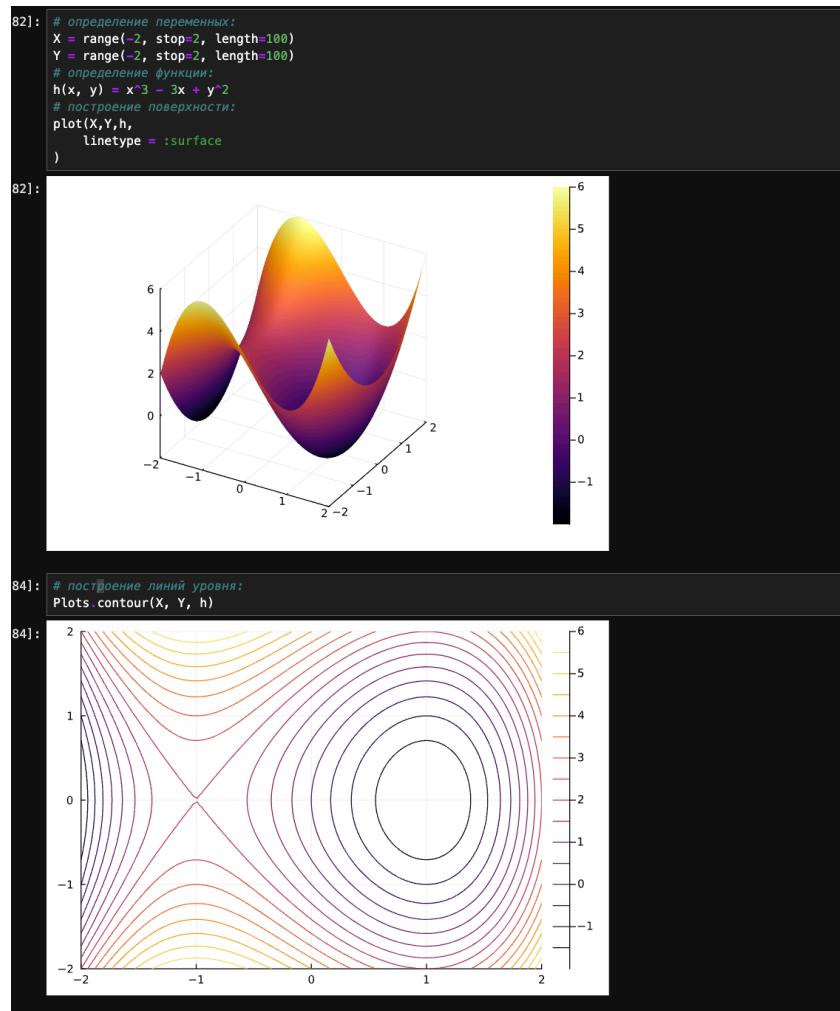
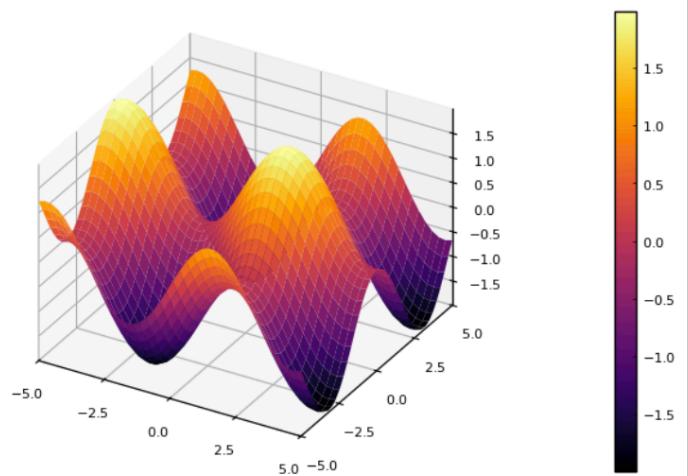


Рис. 4.11: Векторные поля

Анимация

```
[0]: pyplot()
# построение поверхности:
i = 0
X = Y = range(-5,stop=5,length=40)
Plots.surface(X, Y, (x,y) -> sin(x+10sin(i))*cos(y))
# анимация:
X = Y = range(-5,stop=5,length=40)
@for i in range(0,stop=π,length=100)
    Plots.surface(X, Y, (x,y) -> sin(x+10sin(i))*cos(y))
end
```

[Info: Saved animation to /Users/darabeliceva/work/study/julia/tmp.gif



```
[0]:
```

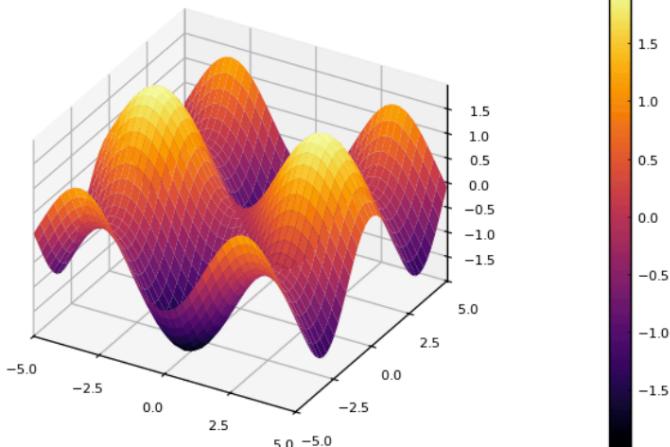


Рис. 4.12: Анимация

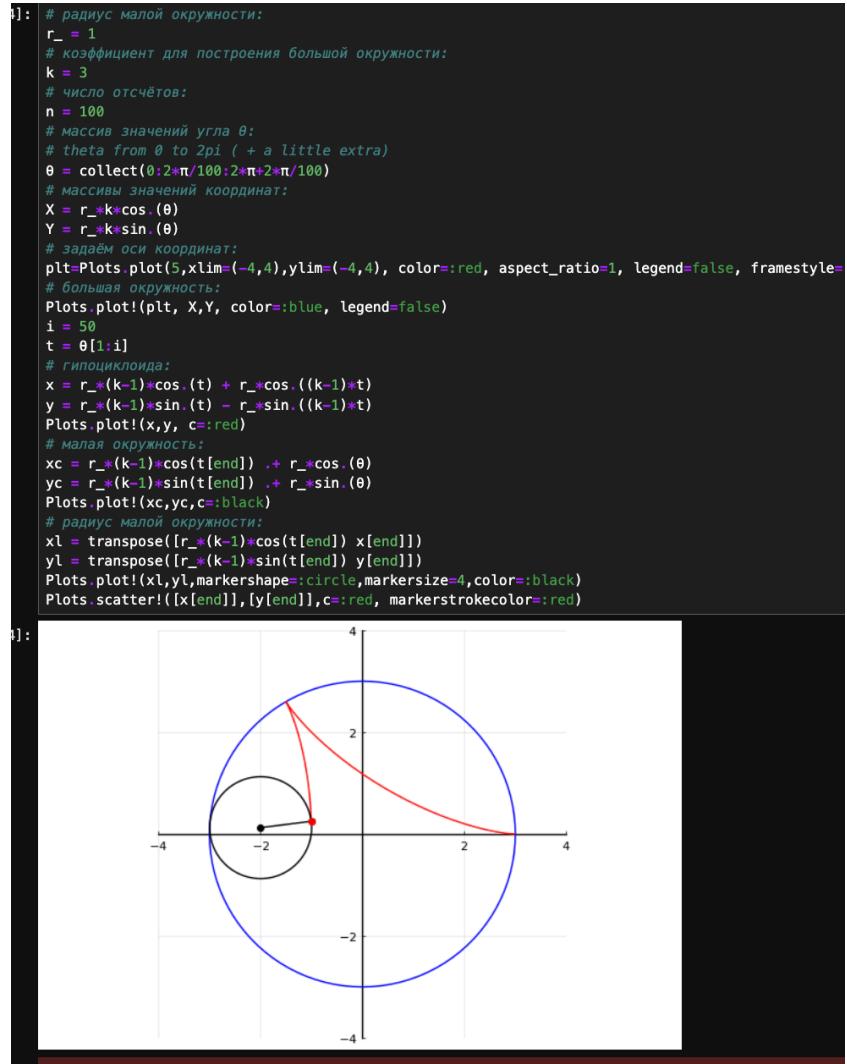


Рис. 4.13: Гипоциклоида

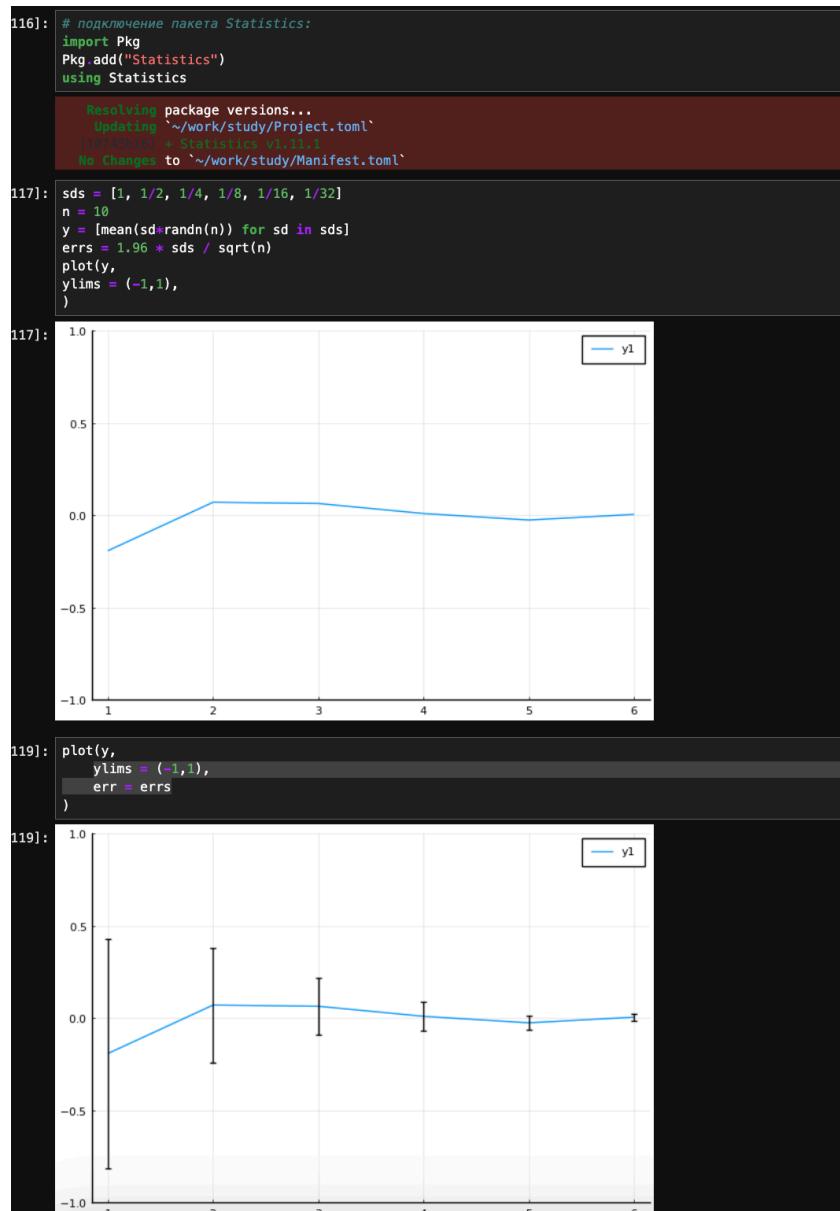


Рис. 4.14: Errorbars

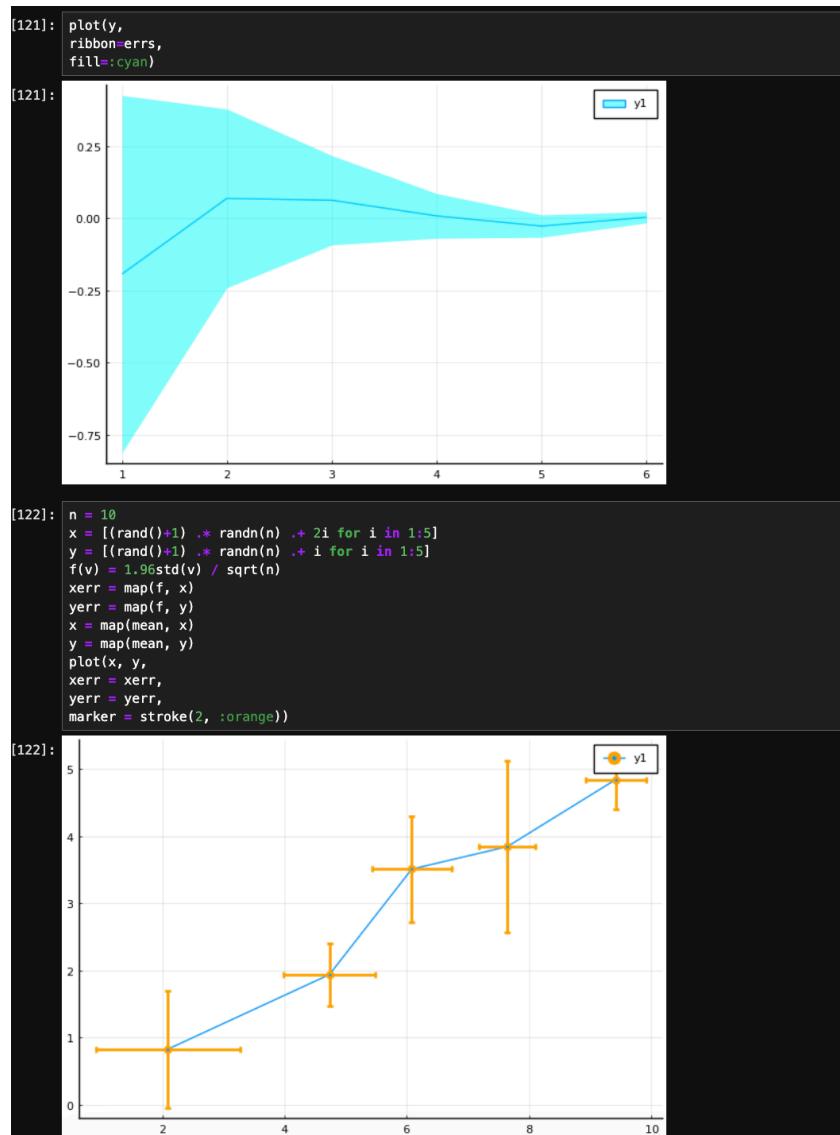


Рис. 4.15: Errorbars

```

[1]: import Pkg
Pkg.add("Distributions")
using Distributions

  Resolving package versions...
  No Changes to `~/work/study/Project.toml'
  No Changes to `~/work/study/Manifest.toml'

[9]: import Pkg; Pkg.add("PlotlyBase")

  Resolving package versions...
  Updating `~/work/study/Project.toml'
  [a03406cd] + PlotlyBase v0.8.19
  No Changes to `~/work/study/Manifest.toml'

[2]: using Plots

[3]: using PyPlot

[5]: pyplot()
ages = rand(15:55,1000)
Plots.histogram(ages)

[5]: 

```

Рис. 4.16: Использование пакета Distributions

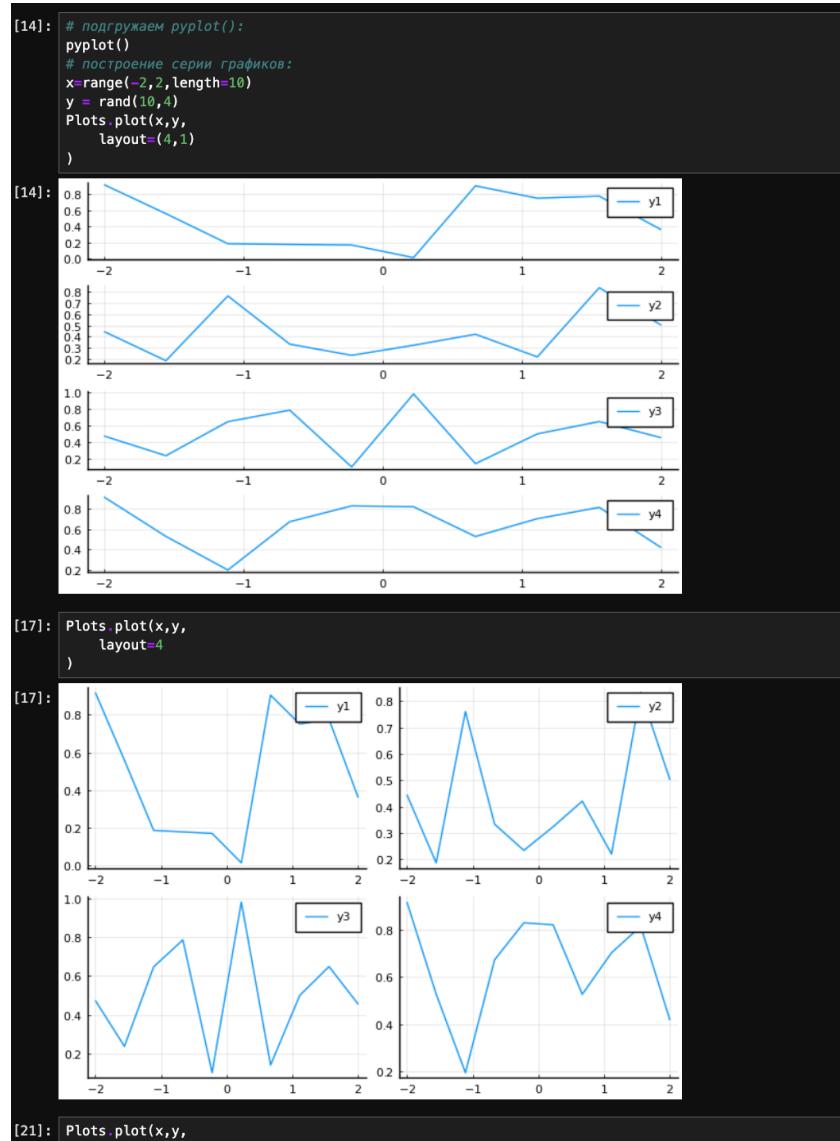


Рис. 4.17: Подграфики

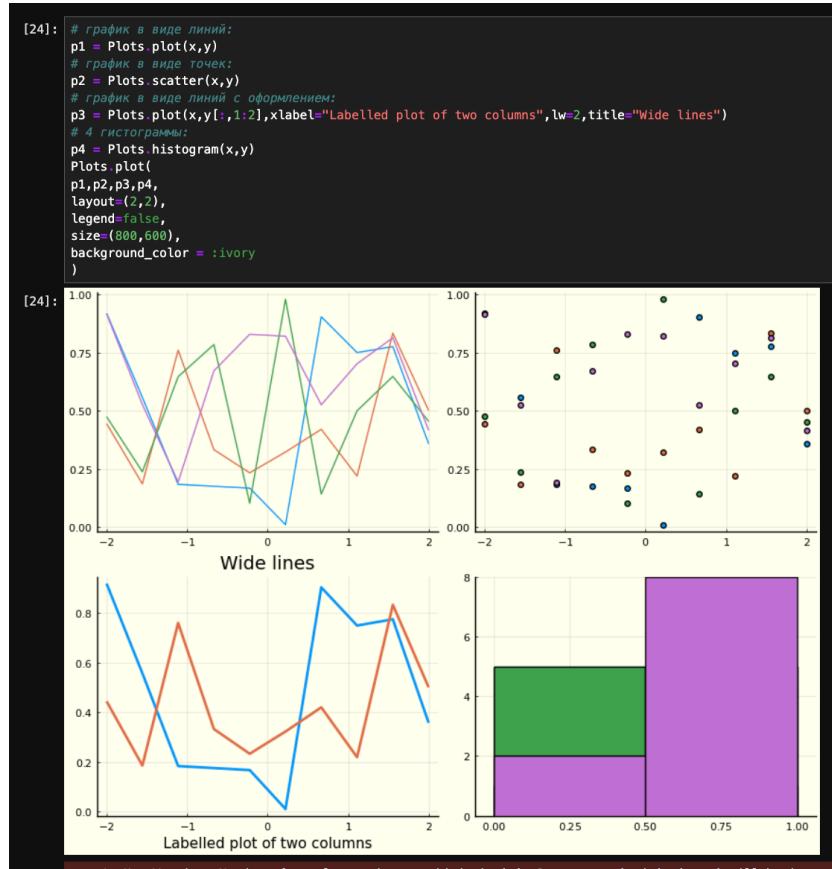


Рис. 4.18: Подграфики



Рис. 4.19: Подграфики

4.1 Задания для самостоятельного выполнения

Выполним задания (рис. 4.20-4.30).

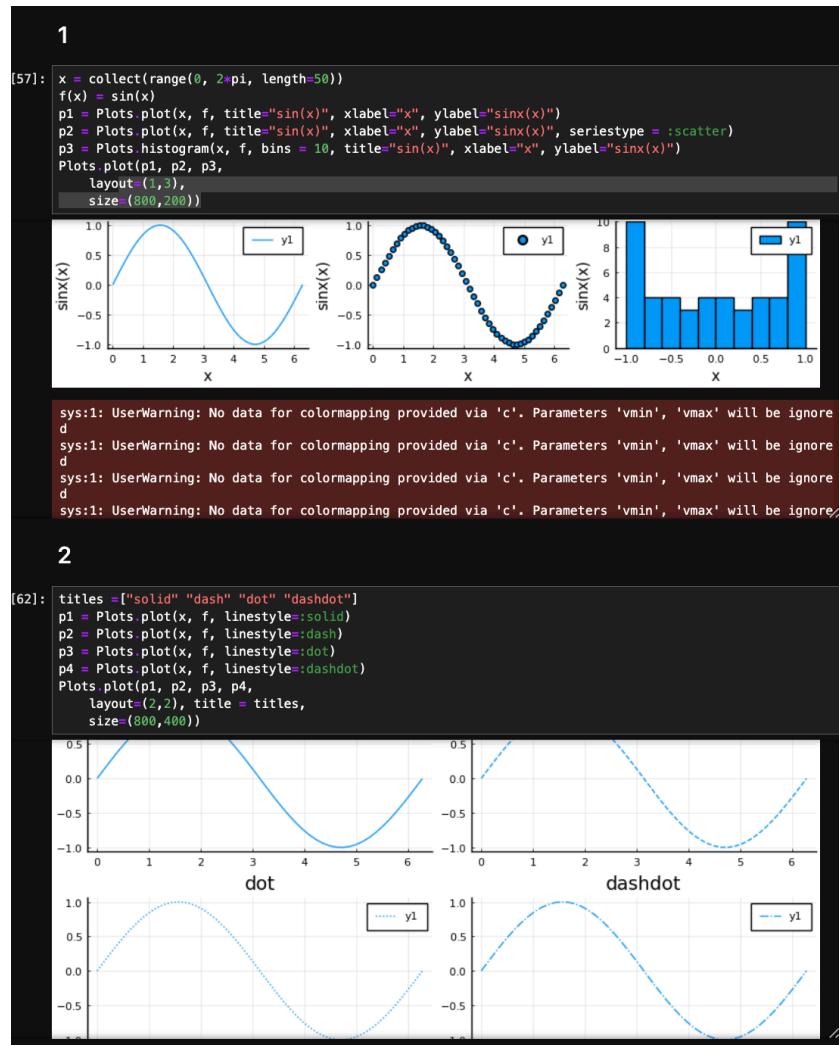


Рис. 4.20: Задание №1 и №2



Рис. 4.21: Задание №3

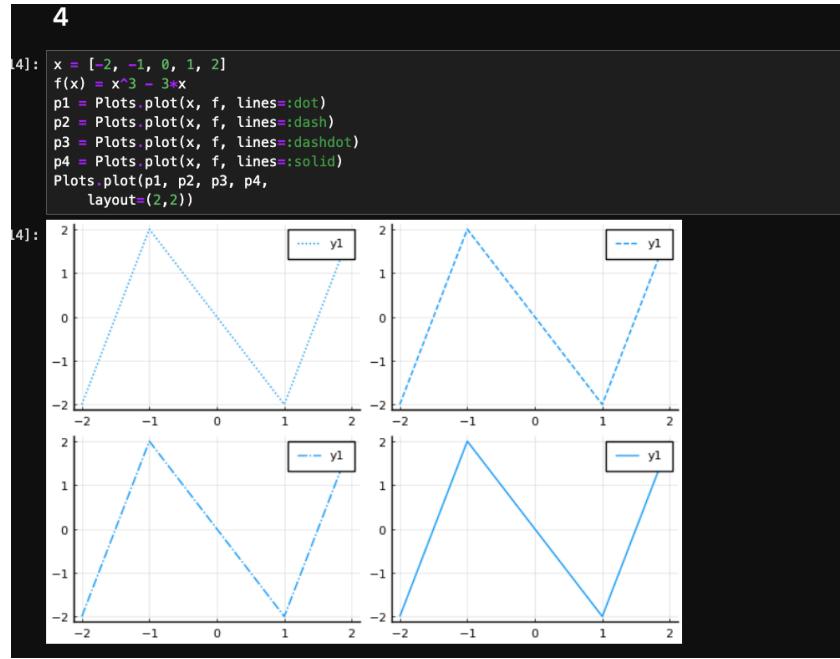


Рис. 4.22: Задание №4

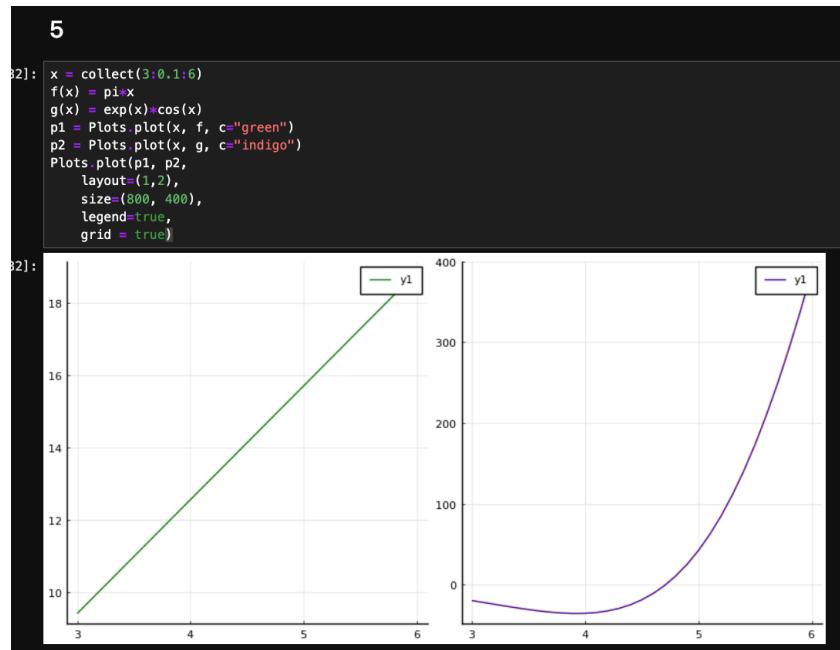


Рис. 4.23: Задание №5

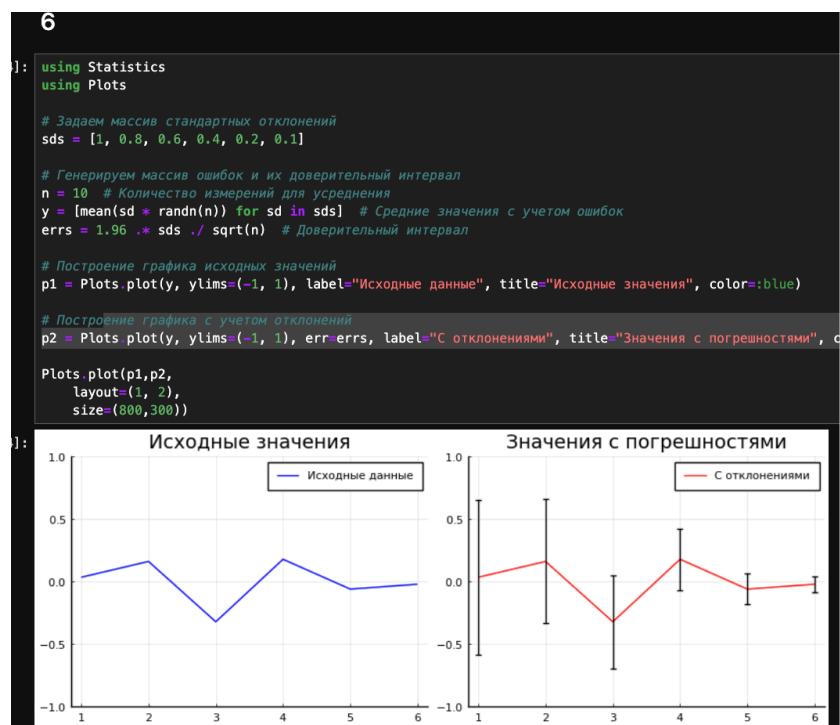


Рис. 4.24: Задание №6

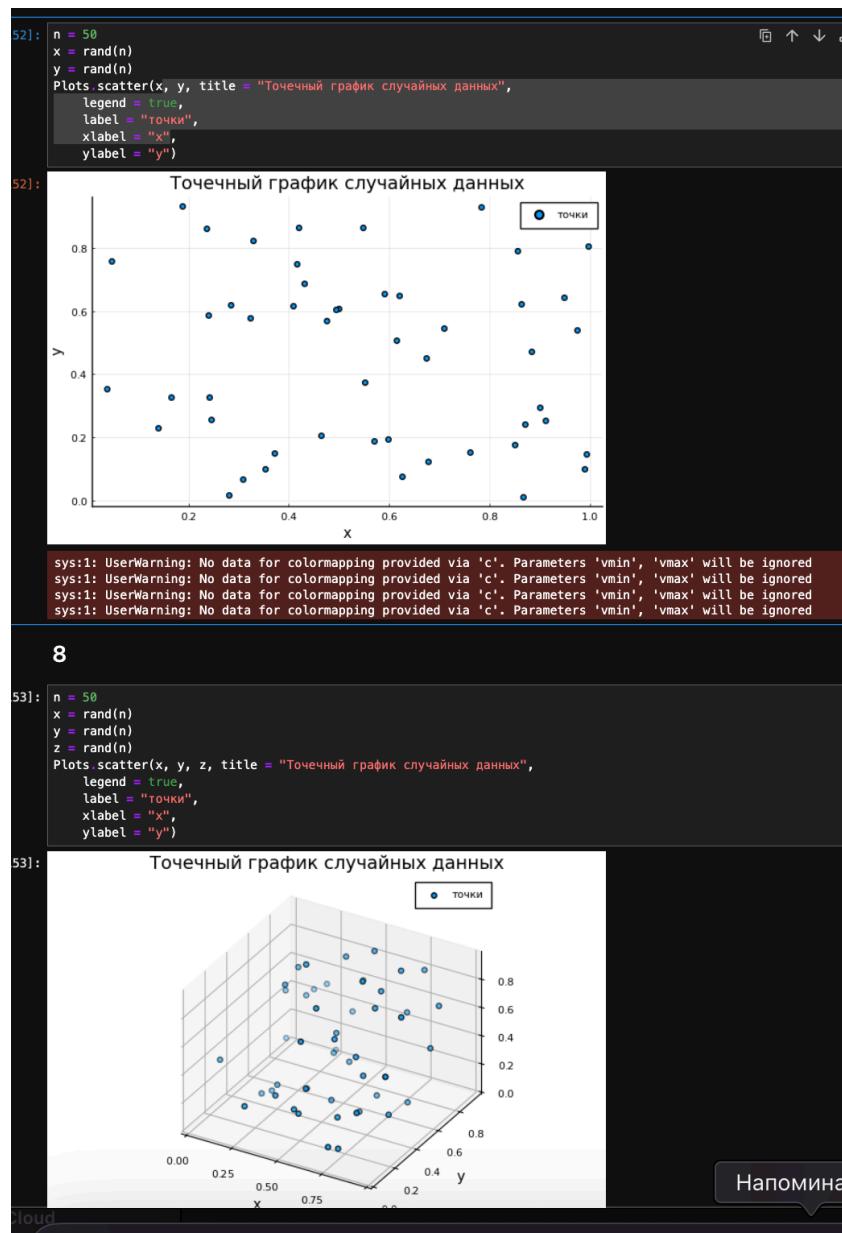


Рис. 4.25: Задание №7 и №8

```

julia> using Plots
julia> gr()

julia> anim = @animate for i in 1:50
julia>     x = 0:0.1:(2π + 0.1 * i)
julia>     y = sin.(x)
julia>     Plots.plot(x, y, label="sin(x)", color=:blue, lw=2, legend=:top, ylim=(-1.5, 1.5))
julia>     xlabel!("x")
julia>     ylabel!("sin(x)")
julia>     title!("Построение синусоиды")
julia> end
julia> Animation("var/folders/3s/wcls7315zq81rzkg_y9b0440000gn/T/jl_d5SKM", ["000001.png", "000002.png", "000003.png",
julia> "000004.png", "000005.png", "000006.png", "000007.png", "000008.png", "000009.png", "000010.png", "000011.png", "000012.png", "000013.png", "000014.png", "000015.png", "000016.png", "000017.png", "000018.png", "000019.png", "000020.png", "000021.png", "000022.png", "000023.png", "000024.png", "000025.png", "000026.png", "000027.png", "000028.png", "000029.png", "000030.png", "000031.png", "000032.png", "000033.png", "000034.png", "000035.png", "000036.png", "000037.png", "000038.png", "000039.png", "000040.png", "000041.png", "000042.png", "000043.png", "000044.png", "000045.png", "000046.png", "000047.png", "000048.png", "000049.png", "000050.png"])
julia> gif(anim, "sine_wave_animation.gif", fps=20)
[ Info: Saved animation to /Users/darabileceva/work/study/julia/sine_wave_animation.gif

```

Рис. 4.26: Задание №9

10

```

5]: using Plots

# Параметры
r_ = 1 # Радиус малой окружности
k=5 # Значения модуля k (два целых и два рациональных)
n = 100 # Число кадров
θ = collect(0:2π/100:2π+2π/100) # Угловые значения для построения

anim = @animate for i in 1:n
    t = θ[1:i] # Логистенное увеличение углов для анимации

    # Большая окружность
    X = r_* * k * cos.(θ)
    Y = r_* * k * sin.(θ)

    # Гипоциклоиды
    x = r_* * (k - 1) * cos.(t) .+ r_* * cos.((k - 1) .* t)
    y = r_* * (k - 1) * sin.(t) .- r_* * sin.((k - 1) .* t)

    # Малая окружность
    xc = r_* * (k - 1) * cos(t[end]) .+ r_* * cos.(θ)
    yc = r_* * (k - 1) * sin(t[end]) .+ r_* * sin.(θ)

    # Построение графиков
    plt = Plots.plot(5, xlim=(-6, 6), ylim=(-6, 6), color=:red, aspect_ratio=1, legend=false, framestyle=:on)
    Plots.plot!(plt, X, Y, color=:blue, legend=false) # Большая окружность
    Plots.plot!(x, y, color=:red) # Гипоциклоиды
    Plots.plot!(xc, yc, color=:black) # Малая окружность
    xl = transpose([r_* * (k - 1) * cos(t[end]) x[end]])
    yl = transpose([r_* * (k - 1) * sin(t[end]) y[end]])
    Plots.plot!(xl, yl, markershape=:circle, markersize=4, color=:black)
    Plots.scatter!([x[end]], [y[end]], color=:red, markerstrokecolor=:red)
end

# Сохранение анимации
gif(anim, "hypocycloid.gif", fps=20)

[ Info: Saved animation to /Users/darabeliceva/work/study/julia/hypocycloid.gif

```

Рис. 4.27: Задание №10

```

]: using Plots

# Параметры
r_ = 1 # Радиус малой окружности
k= 33/4 # Значения модуля k (два целых и два рациональных)
n = 100 # Число кадров
θ = collect(0:2*π/100:2*π+2*π/100) # Угловые значения для построения

anim = @animate for i in 1:n
    t = θ[1:i] # Постепенное увеличение углов для анимации

    # Большая окружность
    X = r_* * k * cos.(θ)
    Y = r_* * k * sin.(θ)

    # Гипоциклоиды
    x = r_* * (k - 1) * cos.(t) .+ r_* * cos.((k - 1) .* t)
    y = r_* * (k - 1) * sin.(t) .- r_* * sin.((k - 1) .* t)

    # Малая окружность
    xc = r_* * (k - 1) * cos(t[end]) .+ r_* * cos.(θ)
    yc = r_* * (k - 1) * sin(t[end]) .+ r_* * sin.(θ)

    # Построение графиков
    plt = Plots.plot(5, xlim=(-10, 10), ylim=(-10, 10), color=:red, aspect_ratio=1, legend=false, framestyle=:origin)
    Plots.plot!(plt, X, Y, color=:blue, legend=false) # Большая окружность
    Plots.plot!(x, y, color=:red) # Гипоциклоиды
    Plots.plot!(xc, yc, color=:black) # Малая окружность
    xl = transpose([r_* * (k - 1) * cos(t[end]) y[end]])
    yl = transpose([r_* * (k - 1) * sin(t[end]) y[end]])
    Plots.plot!(xl, yl, markershape=:circle, markersize=4, color=:black)
    Plots.scatter!([x[end]], [y[end]], color=:red, markerstrokecolor=:red)
end

# Сохранение анимации
gif(anim, "hypocycloid.gif", fps=20)

```

[Info: Saved animation to /Users/darabeliceva/work/study/julia/hypocycloid.gif]:

Рис. 4.28: Задание №10

```

2]: using Plots

# Параметры
r_ = 1 # Радиус малой окружности
k = 3 # Значения модуля k (два целых и два рациональных)
n = 100 # Число кадров
θ = collect(0:2π/100:2π+2π/100) # Угловые значения для построения

anim = @animate for i in 1:n
    t = θ[1:i] # Постепенное увеличение углов для анимации

    # Большая окружность
    X = r_ * k * cos.(θ)
    Y = r_ * k * sin.(θ)

    # Эпциклоида
    x = r_ * (k + 1) * cos.(t) .- r_ * cos.((k + 1) .* t)
    y = r_ * (k + 1) * sin.(t) .- r_ * sin.((k + 1) .* t)

    # Малая окружность
    xc = r_ * (k + 1) * cos(t[end]) .- r_ * cos.(θ)
    yc = r_ * (k + 1) * sin(t[end]) .- r_ * sin.(θ)

    # Построение графиков
    plt = Plots.plot(5, xlim=(-7, 7), ylim=(-7, 7), color=:red, aspect_ratio=1, legend=false, framestyle=:origin)
    Plots.plot!(plt, X, Y, color=:blue, legend=false) # Большая окружность
    Plots.plot!(xc, yc, color=:red) # Эпциклоида
    Plots.plot!(xc, yc, color=:black) # Малая окружность
    xl = transpose([r_ * (k + 1) * cos(t[end]) x[end]])
    yl = transpose([r_ * (k + 1) * sin(t[end]) y[end]])
    Plots.plot!(xl, yl, markershape=:circle, markersize=4, color=:black)
    Plots.scatter!([(x[end]), (y[end])], color=:red, markerstrokecolor=:red)
end

# Сохранение анимации
gif(anim, "epicycloid_k_3.gif", fps=20)

[ Info: Saved animation to /Users/darabbeliceva/work/study/julia/epicycloid_k_3.gif

```

Рис. 4.29: Задание №11

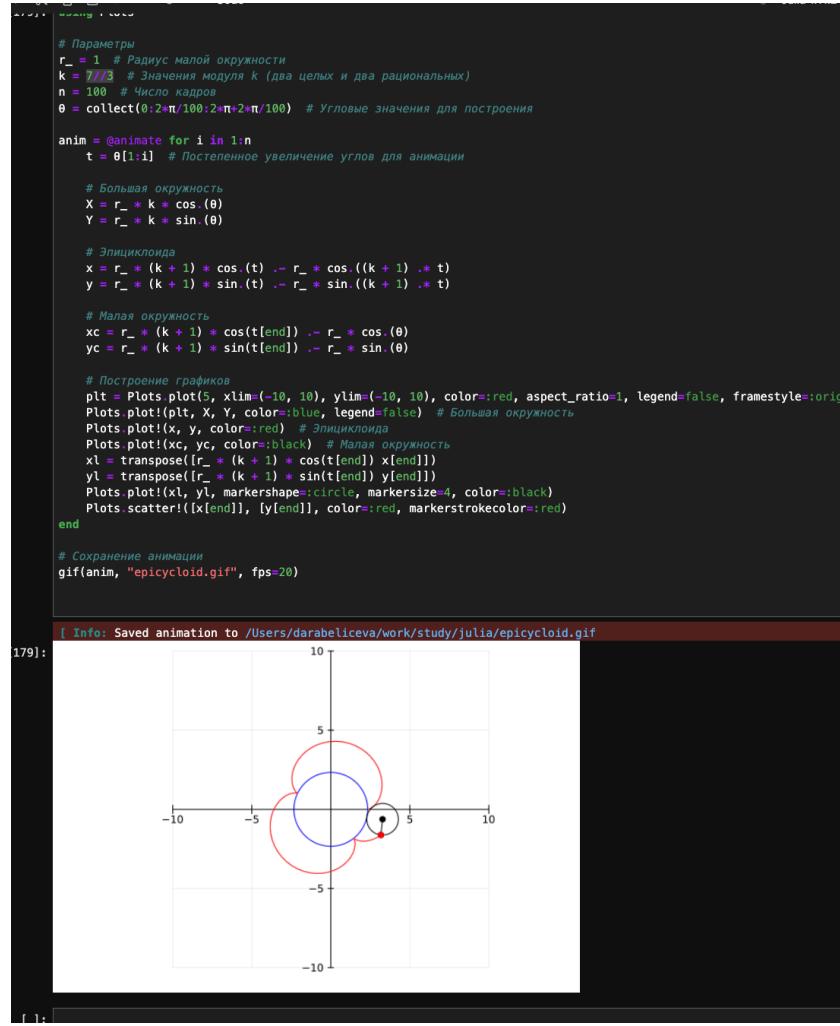


Рис. 4.30: Задание №11

5 Выводы

В результате выполнения данной лабораторной работы я освоила синтаксис языка Julia для построения графиков.

Список литературы

1. JuliaLang [Электронный ресурс]. 2024 JuliaLang.org contributors. URL: <https://julialang.org/> (дата обращения: 11.10.2024).
2. Julia 1.11 Documentation [Электронный ресурс]. 2024 JuliaLang.org contributors. URL: <https://docs.julialang.org/en/v1/> (дата обращения: 11.10.2024).