Лабораторная работа №5.

Евдокимов Иван Андреевич. НФИбд-01-20

2 декабря, 2023, Москва, Россия

Российский Университет Дружбы Народов

Цель лабораторной работы

Цель лабораторной работы

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

Процесс выполнения лабораторной работы

Пункт 1

1. Постройте все возможные типы графиков (простые, точечные, гистограммы и т.д.) функции $y = sin(x)$, $x = 0, 2\pi$. Отобразите все графики в одном графическом окне.

```julia
using Plots

# Задаем значения x от 0 до 2π
x = range(0, stop=2π, length=100)
n = 1

# Вычисляем значения y = sin(x)
y = sin.(x)

# Создаем графическое окно с 2 строками и 2 столбцами
plot_layout = @layout [a b; c d]

# Строим график функции
p1 = plot(x, y, label="sin(x)", xlabel="x", ylabel="y", legend=:topleft)

# Строим точечный график
p2 = scatter(x, y, label="sin(x)", xlabel="x", ylabel="y", legend=:topleft)

# Строим гистограмму
p3 = histogram(n, y, label="sin(x)", xlabel="y", ylabel="Frequency", legend=:topleft)

# Строим столбчатую диаграмму
p4 = bar(x, y, label="sin(x)", xlabel="x", ylabel="y", legend=:topleft)

# Отображаем все графики в одном графическом окне
plot(p1, p2, p3, p4, layout=plot_layout)
```

Рис. 1: Код пункт 1

1.1.

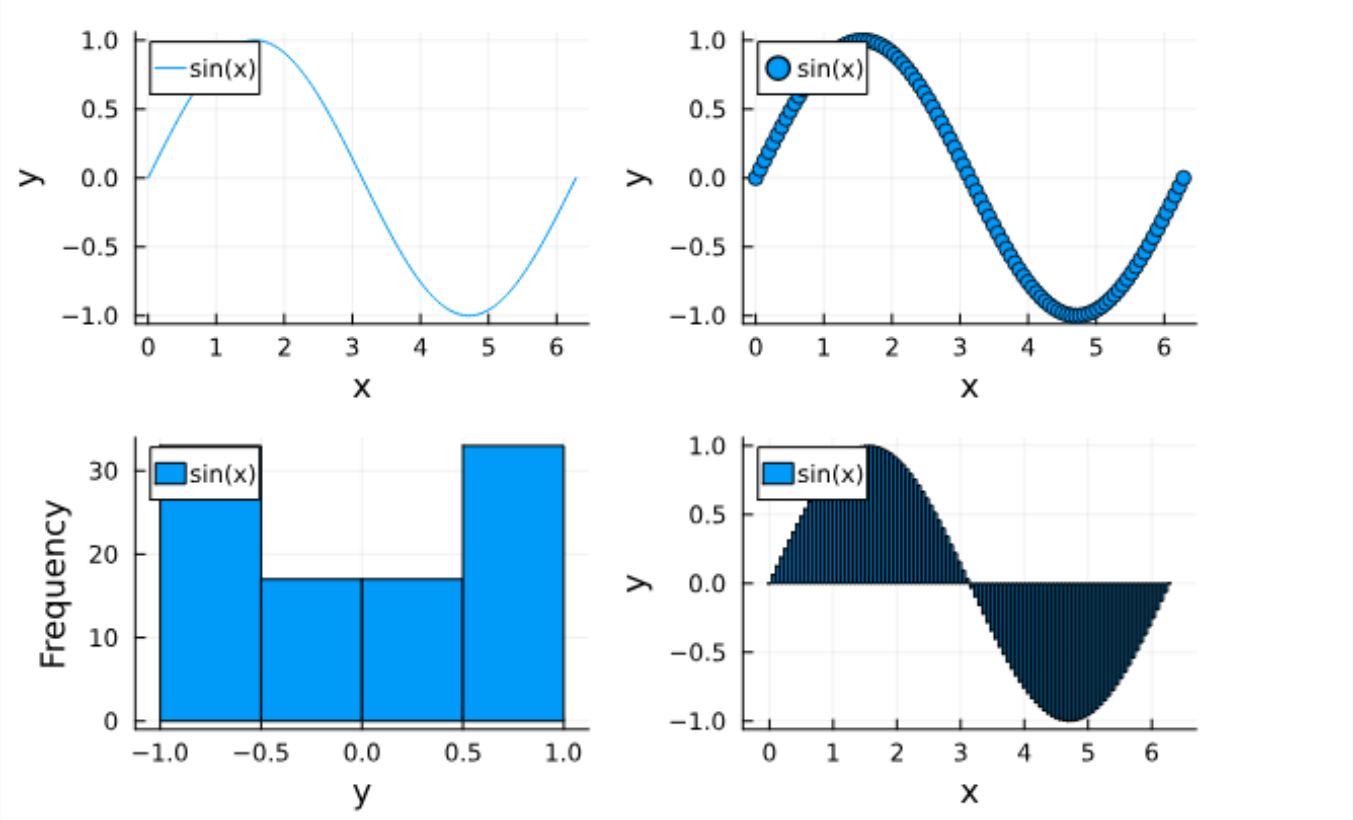

Рис. 2: Пункт 1.1

Пункт 2

2. Постройте графики функции $y = sin(x), x = 0, 2\pi$ со всеми возможными (сколько сможете вспомнить) типами оформления линий графика. Отобразите все графики в одном графическом окне.

```julia
using Plots

x = 0:0.01:2π
y = sin.(x)

linestyles = [:solid, :dot, :dash, :dashdot, :dashdotdot]

plot(x, y, label="sin(x)", linewidth=2, legend=:topleft) # Начальный график с обычной линией

for (i, linestyle) in enumerate(linestyles)
    plot!(x .+ i*0.05, y, label="Line $i", linestyle=linestyle, linewidth=2) # Построение графиков с разными типами линий
end

display(plot!()) # Отображение всех графиков в одном окне
```

Рис. 3: Код пункт 2

2.1.

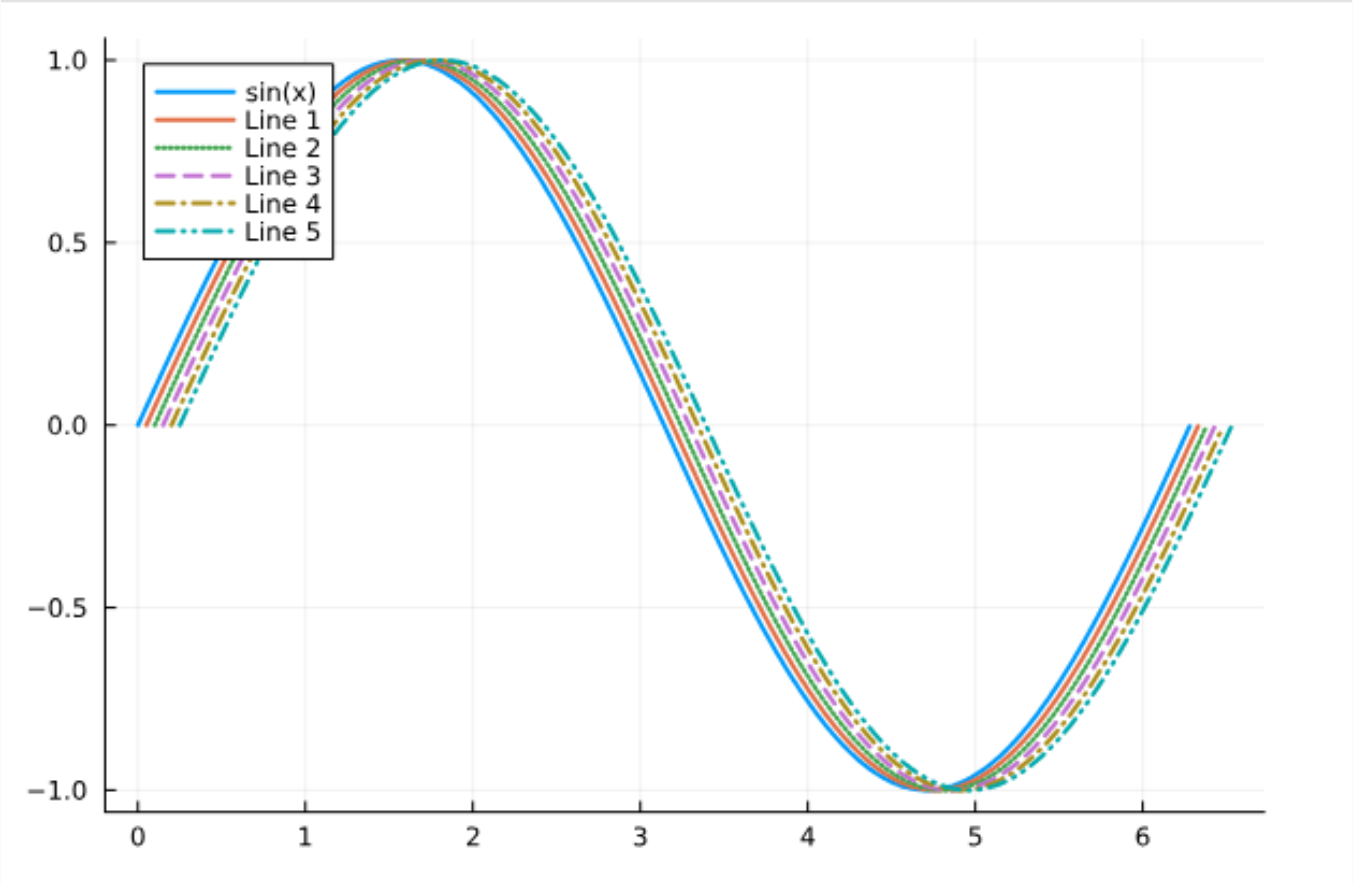

Рис. 4: Пункт 2.1

Пункт 3

3. Постройте график функции $y(x) = \pi x^2 \ln(x)$, назовите оси соответственно. Пусть цвет рамки будет зелёным, а цвет самого графика — красным. Задайте расстояние между надписями и осями так, чтобы надписи полностью умещались в графическом окне. Задайте шрифт надписей. Задайте частоту отметок на осях координат.

```
using Plots

# Определение функции
y1(x) = pi * x^2 * log(x)

# Создание массива значений x
x_values = 0.1:0.1:2.5

# Создание массива значений y
y_values = y1.(x_values)

# Построение графика
plot(x_values, y_values, color=:red, xlabel="x", ylabel="y", label="\$y = \\pi x^{2} \\ln(x)\$",
     title="График функции", linewidth=2, legend=:topright,
     framestyle=:box, grid=:on, size=(600, 400),
     guidefont=font(12, "Arial"), tickfont=font(10, "Arial"))

display(plot!())  # Отображение графика
```

Рис. 5: Код пункт 3

3.1.

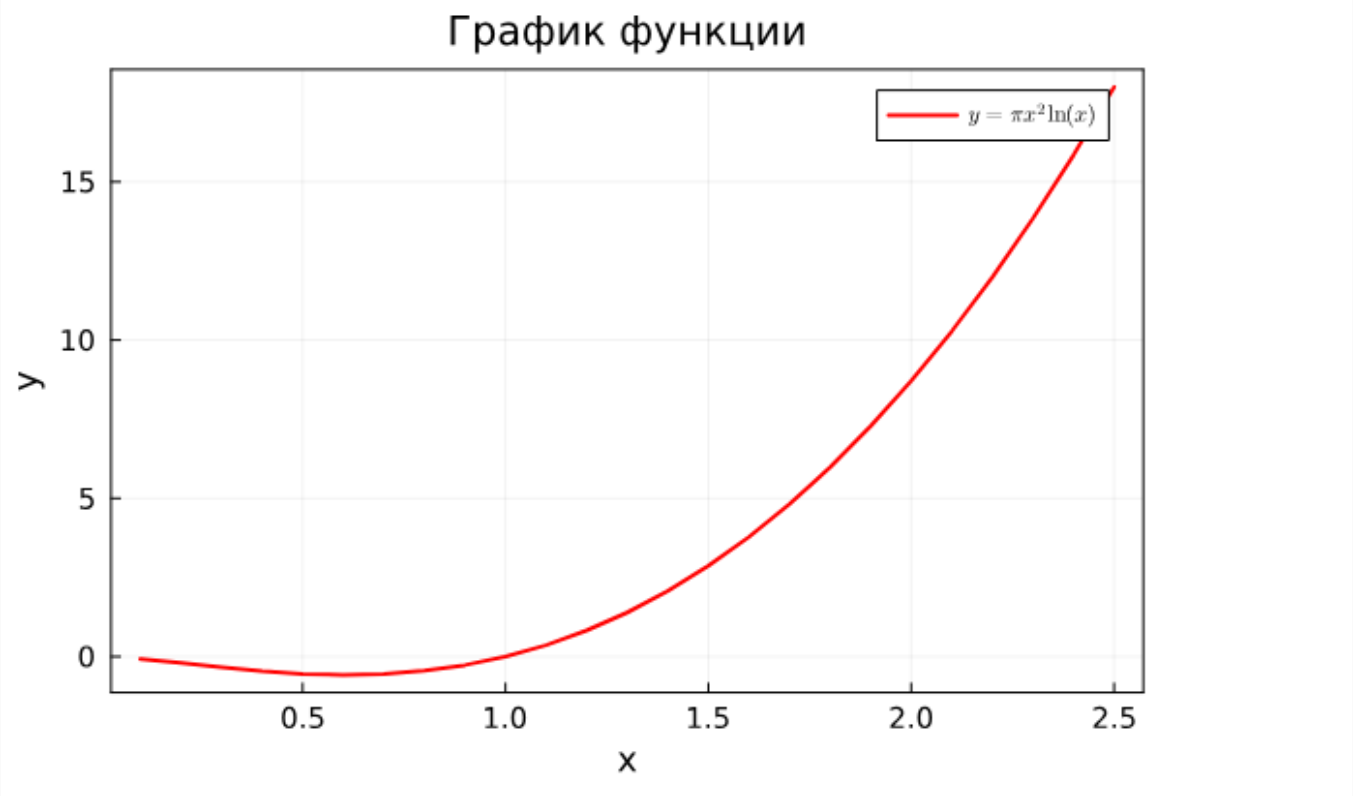

Рис. 6: Пункт 3.1

Пункт 4

4. Задайте вектор $x = (-2, -1, 0, 1, 2)$. В одном графическом окне (в 4-х подокнах) изобразите графически по точкам x значения функции $y(x) = x^3 - 3x$ в виде:(точек, линий, линий и точек, кривой.) Сохраните полученные изображения в файле figure_familiya.png, где вместо familiya укажите вашу фамилию

```julia
using Plots

# Задание вектора x
x = [-2, -1, 0, 1, 2]

# Функция y(x) = x^3 - 3x
y2(x) = x^3 - 3x

# Отображение в 4-х подокнах
plot1 = scatter(x, y2.(x), markersize=4, markercolor=:blue, xlabel="x", ylabel="y", label="Точки")
plot2 = plot(x, y2.(x), line=:line, linewidth=2, linecolor=:green, xlabel="x", ylabel="y", label="Линии")
plot3 = plot(x, y2.(x), line=:line, linewidth=2, linecolor=:orange, markersize=4, markercolor=:red, m=:circle, xlabel="x", ylabel="y", label="Линии и точки")
plot4 = curves(x, y2.(x), line=:auto, linewidth=2, linecolor=:purple, xlabel="x", ylabel="y", label="Кривая")

# Объединение графиков в одно графическое окно
plot(plot1, plot2, plot3, plot4, layout=(2,2), legend=:bottomright, size=(800, 600))

display(plot!())
```

Рис. 7: Код пункт 4

4.1.

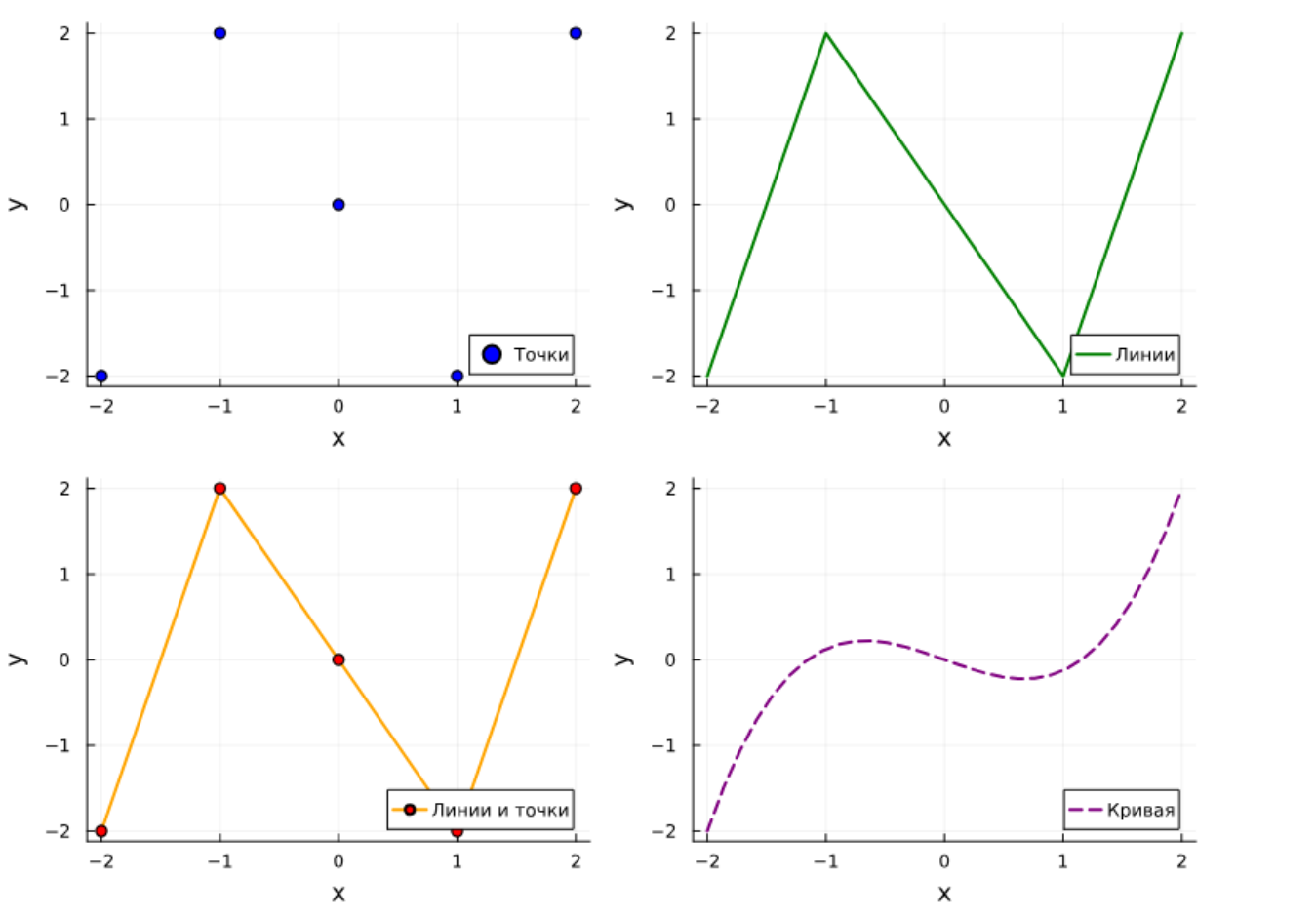

Рис. 8: Пункт 4.1

Пункт 5

5. Задайте вектор $x = (3, 3.1, 3.2, \ldots, 6)$. Постройте графики функций $y_1\,(x) = \pi x$ и $y_2\,(x) = \exp\,(x)\cos\,(x)$ в указанном диапазоне значений аргумента x следующим образом: постройте оба графика разного цвета на одном рисунке, добавьте легенду и сетку для каждого графика; укажите недостатки у данного построения; постройте аналогичный график с двумя осями ординат.

```julia
using Plots

# Задание вектора x
x = 3:0.01:6

# Функции y1(x) и y2(x)
y11(x) = π * x
y22(x) = exp(x) * cos(x)

# Построение графиков на одном рисунке с легендой и сеткой
plot(x, y1.(x), label="y₁(x) = πx", color=:blue, xlabel="x", ylabel="y₁(x)",
     title="Графики функций", linewidth=2, legend=:left, grid=:on)
plot!(x, y2.(x), label="y₂(x) = e*cos(x)", color=:red, linewidth=2)

# Отображение графиков
display(plot!())
```

Рис. 9: Код пункт 5 часть 1

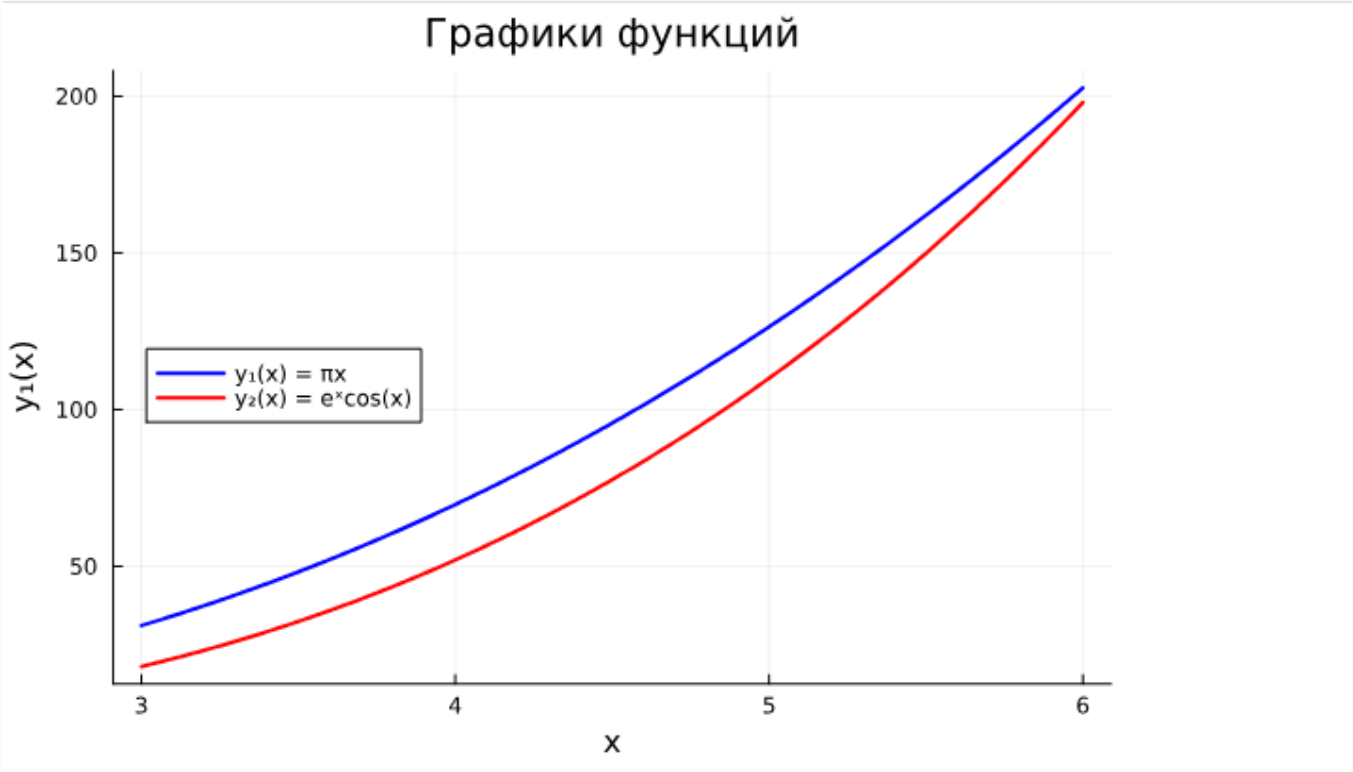

Рис. 10: Пункт 5.1

Код часть 2

```julia
using Plots

# Задание вектора x
x = 3:0.1:6

# Функции y1(x) и y2(x)
y11(x) = π * x
y22(x) = exp(x) * cos(x)

# Построение аналогичного графика с двумя осями ординат
plot(x, y1.(x), label="y₁(x) = πx", color=:blue, xlabel="x", ylabel="y₁(x)",
     title="Графики функций с двумя осями ординат", linewidth=2, legend=:left, grid=:on)
plot!(twinx(), x, y2.(x), secondary=true, label="y₂(x) = eˣcos(x)", color=:red, linewidth=2,
      ylabel="y₂(x)", linestyle=:dot)

# Отображение графиков
display(plot!())
```

Рис. 11: Код пункт 5 часть 2

5.2

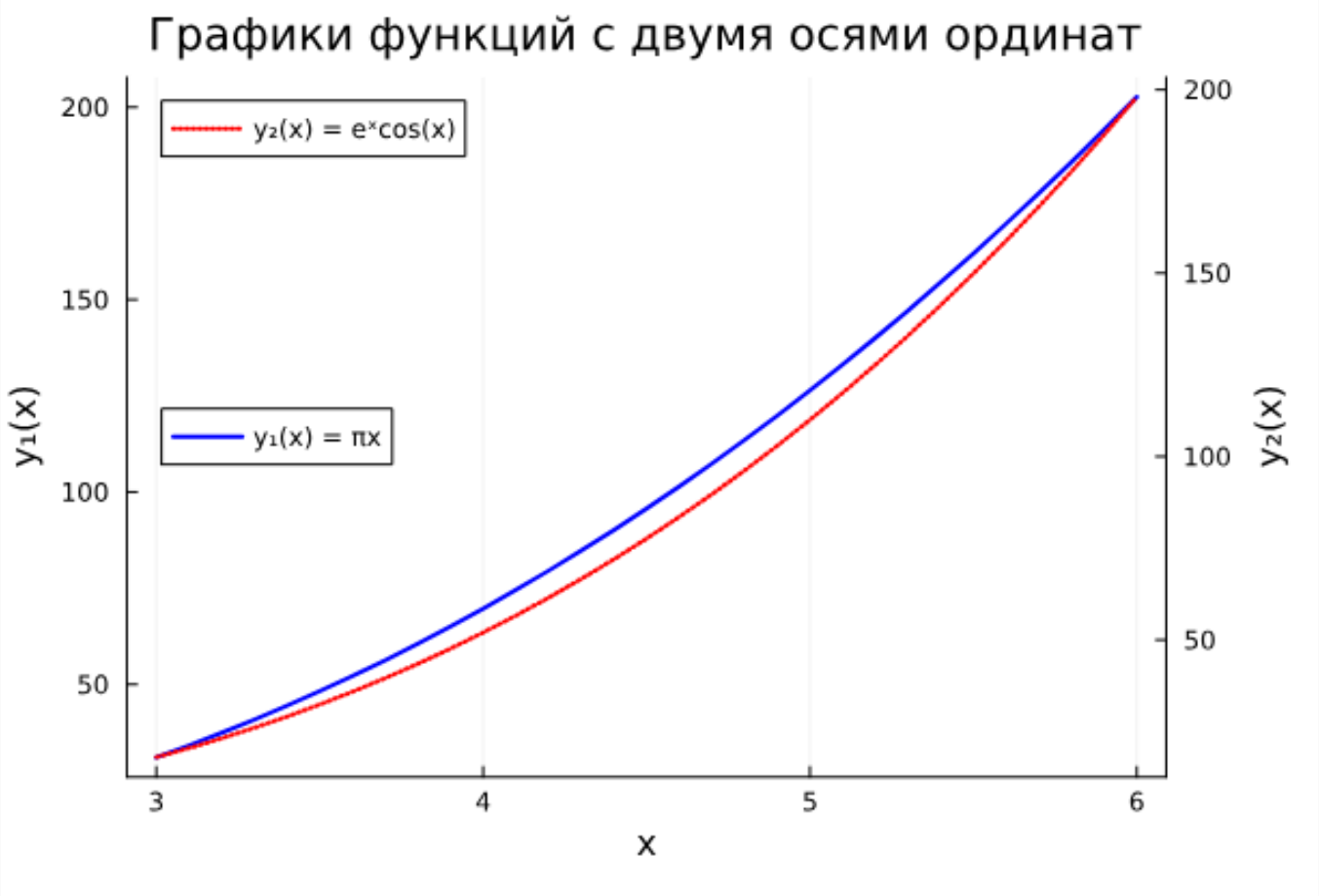

Рис. 12: Пункт 5.2

Пункт 6

6. Постройте график некоторых экспериментальных данных (придумайте сами), учитывая ошибку измерения.

```julia
using Plots
using Random

# Генерация исходных данных
x = 1:0.1:10
y_true = 2 * sin.(x) .+ 0.5 * x  # Исходная зависимость

# Добавление случайной ошибки к данным
error_std = 0.5
y_noisy = y_true .+ randn(size(y_true)) * error_std

# Построение графика с учетом ошибки измерения
plot(x, y_noisy, ribbon=error_std, fillalpha=0.2, label="Измерения с ошибкой",
     xlabel="x", ylabel="y", title="График экспериментальных данных с ошибкой",
     linewidth=2, color=:blue)

plot!(x, y_true, label="Исходная зависимость", linewidth=2, color=:red)

# Отображение графика
display(plot!())
```

Рис. 13: Код пункт 6

6.1.

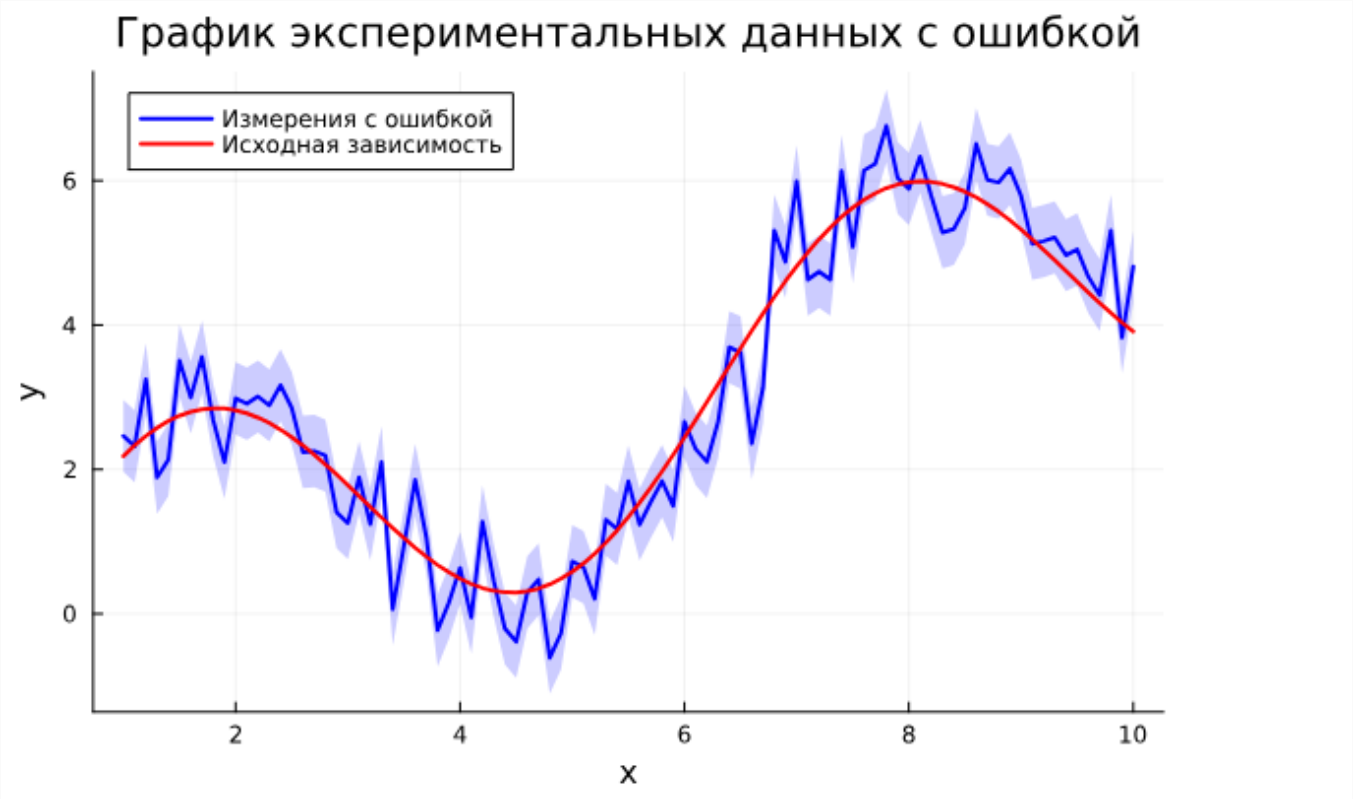

Рис. 14: Пункт 6.1

Пункт 7

7. Постройте точечный график случайных данных. Подпишите оси, легенду, название графика.

```julia
using Plots
using Random

# Генерация случайных данных
x_data = rand(100)  # 100 случайных значений для оси x
y_data = rand(100)  # 100 случайных значений для оси y

# Построение точечного графика
scatter(x_data, y_data, label="Случайные данные", xlabel="Ось X", ylabel="Ось Y", title="Точечный график случайных данных", marker=:circle, color=:blue)

# Отображение графика
display(plot!())
```

Рис. 15: Код пункт 7

7.1.

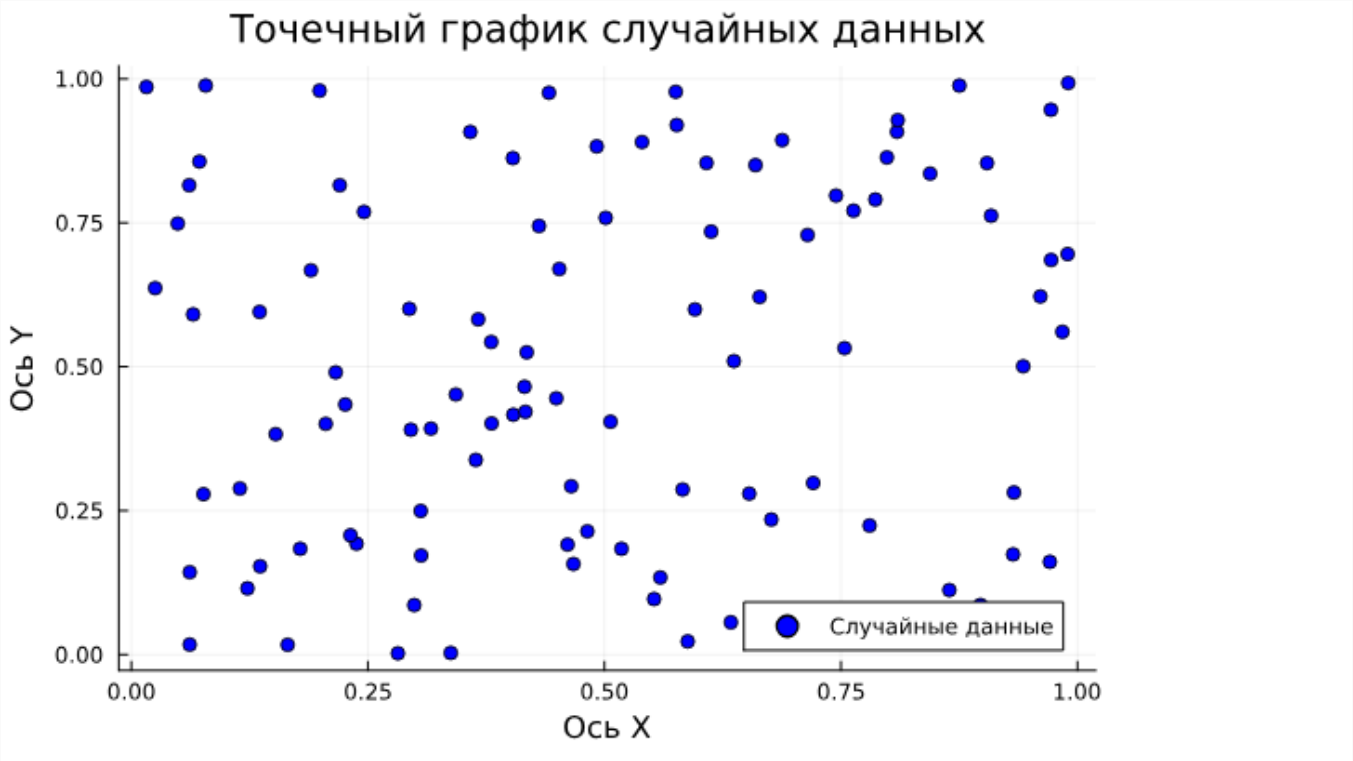

Рис. 16: Пункт 7.1

Пункт 8

8. Постройте 3-мерный точечный график случайных данных. Подпишите оси, легенду, название графика.

```julia
using Plots


# Генерация случайных данных для осей x, y, z
x_data = rand(1:100,100)
y_data = rand(1:100,100)
z_data = rand(1:100,100)

# Построение 3D точечного графика
plot(x_data, y_data, z_data, seriestype=:scatter, markersize = 7, label="Случайные данные", xlabel="Ось X", ylabel="Ось Y", zlabel="Ось Z", title="3D Точечный график случайных данных", marker=:circle, color=:blue)

# Отображение графика
display(plot!())
```

Рис. 17: Код пункт 8

8.1.

Рис. 18: Пункт 8.1

Пункт 9

9. Создайте анимацию с построением синусоиды. То есть вы строите последовательность графиков синусоиды, постепенно увеличивая значение аргумента. После соединит их в анимацию.

```
using Plots

# Создание функции, которую будем анимировать
f(x, ω, t) = sin(ω * x + t)

# Диапазон значений x
x_values = 0:0.1:10

# Диапазон значений времени для анимации
time_values = 0:0.1:2π

# Создание анимации
anim = @animate for t in time_values
    y_values = f.(x_values, 1, t)  # Частота ω = 1
    plot(x_values, y_values, label="Синусоида", xlabel="x", ylabel="y", title="Анимация построения синусоиды",
         ylims=(-1.5, 1.5), linewidth=2, color=:blue)
end

# Сохранение анимации в файл
gif(anim, "sinusoid_animation.gif", fps = 10)
```

Рис. 19: Код пункт 9

9.1.

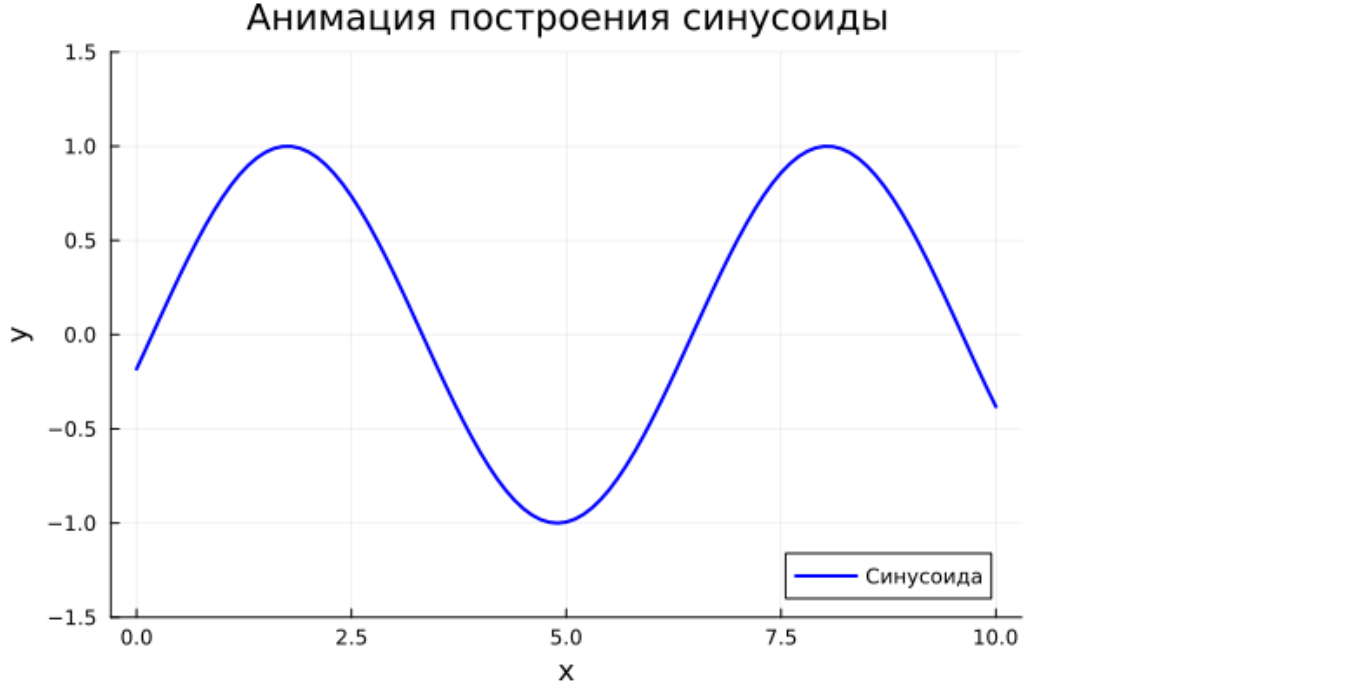

Рис. 20: Пункт 9.1

Пункт 10

10. Постройте анимированную гипоциклоиду для 2 целых значений модуля k и 2 рациональных значений модуля k.

```julia
using Plots

# Функция для создания гипоциклоиды
function hypocycloid(a, b, k)
    θ = range(0, stop=2π, length=100)
    x = (a - b) * cos.(θ) .+ b * k * cos.((a - b) / b * θ)
    y = (a - b) * sin.(θ) .- b * k * sin.((a - b) / b * θ)
    return x, y
end

# Значения модуля k (для целых и рациональных чисел)
k_values = [2, 3//2, 4, 5//2]

# Создание анимации для разных значений k
anim = @animate for k in k_values
    x, y = hypocycloid(1, 0.3, k)

    plot(x, y, aspect_ratio=:equal, label="k = $k", xlabel="x", ylabel="y",
         title="Анимированная гипоциклоида", linewidth=2)
end

# Сохранение анимации в файл
gif(anim, "hypocycloid_animation.gif", fps=5)
```

Рис. 21: Код пункт 10

9.1.

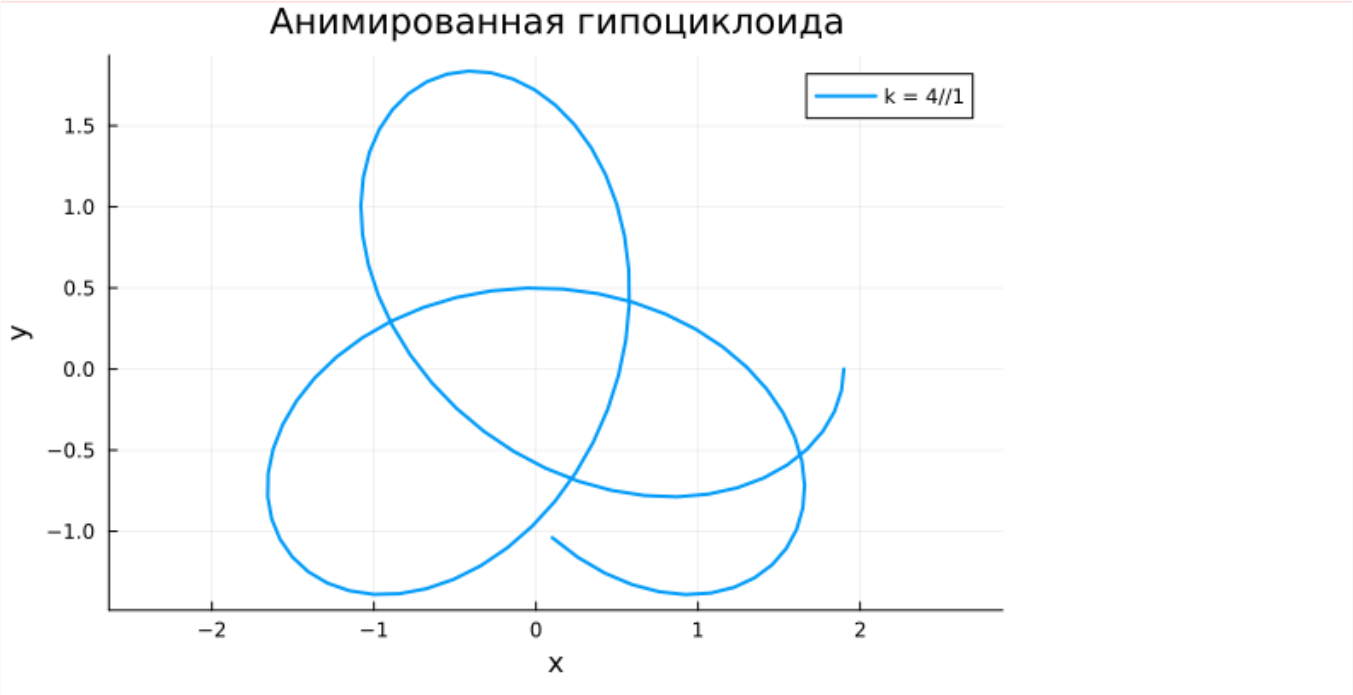

Рис. 22: Пункт 10.1

Пункт 11

11. Постройте анимированную эпициклоиду для 2 целых значений модуля k и 2 рациональных значений модуля k.

```julia
using Plots

# Функция для создания эпициклоиды
function epicycloid(a, b, k)
    θ = range(0, stop=2π, length=100)
    x = (a + b) * cos.(θ) .- b * k * cos.((a + b) / b * θ)
    y = (a + b) * sin.(θ) .- b * k * sin.((a + b) / b * θ)
    return x, y
end

# Значения модуля k (для целых и рациональных чисел)
k_values = [2, 3//2, 4, 5//2]

# Создание анимации для разных значений k
anim = @animate for k in k_values
    x, y = epicycloid(1, 0.3, k)

    plot(x, y, aspect_ratio=:equal, label="k = $k", xlabel="x", ylabel="y",
         title="Анимированная эпициклоида", linewidth=2)
end

# Сохранение анимации в файл
gif(anim, "epicycloid_animation.gif", fps=5)
```

Рис. 23: Код пункт 11

11.1.

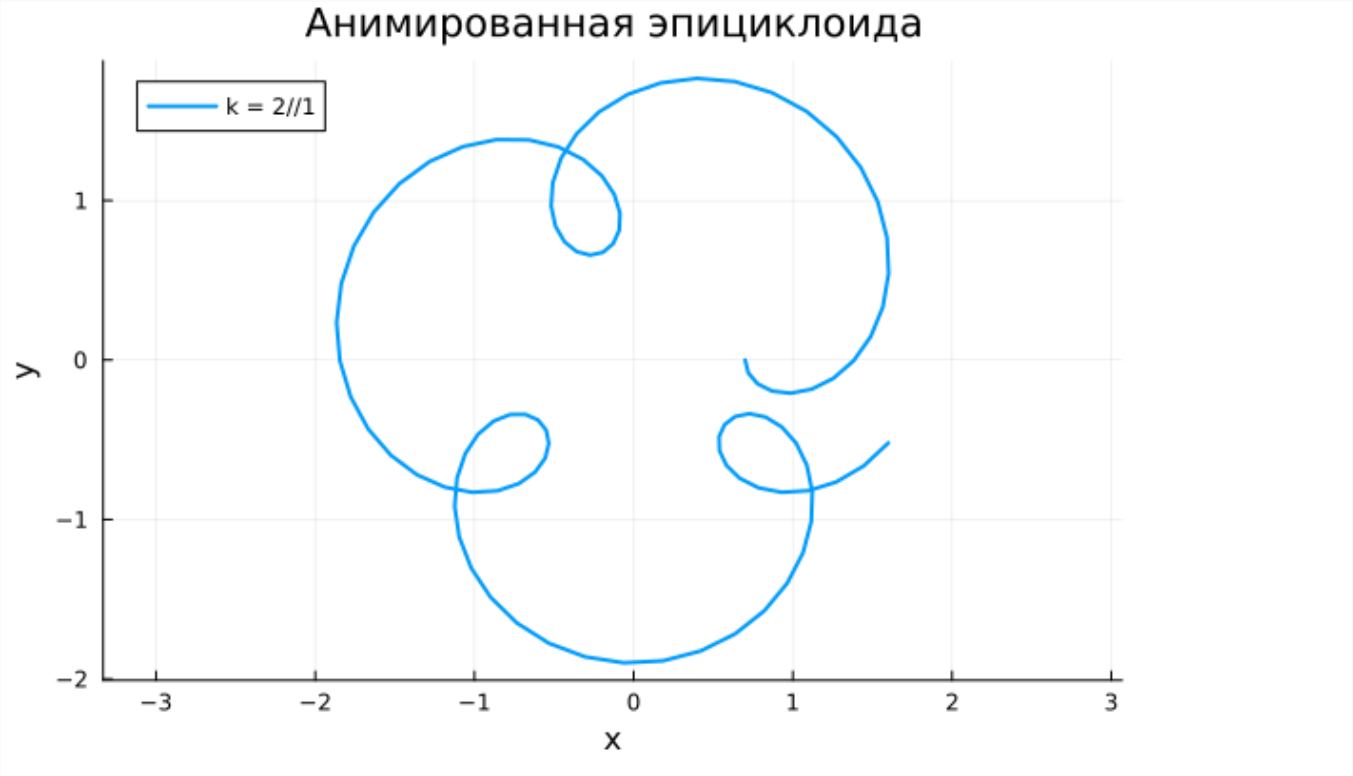

Рис. 24: Пункт 11.1

Выводы

Выводы

Мною были освоены основы синтаксисиса языка Julia для построения графиков.