

Лабораторная работа 5

Построение графиков

Ланцова Яна Игоревна

Содержание

1 Цель работы	5
2 Задание	6
3 Выполнение лабораторной работы	7
3.1 Задание 1	10
3.2 Задание 2	11
3.3 Задание 3	12
3.4 Задание 4	13
3.5 Задание 5	14
3.6 Задание 6	15
3.7 Задание 7	16
3.8 Задание 8	16
3.9 Задание 9	17
3.10 Задание 10	17
3.11 Задание 11	19
4 Выводы	22

Список иллюстраций

3.1 Основные пакеты для работы с графиками в Julia	7
3.2 Основные пакеты для работы с графиками в Julia	7
3.3 Опции при построении графика	8
3.4 Точечный график	8
3.5 Две оси ординат	8
3.6 Полярные координаты	9
3.7 Параметрический график	9
3.8 Векторные поля	9
3.9 Анимация	10
3.10 Подграфики	10
3.11 Задание №1	11
3.12 Графики №1	11
3.13 Задание №2	12
3.14 Графики №2	12
3.15 Задание №3	13
3.16 Задание №4	13
3.17 Графики №4	14
3.18 Задание №5	15
3.19 Задание №5	15
3.20 Задание №6	16
3.21 Задание №7	16
3.22 Задание №8	17
3.23 Задание №9	17
3.24 Задание №10	18
3.25 Задание №10	18
3.26 Задание №10	18
3.27 Задание №10	19
3.28 Задание №10	19
3.29 Задание №11	20
3.30 Задание №11	20
3.31 Задание №11	20
3.32 Задание №11	21
3.33 Задание №11	21

Список таблиц

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 Выполнение лабораторной работы

Выполним примеры из лабораторной работы для знакомства с пакетами по отрисовки графиков и их функциями (рис. 3.1 - 3.10).

```
[*:]: using Pkg  
Pkg.add("Plots")  
Pkg.add("PyPlot")  
Pkg.add("UnicodePlots")  
# подключаем для использования Plots:  
using Plots  
  
Updating registry at `~/.julia/registries/General.toml'  
Resolving package versions...  
Installed x265_jll └── v4.1.0+0  
Installed GR_jll └── v0.73.18+0  
Installed Cairo_jll └── v0.1.1+0  
Installed JpegTurbo_jll └── v2.3.1.3+0  
Installed Libmount_jll └── v2.41.2+0  
Installed LERC_jll └── v4.0.1+0  
Installed libdecor_jll └── v0.2.2+0  
Installed LoggingExtras └── v1.2.0  
Installed OpenBLAS_jll └── v1.5.2+0  
Installed ConcurrentUtilities └── v0.1.0+0  
Installed Measure └── v0.3.3  
Installed RelocatableFolders └── v1.0.1  
Installed Xorg_xkbcomp_jll └── v1.4.7+0  
Installed Contour └── v0.6.3  
Installed Gravitation └── v1.0.2  
Installed Xorg_xcb_util_image_jll └── v1.1.1+0  
Installed Xorg_xcb_util_wm_jll └── v0.4.2+0  
Installed PlotUtils └── v1.4.3  
Installed RecipesPipeline └── v0.6.12  
Installed OpenSSL └── v1.6.0  
Installed DelimitedFiles └── v1.9.1
```

Рис. 3.1: Основные пакеты для работы с графиками в Julia

Основные пакеты для работы с графиками в Julia

```
[3]: # задание функции:  
f(x) = (3x.^2 + 6x - 9).exp.(-0.3x)  
# 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))  
# генерирование массива значений y:  
y = f(x);
```

```
[4]: # указывается, что для построения графика используется gr():  
gr()  
# 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")
```

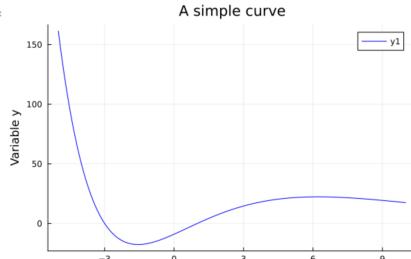


Рис. 3.2: Основные пакеты для работы с графиками в Julia

Опции при построении графика

```
[6]: # задание функции sin(x);
sin_theor(x) = sin(x)
# задание функции разложения исходной функции в ряд Тейлора:
sin_taylor(x) = [(-1)^i*x^(2*i+1)/factorial(2*i+1) for i in 0:4] |> sum
```

```
[6]: sin_taylor (generic function with 1 method)
```

```
[8]: # построение двух функций на одном графике:
plot(sin_theor)
plot(sin_taylor)
```

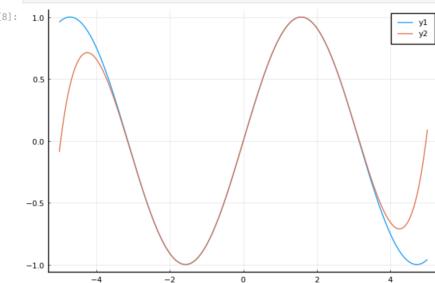


Рис. 3.3: Опции при построении графика

Точечный график

```
[10]: # параметры распределения точек на плоскости:
x = range(1,10,length=10)
y = rand(10)
# параметры построения графика:
plot(x, y, seriestype=:scatter, title = "Точечный график")
```

```
[10]: Точечный график
```

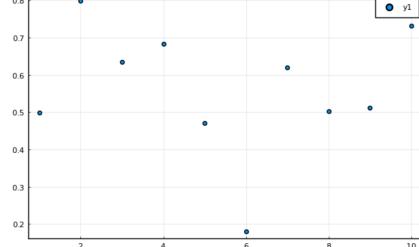


Рис. 3.4: Точечный график

Две оси ординат

```
[21]: # пример случайной траектории
# (заданы обозначение траектории, легенда вверху справа, без сетки)
plot(randn(100), ylabel="y1", legend=:topright, grid=:off,)
```

```
[21]:
```

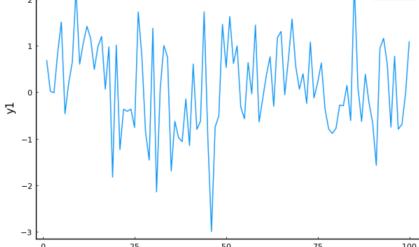


Рис. 3.5: Две оси ординат

Полярные координаты

```
[22]: # функция в полярных координатах:
r(theta) = 1 + cos(theta) * sin(theta)^2
# полярная система координат:
theta = range(0, stop=2pi, length=50)
# график функции, заданной в полярных координатах:
plot(theta, r(theta), proj=polar, lims=(0,1.5))
```

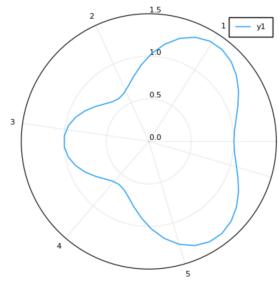


Рис. 3.6: Полярные координаты

Параметрический график

```
[24]: # параметрическое уравнение:
xx(t) = sin(t)
yy(t) = sin(2t)
# построение графика:
plot(xx, yy, theta, 2pi, leg=false, fill=(0,:orange))
```

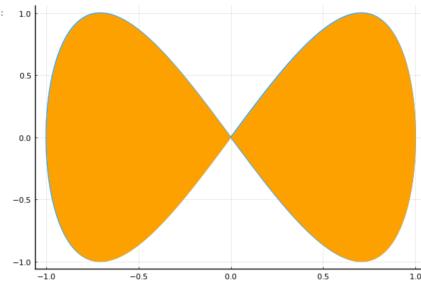


Рис. 3.7: Параметрический график

Векторные поля

```
[25]: # определение переменных:
X = range(-2, stop=2, length=100)
Y = range(-2, stop=2, length=100)
# определение функции:
h(x, y) = x^3 - 3x * y^2
# построение поверхности:
plot(X,Y,h,linetype = :surface)
# построение линий уровня:
contour(X, Y, h)
```

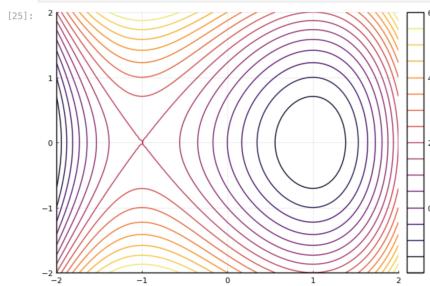


Рис. 3.8: Векторные поля

Анимация

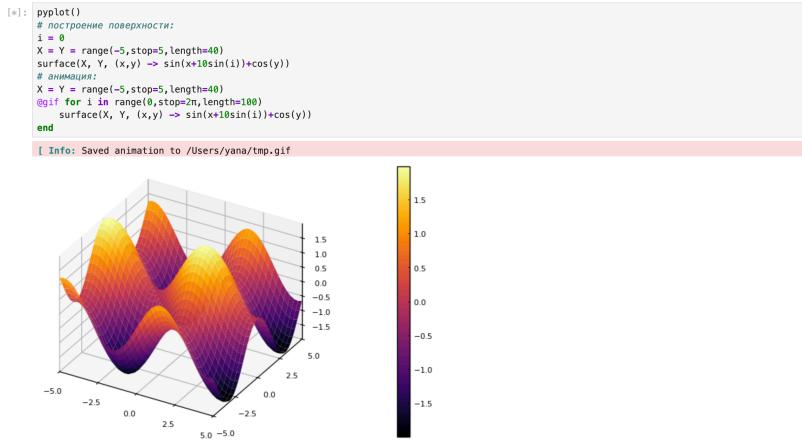


Рис. 3.9: Анимация

Подграфики

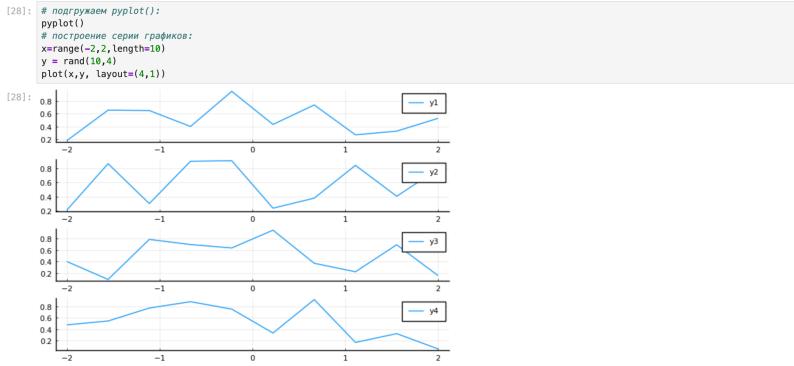


Рис. 3.10: Подграфики

Теперь перейдем к заданиям для самостоятельного выполнения

3.1 Задание 1

Постройте все возможные типы графиков (простые, точечные, гистограммы и т.д.) функции

$$y = \sin(x)$$

$x = 0, 2\pi$. Отобразим все графики в одном графическом окне. (рис. 3.11 - 3.12).

Задания для самостоятельного выполнения

1. Все возможные типы графиков функции $y = \sin(x)$

```
[69]: using Plots
[67]: x = collect(range(0, 2*pi, length=100))
y = sin.(x);

[88]: # Создаем макет 3x3 для отображения всех графиков
p1 = Plots.plot(x, y, title="Линейный график", xlabel="x", ylabel="sin(x)", legend=false)
p2 = scatter(x, y, title="Точечный график", xlabel="x", ylabel="sin(x)", legend=false)
p3 = Plots.plot(x, y, seriestype=:step, title="Ступенчатый график", xlabel="x", ylabel="sin(x)", legend=false)

x_hist = range(0, 2*pi, length=20)
y_hist = sin.(x_hist)
p4 = histogram(y_hist, title="Гистограмма", xlabel="sin(x)", ylabel="Частота", legend=false)

p5 = plot(x, y, seriestype=:path, fill=(0, :lightblue), title="График с областями", xlabel="x", ylabel="sin(x)", legend=false)
p6 = plot(x, y, seriestype=:stick, title="График с палочками", xlabel="x", ylabel="sin(x)", legend=false)

Plots.plot(p1, p2, p3, p4, p5, p6, layout=(3,2), size=(1000, 700))
```

Рис. 3.11: Задание №1

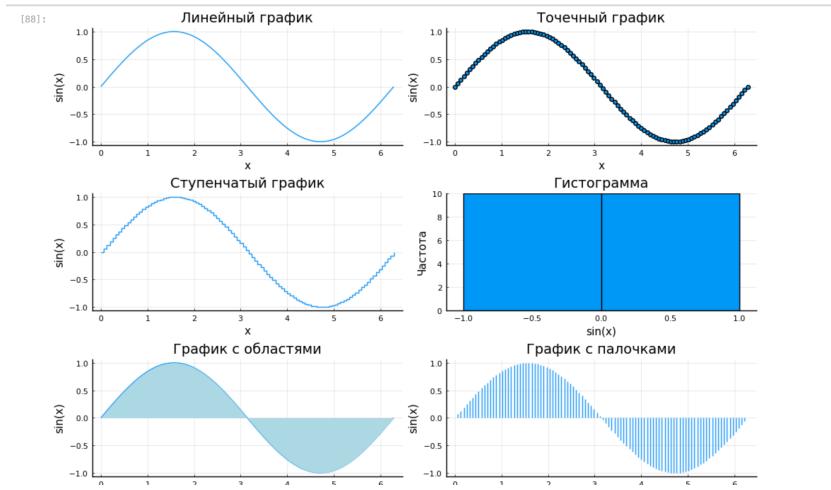


Рис. 3.12: Графики №1

3.2 Задание 2

Постройте графики функции

$$y = \sin(x)$$

$x = 0, 2\pi i$ со всеми возможными (сколько сможете вспомнить) типами оформления линий графика. Отобразите все графики в одном графическом окне (рис. 3.13 - 3.14).

```

2. Все возможные типы оформления линий

[93]: line_styles = [:solid, :dash, :dot, :dashdot, :auto]
line_colors = [blue, red, green, orange, purple]
line_widths = [1, 2, 3, 4, 5]
marker_shapes = [:circle, :rect, :diamond, :triangle, :star5];

[94]: plots_list = []
for i in 1:5
    p = plot(x, y,
              line.line_styles[i],
              color=line_colors[i],
              linewidth=line_widths[i],
              marker=marker_shapes[i],
              markersize=4,
              markercolor=marker_colors[i],
              title="Строка: $(line_styles[i])",
              xlabel="x",
              ylabel="sin(x)",
              legend=false)
    push!(plots_list, p)
end

```

Рис. 3.13: Задание №2

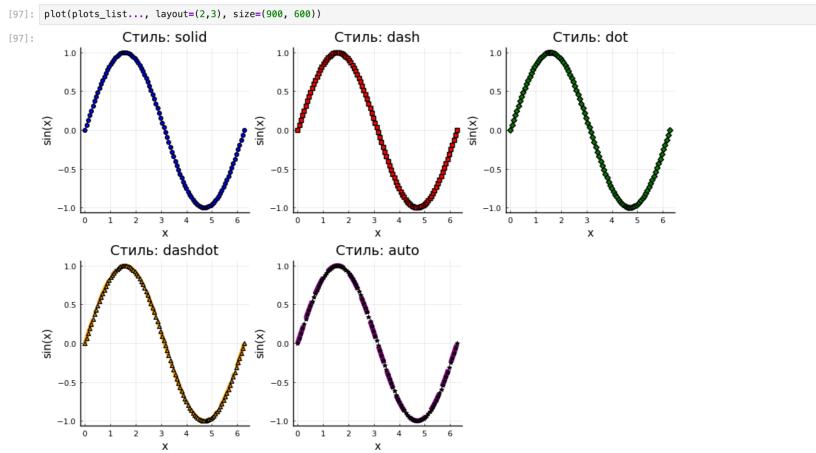


Рис. 3.14: Графики №2

3.3 Задание 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 (рис. 3.16).

3. Построить график функции $y(x) = \pi x^2 \ln(x)$. Пусть цвет рамки будет зеленым, 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.

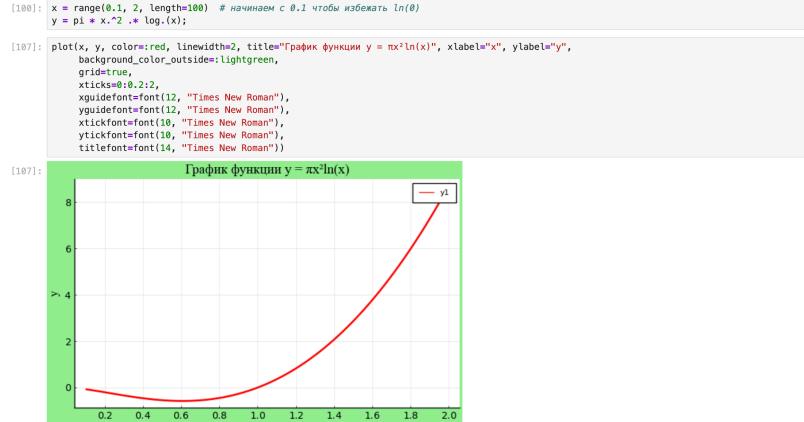


Рис. 3.15: Задание №3

3.4 Задание 4

Задайте вектор $x = (-2, -1, 0, 1, 2)$. В одном графическом окне (в 4-х подокнах).

Изобразите графически по точкам x значения функции

$$y(x) = x^3 - 3 * x$$

в виде: точек, линий, линий и точек, кривой. Сохраним полученные изображения в файле figure_Lantsova.png (рис. 3.16 - 3.17).

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



Рис. 3.16: Задание №4

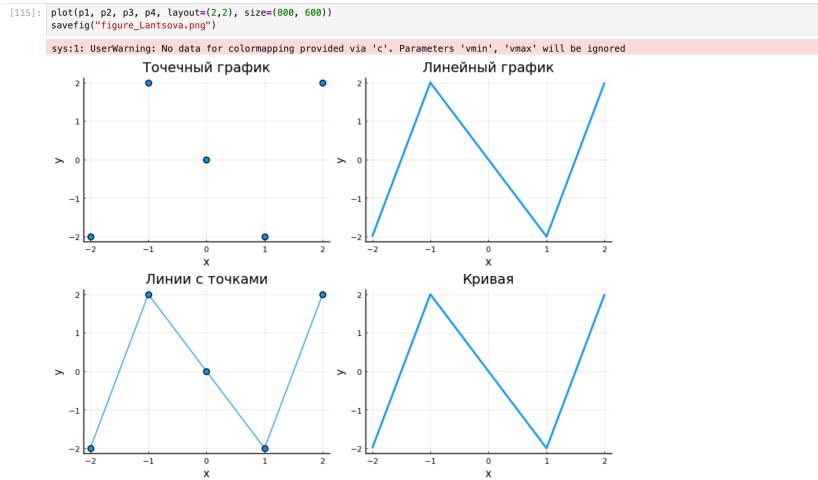


Рис. 3.17: Графики №4

3.5 Задание 5

Задайте вектор $x = (3, 3.1, 3.2, \dots, 6)$. Постройте графики функций

$$y_1(x) = pi * x$$

и

$$y_2(x) = exp(x) * cos(x)$$

в указанном диапазоне значений аргумента x следующим образом (рис. 3.18 - 3.19):

- постройте оба графика разного цвета на одном рисунке, добавьте легенду и сетку для каждого графика; укажите недостатки у данного построения;

5. Задайте вектор $x = (3, 3.1, 3.2, \dots, 6)$. Постройте графики функций 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.

```
[118]: x = range(3, 6, length=100)
y1 = pi * x
y2 = exp(x) * cos(x)

[126]: p1 = plot(x, y1, linewidth=2, color=blue)
p2 = plot(x, y2, linewidth=2, color:red)
plot(p1, p2, layout=(1,2), size=(600, 400), legend=true, grid=true)
```

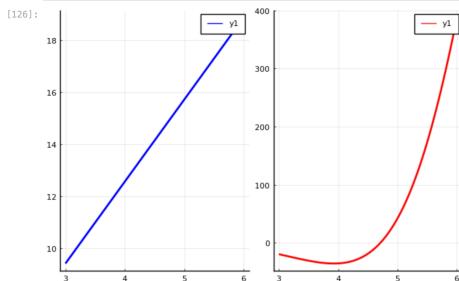
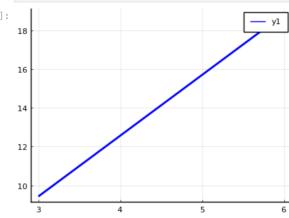


Рис. 3.18: Задание №5

- постройте аналогичный график с двумя осями ординат

```
[132]: plot(x, y1, linewidth=2, color=blue, size=(400, 300))
```



```
[133]: plot(x, y2, linewidth=2, color:red, size=(400, 300))
```



Рис. 3.19: Задание №5

3.6 Задание 6

Постройте график некоторых экспериментальных данных (придумайте сами), учитывая ошибку измерения (рис. 3.20).

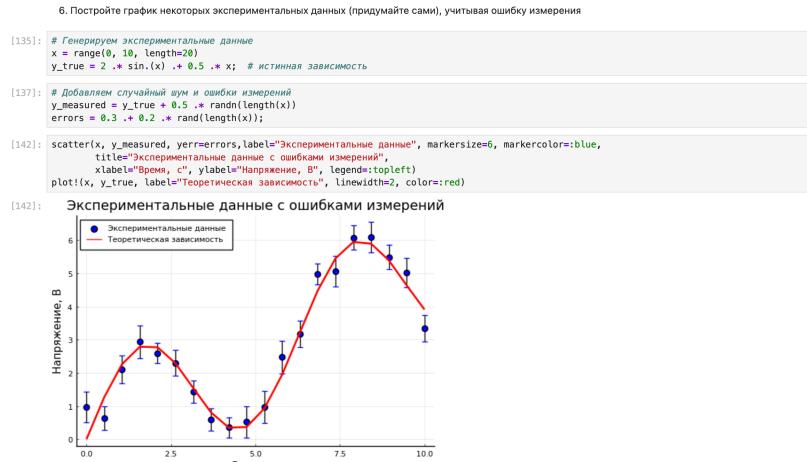


Рис. 3.20: Задание №6

3.7 Задание 7

Постройте точечный график случайных данных. Подпишите оси, легенду, название графика (рис. 3.21).

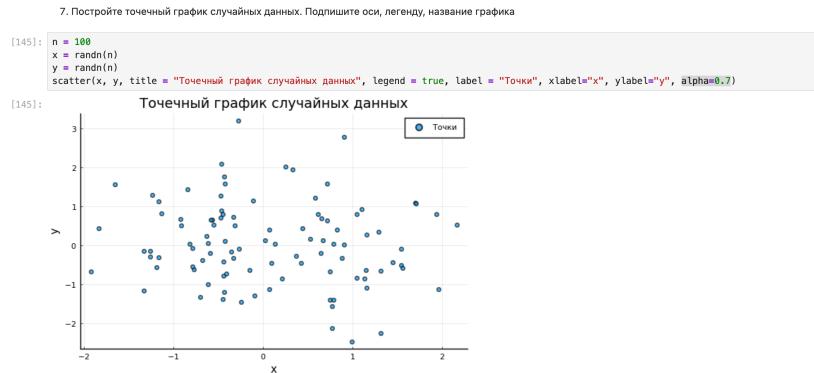


Рис. 3.21: Задание №7

3.8 Задание 8

Постройте 3-мерный точечный график случайных данных. Подпишите оси, легенду, название графика (рис. 3.22).



Рис. 3.22: Задание №8

3.9 Задание 9

Создайте анимацию с построением синусоиды. То есть вы строите последовательность графиков синусоиды, постепенно увеличивая значение аргумента. После соедините их в анимацию (рис. 3.23).

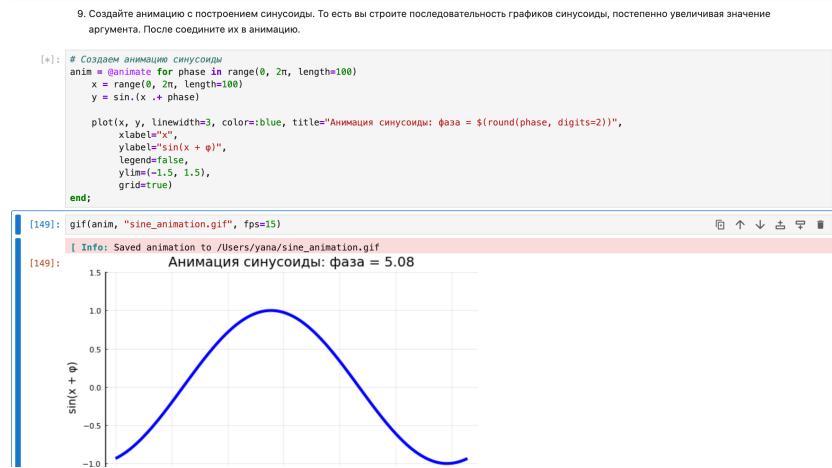


Рис. 3.23: Задание №9

3.10 Задание 10

Постройте анимированную гипоциклоиду для 2 целых значений модуля k и 2 рациональных значений модуля k (рис. 3.24 - 3.28).

```

10. Анимированная гипоциклоида

[=]: function plot_hypocycloid(r, k, n_frames=100)
    theta = range(0, 2pi, length=n_frames)
    anim = @animate for i in 1:n_frames
        t = theta[i]
        # Координаты гипоциклоиды
        x = r * (k-1) * cos(t) + r * cos((k-1) * t)
        y = r * (k-1) * sin(t) - r * sin((k-1) * t)
        # Большая окружность
        X_big = r + k * cos(theta)
        Y_big = r + k * sin(theta)
        plot(X_big, Y_big, color=:blue, linewidth=1, label="Большая окружность",
              aspect_ratio=:equal,
              xlim=(-r+k-1, r+k+1),
              ylim=(-r+k-1, r+k+1))
        plot!(x, y, color=:red, linewidth=2, label="Гипоциклоида (k=k)")
        title!("Гипоциклоида: k = $k")
    end
    return anim
end

# Целые значения k
anim1 = plot_hypocycloid(1, 3)
anim2 = plot_hypocycloid(1, 5)

# Рациональные значения k
anim3 = plot_hypocycloid(1, 2.5)
anim4 = plot_hypocycloid(1, 3.5)

```

Рис. 3.24: Задание №10

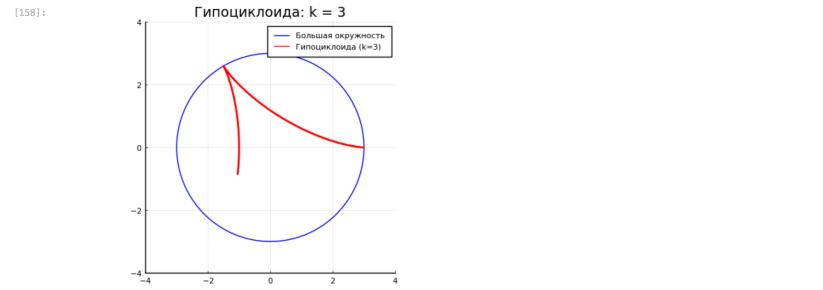


Рис. 3.25: Задание №10

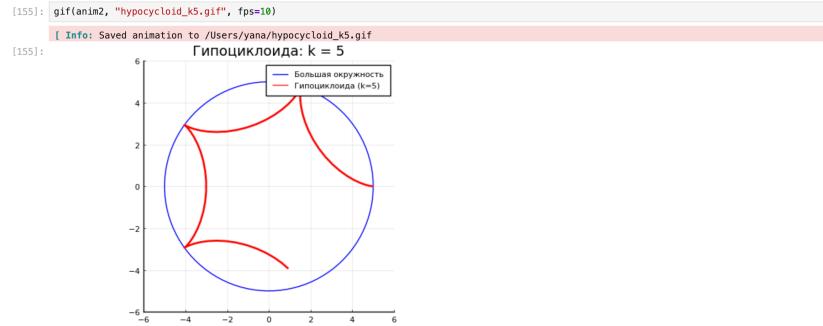


Рис. 3.26: Задание №10

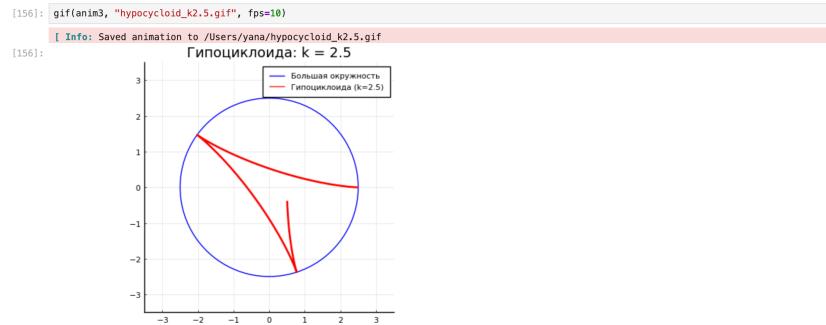


Рис. 3.27: Задание №10

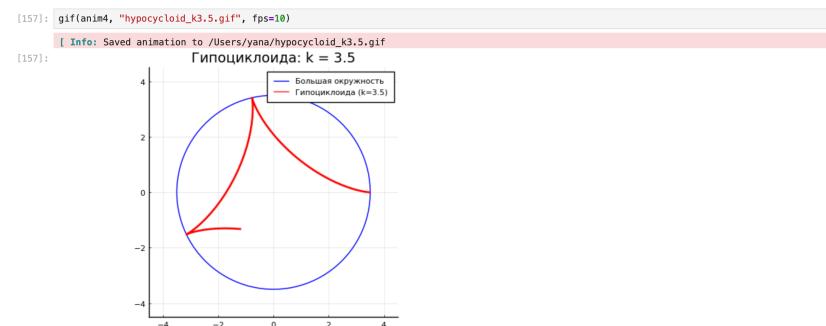


Рис. 3.28: Задание №10

3.11 Задание 11

Постройте анимированную эпициклоиду для 2 целых значений модуля k и 2 рациональных значений модуля k (рис. 3.29 - 3.33).

```

11. Анимированная эпициклоида

[4]: function plot_epicycloid(r, k, n_frames=100)
    theta = range(0, 2pi, length=n_frames)

    anim = @animate for i in 1:n_frames
        t = 0[i:i]

        # Координаты эпициклоиды
        x = r * (k+1) * cos.(t) - r * cos.((k+1) * t)
        y = r * (k+1) * sin.(t) - r * sin.((k+1) * t)

        # Большая окружность
        X_big = r * k * cos.(theta)
        Y_big = r * k * sin.(theta)

        plot(X_big, Y_big, color=:blue, linewidth=1, label="Неподвижная окружность", aspect_ratio=:equal,
              xlim=(-r*(k+2), r*(k+2)),
              ylim=(-r*(k+2), r*(k+2)))

        plot!(x, y, color=:green, linewidth=2, label="Эпициклоида (k=$k)")

        title!("Эпициклоида: k = $k")
    end

    return anim
end

# Использование k
anim1 = plot_epicycloid(1, 2)
anim2 = plot_epicycloid(1, 4)

# Рациональные значения k
anim3 = plot_epicycloid(1, 1.5)
anim4 = plot_epicycloid(1, 2.5)

```

Рис. 3.29: Задание №11

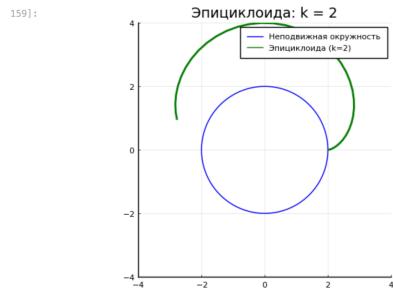


Рис. 3.30: Задание №11

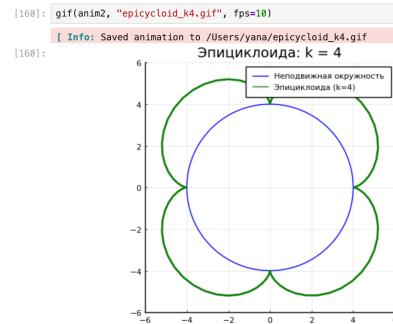


Рис. 3.31: Задание №11

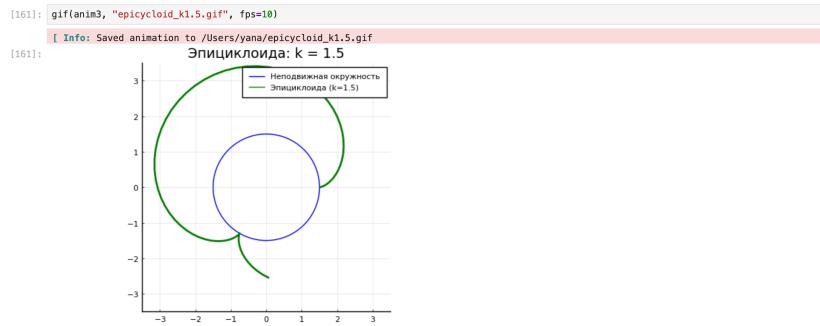


Рис. 3.32: Задание №11

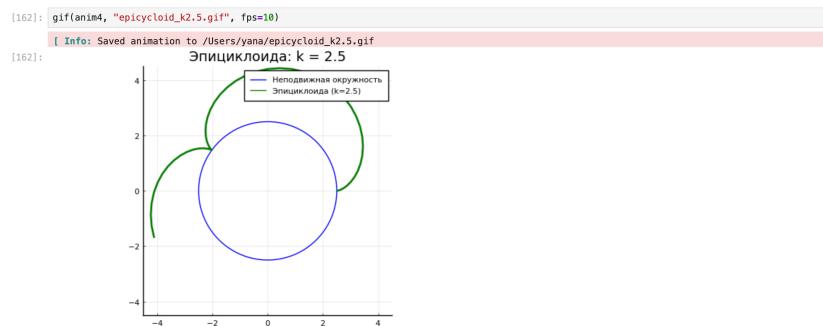


Рис. 3.33: Задание №11

4 Выводы

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