Презентация по лабораторной работе №5 по предмету Компьютерный практикум по статистическому анализу данных

СТУДЕНТ ГРУППЫ НФИБД-01-20 ЕВДОКИМОВ МАКСИМ МИХАЙЛОВИЧ (1032203019)

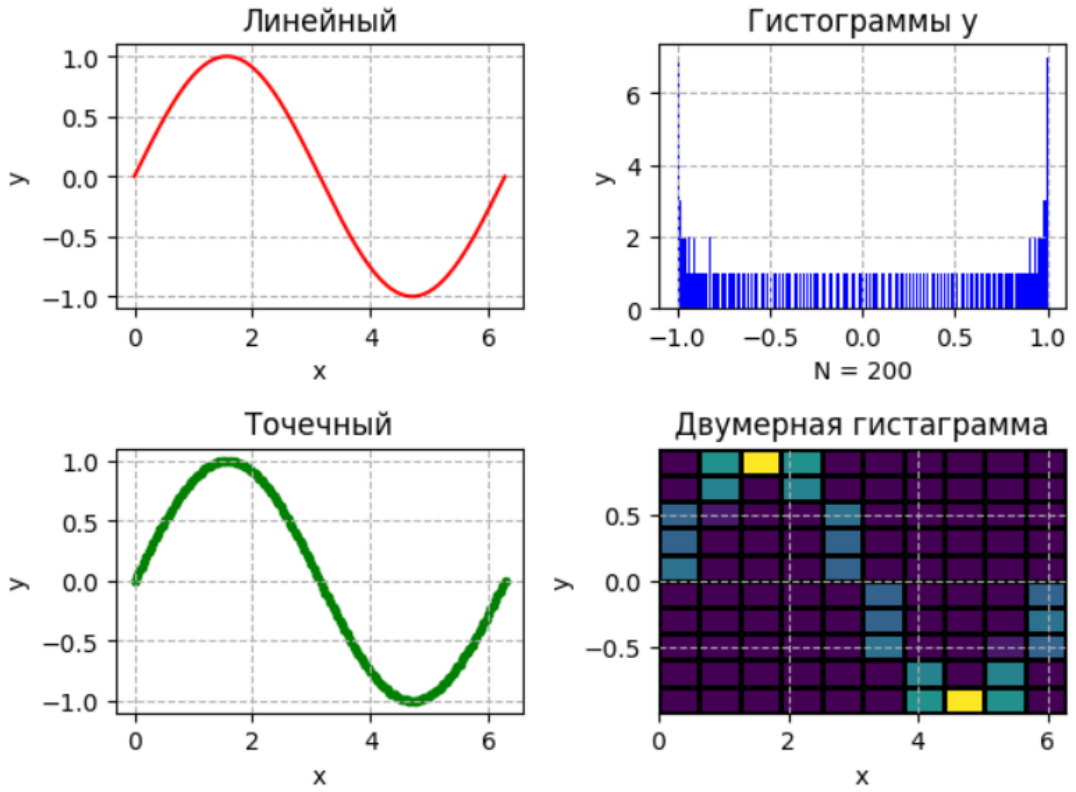
Цель работы

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

Задание 1

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

```julia
using PyCall
plt = pyimport("matplotlib.pyplot")
# 1
x = [i for i in 0:0.04:2*pi]
y = [sin(i) for i in x]
fig, axs = plt.subplots(ncols=2, nrows=2)
axs[1, 1].set_title("Линейный")
axs[1, 1].set_xlabel("x")
axs[1, 1].set_ylabel("y")
axs[1, 1].grid(visible=true, linestyle="--")
axs[1, 1].plot(x, y, color="r", linestyle="-")

axs[2, 1].set_title("Точечный")
axs[2, 1].set_xlabel("x")
axs[2, 1].set_ylabel("y")
axs[2, 1].grid(visible=true, linestyle="--")
axs[2, 1].scatter(x, y, color="g", marker=".")
n_bins = 200
axs[1, 2].set_title("Гистограммы y")
axs[1, 2].set_xlabel("N = $(n_bins)")
axs[1, 2].set_ylabel("y")
axs[1, 2].grid(visible=true, linestyle="--")
axs[1, 2].hist(y, bins=n_bins, color="b")

axs[2, 2].set_title("Двумерная гистаграмма")
axs[2, 2].set_xlabel("x")
axs[2, 2].set_ylabel("y")
axs[2, 2].grid(visible=true, linestyle="--")
axs[2, 2].hist2d(x, y, color="k")
fig.tight_layout()
```



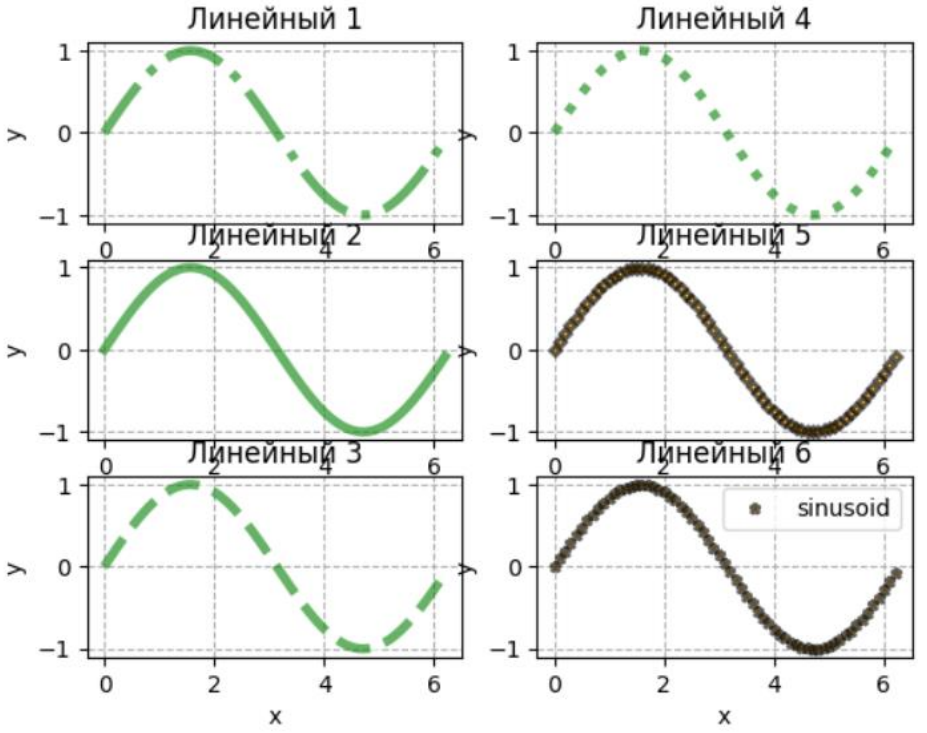
Задание 1: код и результат

Я решил данное задание используя matplotlib из Python, так как абсолютное большинство графических интерпретаторов на Julia используют в своей основе его (PyPlot, Plots, Plotly и UnicodePlots).

Задание 2

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

```
# 2
x = [i for i in 0:0.1:2*pi]
y = [sin(i) for i in x]
fig, axs = plt.subplots(ncols=2, nrows=3)
axs[1, 1].set_title("Линейный 1")
axs[1, 1].set_xlabel("x")
axs[1, 1].set_ylabel("y")
axs[1, 1].grid(visible=true, linestyle="--")
axs[1, 1].plot(x, y, color="g", marker="D",  markeredgecolor="k", linestyle="-.", linewidth=4, markersize=0, alpha=0.6,
               gapcolor="w", markeredgewidth=2, label="sinusoid", fillstyle="full", markerfacecolor="orange")

axs[2, 1].set_title("Линейный 2")
axs[2, 1].set_xlabel("x")
axs[2, 1].set_ylabel("y")
axs[2, 1].grid(visible=true, linestyle="--")
axs[2, 1].plot(x, y, color="g", marker="D",  markeredgecolor="k", linestyle="-", linewidth=4, markersize=0, alpha=0.6,
               gapcolor="w", markeredgewidth=2, label="sinusoid", fillstyle="full", markerfacecolor="orange")

axs[3, 1].set_title("Линейный 3")
axs[3, 1].set_xlabel("x")
axs[3, 1].set_ylabel("y")
axs[3, 1].grid(visible=true, linestyle="--")
axs[3, 1].plot(x, y, color="g", marker="D",  markeredgecolor="k", linestyle="--", linewidth=4, markersize=0, alpha=0.6,
               gapcolor="w", markeredgewidth=2, label="sinusoid", fillstyle="full", markerfacecolor="orange")

axs[1, 2].set_title("Линейный 4")
axs[1, 2].set_xlabel("x")
axs[1, 2].set_ylabel("y")
axs[1, 2].grid(visible=true, linestyle="--")
axs[1, 2].plot(x, y, color="g", marker="D",  markeredgecolor="k", linestyle=":", linewidth=4, markersize=0, alpha=0.6,
               gapcolor="w", markeredgewidth=2, label="sinusoid", fillstyle="full", markerfacecolor="orange")

axs[2, 2].set_title("Линейный 5")
axs[2, 2].set_xlabel("x")
axs[2, 2].set_ylabel("y")
axs[2, 2].grid(visible=true, linestyle="--")
axs[2, 2].plot(x, y, color="g", marker="D",  markeredgecolor="k", linestyle="", linewidth=4, markersize=3, alpha=0.6,
               gapcolor="w", markeredgewidth=2, label="sinusoid", fillstyle="full", markerfacecolor="orange")

axs[3, 2].set_title("Линейный 6")
axs[3, 2].set_xlabel("x")
axs[3, 2].set_ylabel("y")
axs[3, 2].grid(visible=true, linestyle="--")
axs[3, 2].plot(x, y, color="g", marker="*",  markeredgecolor="k", linestyle=" ", linewidth=4, markersize=5, alpha=0.6,
               gapcolor="w", markeredgewidth=2, label="sinusoid", fillstyle="full", markerfacecolor="orange")
fig.tight_layout()
plt.legend()
```

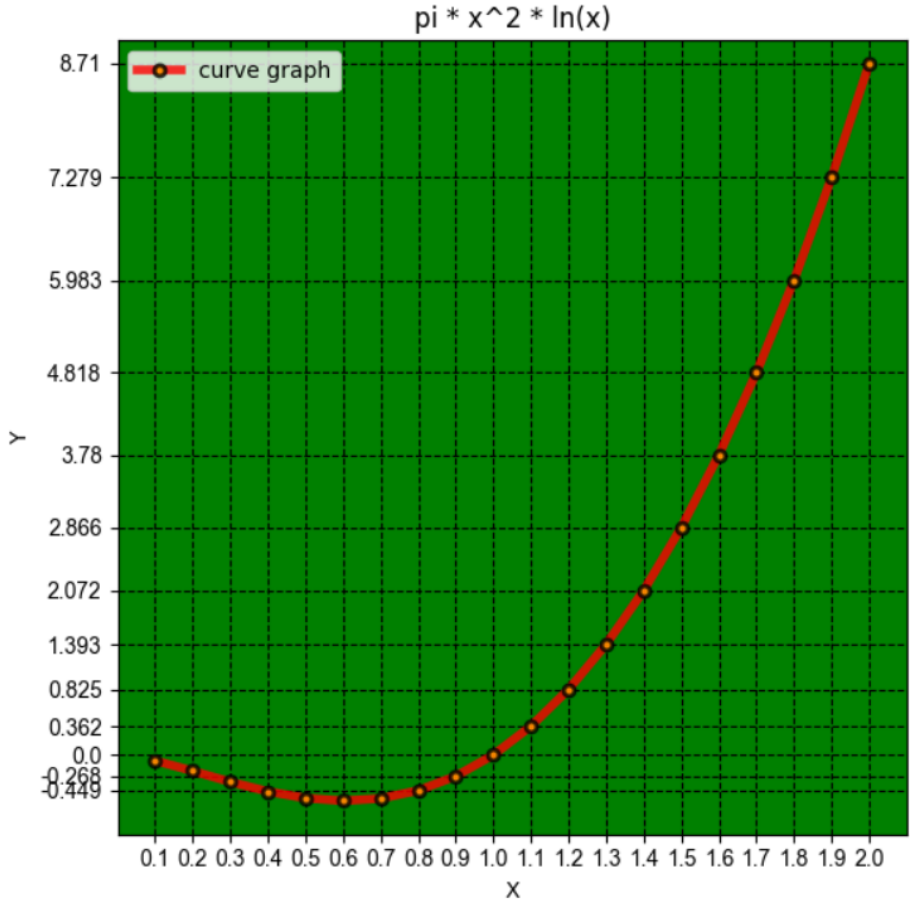


Задание 2: код и результат

Тот же принцип что и в первом на matplotlib.

Задание 3

- Постройте график функции y(x) = πx2*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.




```
# 3
e = 2.71828
x = [i for i in 0.1:0.1:2]
y = [pi*i^2*log(e,i) for i in x]
fig = plt.figure(figsize=[6, 6], tight_layout=true)
ax = fig.add_subplot(ylabel="Y", xlabel="X", xlim=[0, 2.1],ylim=[-1, 9])
ax.set_title("pi * x^2 * ln(x)")
ax.set_facecolor("g")
ax.grid(color="k", visible=true, linestyle="--")
ax.set_xticks(x, ["$i" for i in x])
ax.set_yticks(y[8:end], ["$(round(i, digits=3))" for i in y[8:end]])
ax.tick_params(axis="both", labelfontfamily="Arial")
ax.plot(x, y, color="r", marker="o",  markeredgecolor="k", linestyle="-", linewidth=4, markersize=5, alpha=0.8,
        gapcolor="w", markeredgewidth=2, label="curve graph", fillstyle="full", markerfacecolor="orange")
ax.legend()
```

Задание 3: код и результат

Описание.

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) = x3 − 3x в виде:

– точек,

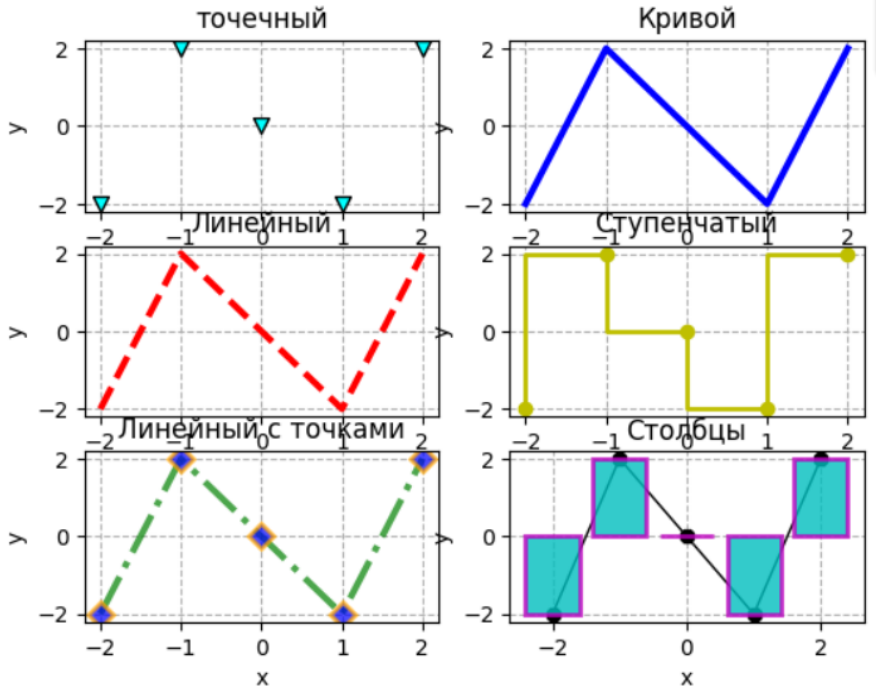
– линий,

– линий и точек,

– кривой..

- Сохраните полученные изображения в файле figure_familiya.png, где вместо familiya укажите вашу фамилию.

```julia
# 4
x = Vector{Float64}([-2, -1, 0, 1, 2])
y = [i^3 - 3*i for i in x]

fig, axs = plt.subplots(ncols=2, nrows=3)
axs[1, 1].set(title="точечный", xlabel="x", ylabel="y")
axs[1, 1].grid(visible=true, linestyle="--")
axs[1, 1].scatter(x, y, c="cyan", marker="v", s=50, edgecolors="k", zorder=2)

axs[2, 1].set(title="Линейный", xlabel="x", ylabel="y")
axs[2, 1].grid(visible=true, linestyle="--")
axs[2, 1].plot(x, y, color="r", marker="None", linestyle="--", linewidth=3)

axs[3, 1].set(title="Линейный с точками", xlabel="x", ylabel="y")
axs[3, 1].grid(visible=true, linestyle="--")
axs[3, 1].plot(x, y, color="g", marker="D",  markeredgecolor="orange", linestyle="-.", linewidth=3, markersize=8, alpha=0.7,
               markeredgewidth=2, fillstyle="full", markerfacecolor="b")

axs[1, 2].set(title="Кривой", xlabel="x", ylabel="y")
axs[1, 2].grid(visible=true, linestyle="--")
axs[1, 2].plot(x, y, color="b", marker="None", linestyle="-", linewidth=3)

axs[2, 2].set(title="Ступенчатый", xlabel="x", ylabel="y")
axs[2, 2].grid(visible=true, linestyle="--")
axs[2, 2].step(x, y, "o-y", linewidth=2)

axs[3, 2].set(title="Столбцы", xlabel="x", ylabel="y")
axs[3, 2].grid(visible=true, linestyle="--")
axs[3, 2].plot(x, y, color="k", marker="8", linestyle="-", linewidth=1)
axs[3, 2].bar(x, y, color="c", edgecolor="m", linewidth=2, alpha=0.8, zorder=2)
fig.tight_layout()
plt.savefig("data/figure_Evdokimov.png")
```



Задание 4: код и результат

Описание.

Задание 5

- Задайте вектор x = (3, 3.1, 3.2, …, 6). Постройте графики функций y1(x) = πx и y2(x) = exp(x)*cos(x) в указанном диапазоне значений аргумента x следующим образом:

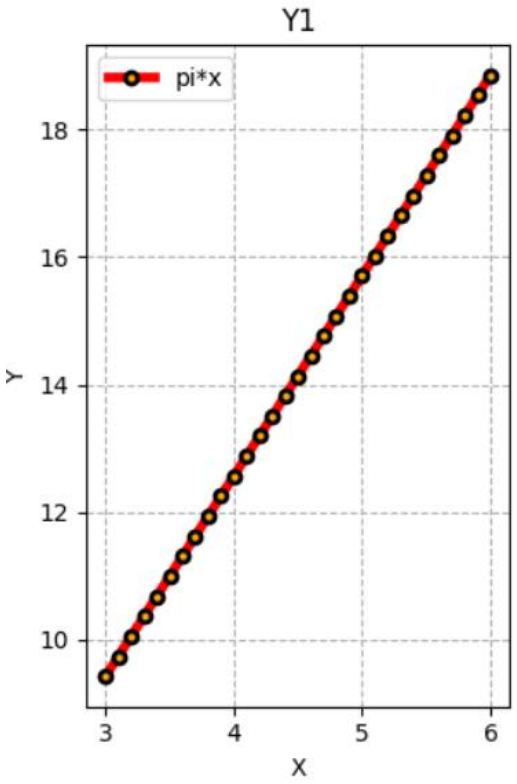
– постройте оба графика разного цвета на одном рисунке, добавьте легенду и сетку для каждого графика; укажите недостатки у данного построения;

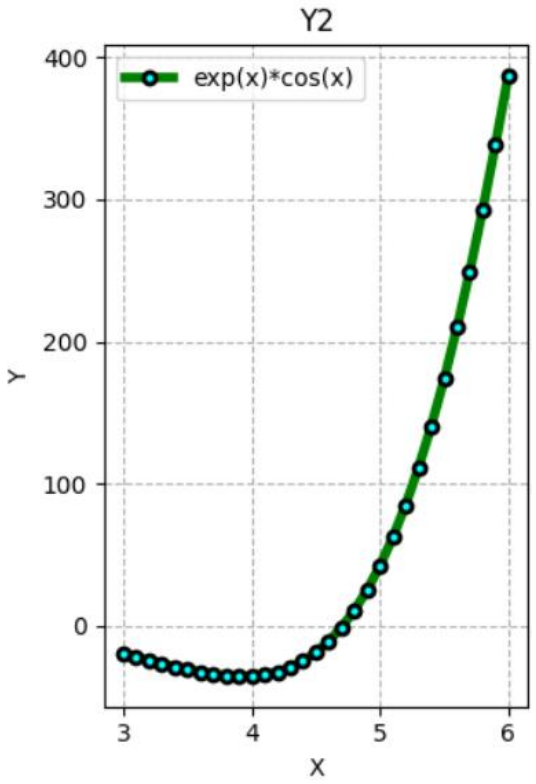
– постройте аналогичный график с двумя осями ординат.

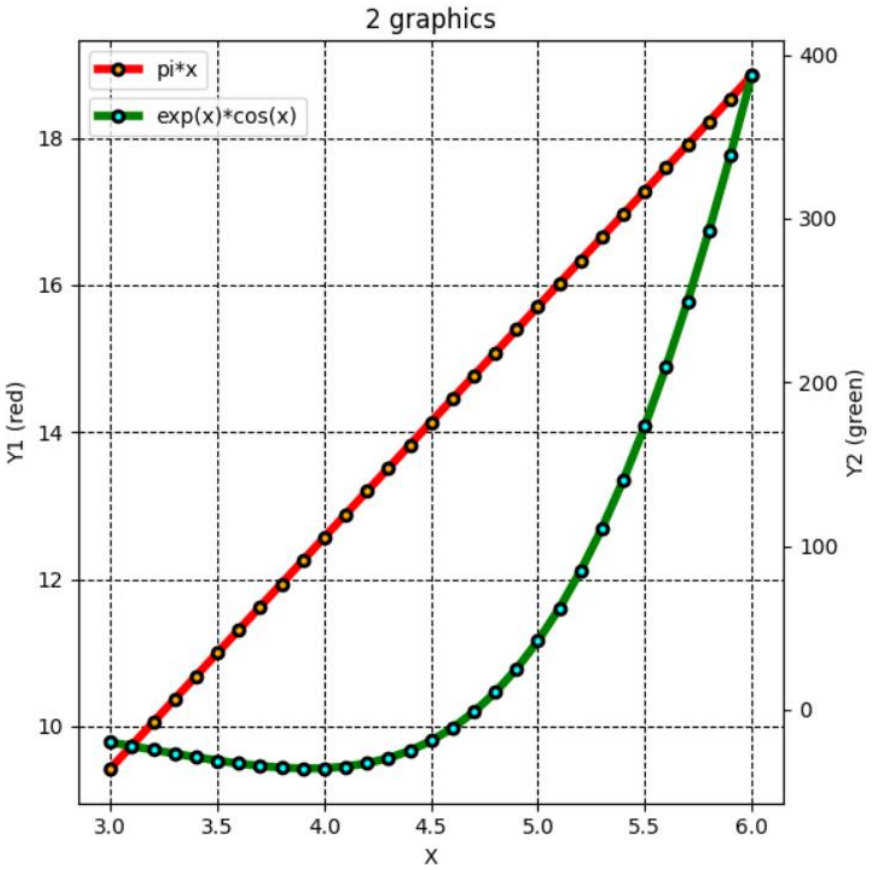
```
# 5
x = [i for i in 3:0.1:6]
y1, y2 = [pi*i for i in x], [exp(i)*cos(i) for i in x]

fig, axs = plt.subplots(ncols=2, nrows=1)
axs[1, 1].set(title="Y1", xlabel="X", ylabel="Y")
axs[1, 1].grid(visible=true, linestyle="--")
axs[1, 1].plot(x, y1, color="r", marker="o",  markeredgecolor="k", linestyle="-", linewidth=4, markersize=5,
        markeredgewidth=2, label="pi*x", fillstyle="full", markerfacecolor="orange")
axs[1, 1].legend()

axs[2, 1].set(title="Y2", xlabel="X", ylabel="Y")
axs[2, 1].grid(visible=true, linestyle="--")
axs[2, 1].plot(x, y2, color="g", marker="o",  markeredgecolor="k", linestyle="-", linewidth=4, markersize=5,
        markeredgewidth=2, label="exp(x)*cos(x)", fillstyle="full", markerfacecolor="cyan")
axs[2, 1].legend()
plt.show()

fig = plt.figure(figsize=[6, 6], tight_layout=true)
ax = fig.add_subplot(ylabel="Y1 (red)", xlabel="X", title="2 graphics")
ax.grid(color="k", visible=true, linestyle="--")
ax.plot(x, y1, color="r", marker="o",  markeredgecolor="k", linestyle="-", linewidth=4, markersize=5,
        markeredgewidth=2, label="pi*x", fillstyle="full", markerfacecolor="orange")
ax2 = ax.twinx()
ax2.set_ylabel("Y2 (green)")
ax2.plot(x, y2, color="g", marker="o",  markeredgecolor="k", linestyle="-", linewidth=4, markersize=5,
        markeredgewidth=2, label="exp(x)*cos(x)", fillstyle="full", markerfacecolor="cyan")

ax.legend(loc=2)
ax2.legend(loc=6, bbox_to_anchor=(0.0, 0.9))
```

Задание 5: код

Описание.







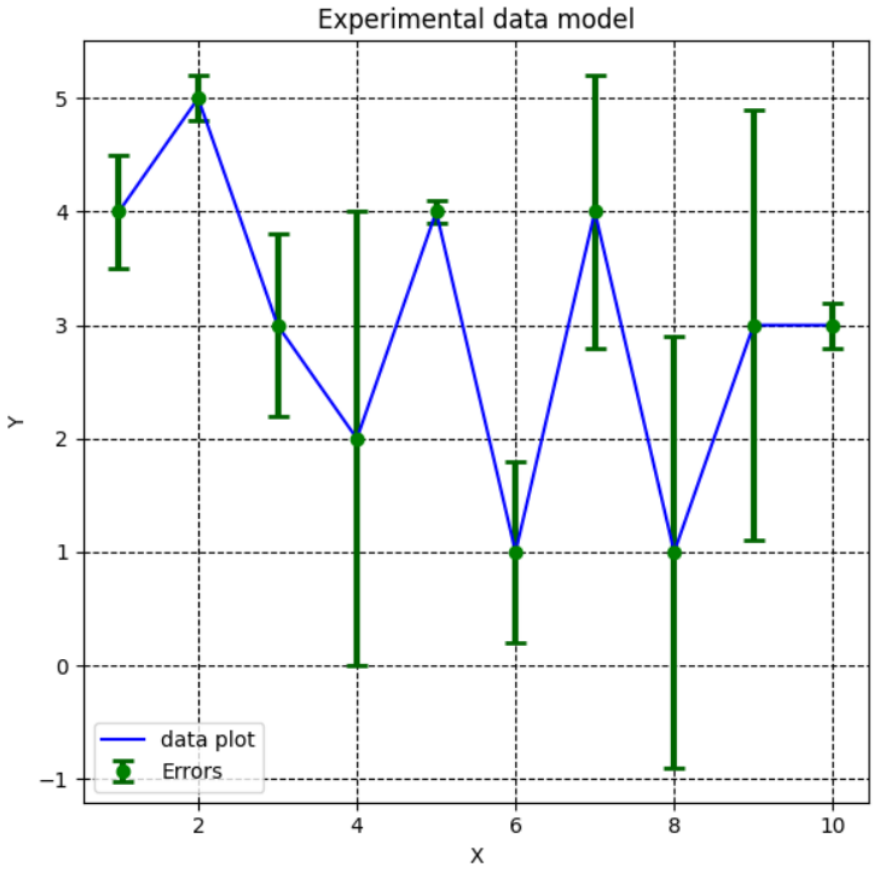
Задание 5: результат

Описание.

Задание 6

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

```
# 6
x, y, error = Vector{Float64}(1:10), rand(1:5, 10), rand(0.1:0.1:2, 10)
fig = plt.figure(figsize=[6, 6], tight_layout=true)
ax = fig.add_subplot(ylabel="Y", xlabel="X", title="Experimental data model")
ax.grid(color="k", visible=true, linestyle="--")
ax.errorbar(x, y, yerr=error, color="g", fmt="o", ecolor="darkgreen", elinewidth=3,
            capthick=2, capsize=5, label="Errors")
ax.plot(x, y, color="b", label="data plot")
fig.tight_layout()
ax.legend()
```



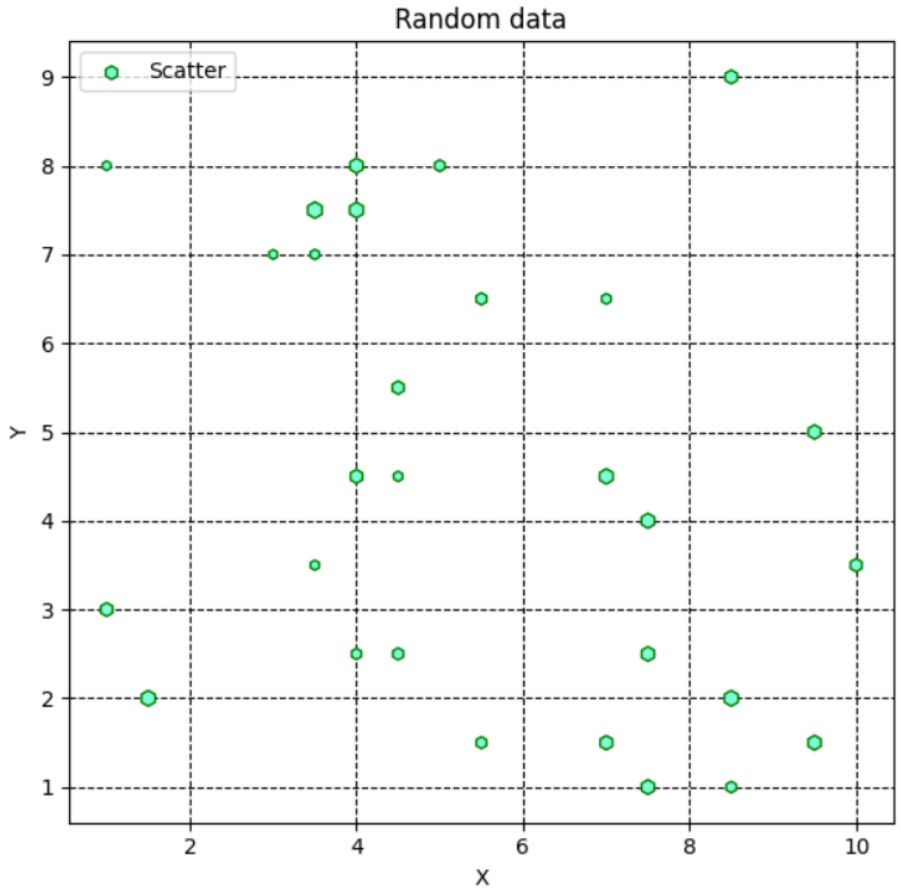
Задание 6: код и результат

Описание.

Задание 7

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

```
# 7
x, y, sz = rand(1:0.5:10, 30), rand(1:0.5:10, 30), rand(20:60, 30)
fig = plt.figure(figsize=[6, 6], tight_layout=true)
ax = fig.add_subplot(ylabel="Y", xlabel="X", title="Random data")
ax.grid(color="k", visible=true, linestyle="--")
ax.scatter(x, y, s=sz, c="aquamarine", marker="h", label="Scatter", edgecolors="g", zorder=2)
fig.tight_layout()
ax.legend()
```

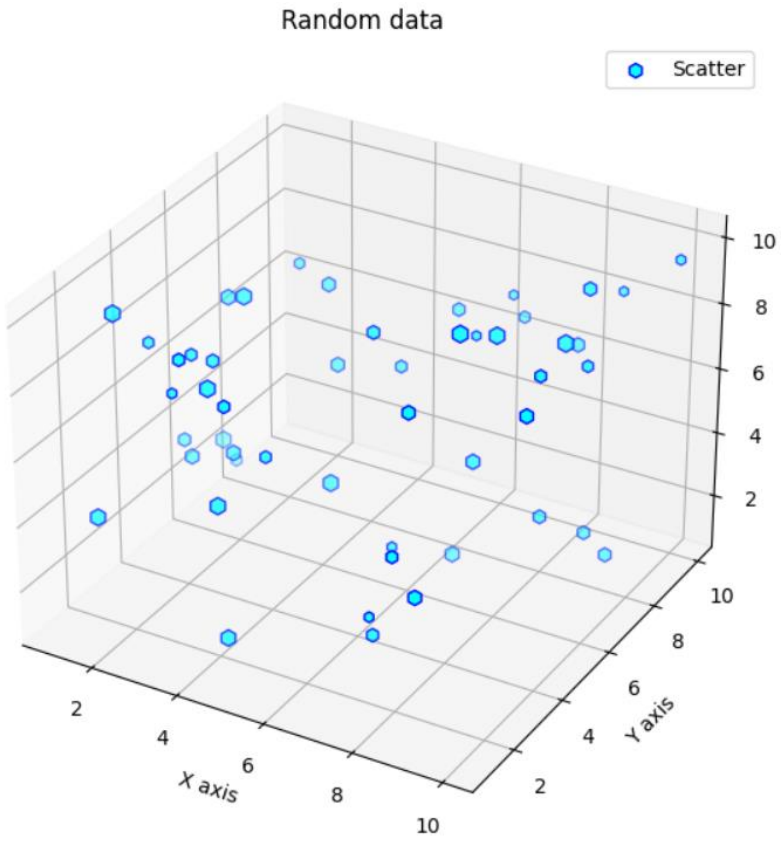


Задание 7: код и результат

Описание.

Задание 8

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




```
# 8
n = 50
x, y, z, sz = rand(1:0.5:10, n), rand(1:0.5:10, n), rand(1:0.5:10, n), rand(25:75, n)
fig = plt.figure(figsize=[6, 6], tight_layout=true)
ax = fig.add_subplot(projection="3d", ylabel="Y axis", xlabel="X axis", zlabel="Z axis", title="Random data")
ax.grid(color="gray", visible=true, linestyle="--")
ax.scatter(x, y, z, s=sz, c="cyan", marker="h", label="Scatter", edgecolors="b", zorder=2)
fig.tight_layout()
ax.legend()
plt.show()
```

Задание 8: код и результат

Описание.

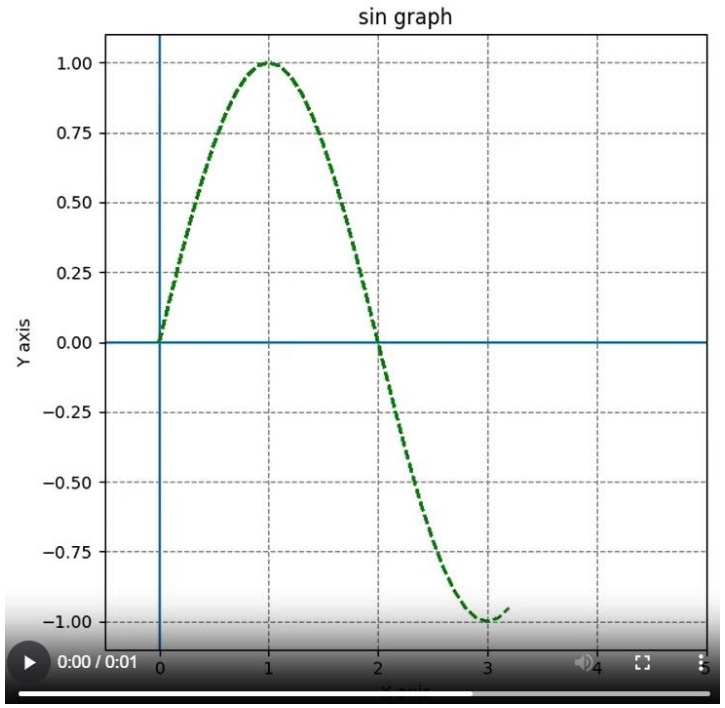
Задание 9

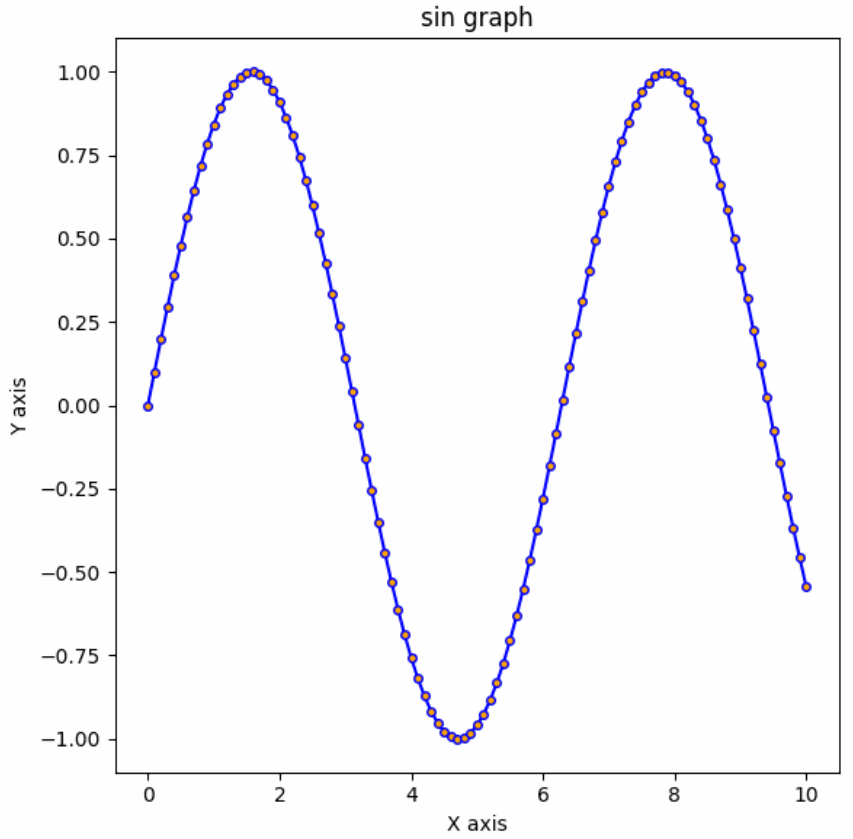
- Создайте анимацию с построением синусоиды. То есть вы строите последовательность графиков синусоиды, постепенно увеличивая значение аргумента. После соедините их в анимацию.

```
# 9v1
animation = pyimport("matplotlib.animation")
using Base64

x = Vector{Float64}(0:0.1:10)
y = [sin(i*pi/2) for i in x]

fig = plt.figure(figsize=[6, 6], tight_layout=true)
ax = fig.add_subplot(ylabel="Y axis", xlabel="X axis", title="sin graph", xlim=[-0.5, 5], ylim=[-1.1, 1.1])
ax.grid(color="gray", visible=true, linestyle="--")
ax.axhline(0); ax.axvline(0)

function showanim(filename)
    base64_video = base64encode(open(filename))
    display("text/html", """<video controls src="data:video/x-m4v;base64,$base64_video">""")
end

function update(frame)
    pl = ax.plot(x[1:frame], y[1:frame], color="green", linestyle="--")
    return pl
end

anima = animation.FuncAnimation(fig=fig, func=update, frames=50, interval=30)
anima[:save]("data/graph_9v1.mp4", bitrate=-1, extra_args=["-vcodec", "libx264", "-pix_fmt", "yuv420p"])
showanim("data/graph_9v1.mp4")
```

Задание 9: код

Описание.





Задание 9: результат

Описание.

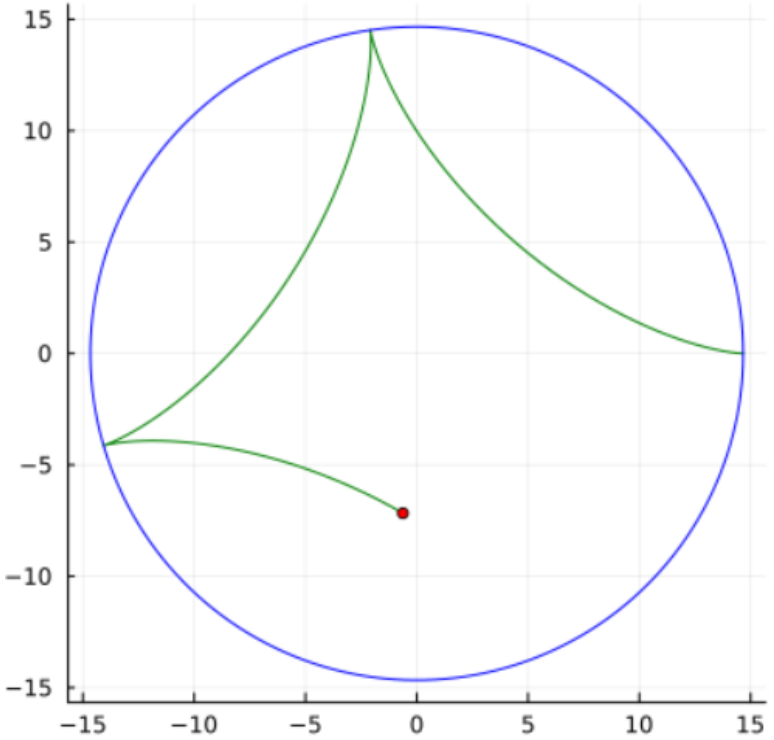
Задание 10

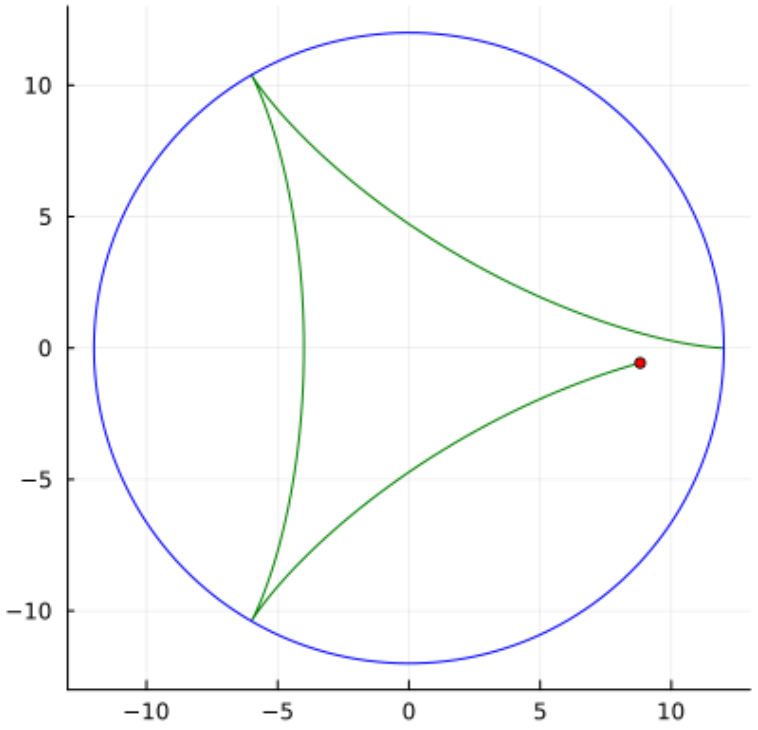
- Постройте анимированную гипоциклоиду для 2 целых значений модуля k и 2 рациональных значений модуля k.

```julia
# 10
using Plots
using Statistics

function hypocycloid(k)
    step, r = pi/100, 4, 1
    t_range, R = 0:step:2*pi, r*k+1
    z = exp.(range(0, 2π, 65)im)
    x(t) = r*(k-1)*cos.(t) + r*cos.((k-1)*t)
    y(t) = r*(k-1)*sin.(t) - r*sin.((k-1)*t)
    x_vals = Float64[]
    y_vals = Float64[]
    for t in t_range
        push!(x_vals, x(t))
        push!(y_vals, y(t))
    end
    cx, cy = zeros(200), zeros(200)
    t = LinRange(0, 2*pi, 200)
    cx = cx .+ k*r*cos.(t)
    cy = cy .+ k*r*sin.(t)

    anim = @animate for i in 1:length(t_range)
        plot(x_vals[1:i], y_vals[1:i], c=:green, xlims=(-R, R), ylims=(-R, R), aspect_ratio=:equal, legend=false)
        plot!(cx, cy, c=:blue)
        scatter!([x_vals[i]], [y_vals[i]], markersize=3, markershape=:circle, markercolor=:red)
    end
    gif(anim, "data//hypocycloid$(round(k, digits=5)).gif")
end

hypocycloid(3)
hypocycloid(5)
hypocycloid(4.222)
hypocycloid(11//3)
```

Задание 10: код

Описание.





Задание 10: результат

Слева k=Float, справа k=Integer.

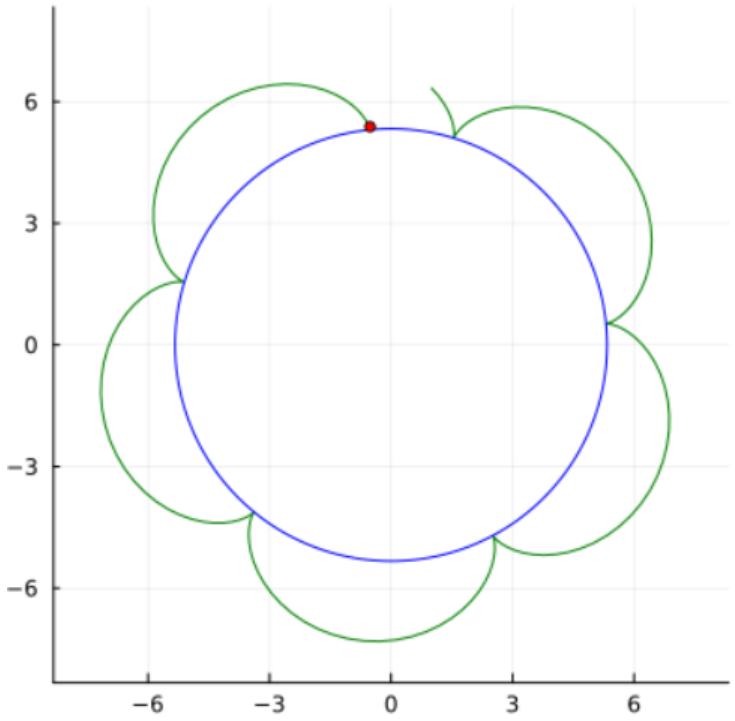
Задание 11

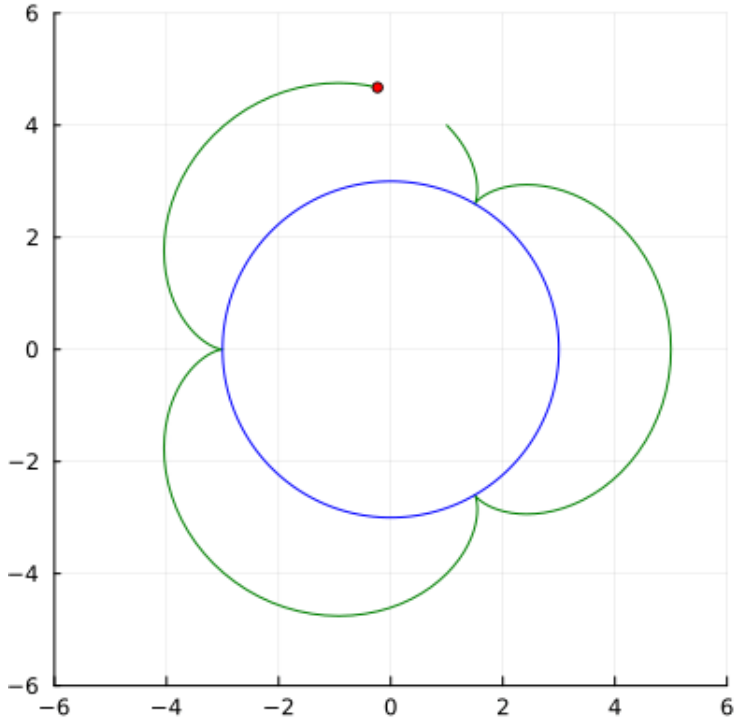
- Постройте анимированную эпициклоиду для 2 целых значений модуля k и 2 рациональных значений модуля k.

```julia
# 11
using Plots
using Statistics

function epicycloid(k)
    step, r = pi/100, 1
    t_range, R = 0:step:2*pi, r*k+1
    z = exp.(range(0, 2π, 65)im)
    x(t) = r*(k-1)*sin.(t) + r*cos.((k-1)*t)
    y(t) = r*(k-1)*cos.(t) - r*sin.((k-1)*t)
    x_vals = Float64[]
    y_vals = Float64[]
    for t in t_range
        push!(x_vals, x(t))
        push!(y_vals, y(t))
    end
    cx, cy = zeros(200), zeros(200)
    t = LinRange(0, 2*pi, 200)
    cx = cx .+ (k-2*r)*cos.(t)
    cy = cy .+ (k-2*r)*sin.(t)

    anim = @animate for i in 1:length(t_range)
        plot(x_vals[1:i], y_vals[1:i], c=:green, xlims=(-R, R), ylims=(-R, R), aspect_ratio=:equal, legend=false)
        plot!(cx, cy, c=:blue)
        scatter!([x_vals[i]], [y_vals[i]], markersize=3, markershape=:circle, markercolor=:red)
    end
    gif(anim, "data//epicycloid$(round(k, digits=5)).gif")
end

epicycloid(5)
epicycloid(7)
epicycloid(6.222)
epicycloid(22//3)
```

Задание 11: код

Описание.





Задание 11: результат

Слева k=Float, справа k=Integer.

Заключение

- В ходе выполнения лабораторной работы были изучены основные методы графического вывода разных типов графиков с использованием Plots и matplotlib), также создания анимации и сохранения построенных моделей.