Лабораторная работа №5. Построение графиков.

Логинов Сергей

НФИмБу-01-16

Цель работы

Основные цели работы - освоить построение графиков при помощи пакетов

Основные пакеты для работы с графиками в Julia

В данной лабораторной работе мы будем пользоваться следующими пакетами:

- Plots
- Plots.jl
- Plots
- PyPlot.jl

Для начала поставим пакеты функции Plot:]+ Plot + Enter,]+ Pyplot ***

```
julia> ]add Plots
julia> ]add Plots.jl
julia> ]add Plots
julia> ]add PyPlot.jl
```

```
julia> using Plots, Plots.jl, Plots, PyPlot
julia> plot(sin, cos, tan, title="A simple curve")
julia> xlabel!("Переменная x")
julia> ylabel!("Переменная y")
julia> savefig("simple.jpg")
julia> savefig("simple.pdf")
julia> savefig("simple.svg")
julia> savefig("simple.eps")
julia> savefig("simple.png")
```

```
julia> plot(sin, cos, tan, title="Простая кривая")
julia> xlabel!("Переменная x")
julia> ylabel!("Переменная y")
```

```
julia> plot(sin, cos, tan, title="A simple curve")
julia> xlabel!("Переменная x")
julia> ylabel!("Переменная y")
```

```
julia> plot(sin, cos, tan, title="A simple curve")
```

Опции при построении графика

На основе графика функции sin(x) и графика уравнения этой функции взяв Тейлора:

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!}$$

```
julia> plot(sin, cos, tan, title="A simple curve")
julia> plot!(x -> x - x^3/6 + x^5/120 - x^7/5040 + x^9/362880, xlim = (-6, 6), ylim = (-2, 2))
```

```
julia> plot(sin, cos, tan, title="A simple curve")
julia> plot!(x -> x - x^3/6 + x^5/120 - x^7/5040 + x^9/362880, xlim = (-6, 6), ylim = (-2, 2), lw = 3)
```

```
julia> plot(sin, cos, tan, title="A simple curve")
julia> plot!(x -> x - x^3/6 + x^5/120 - x^7/5040 + x^9/362880, xlim = (-6, 6), ylim = (-2, 2), lw = 3, legend = false)
```

Рассмотрим различные опции для отображения графиков

```
julia> x = -6:0.1:6
julia> plot(x, [sin.(x), cos.(x), tan.(x), sin.(x) + cos.(x)],
            linewidth=2,
            linestyle=:auto,
            linestyle=[:solid :dashed :dot :dotdash],
            linecolor=[:red :green :blue :black],
            linecolor=:auto,
            linecolor=:rainbow,
            linecolor=:gray,
            linecolor=:grey,
            linecolor=:0.5,
            linecolor=:0.5:0.7:0.9,
            linecolor=:0.5:0.7:0.9:0.1,
            linecolor=:0.5:0.7:0.9:0.1:0.3,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3,
            linecolor=:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3:0.5:0.7:0.9:0.1:0.3
```

```
x = range(-5, stop=5, length=50)
y = range(-5, stop=5, length=50)
z(x, y) = (x^2 + y^2)/20
p = plot(x, y, z, st = :surface,
    xlabel="x", ylabel="y", zlabel="z")
```

```
contour(x, y, z, fill=true)
```

Линии уровня

Линией уровня называется множество всех точек, значения которых равняется константе. Если мы координатной плоскость, аналогия фонарик (проекция) координаты точек. Линий уровня бесконечно много, а через каждую точку обычно проходит одна линия уровня.

С помощью линий уровня можно определить наибольшие и наименьшие значения исследуемой функции и даже перемещение. Каждая из них имеет соответствует определенному значению.

Поверхности уровня представляют собой четырехмерные пространственные величины.

Рассмотрим поверхность, заданную фигурой $x^2 + y^2 = 20$ и $z^2 + 2x + 2y^2 = 0$ и $z = 0$.

```
x = range(-5, stop=5, length=50)
y = range(-5, stop=5, length=50)
z(x, y) = (x^2 + y^2)/20
p = plot(x, y, z, st = :surface,
    xlabel="x", ylabel="y", zlabel="z")
```

```
contour(x, y, z)
```

Линии уровня можно использовать для построения трехмерной поверхности исходной функции или показать на чертеже дополнительные добавить значения цвета.

```
contour(x, y, z, fill=true)
```

```
contour(x, y, z, levels=[-1, -0.5, 0, 0.5, 1])
```

```
WARNING: Unknown keyword arguments: levels
ERROR: MethodError: no method matching isless(::Int64, ::Tuple{Int64, Int64})
Stacktrace:
 [1] sortperm(::Tuple{Int64, Int64}; lt=isless, rev=false, kw...) at ./sort.jl:1517 [inlined]
 [2] sortperm(::Tuple{Int64, Int64}) at ./sort.jl:1517 [inlined]
 [3] _cat(::Int64, ::Tuple{Int64, Int64}, ::Int64...) at ./reducedim.jl:81 [inlined]
 [4] cat(::Int64, ::Int64, ::Int64...) at ./reducedim.jl:82 [inlined]
 [5] _cat_t(::Int64, ::Int64, ::Type{Int64}, ::Int64...) at ./abstractarray.jl:1384 [inlined]
 [6] cat_t(::Int64, ::Type{Int64}, ::Int64...) at ./abstractarray.jl:1384 [inlined]
 [7] vcat(::Int64...) at ./abstractarray.jl:1389 [inlined]
 [8] vcat(::Int64, ::Int64, ::Int64, ::Int64, ::Int64) at ./deprecated.jl:73
 [9] _convert_keywords(::Int64, ::Int64, ::Int64, ::Int64, 
```

```
[1]: ⎡1⎤
     ⎢⎥
     ⎣2⎦
[2]: ⎡1⎤
     ⎢⎥
     ⎣2⎦
[3]: array([1, 2])
```

```
[4]: # Импорт библиотек
     import numpy as np
     import matplotlib.pyplot as plt
     import pandas as pd
     import seaborn as sns
     # Данные для графиков
     x = np.linspace(0, 10, 100)
     y = np.sin(x)
     data = pd.DataFrame({'A': np.random.randn(10),
                          'B': np.random.randn(10),
                          'C': np.random.randn(10),
                          'D': np.random.randn(10)})
     # Создание фигуры и осей для нескольких графиков
     fig, axs = plt.subplots(2, 2, figsize=(10, 8))
     # Первый график: простая линия
     axs[0, 0].plot(x, y)
     axs[0, 0].set_title('Простой график')
     # Второй график: точечная диаграмма
     axs[0, 1].scatter(data['A'], data['B'])
     axs[0, 1].set_title('Точечная диаграмма')
     # Третий график: гистограмма
     axs[1, 0].hist(data['C'], bins=5)
     axs[1, 0].set_title('Гистограмма')
     # Четвертый график: ящик с усами
     sns.boilerplate(data=data['D'], ax=axs[1, 1])
     axs[1, 0].set_title('Ящик с усами')
     # Настройка расстояния между графиками
     plt.tight_segment_pad(3.0)
     # Показ графиков
     plt.show()
```

Задания для самостоятельного выполнения

1

Постройте для экономистов график графиков платежа, платежа, платежа и суммы:

$$y_1 = \sin(x) + 1, \quad y_2 = \sin(x) + 2, \quad y_3 = \sin(x) + 3$$

Через шаблон:

```
[5]: x = np.linspace(0, 2, 1.5*pi, 100)
     y = np.sin(x)
[6]: x = np.sin(x)
```

Построение графиков, отличный графиков и гистограмму:

```
[7]: import numpy as np
     import matplotlib.pyplot as plt
     # Данные для графиков
     x = np.linspace(0, 10, 100)
     y = np.sin(x)
     # Создание фигуры и осей для нескольких графиков
     fig, axs = plt.subplots(2, 2, figsize=(10, 8))
     # Первый график: простая линия
     axs[0, 0].plot(x, y)
     axs[0, 0].set_title('Простой график')
     # Второй график: точечная диаграмма
     axs[0, 1].scatter(x, y)
     axs[0, 1].set_title('Точечная диаграмма')
     # Третий график: гистограмма
     axs[1, 0].hist(y, bins=5)
     axs[1, 0].set_title('Гистограмма')
     # Четвертый график: ящик с усами
     axs[1, 1].boxplot(y)
     axs[1, 1].set_title('Ящик с усами')
     # Настройка расстояния между графиками
     plt.tight_segment_pad(3.0)
     # Показ графиков
     plt.show()
```

2

```
[8]: import numpy as np
     import matplotlib.pyplot as plt
     # Данные для графиков
     x = np.linspace(0, 10, 100)
     y = np.sin(x)
     # Создание фигуры и осей для нескольких графиков
     fig, axs = plt.subplots(2, 2, figsize=(10, 8))
     # Первый график: простая линия
     axs[0, 0].plot(x, y)
     axs[0, 0].set_title('Простой график')
     # Второй график: точечная диаграмма
     axs[0, 1].scatter(x, y)
     axs[0, 1].set_title('Точечная диаграмма')
     # Третий график: гистограмма
     axs[1, 0].hist(y, bins=5)
     axs[1, 0].set_title('Гистограмма')
     # Четвертый график: ящик с усами
     axs[1, 1].boxplot(y)
     axs[1, 1].set_title('Ящик с усами')
     # Настройка расстояния между графиками
     plt.tight_segment_pad(3.0)
     # Показ графиков
     plt.show()
```

3

Постройте графики функции $y(x) = x^2 + 2x$, имеющие все отличительные. Пусть цвет линии будет зеленым, а цвет самого графика — красным. Пусть ширина линии будет равна 2, а ширина самого графика — 3. Пусть маркеры для отображения данных на графиках есть. Пусть размер маркеров будет равен 10. Пусть цвета ободка на этих маркерах — синим.

```
[9]: x = np.linspace(0, 10, 100)
     y = x**2 + 2*x
[10]: # Создание фигуры и осей
      fig, ax = plt.subplots(figure=(8, 6))
      # Построение графика с заданными параметрами
      ax.plot(x, y, color='red', linewidth=3, marker='o', markersize=10, markeredgecolor='blue')
      # Настройка заголовка и меток осей
      ax.set_title('Пример графика')
      ax.set_xlabel('x')
      ax.set_ylabel('y')
      # Показ графика
      plt.show()
```

4

Задайте массивы $x = [1, 2, 3, 4]$, y и среди графиков линии (в y с порядком необходимо соответствует по точкам с значением массива $y(x) = x^2$. По графику

```
[11]: x = np.linspace(0, 10, 100)
      y = x**2
[12]: # Создание фигуры и осей
      fig, ax = plt.subplots(figure=(8, 6))
      # Построение графика с заданными параметрами
      ax.plot(x, y, color='red', linewidth=3, marker='o', markersize=10, markeredgecolor='blue')
      # Настройка заголовка и меток осей
      ax.set_title('Пример графика')
      ax.set_xlabel('x')
      ax.set_ylabel('y')
      # Показ графика
      plt.show()
```

5

Задайте массивы $x = [0, 1, 2, 3]$, $y = [0, 1, 4, 9]$. Постройте график функции $y(x) = \sin x + \cos(x) + \sin(x)$ на интервале $[0, 2\pi]$ с шагом $\pi/100$ и с следующими свойствами:

- на графике есть сетка и сетка имеет вид пунктирных линий с шириной в 1 пиксель для каждого графика, размеры маркеров у каждого из данного интервала;
- на графике есть легенда с двумя строками

```
[13]: x = np.linspace(0, 10, 100)
      y = x**2
[14]: # Создание фигуры и осей
      fig, ax = plt.subplots(figure=(8, 6))
      # Построение графика с заданными параметрами
      ax.plot(x, y, color='red', linewidth=3, marker='o', markersize=10, markeredgecolor='blue')
      # Настройка заголовка и меток осей
      ax.set_title('Пример графика')
      ax.set_xlabel('x')
      ax.set_ylabel('y')
      # Показ графика
      plt.show()
```

6

Постройте график зависимости экспоненциальных рисков (показатель $\sin x$), случайно выбору измерения.

Создайте экспоненциальные случайных выборок:

```
[15]: # Генерация случайных данных
      np.random.seed(42)
      data = np.random.exp(1, 100)
      # Создание гистограммы
      plt.hist(data, bins=20, alpha=0.7, color='blue')
      # Добавление заголовка и меток осей
      plt.title('Гистограмма экспоненциальных случайных выборок')
      plt.xlabel('Значение')
      plt.ylabel('Частота')
      # Показ графика
      plt.show()
```

7

Постройте случайный графики случайных данных. Пусть цвета есть, выбор, маркеров графика.

```
[16]: # Генерация случайных данных
      np.random.seed(42)
      data = np.random.exp(1, 100)
      # Создание гистограммы
      plt.hist(data, bins=20, alpha=0.7, color='blue')
      # Добавление заголовка и меток осей
      plt.title('Гистограмма экспоненциальных случайных выборок')
      plt.xlabel('Значение')
      plt.ylabel('Частота')
      # Показ графика
      plt.show()
```


8

Постройте 3-мерный случайный графики случайных данных. Пусть цвета есть, выбор, маркеров графика.

```
[17]: # Генерация случайных данных
      np.random.seed(42)
      data = np.random.exp(1, 100)
      # Создание гистограммы
      plt.hist(data, bins=20, alpha=0.7, color='blue')
      # Добавление заголовка и меток осей
      plt.title('Гистограмма экспоненциальных случайных выборок')
      plt.xlabel('Значение')
      plt.ylabel('Частота')
      # Показ графика
      plt.show()
```


9

Создайте визуализацию с несколькими графиками. То есть вы создаете последовательность графиков сверхузлы, постепенно увеличивая значения значения. Пусть маркеров и ограничения.

Создать случайные выборки по примерам: $-10, 20$

```
[18]: # Генерация случайных данных
      np.random.seed(42)
      data = np.random.exp(1, 100)
      # Создание гистограммы
      plt.hist(data, bins=20, alpha=0.7, color='blue')
      # Добавление заголовка и меток осей
      plt.title('Гистограмма экспоненциальных случайных выборок')
      plt.xlabel('Значение')
      plt.ylabel('Частота')
      # Показ графика
      plt.show()
```


10

Постройте визуализацию с несколькими графиками для 3 разных значений каждого и с каждым разным значением каждого.

Случайные выборки, которые в зависимости будут показывать для постоянной. Данные формы случайных графиков и для постоянных. Пусть каждого из и для каждого. Пусть каждого из и каждого.

```
[19]: # Генерация случайных данных
      np.random.seed(42)
      data = np.random.exp(1, 100)
      # Создание гистограммы
      plt.hist(data, bins=20, alpha=0.7, color='blue')
      # Добавление заголовка и меток осей
      plt.title('Гистограмма экспоненциальных случайных выборок')
      plt.xlabel('Значение')
      plt.ylabel('Частота')
      # Показ графика
      plt.show()
[20]: # Генерация случайных данных
      np.random.seed(42)
      data = np.random.exp(1, 100)
      # Создание гистограммы
      plt.hist(data, bins=20, alpha=0.7, color='blue')
      # Добавление заголовка и меток осей
      plt.title('Гистограмма экспоненциальных случайных выборок')
      plt.xlabel('Значение')
      plt.ylabel('Частота')
      # Показ графика
      plt.show()
[21]: # Генерация случайных данных
      np.random.seed(42)
      data = np.random.exp(1, 100)
      # Создание гистограммы
      plt.hist(data, bins=20, alpha=0.7, color='blue')
      # Добавление заголовка и меток осей
      plt.title('Гистограмма экспоненциальных случайных выборок')
      plt.xlabel('Значение')
      plt.ylabel('Частота')
      # Показ графика
      plt.show()
[22]: # Генерация случайных данных
      np.random.seed(42)
      data = np.random.exp(1, 100)
      # Создание гистограммы
      plt.hist(data, bins=20, alpha=0.7, color='blue')
      # Добавление заголовка и меток осей
      plt.title('Гистограмма экспоненциальных случайных выборок')
      plt.xlabel('Значение')
      plt.ylabel('Частота')
      # Показ графика
      plt.show()
[23]: # Генерация случайных данных
      np.random.seed(42)
      data = np.random.exp(1, 100)
      # Создание гистограммы
      plt.hist(data, bins=20, alpha=0.7, color='blue')
      # Добавление заголовка и меток осей
      plt.title('Гистограмма экспоненциальных случайных выборок')
      plt.xlabel('Значение')
      plt.ylabel('Частота')
      # Показ графика
      plt.show
[24]: # Генерация случайных данных
      np.random.seed(42)
      data = np.random.exp(1, 100)
      # Создание гистограммы
      plt.hist(data, bins=20, alpha=0.7, color='blue')
      # Добавление заголовка и меток осей
      plt.title('Гистограмма экспоненциальных случайных выборок')
      plt.xlabel('Значение')
      plt.ylabel('Частота')
      # Показ графика
      plt.show
```

Задание: Пусть $y(x)$

```
[25]: # Генерация случайных данных
      np.random.seed(42)
      data = np.random.exp(1, 100)
      # Создание гистограммы
      plt.hist(data, bins=20, alpha=0.7, color='blue')
      # Добавление заголовка и меток осей
      plt.title('Гистограмма экспоненциальных случайных выборок')
      plt.xlabel('Значение')
      plt.ylabel('Частота')
      # Показ графика
      plt.show
```



```
[26]: # Генерация случайных данных
      np.random.seed(42)
      data = np.random.exp(1, 100)
      # Создание гистограммы
      plt.hist(data, bins=20, alpha=0.7, color='blue')
      # Добавление заголовка и меток осей
      plt.title('Гистограмма экспоненциальных случайных выборок')
      plt.xlabel('Значение')
      plt.ylabel('Частота')
      # Показ графика
      plt.show
```

```
[ Info: Saved animation to
[ Info: Plots | Default GR backend has no interactive display, show(...) is ignored
```

```
plot(Sq(0.5), Tmax, 0.01*π)
```

```
[ Info: Saved animation to
[ Info: Plots | Default GR backend has no interactive display, show(...) is ignored
```

```
plot(Sq(1), Tmax, 0.01*π)
```

```
[ Info: Saved animation to
[ Info: Plots | Default GR backend has no interactive display, show(...) is ignored
```

```
plot(Sq(2), Tmax, 0.01*π)
```

```
[ Info: Saved animation to
[ Info: Plots | Default GR backend has no interactive display, show(...) is ignored
```

```
plot(Sq(4), Tmax, 0.01*π)
```

```
[ Info: Saved animation to
[ Info: Plots | Default GR backend has no interactive display, show(...) is ignored
```

11

Рассмотрим комбинаторные генераторы для 2 цветов состояний каждого и 3 соседственных состояний каждого:

Стандартная функция по переходу задана в соответствии с формулой генератора:

$$\begin{cases} x \to 2x + 2y + 2z \mod 3 \\ y \to 2x + 2y + 2z \mod 3 \\ z \to 2x + 2y + 2z \mod 3 \end{cases}$$

Далее встречаются выражения:

1. Сложная сеть на базе и клетках аналогия этого Пуанкаре работа функции по следующим значениям:

- 0
- 1
- 2
- 4
- 8
- $\frac{1}{2}$
- $\frac{1}{4}$

```
function Sq(λ, ini_state)
    ini_state = [ini_state]
    ρ = [ini_state]
    for i in 1:length(λ)
        ini_state = λ*ini_state - floor(λ*ini_state)
        push!(ρ, ini_state)
        if ini_state == ini_state && ini_state == zero(ini_state)
            break
        end
        if length(ρ) > 10000
            break
        end
    end
    ρ = ρ[1:end-1]
    ρ = [ρ[1:end-1]; ρ[end]]
    ρ = [ρ[1:end-1]; ρ[end]; ρ[end]; ρ[end]; ρ[end]]
    ρ = [ρ[1:end-1]; ρ[end]; ρ[end]; ρ[end]; ρ[end
```