РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ № 5

дисциплина: Компьютерный практикум по математическому моделированию

Студент: Саинт-Амур Измаэль

Группа: НПИбд-01-20

МОСКВА

2023 г.

Постановка задачи

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

Выполнение работы

5.2. Предварительные сведения

5.2.1. Основные пакеты для работы с графиками в Julia

Julia поддерживает несколько пакетов для работы с графиками. Использование того или иного пакета зависит от целей, преследуемых пользователем при построении.

Стандартным для Julia является пакет Plots.jl.

Перед использованием графических пакетов следует их установить и подключить

в Julia:

```
using Pkg
Pkg.add("Plots")
Pkg.add("PyPlot")
Pkg.add("Plotly")
Pkg.add("UnicodePlots")
# подключаем для использования Plots:
using Plots.
```

In [1]:
```
using Pkg
```

In [2]:
```
Pkg.add("Plots")
```

```
 Updating registry at `~/.julia/registries/General.toml`
Resolving package versions...
Installed libfdk_aac_jll ──────────────── v2.0.2+0
Installed JpegTurbo_jll ──────────────── v3.0.1+0
Installed Libmount_jll ──────────────── v2.35.0+0
Installed x265_jll ──────────────── v3.5.0+0
Installed LERC_jll ──────────────── v3.0.0+1
Installed GR_jll ──────────────── v0.72.10+0
Installed Opus_jll ──────────────── v1.3.2+0
Installed LoggingExtras ──────────────── v1.0.3
                                  jll ──────────────── v1.4.6+0
Installed ConcurrentUtilities ──────────── v2.3.0
Installed Unitful ──────────────── v1.19.0
Installed Measures ──────────────── v0.3.2
Installed RelocatableFolders ──────────── v1.0.1
Installed Grisu ──────────────── v1.0.2
Installed Xorg_xcb_util_wm_jll ──────────── v0.4.1+1
Installed Formatting ──────────────── v0.4.2
Installed Contour ──────────────── v0.6.2
```

click to unscroll output; double click to hide

In [3]:
```
Pkg.add("Plots")
```

In [4]:
```
Pkg.add("Plotly")
```

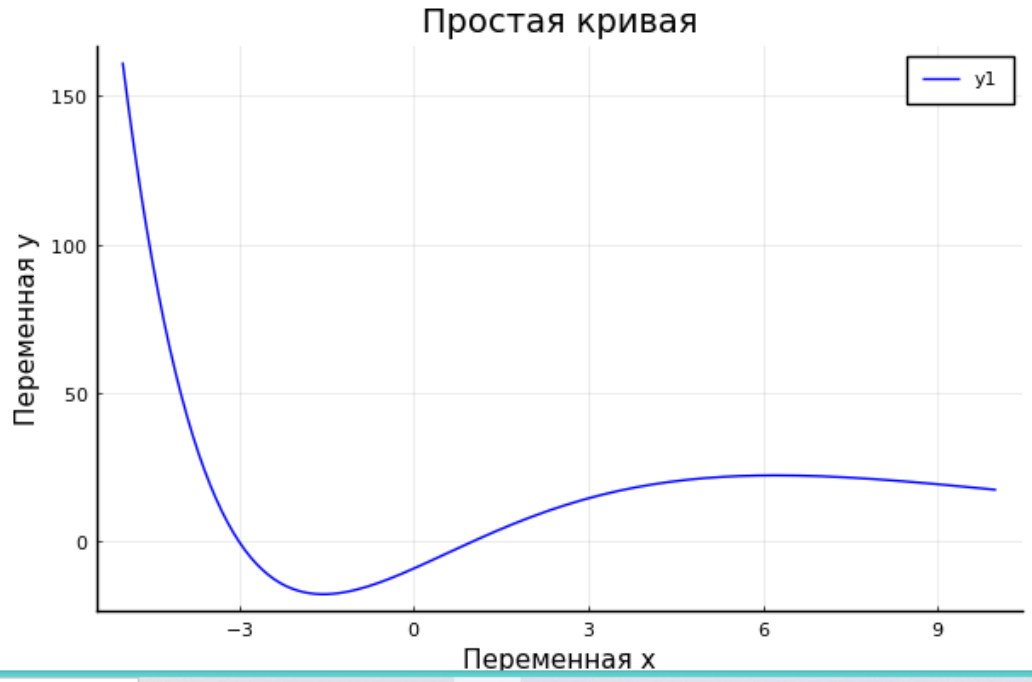
```
Resolving package versions...
Installed URIParser ──────────────── v0.4.1
Installed AssetRegistry ──────────────── v0.1.0
Installed PlotlyBase ──────────────── v0.8.19
Installed Pidfile ──────────────── v1.3.0
Installed BinDeps ──────────────── v1.0.2
Installed HTTP ──────────────── v0.9.17
Installed Hiccup ──────────────── v0.2.2
Installed Kaleido_jll ──────────────── v0.2.1+0
Installed WebSockets ──────────────── v1.5.9
Installed Lazy ──────────────── v0.15.1
Installed Blink ──────────────── v0.12.5
Installed TableTraits ──────────────── v1.0.1
Installed DataValueInterfaces ──────────── v1.0.0
Installed Plotly ──────────────── v0.4.1
Installed UnPack ──────────────── v1.0.2
Installed Mustache ──────────────── v1.0.19
Installed JSExpr ──────────────── v0.5.4
Installed FunctionalCollections ──────── v0.5.0
```

In [5]:
```
Pkg.add("UnicodePlots")
```

```
Resolving package versions...
```

```
# (название кривой, подписи по осям, цвет графика):
plot(x,y,
title="Простая кривая",
xlabel="Переменная x",
ylabel="Переменная y",
color="blue")
```

18]:



5.2.2. Опции при построении графика

рассмотрим дополнительные возможности пакетов для работы с графикой.

Используем pyplot():

указывается, что для построения графика используется pyplot():

pyplot()

Задаём функцию:

задание функции sin(x):

sin_theor(x) = sin(x)

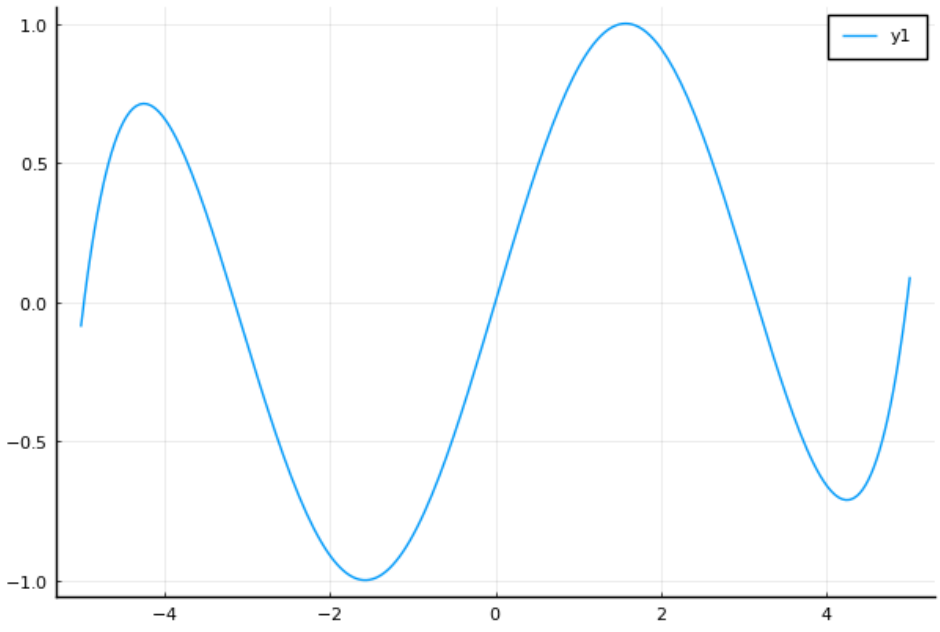
Строим график функции (рис. 5.3):

построение графика функции sin(x):

plot(sin_theor)

```
In [22]: # построение графика функции sin_taylor(x):
         plot(sin_taylor)
```

Out[22]:



5.2.3. Точечный график

Графики в виде точек на плоскости или в пространстве часто используются в статисти□ческих исследованиях.

5.2.3.1. Простой точечный график

Как и построении обычного графика для точечного графика необходимо задать массив

значений x, посчитать или задать значения y, задать опции построения графика:

параметры распределения точек на плоскости:

x = range(1,10,length=10)

y = rand(10)

параметры построения графика:

plot(x, y,

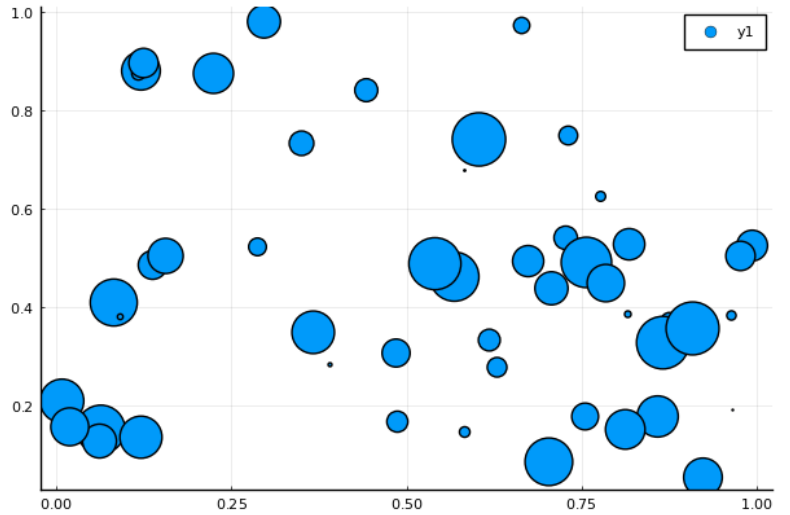
seriestype = :scatter,

title = "Точечный график"

)

```
In [31]: # параметры построения графика:
         scatter(x, y, markersize=ms)
```

Out[31]:

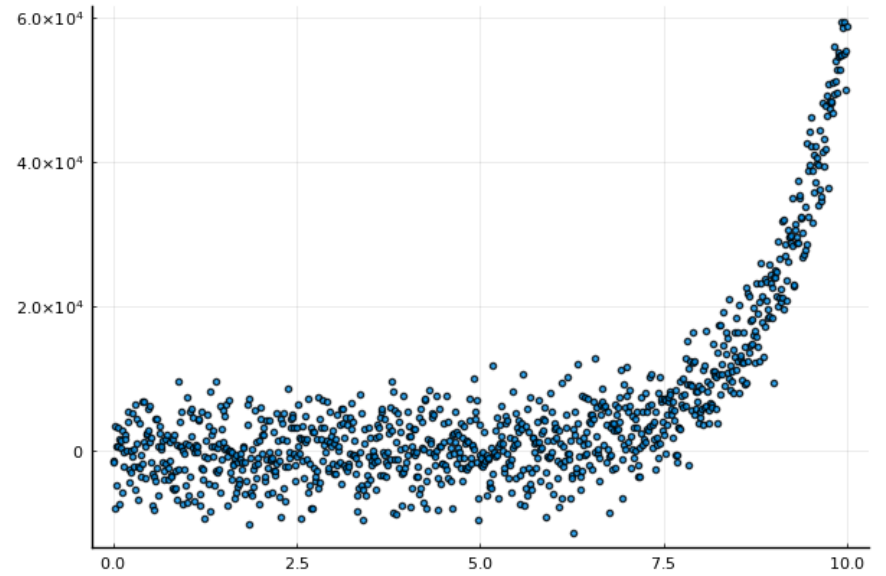


5.2.4. Аппроксимация данных

Аппроксимация — научный метод, состоящий в замене объектов их более простыми
аналогами, сходными по своим свойствам.

```
# экспоненциальная функция со случайным сдвигом значений:
y = exp.(ones(1000)+x) + 4000*randn(1000)
# построение графика:
scatter(x,y,markersize=3,alpha=.8,legend=false)
```

Out[35]:



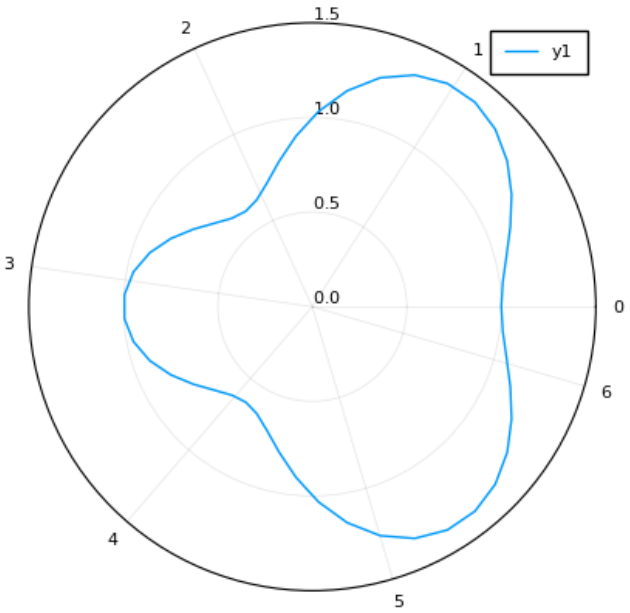
5.2.5. Две оси ординат

Иногда требуется на один график вывести несколько траекторий с существенными

отличиями в значениях по оси ординат.

```
# график функции, заданной в полярных координатах:
plot(θ, r.(θ),
proj=:polar,
lims=(0,1.5)
)
```

ut[46]:



5.2.7. Параметрический график

При параметрическом представлении графика некоторой функции координаты на

графике задаются как функции от некоторого набора свободных параметров. В случае

одного параметра получим параметрическое уравнение кривой. Выражая координаты

точек поверхности через два свободных параметра, получим параметрическое задание

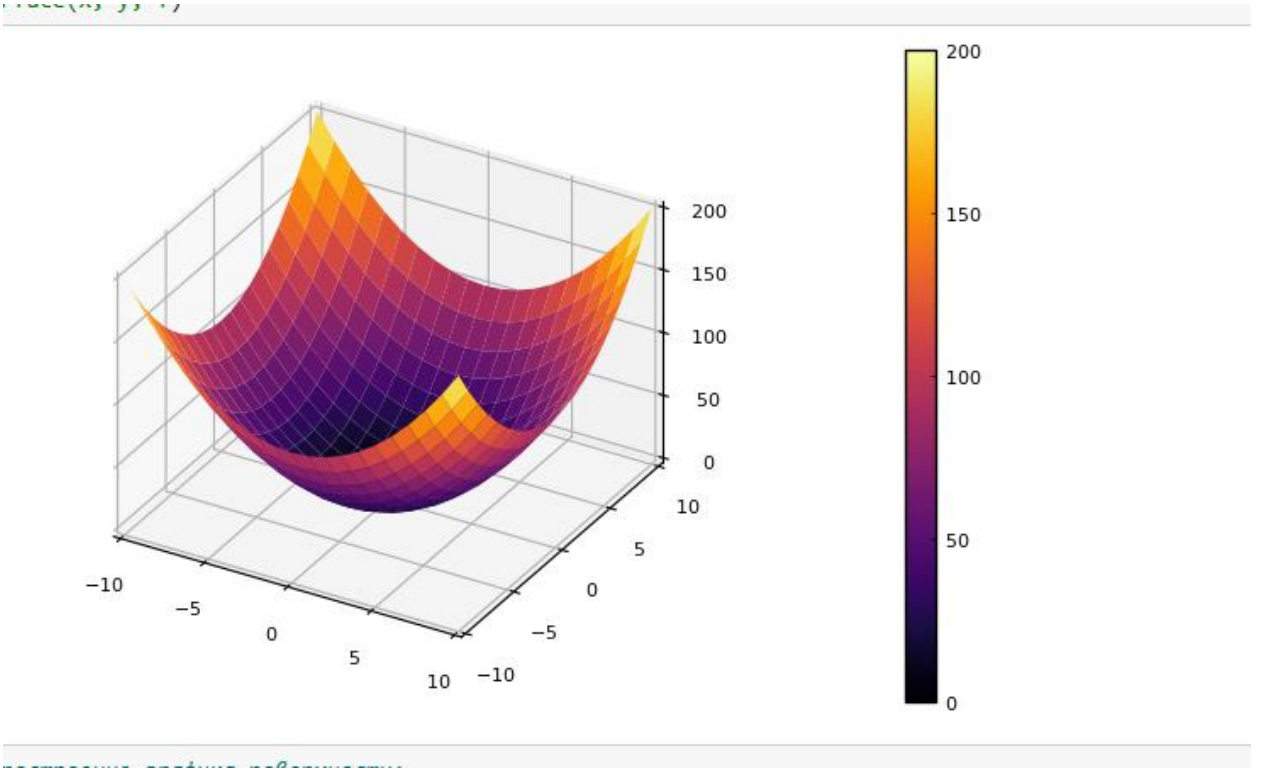
поверхности.

5.2.7.1. Параметрический график кривой на плоскости

Приведём пример построения графика параметрически заданной кривой на плоскости

(рис. 5.15). Кривая задана выражениями $x(t) = \sin(t)$, $y(t) = \sin(2t)$, $0 \leqslant t \leqslant$

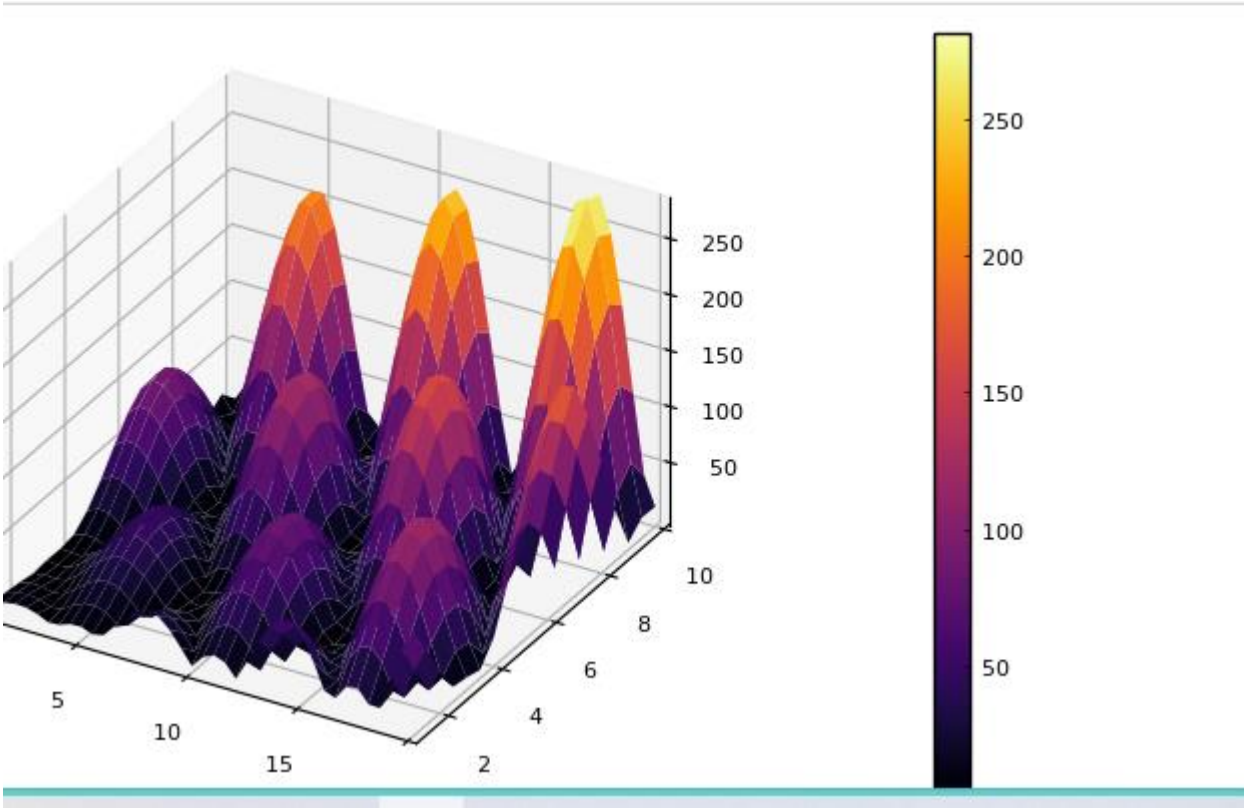
2π:



5.2.9. Линии уровня

Линией уровня некоторой функции от двух переменных называется множество точек
на координатной плоскости, в которых функция принимает одинаковые значения. Линий
уровня бесконечно много, и через каждую точку области определения можно провести
линию уровня.

С помощью линий уровня можно определить наибольшее и наименьшее значение
исходной функции от двух переменных. Каждая из этих линий соответствует определённому значению высоты.

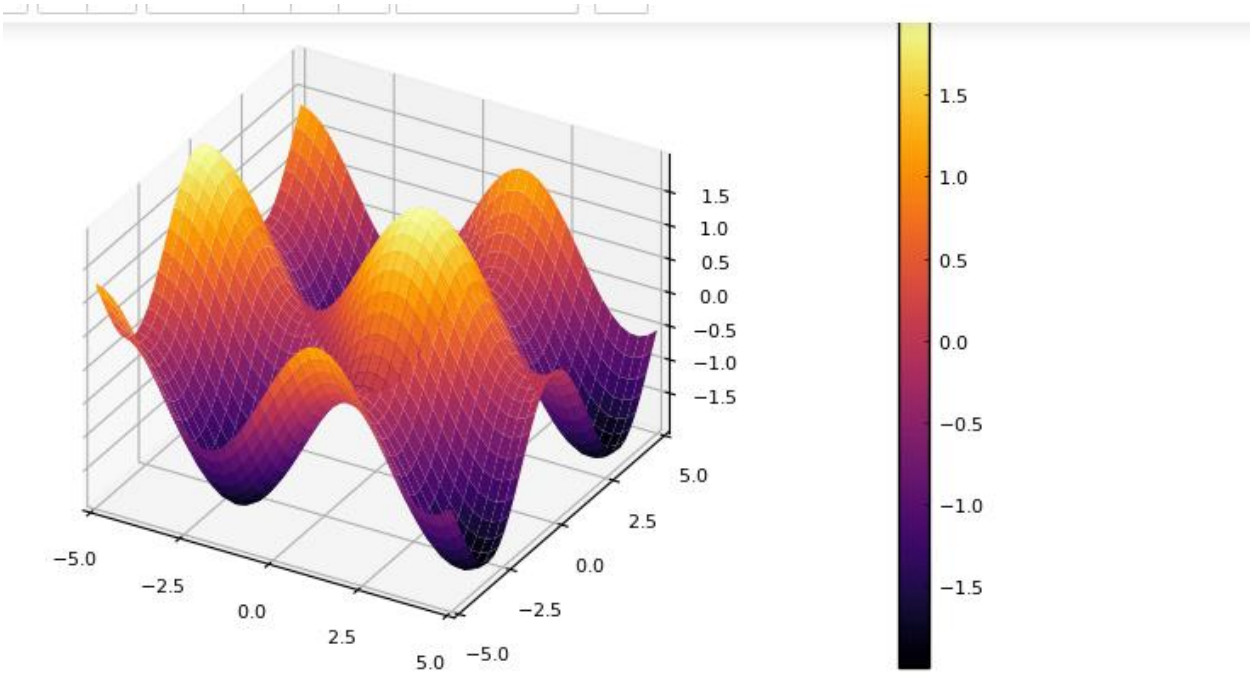


5.2.11. Анимация

Технически анимированное изображение представляет собой несколько наложенных

изображений (или построенных в разных точках графиках) в одном файле.

5.2.11.1. Gif-анимация



5.2.12. Errorbars

В исследованиях часто требуется изобразить графики погрешностей измерения.

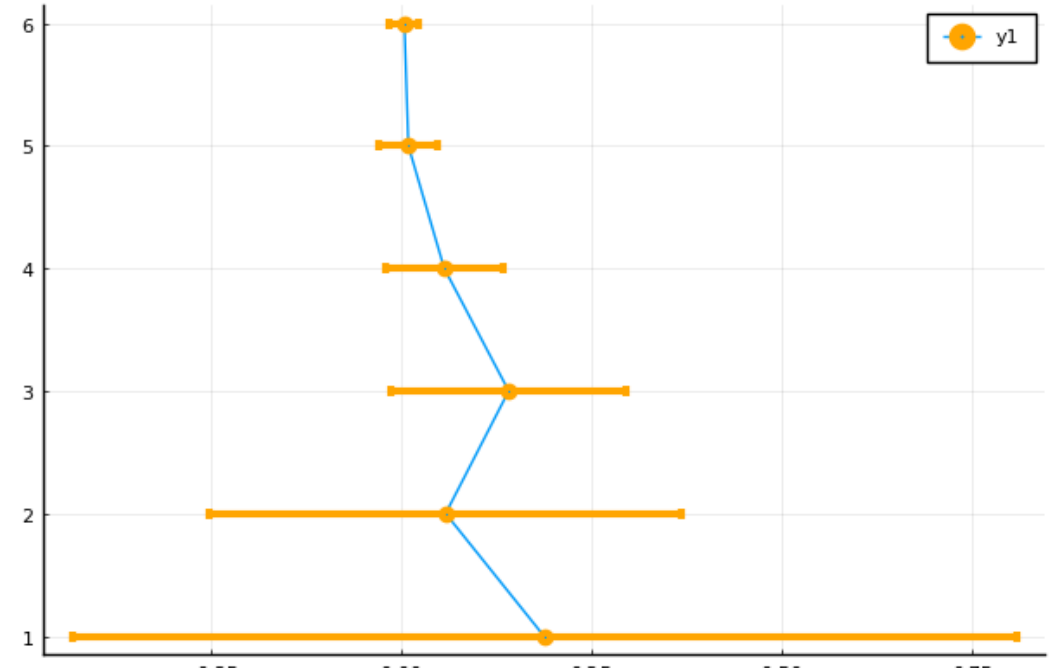
Подключим пакет Statistics:

 # подключение пакета Statistics:

import Pkg

Pkg.add("Statistics")

using Statistics



5.2.13. Использование пакета Distributions

Подгружаем пакет распределений:

import Pkg

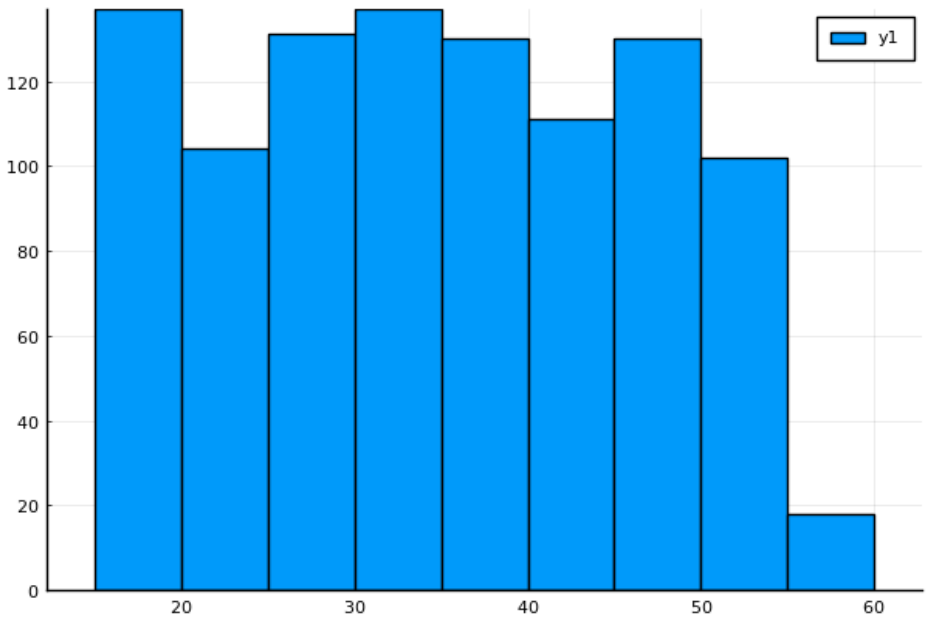
Pkg.add("Distributions")

using Distributions

Подгружаем pyplot():

pyplot()

```
In [84]: histogram(ages)
Out[84]:
```

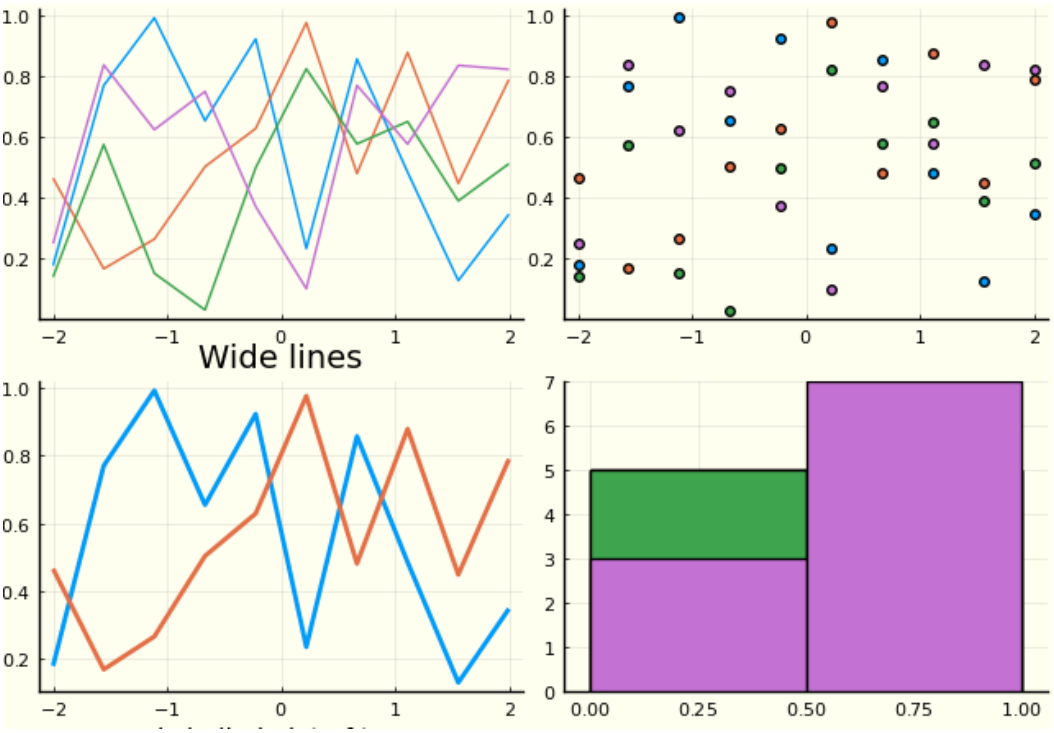


5.2.14. Подграфики

Определим макет расположения графиков. Команда layout принимает кортеж layout

= (N, M), который строит сетку графиков NxM. Например, если задать layout = (4,1)

на графике четыре серии, то получим четыре ряда графиков (рис. 5.44)



Выводы

В ходе полученные навыки в синтаксис языка Julia для построения графиков.

Начало работы

темы

1. Основные пакеты для работы с графиками в Julia
2. Опции при построении графика
3. Точечный график
4. Простой точечный график
5. Точечный график с кодированием значения размером точки
6. Аппроксимация данных
7. Две оси ординат
8. Полярные координаты
9. Параметрический график
10. Параметрический график кривой на плоскости
11. Параметрический график кривой в пространстве
12. График поверхности
13. Линии уровня
14. Векторные поля
15. Анимация
16. Gif-анимация
17. Гипоциклоида
18. Errorbars
19. Использование пакета Distributions
20. Подграфики
21. Задания для самостоятельного выполнения

5.2. Предварительные сведения

5.2.1. Основные пакеты для работы с графиками в Julia

In [1]:
```
using Pkg
```

In [2]:
```
Pkg.add("Plots")
```

```
  Updating registry at `~/.julia/registries/General.toml`
 Resolving package versions...
 Installed libfdk_aac_jll ──────────────── v2.0.2+0
 Installed JpegTurbo_jll ──────────────── v3.0.1+0
 Installed Libmount_jll ──────────────── v2.35.0+0
 Installed x265_jll ──────────────── v3.5.0+0
 Installed LERC_jll ──────────────── v3.0.0+1
 Installed GR_jll ──────────────── v0.72.10+0
 Installed Opus_jll ──────────────── v1.3.2+0
 Installed LoggingExtras ──────────────── v1.0.3
 Installed Xorg_xkbcomp_jll ──────────────── v1.4.6+0
 Installed ConcurrentUtilities ──────────── v2.3.0
 Installed Unitful ──────────────── v1.19.0
 Installed Measures ──────────────── v0.3.2
 Installed RelocatableFolders ──────────── v1.0.1
 Installed Grisu ──────────────── v1.0.2
 Installed Xorg_xcb_util_wm_jll ──────────── v0.4.1+1
 Installed Formatting ──────────────── v0.4.2
 Installed Contour ──────────────── v0.6.2
 Installed Xorg_xcb_util_image_jll ──────── v0.4.0+1
```

In [3]:
```
Pkg.add("Plots")
```

```
 Resolving package versions...
No Changes to `~/.julia/environments/v1.9/Project.toml`
No Changes to `~/.julia/environments/v1.9/Manifest.toml`
```

In [4]:
```
Pkg.add("Plotly")
```

```
 Resolving package versions...
 Installed URIParser ──────────────── v0.4.1
 Installed AssetRegistry ──────────────── v0.1.0
 Installed PlotlyBase ──────────────── v0.8.19
 Installed Pidfile ──────────────── v1.3.0
 Installed BinDeps ──────────────── v1.0.2
 Installed HTTP ──────────────── v0.9.17
 Installed Hiccup ──────────────── v0.2.2
 Installed Kaleido_jll ──────────────── v0.2.1+0
 Installed WebSockets ──────────────── v1.5.9
 Installed Lazy ──────────────── v0.15.1
 Installed Blink ──────────────── v0.12.5
 Installed TableTraits ──────────────── v1.0.1
 Installed DataValueInterfaces ──────────── v1.0.0
 Installed Plotly ──────────────── v0.4.1
 Installed UnPack ──────────────── v1.0.2
 Installed Mustache ──────────────── v1.0.19
 Installed JSExpr ──────────────── v0.5.4
 Installed FunctionalCollections ──────── v0.5.0
 Installed Widgets ──────────────── v0.6.6
```

In [5]:
```
Pkg.add("UnicodePlots")
```

```
   Resolving package versions...
   Installed MarchingCubes ──── v0.1.9
   Installed StaticArraysCore ─ v1.4.2
   Installed Crayons ─────────── v4.1.1
   Installed StaticArrays ────── v1.7.0
   Installed UnicodePlots ────── v3.6.0
    Updating `~/.julia/environments/v1.9/Project.toml`
  [b8865327] + UnicodePlots v3.6.0
    Updating `~/.julia/environments/v1.9/Manifest.toml`
  [a8cc5b0e] + Crayons v4.1.1
  [299715c1] + MarchingCubes v0.1.9
  [90137ffa] + StaticArrays v1.7.0
  [1e83bf80] + StaticArraysCore v1.4.2
  [b8865327] + UnicodePlots v3.6.0
Precompiling project...
  ✓ StaticArraysCore
  ✓ Crayons
  ✓ StaticArrays
  ✓ StaticArrays → StaticArraysStatisticsExt
  ✓ MarchingCubes
  ✓ UnicodePlots
  ✓ UnicodePlots → UnitfulExt
  7 dependencies successfully precompiled in 317 seconds. 173 alrea
dy precompiled.
```

In [6]:
```
# подключаем для использования Plots:
using Plots
```

In [7]:
```
# задание функции:
f(x) = (3x.^2 + 6x .- 9).*exp.(-0.3x)
```

Out[7]:
```
f (generic function with 1 method)
```

In [8]:
```
# генерирование массива значений x в диапазоне от -5 до 10 с шагом 0,
# (шаг задан через указание длины массива):
x = collect(range(-5,10,length=151))
```

Out[8]:
```
151-element Vector{Float64}:
 -5.0
 -4.9
 -4.8
 -4.7
 -4.6
 -4.5
 -4.4
 -4.3
 -4.2
 -4.1
 -4.0
 -3.9
 -3.8
  ⋮
  8.9
  9.0
  9.1
  9.2
  9.3
  9.4
  9.5
  9.6
  9.7
  9.8
  9.9
 10.0
```

In [9]:
```
# генерирование массива значений y:
y = f(x)
```

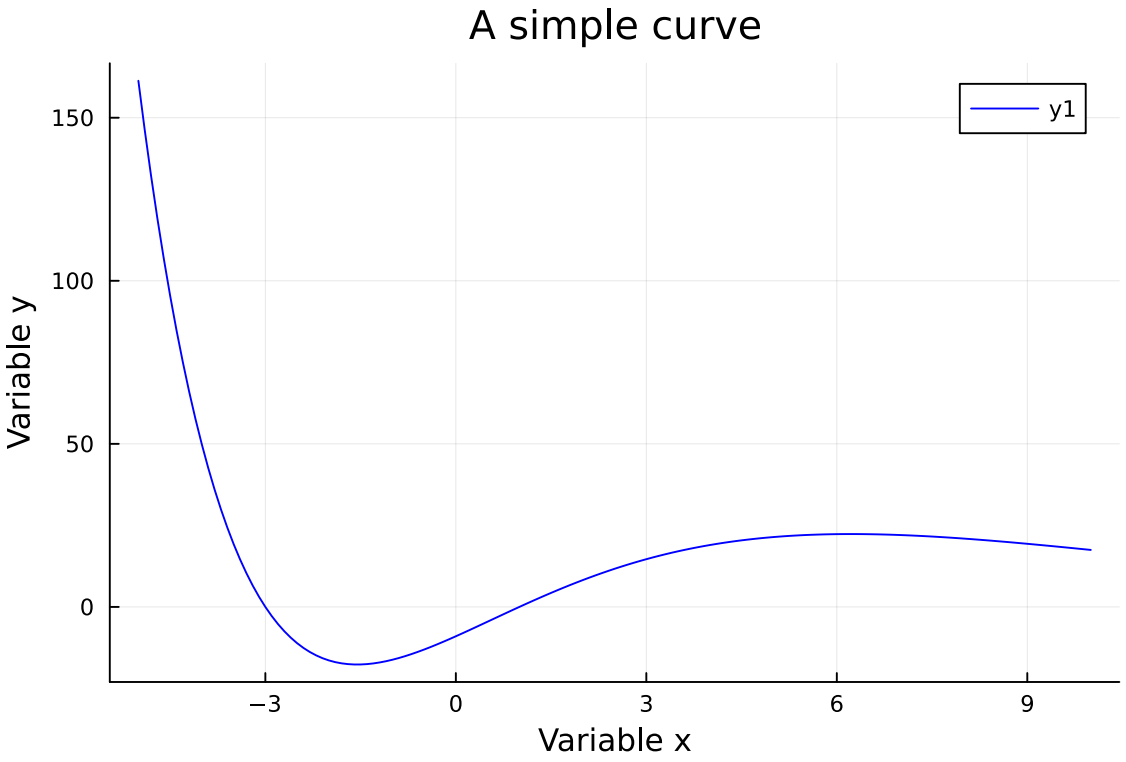
Out[9]:
```
151-element Vector{Float64}:
 161.34080653217032
 146.26477779394003
 132.19219298833204
 119.06942359634911
 106.8453557470588
  95.47128188475011
  84.9007968362764
  75.08969810741056
  65.99589024347995
  57.579293095755176
  49.80175384104821
  42.62696260773204
  36.02037156694452
   ⋮
  19.531205103994854
  19.355187669047936
  19.176427741119536
  18.995125520095375
  18.811475185747092
  18.625664977689375
  18.437877278854756
  18.248288702120195
  18.057070179740368
  17.86438705526336
  17.6703991776229
  17.475260997120245
```

In [10]:
```
# указывается, что для построения графика используется gr():
gr()
```

Out[10]:
```
Plots.GRBackend()
```

In [13]:
```
#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")
```

Out[13]:



In [15]:
```
import Pkg;
```

In [16]:
```
  Pkg.add("PyPlot")
```

```
   Resolving package versions...
   Installed PyPlot ─ v2.11.2
   Installed PyCall ─ v1.96.3
    Updating `~/.julia/environments/v1.9/Project.toml`
  [d330b81b] + PyPlot v2.11.2
    Updating `~/.julia/environments/v1.9/Manifest.toml`
  [438e738f] + PyCall v1.96.3
  [d330b81b] + PyPlot v2.11.2
    Building PyCall → `~/.julia/scratchspaces/44cfe95a-1eb2-52ea-b672-e2afdf69b78f/c9932f1c60d2e653df4f06d76108af8fde2200c0/build.log`
Precompiling project...
  ✓ PyCall
  ✓ PyPlot
  2 dependencies successfully precompiled in 72 seconds. 180 already precompiled.
```

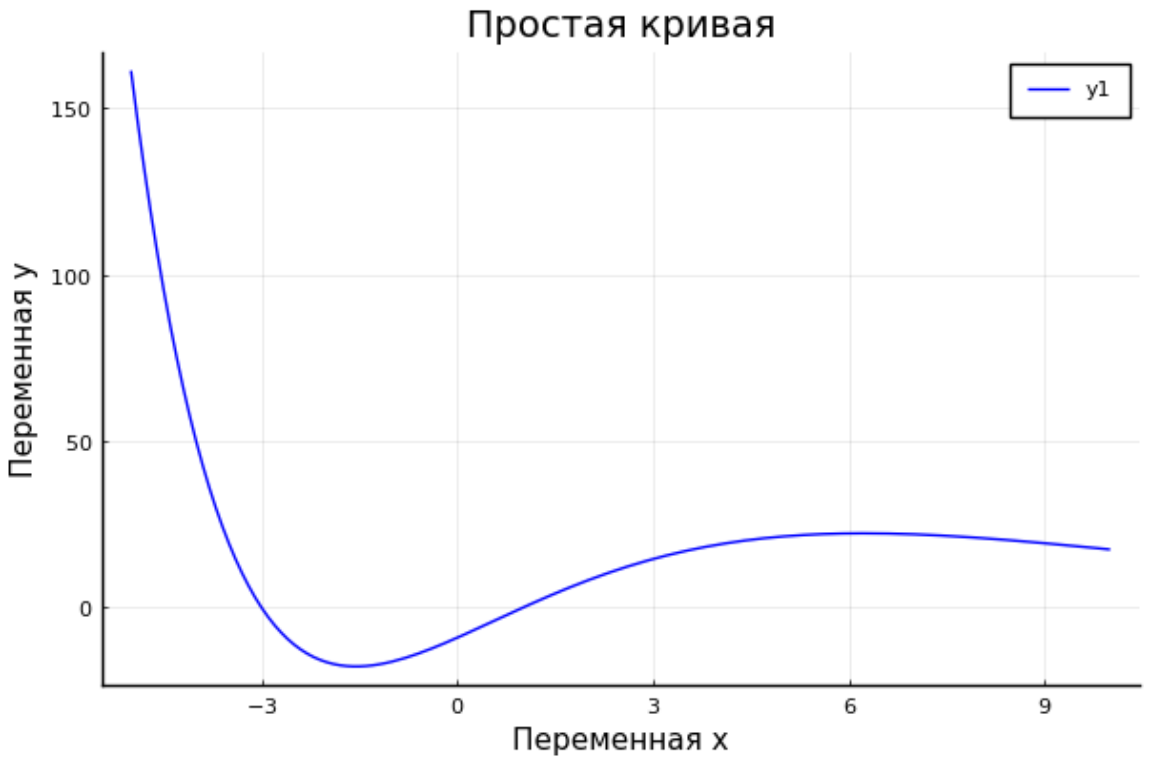
In [17]:
```
# указывается, что для построения графика используется pyplot():
pyplot()
```

```
Matplotlib is building the font cache; this may take a moment.
```

Out[17]:
```
Plots.PyPlotBackend()
```

In [18]:
```
# задание опций при построении графика
# (название кривой, подписи по осям, цвет графика):
plot(x,y,
title="Простая кривая",
xlabel="Переменная x",
ylabel="Переменная y",
color="blue")
```

Out[18]:

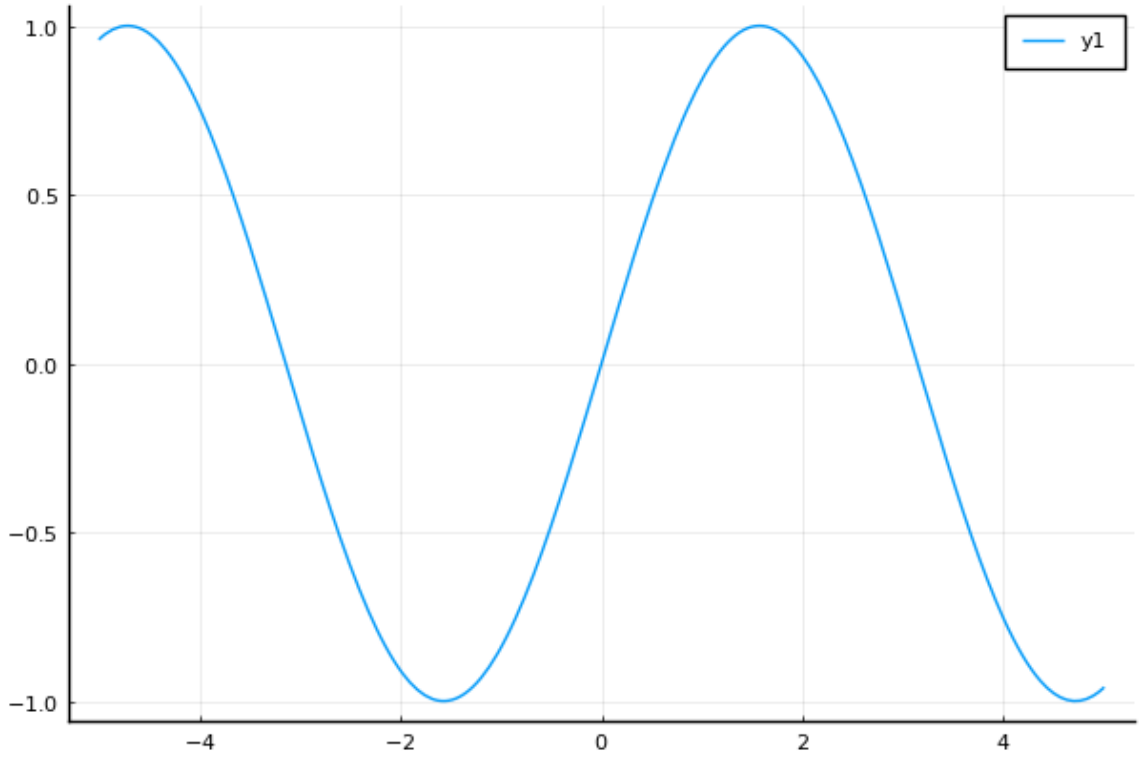


In [19]:
```
# задание функции sin(x):
sin_theor(x) = sin(x)
```

Out[19]:
```
sin_theor (generic function with 1 method)
```

In [20]:
```
# построение графика функции sin(x):
plot(sin_theor)
```

Out[20]:



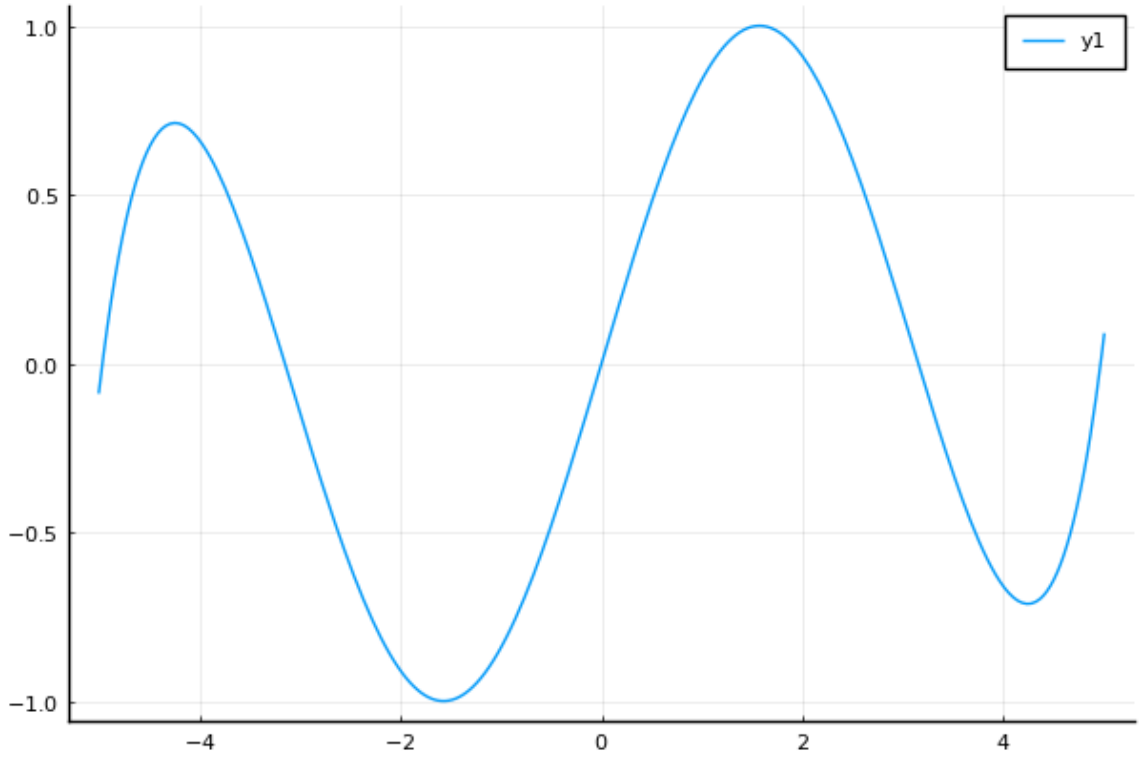
5.2.2. Опции при построении графика

In [21]:
```
# 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] |> s
```

Out[21]:
```
sin_taylor (generic function with 1 method)
```

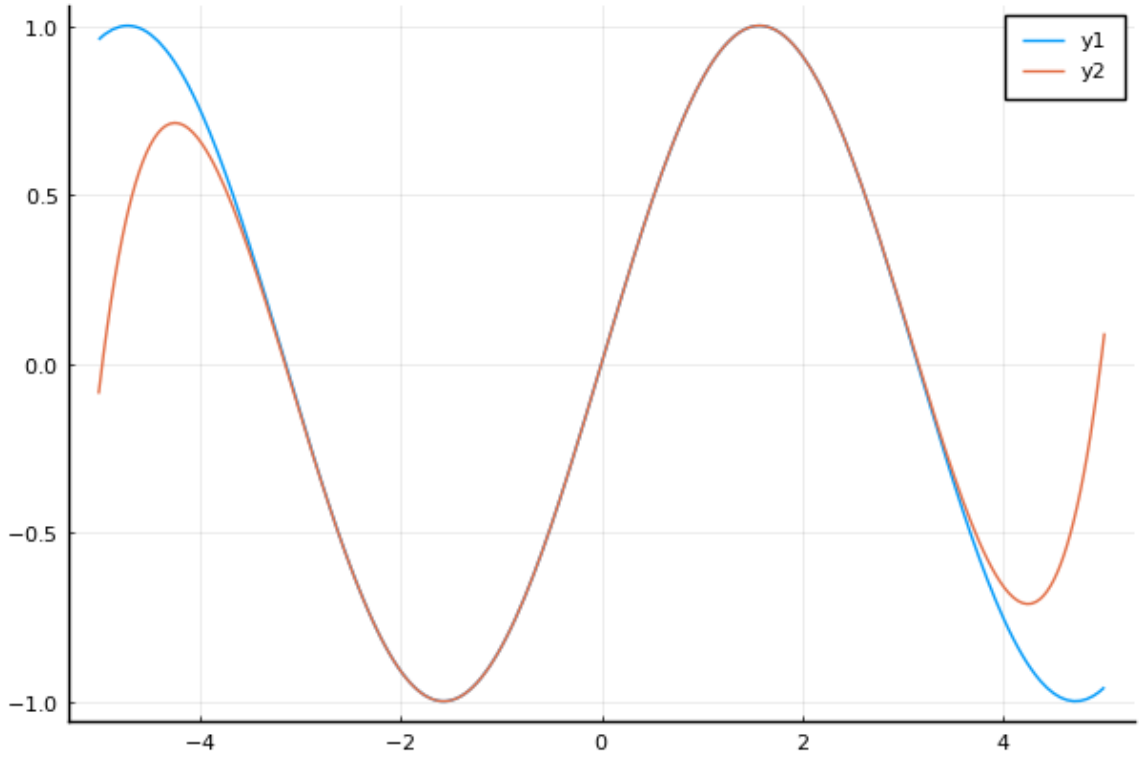
In [22]:
```
# построение графика функции sin_taylor(x):
plot(sin_taylor)
```

Out[22]:



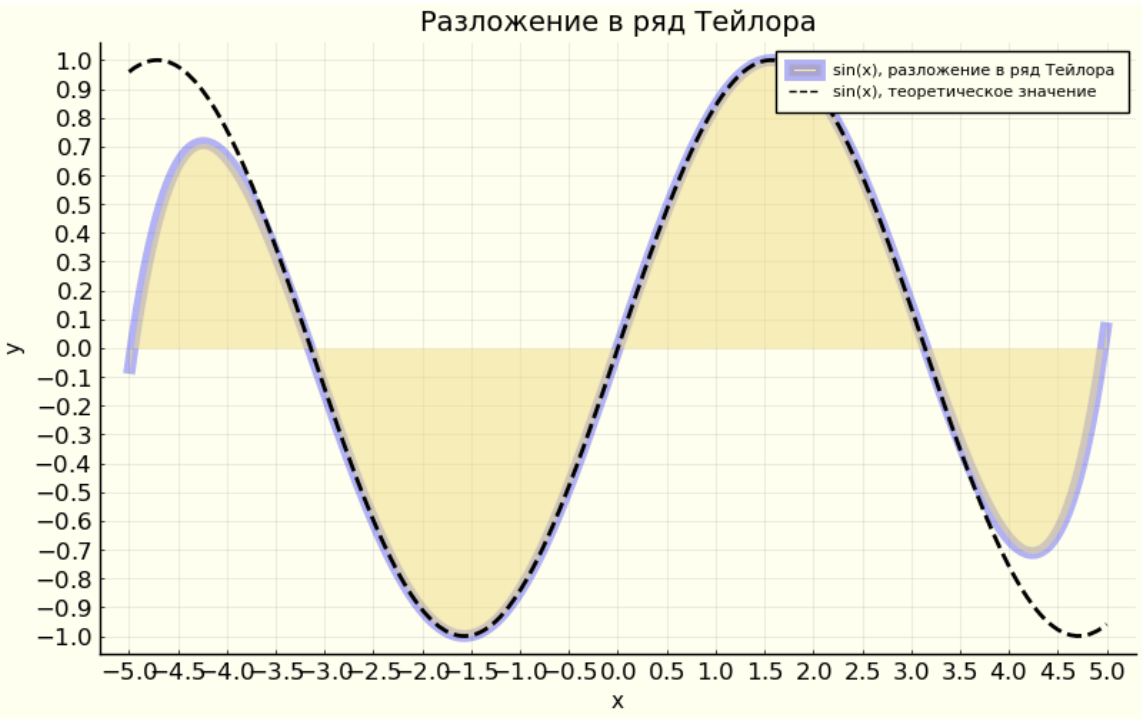
In [23]:
```
# построение двух функций на одном графике:
plot(sin_theor)
plot!(sin_taylor)
```

Out[23]:



In [25]:
```
plot(
# функция sin(x):
sin_taylor,
# подпись в легенде, цвет и тип линии:
label = "sin(x), разложение в ряд Тейлора",
line=(:blue, 0.3, 6, :solid),
# размер графика:
size=(800, 500),
# параметры отображения значений по осям
xticks = (-5:0.5:5),
yticks = (-1:0.1:1),
xtickfont = font(12, "Times New Roman"),
ytickfont = font(12, "Times New Roman"),
    # подписи по осям:
ylabel = "y",
xlabel = "x",
# название графика:
title = "Разложение в ряд Тейлора",
# поворот значений, заданный по оси x:
xrotation = rad2deg(pi/4),
# заливка области графика цветом:
fillrange = 0,
fillalpha = 0.5,
fillcolor = :lightgoldenrod,
# задание цвета фона:
background_color = :ivory
)
plot!(
# функция sin_theor:
sin_theor,
# подпись в легенде, цвет и тип линии:
label = "sin(x), теоретическое значение",
line=(:black, 1.0, 2, :dash))
```

Out[25]:




```
findfont: Font family ['Times New Roman'] not found. Falling back t
o DejaVu Sans.
findfont: Font family ['Times New Roman'] not found. Falling back t
o DejaVu Sans.
```

In [26]:
```
# сохранение графика в файле в формате pdf или png:
savefig("taylor.pdf")
savefig("taylor.png")
```

Out[26]:
```
"/home/marc/Desktop/taylor.png"
```

5.2.3. Точечный график

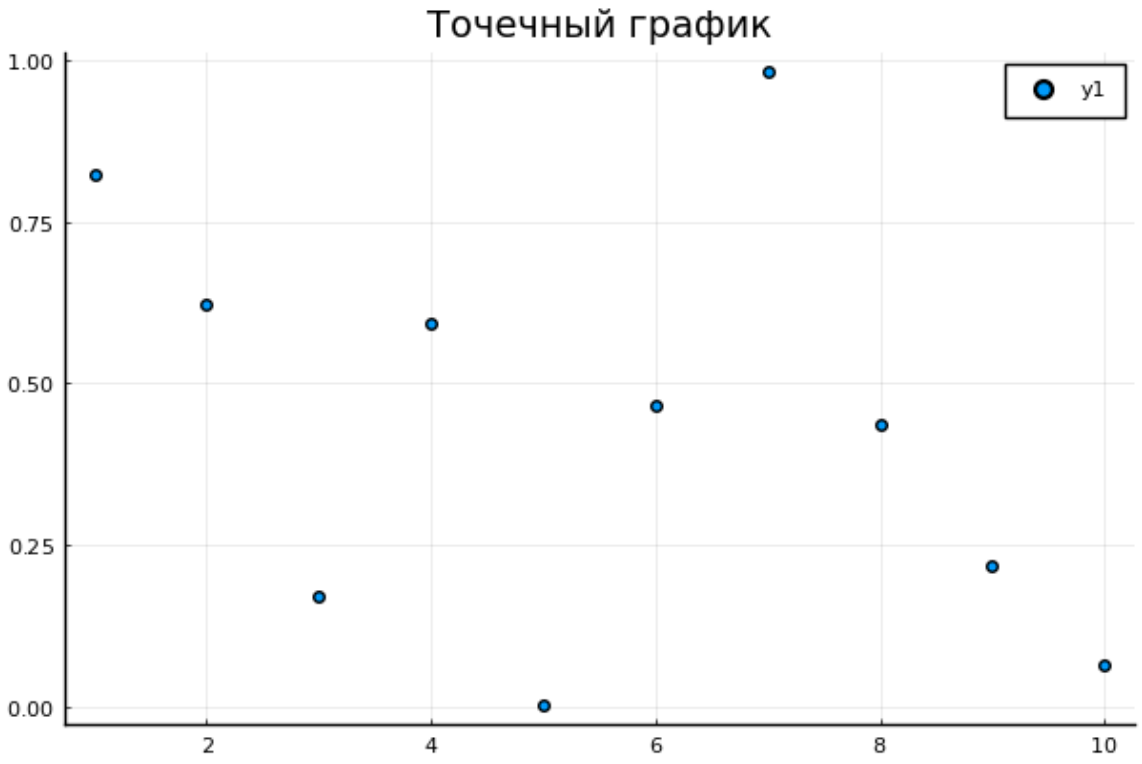
Графики в виде точек на плоскости или в пространстве часто используются в статисти- ческих исследованиях. **5.2.3.1. Простой точечный график** Как и построении обычного графика для точечного графика необходимо задать массив значений x, посчитать или задать значения y, задать опции построения графика:

In [27]:
```
# параметры распределения точек на плоскости:
x = range(1,10,length=10)
y = rand(10)
```

Out[27]:
```
10-element Vector{Float64}:
 0.822436754295574
 0.6219931715017678
 0.17007607071121156
 0.5911302592552476
 0.001716920472386918
 0.4645295645663181
 0.9810646316411372
 0.4353186021307912
 0.21847157274803497
 0.064706387777557
```

In [28]:
```
# параметры построения графика:
plot(x, y,
seriestype = :scatter,
title = "Точечный график"
)
```

Out[28]:



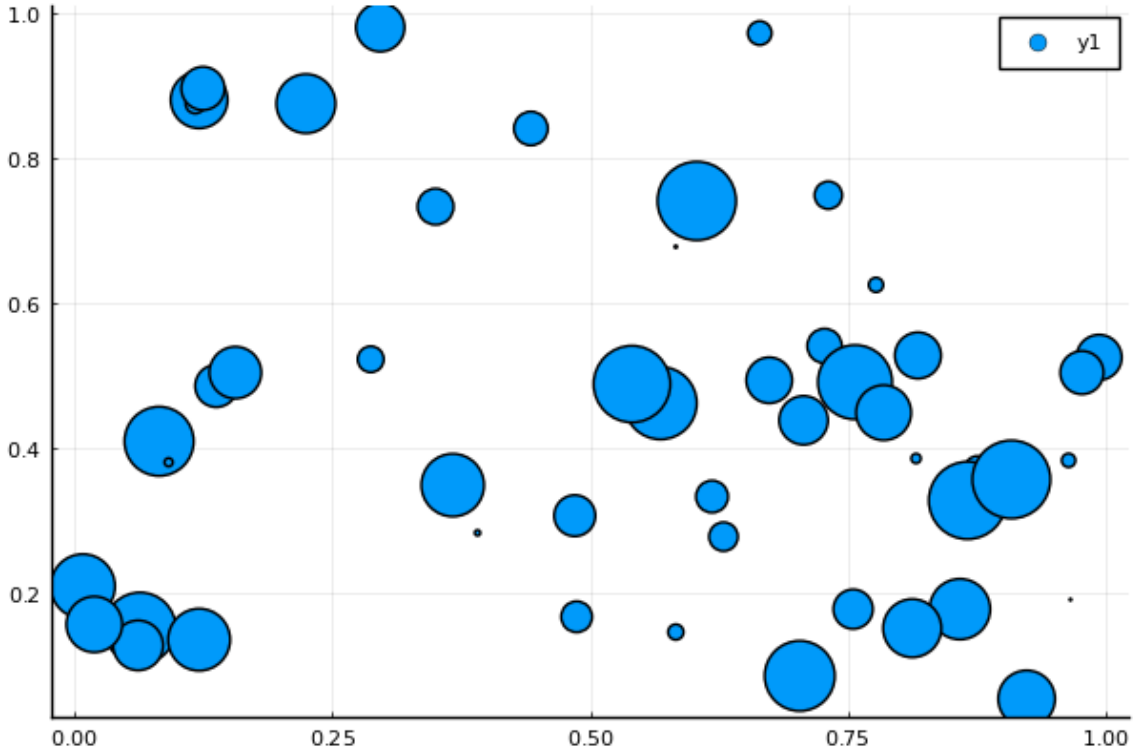
5.2.3.2. Точечный график с кодированием значения размером точки

In [29]:
```
# параметры распределения точек на плоскости:
n = 50
x = rand(n)
y = rand(n)
ms = rand(50) * 30
```

Out[29]:
```
50-element Vector{Float64}:
 21.483231261341434
  9.025418876563593
  9.015342924918379
  5.1696674298952585
 10.43203146839884
 24.281005750582292
 13.055751411328087
 27.362053104521344
 22.894631475265147
 21.453716082286743
 15.972769460486823
 13.650842440424945
  2.138315687556107
  ⋮
 12.779263509354212
 22.3220713441298
 21.06072039830772
 16.326938749300997
 26.49783028847362
 20.781811666974498
 16.321624080484444
  5.500762472232472
 28.940443845357713
 18.40871837036443
 19.58877604023983
 23.429611305712807
```

In [31]:
```
# параметры построения графика:
scatter(x, y, markersize=ms)
```

Out[31]:

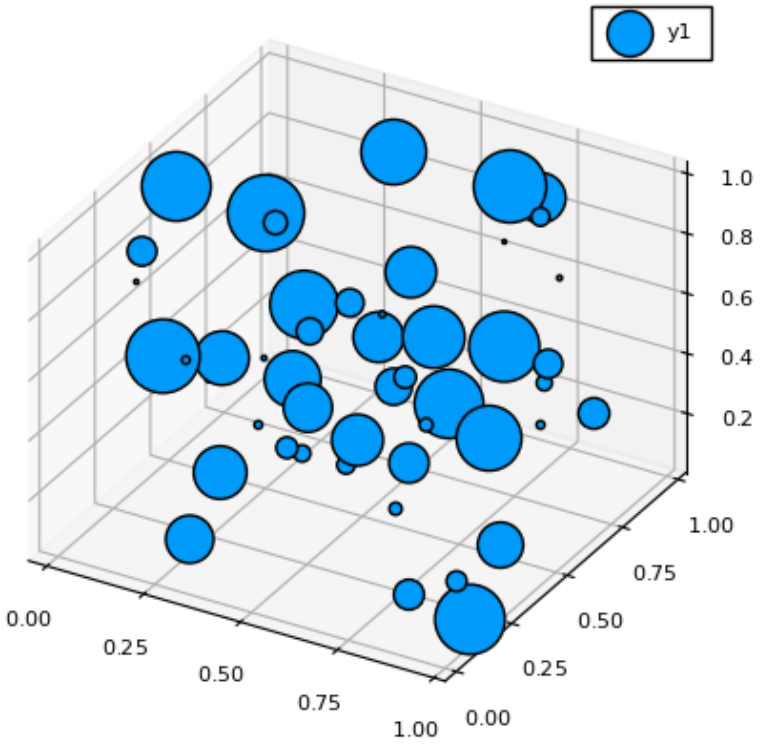


In [32]:
```
# параметры распределения точек в пространстве:
n = 50
x = rand(n)
y = rand(n)
z = rand(n)
ms = rand(50) * 30
```

Out[32]:
```
50-element Vector{Float64}:
 17.227419978497206
 28.974413220240763
 18.06747331247271
  3.1370286062755524
 24.590517751039982
 23.416666565194568
 24.60953555707589
  1.738270343729471
  8.349884443437741
  6.833157655464527
  6.354787356839982
 10.117483078300276
  3.09770057768441
  ⋮
 15.051420840519803
 27.826906306042563
 21.279019284433723
 13.892721277232496
 16.274333456127813
 26.051770757824848
 19.287815229723144
 11.449674948948322
 27.421844143292702
 11.741868582519645
  1.528646458560341
  6.8108695060764255
```

In [33]:
```
# параметры построения графика:
scatter(x, y, z, markersize=ms)
```

Out[33]:



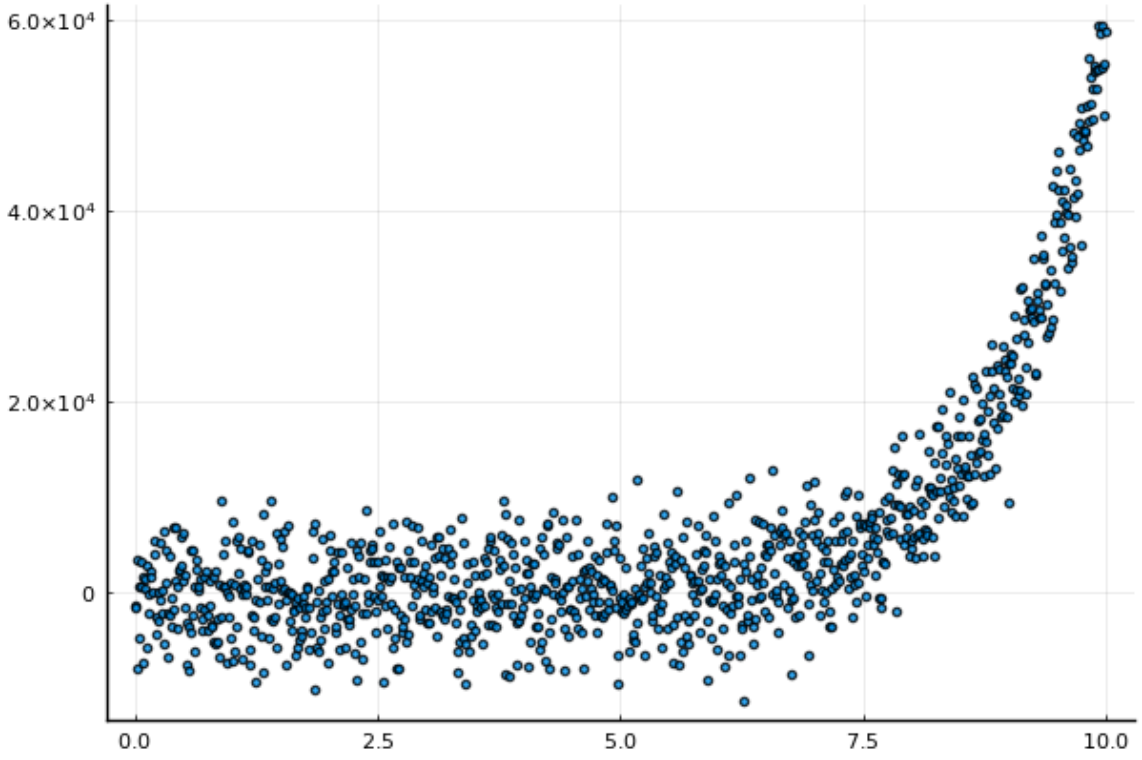
5.2.4. Аппроксимация данных

In [34]:
```
# массив данных от 0 до 10 с шагом 0.01:
x = collect(0:0.01:9.99)
```

Out[34]:
```
1000-element Vector{Float64}:
 0.0
 0.01
 0.02
 0.03
 0.04
 0.05
 0.06
 0.07
 0.08
 0.09
 0.1
 0.11
 0.12
 ⋮
 9.88
 9.89
 9.9
 9.91
 9.92
 9.93
 9.94
 9.95
 9.96
 9.97
 9.98
 9.99
```

In [35]:
```
# массив данных от 0 до 10 с шагом 0.01:
x = collect(0:0.01:9.99)
# экспоненциальная функция со случайным сдвигом значений:
y = exp.(ones(1000)+x) + 4000*randn(1000)
# построение графика:
scatter(x,y,markersize=3,alpha=.8,legend=false)
```

Out[35]:



In [37]:
```
# определение массива для нахождения коэффициентов полинома:
A = [ones(1000) x x.^2 x.^3 x.^4 x.^5]
```

Out[37]:
```
1000×6 Matrix{Float64}:
 1.0  0.0    0.0       0.0          0.0             0.0
 1.0  0.01   0.0001    1.0e-6       1.0e-8          1.0e-10
 1.0  0.02   0.0004    8.0e-6       1.6e-7          3.2e-9
 1.0  0.03   0.0009    2.7e-5       8.1e-7          2.43e-8
 1.0  0.04   0.0016    6.4e-5       2.56e-6         1.024e-7
 1.0  0.05   0.0025    0.000125     6.25e-6         3.125e-7
 1.0  0.06   0.0036    0.000216     1.296e-5        7.776e-7
 1.0  0.07   0.0049    0.000343     2.401e-5        1.6807e-6
 1.0  0.08   0.0064    0.000512     4.096e-5        3.2768e-6
 1.0  0.09   0.0081    0.000729     6.561e-5        5.9049e-6
 1.0  0.1    0.01      0.001        0.0001          1.0e-5
 1.0  0.11   0.0121    0.001331     0.00014641      1.61051e-5
 1.0  0.12   0.0144    0.001728     0.00020736      2.48832e-5
 ⋮                                                 ⋮
 1.0  9.88  97.6144  964.43      9528.57        94142.3
 1.0  9.89  97.8121  967.362     9567.21        94619.7
 1.0  9.9   98.01    970.299     9605.96        95099.0
 1.0  9.91  98.2081  973.242     9644.83        95580.3
 1.0  9.92  98.4064  976.191     9683.82        96063.5
 1.0  9.93  98.6049  979.147     9722.93        96548.7
 1.0  9.94  98.8036  982.108     9762.15        97035.8
 1.0  9.95  99.0025  985.075     9801.5         97524.9
 1.0  9.96  99.2016  988.048     9840.96        98015.9
 1.0  9.97  99.4009  991.027     9880.54        98509.0
 1.0  9.98  99.6004  994.012     9920.24        99004.0
 1.0  9.99  99.8001  997.003     9960.06        99501.0
```

In [38]:
```
# решение матричного уравнения:
c = A\y
```

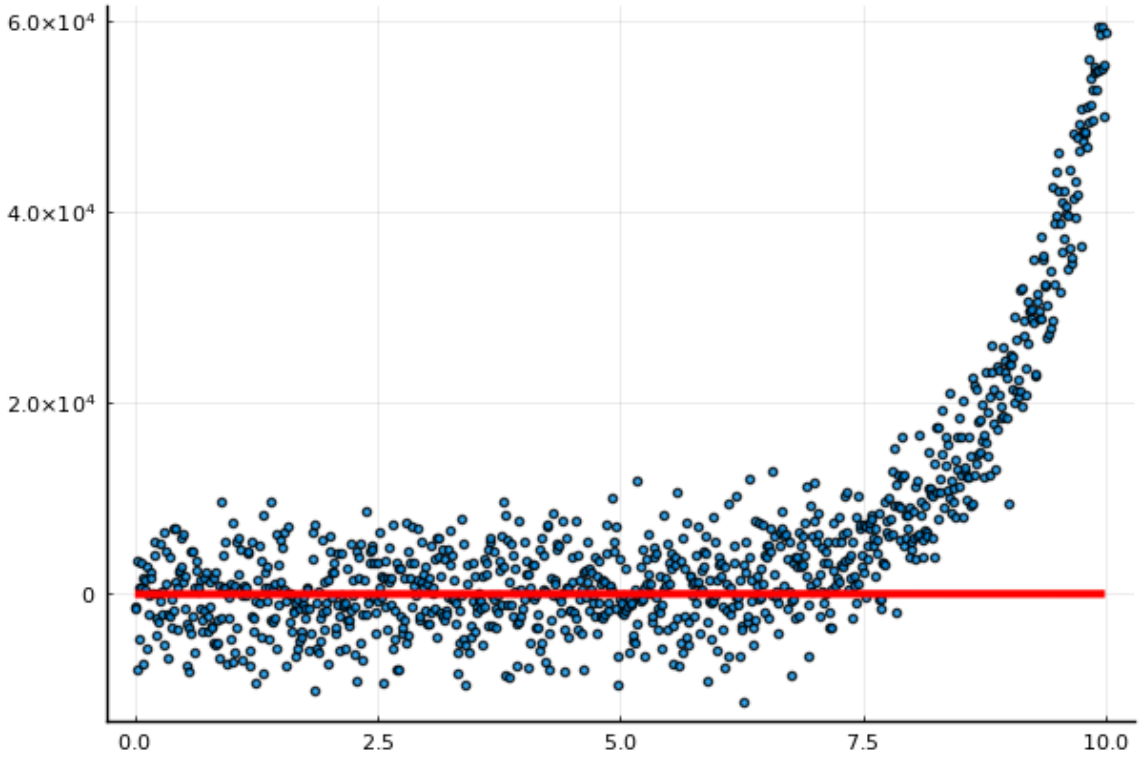
Out[38]:
```
6-element Vector{Float64}:
  -813.1544762143986
  2208.5168801188497
 -2118.8126399987154
   783.9498290320296
  -120.86148779277079
     6.724495820686029
```

In [41]:
```
# построение полинома:
f = c[1]*ones(1000) + c[2]*x + c[3]*x.^2 + c[4]*x.^3 + c[5]*x.^4 +

c[6]*x.^5
```

```
invalid redefinition of constant f

Stacktrace:
 [1] top-level scope
   @ In[41]:2
```

In [42]:
```
# построение графика аппроксимирующей функции:
plot!(x,f,linewidth=3, color=:red)
```

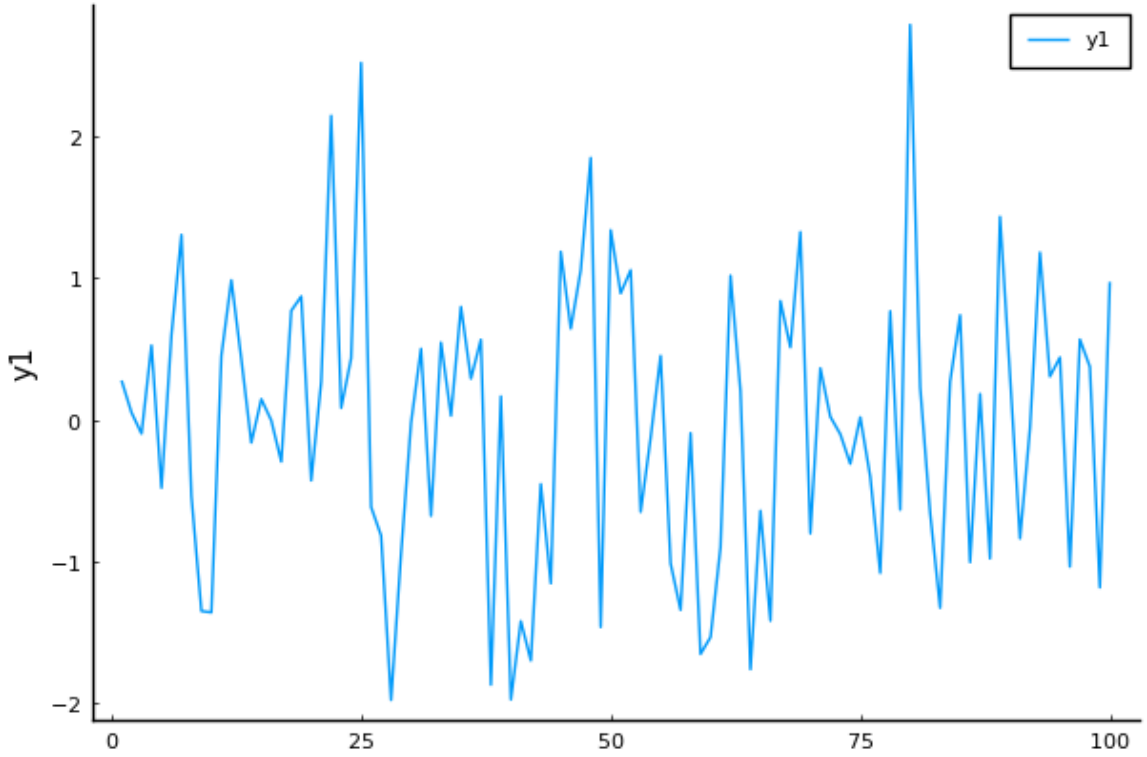
Out[42]:



5.2.5. Две оси ординат

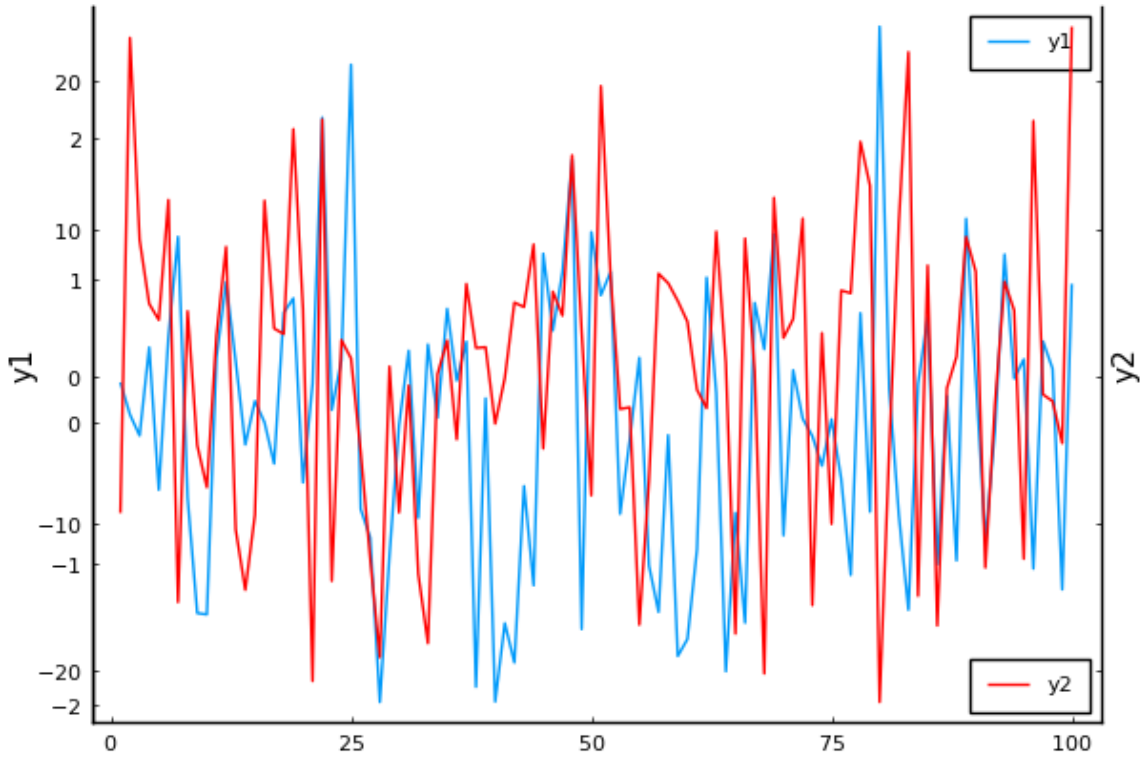
In [43]:
```
# пример случайной траектории
# (заданы обозначение траектории, легенда вверху справа, без сетки)
plot(randn(100),
ylabel="y1",
leg=:topright,
grid = :off,
)
```

Out[43]:



In [45]:
```
# пример добавления на график второй случайной траектории
# (задано обозначение траектории и её цвет, легенда снизу справа, бе
# задана рамка графика
plot!(twinx(), randn(100)*10,
c=:red,
ylabel="y2",
leg=:bottomright,
grid = :off,
box = :on,
#
size=(600, 400)
)
```

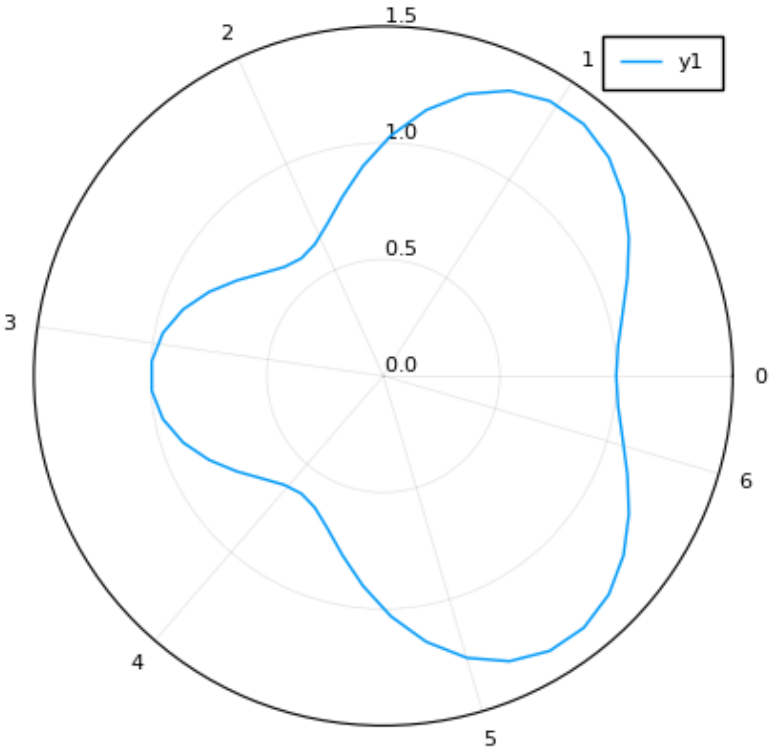
Out[45]:



5.2.6. Полярные координаты

In [46]:
```
# функция в полярных координатах:
r(θ) = 1 + cos(θ) * sin(θ)^2
# полярная система координат:
θ = range(0, stop=2π, length=50)
# график функции, заданной в полярных координатах:
plot(θ, r.(θ),
proj=:polar,
lims=(0,1.5)
)
```

Out[46]:

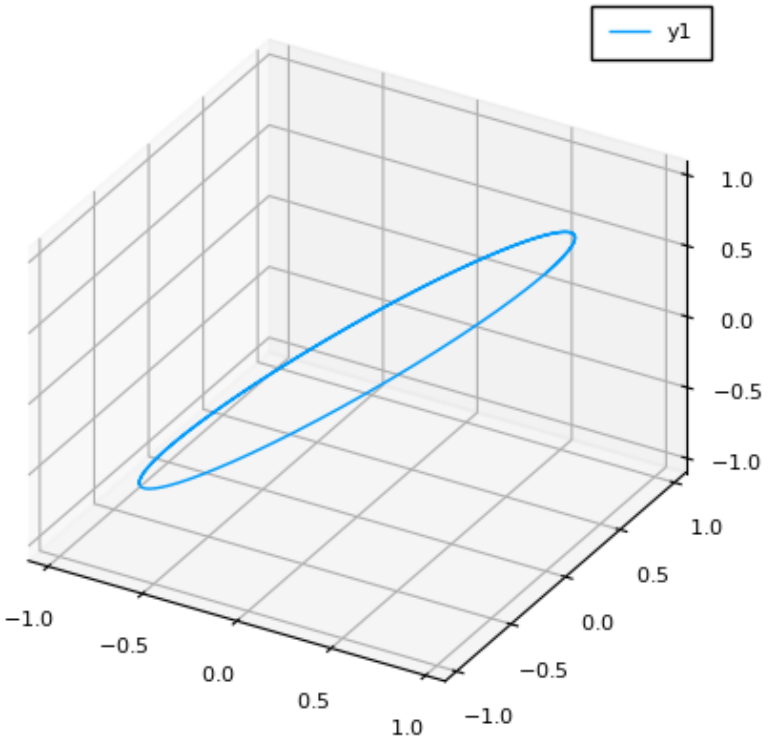


5.2.7. Параметрический график

При параметрическом представлении графика некоторой функции координаты на графике задаются как функции от некоторого набора свободных параметров. В случае одного параметра получим параметрическое уравнение кривой. Выражая координаты точек поверхности через два свободных параметра, получим параметрическое задание поверхности. **5.2.7.1. Параметрический график кривой на плоскости**

In [49]:
```
# параметрическое уравнение
t = range(0, stop=10, length=1000)
x = cos.(t)
y = sin.(t)
z = sin.(5t)
# построение графика:
plot(x, y, x)
```

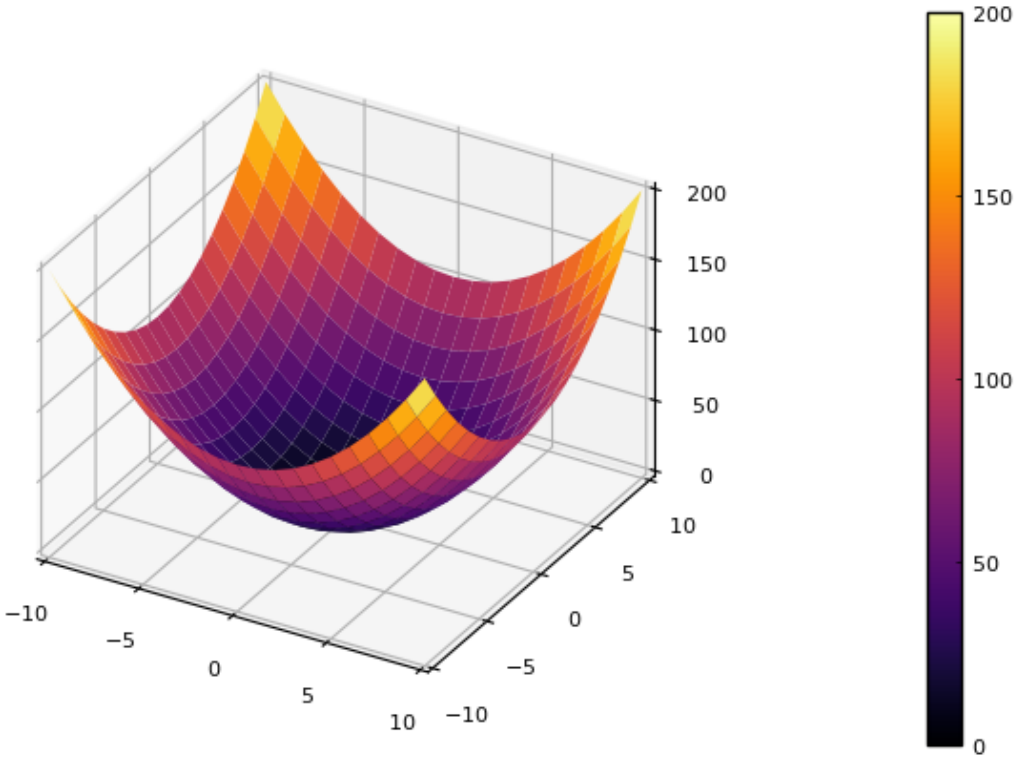
Out[49]:



5.2.8. График поверхности

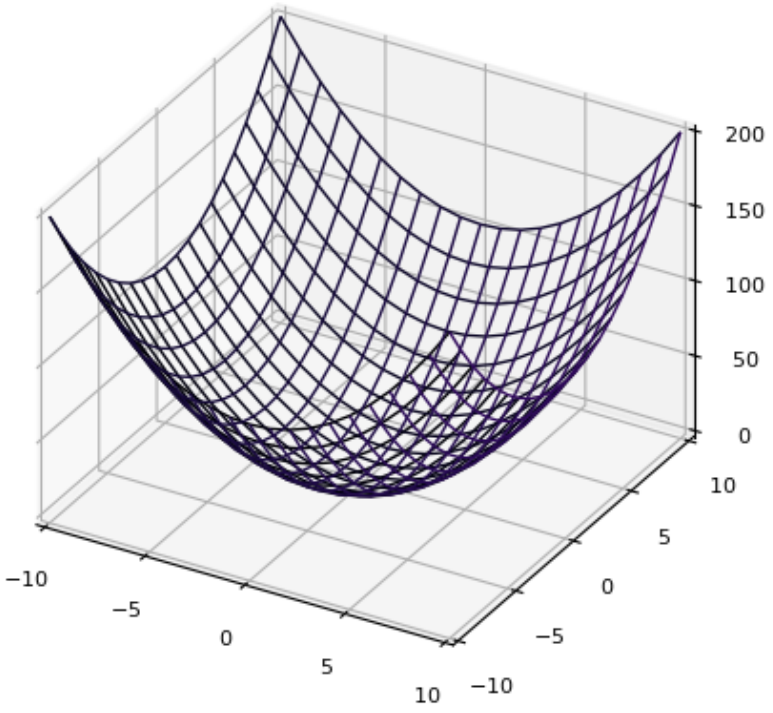
In [52]:
```
# построение графика поверхности:
f(x,y) = x^2 + y^2
x = -10:10
y = x
surface(x, y, f)
```

Out[52]:



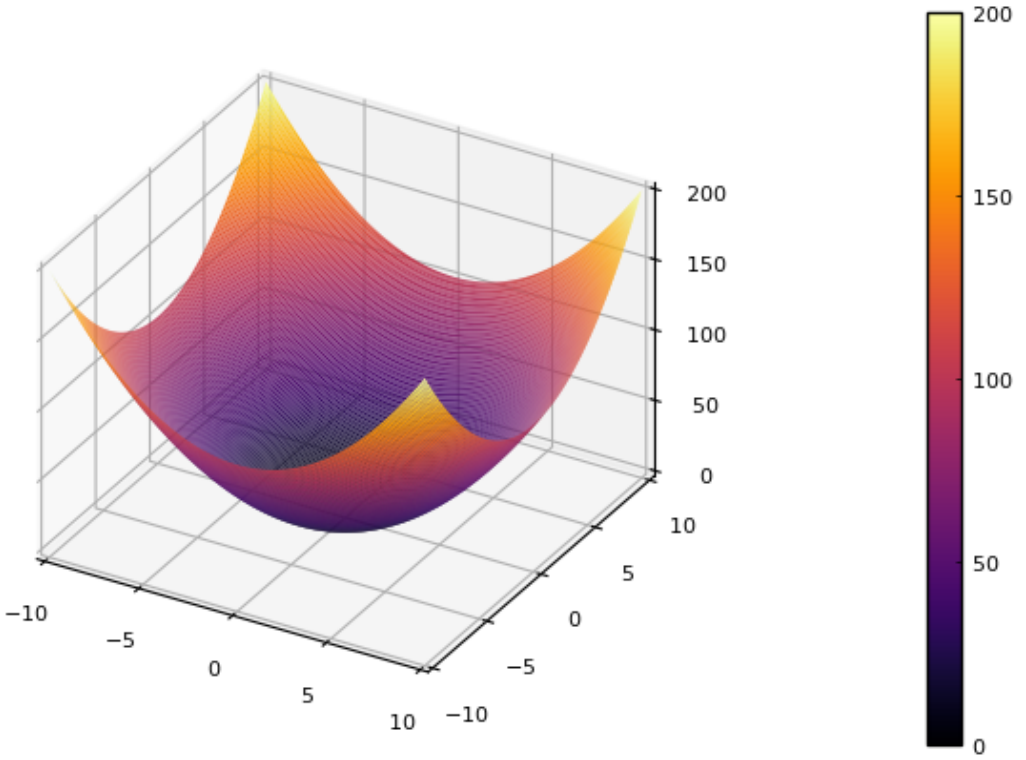
In [53]:
```
# построение графика поверхности:
f(x,y) = x^2 + y^2
x = -10:10
y = x
plot(x, y, f,
linetype=:wireframe
)
```

Out[53]:



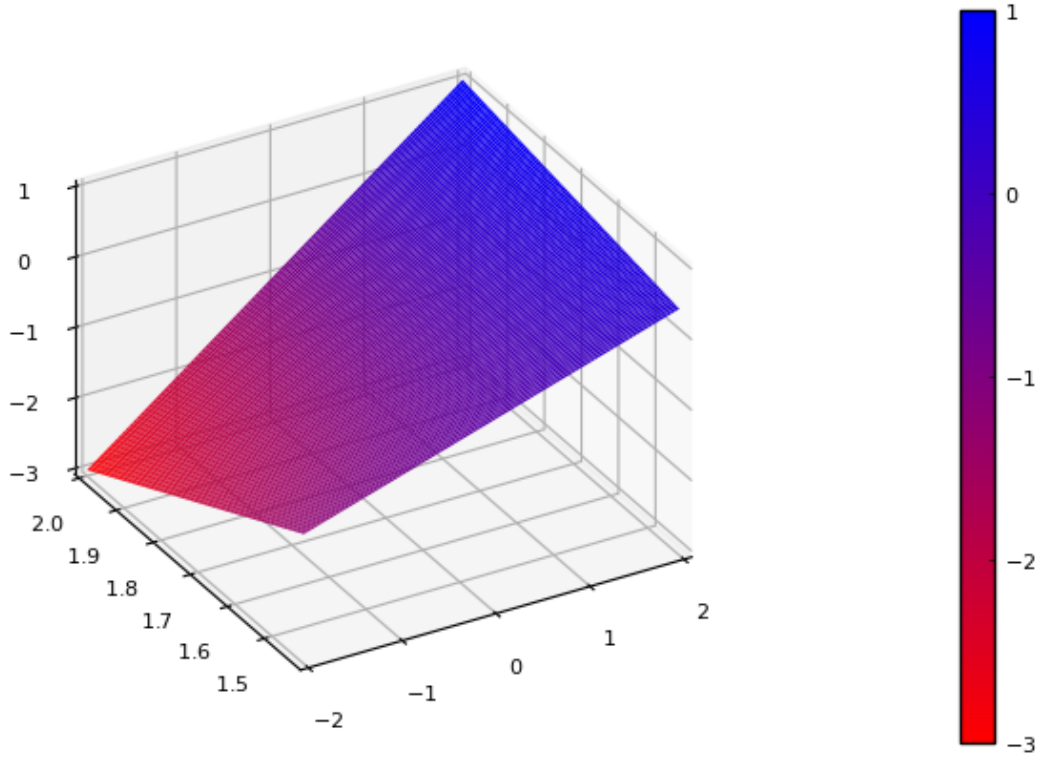
In [54]:
```
f(x,y) = x^2 + y^2
x = -10:0.1:10
y = x
plot(x, y, f,
linetype = :surface
)
```

Out[54]:



In [55]:
```
x=range(-2,stop=2,length=100)
y=range(sqrt(2),stop=2,length=100)
f(x,y) = x*y-x-y+1
plot(x,y,f,
linetype = :surface,
c=cgrad([:red,:blue]),
camera=(-30,30),
)
```

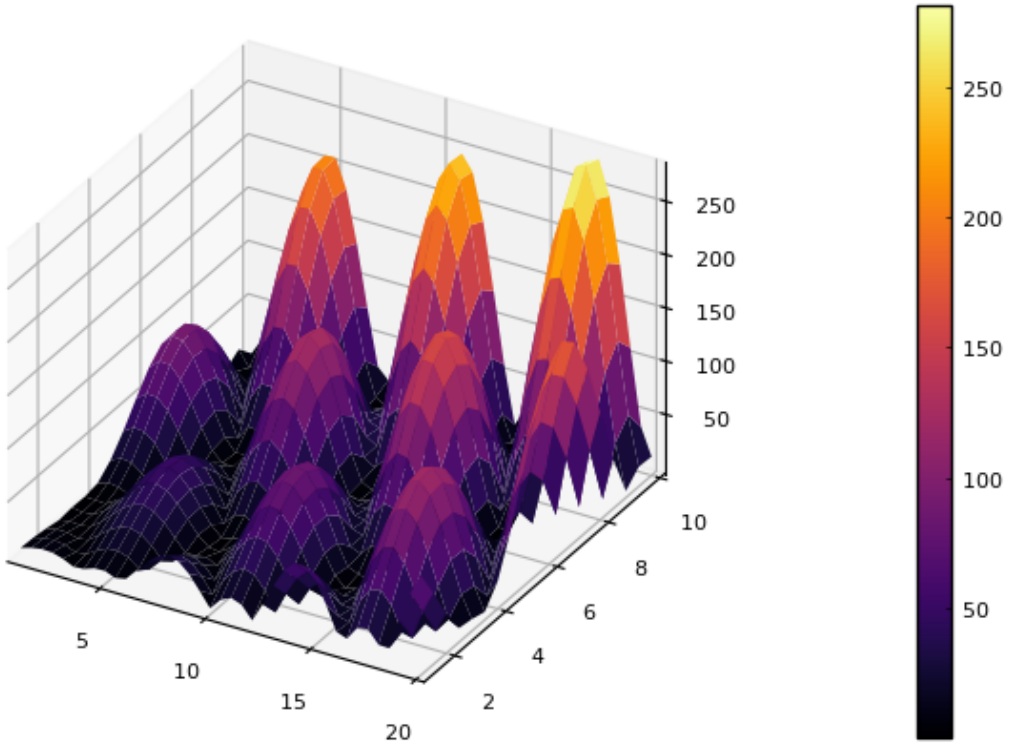
Out[55]:



5.2.9. Линии уровня

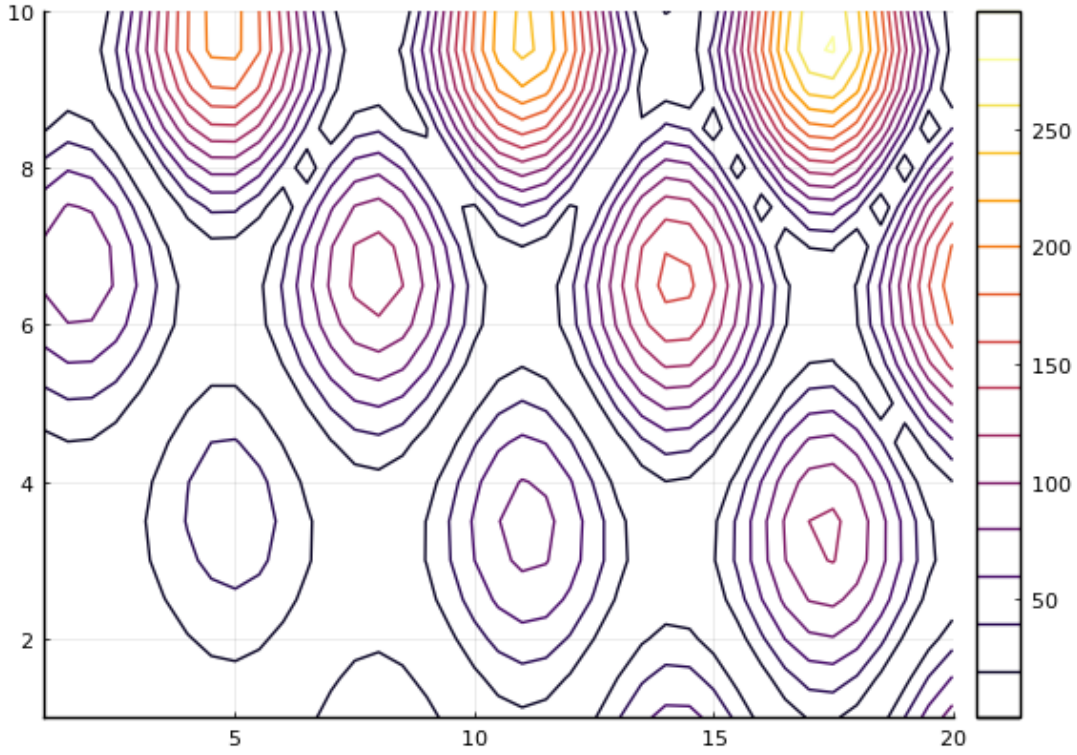
In [56]:
```
x = 1:0.5:20
y = 1:0.5:10
g(x, y) = (3x + y ^ 2) * abs(sin(x) + cos(y))
plot(x,y,g,
linetype = :surface,
)
```

Out[56]:



In [57]:
```
contour(x, y, g)
```

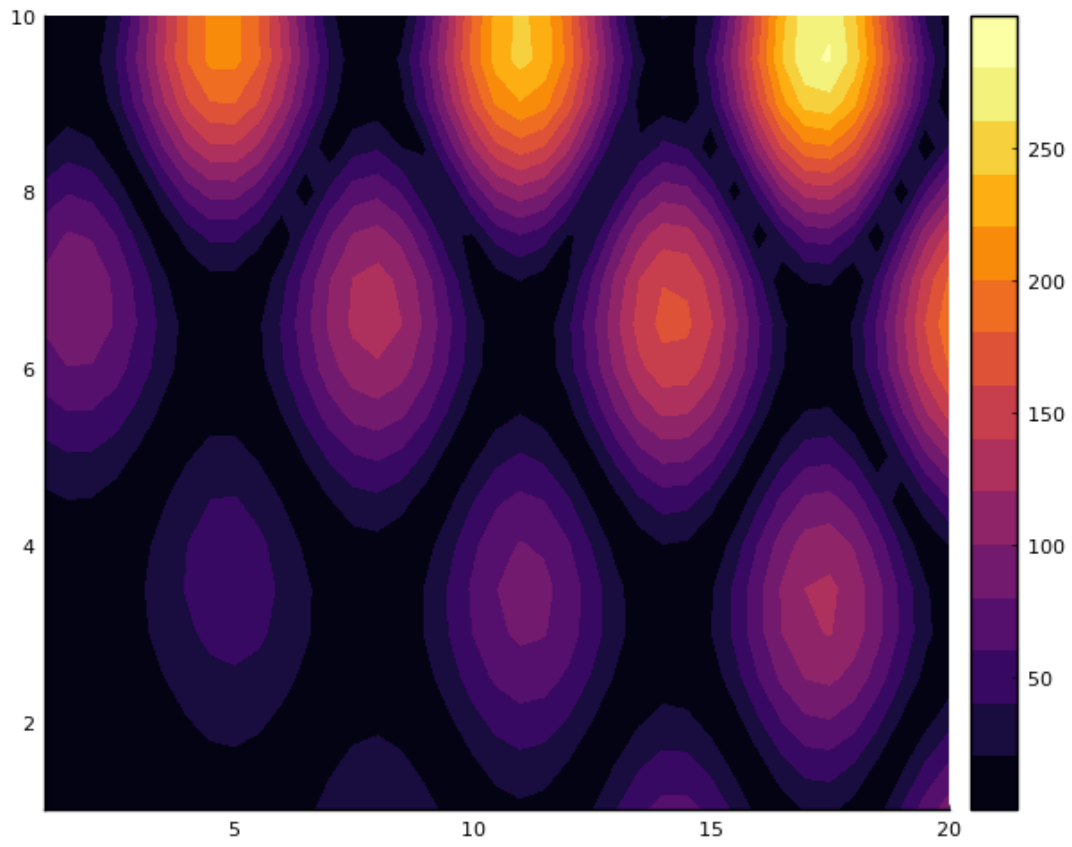
Out[57]:




```
sys:1: UserWarning: The following kwargs were not used by contour:
'label'
```

In [58]:
```
p = contour(x, y, g,
fill=true)
plot(p)
```

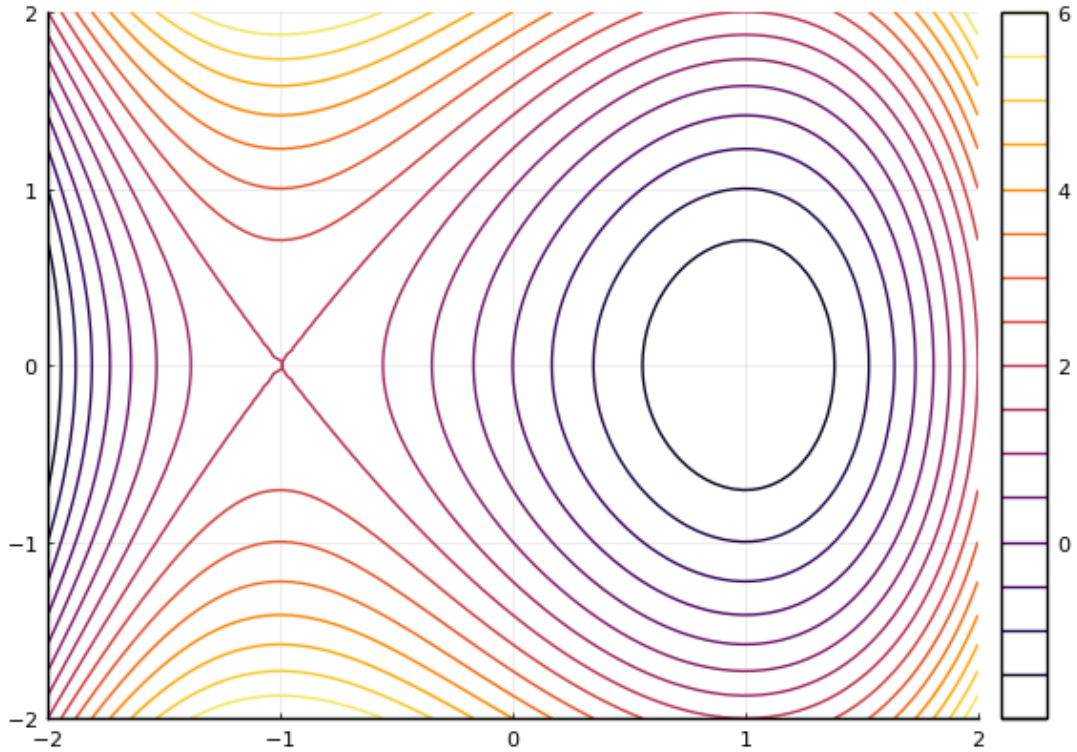
Out[58]:



5.2.10. Векторные поля

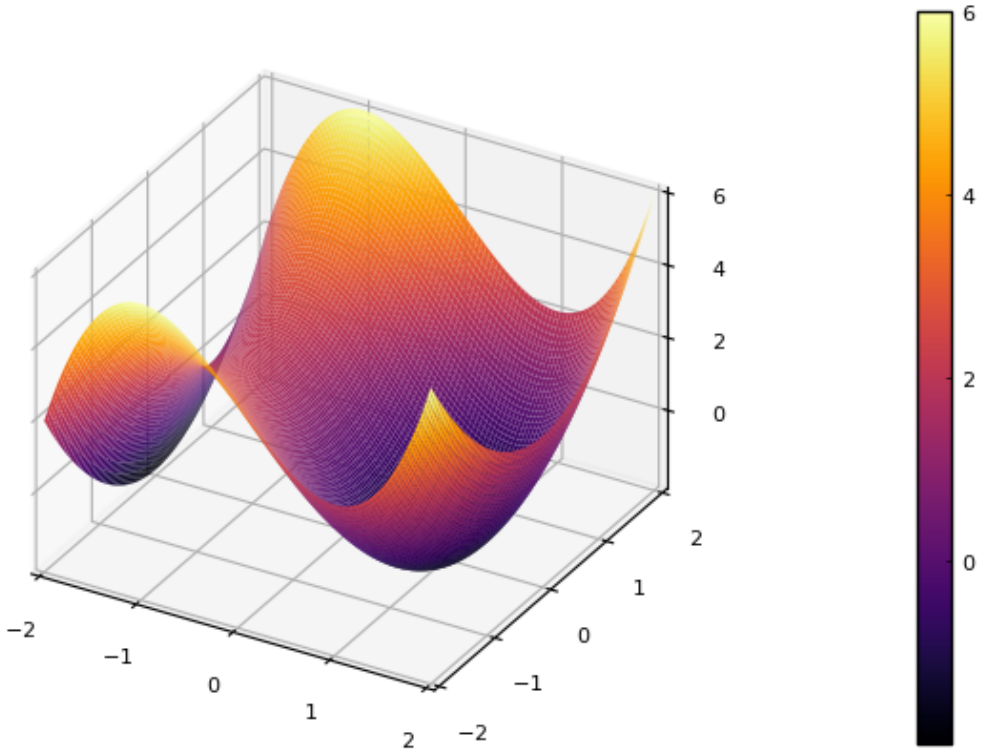
In [59]:
```
# построение линий уровня:
contour(X, Y, h)
```

Out[59]:



In [60]:
```
# 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
)
```

Out[60]:

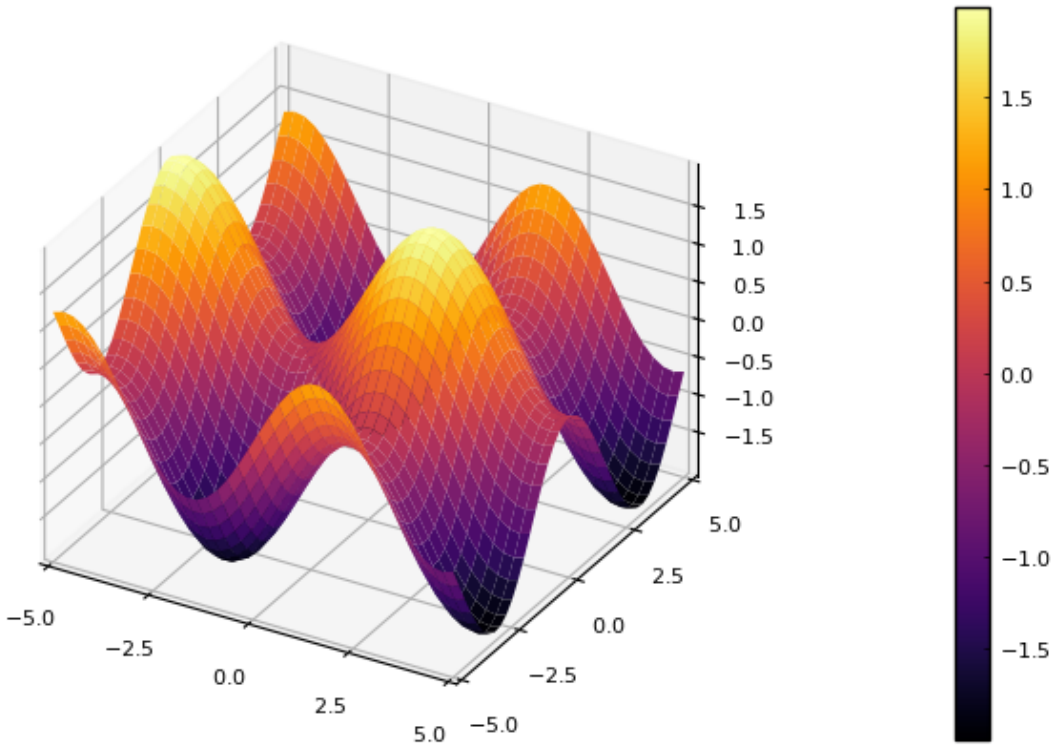


5.2.11. Анимация

Технически анимированное изображение представляет собой несколько наложенных изображений (или построенных в разных точках графиках) в одном файле. **5.2.11.1. Gif-анимация **

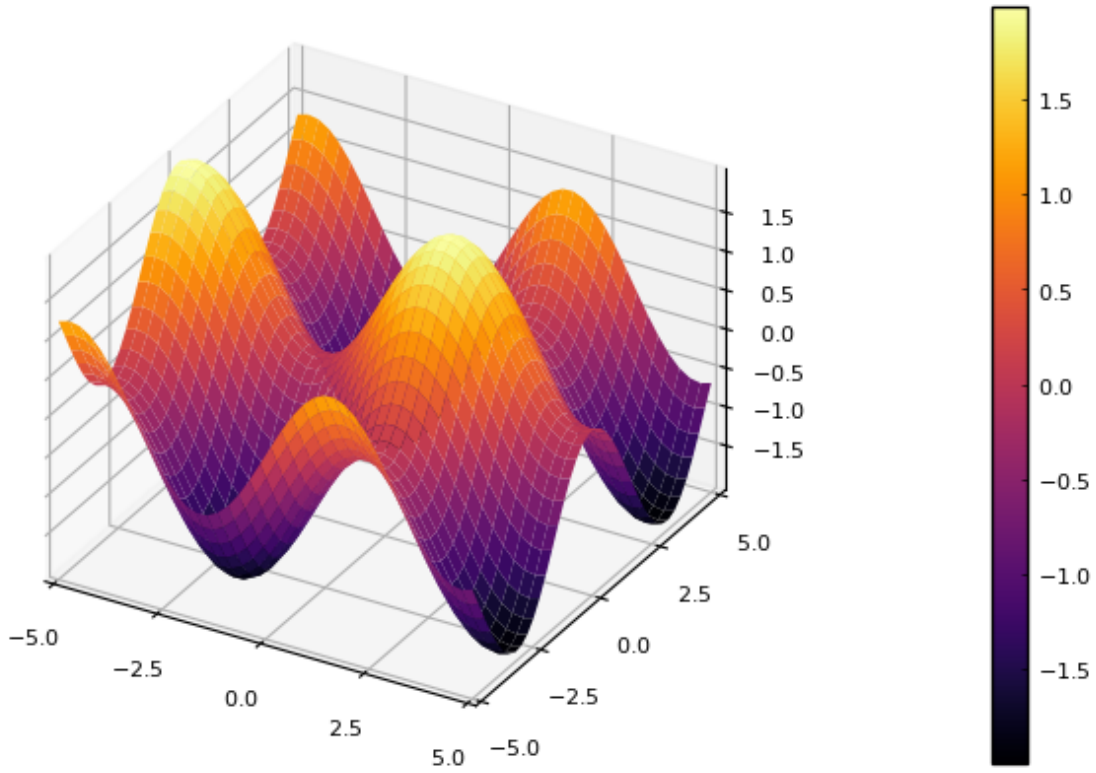
In [64]:
```
# построение поверхности:
i = 0
X = Y = range(-5,stop=5,length=40)
surface(X, Y, (x,y) -> sin(x+10sin(i))+cos(y))
```

Out[64]:

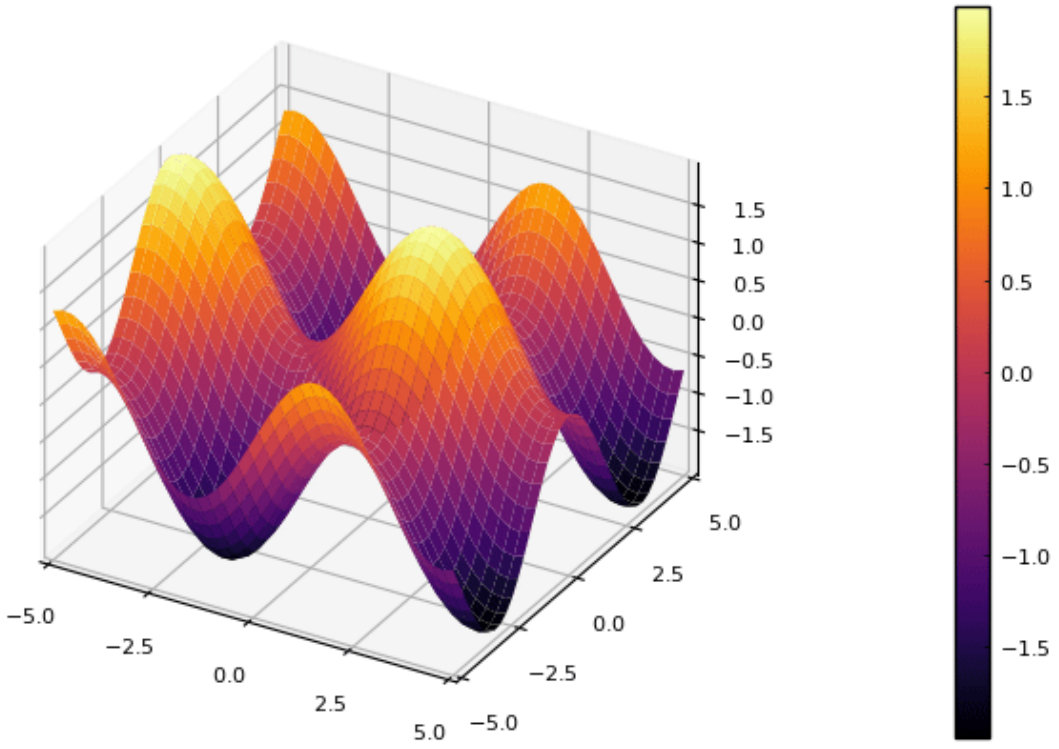


In [65]:
```
X = Y = range(-5,stop=5,length=40)
@gif for i in range(0,stop=2π,length=100)
surface(X, Y, (x,y) -> sin(x+10sin(i))+cos(y))
end
```

```
[ Info: Saved animation to /home/marc/Desktop/tmp.gif
```



Out[65]:



5.2.12. Errorbars

In [71]:
```
# подключение пакета Statistics:
import Pkg
Pkg.add("Statistics")
using Statistics
```

```
   Resolving package versions...
    Updating `~/.julia/environments/v1.9/Project.toml`
  [10745b16] + Statistics v1.9.0
  No Changes to `~/.julia/environments/v1.9/Manifest.toml`
```

In [72]:
```
sds = [1, 1/2, 1/4, 1/8, 1/16, 1/32]
```

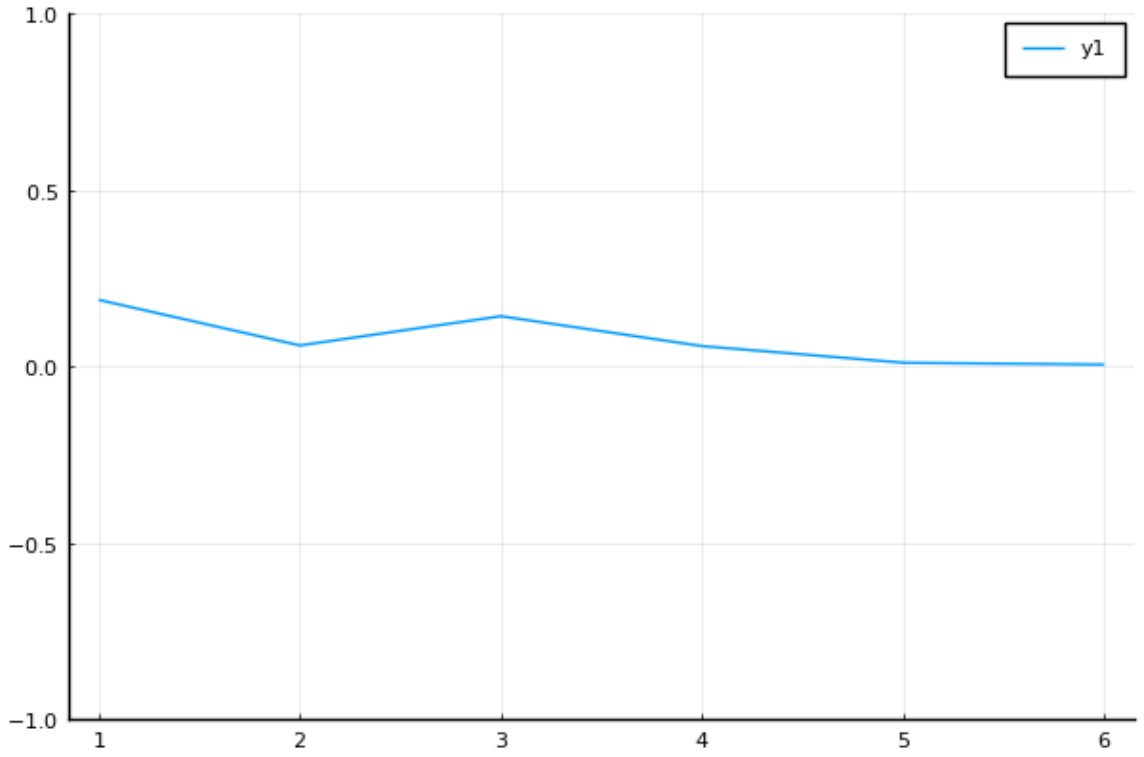
Out[72]:
```
6-element Vector{Float64}:
 1.0
 0.5
 0.25
 0.125
 0.0625
 0.03125
```

In [73]:
```
n = 10
y = [mean(sd*randn(n)) for sd in sds]
errs = 1.96 * sds / sqrt(n)
```

Out[73]:
```
6-element Vector{Float64}:
 0.6198064213930023
 0.3099032106965012
 0.1549516053482506
 0.0774758026741253
 0.03873790133706265
 0.019368950668531323
```

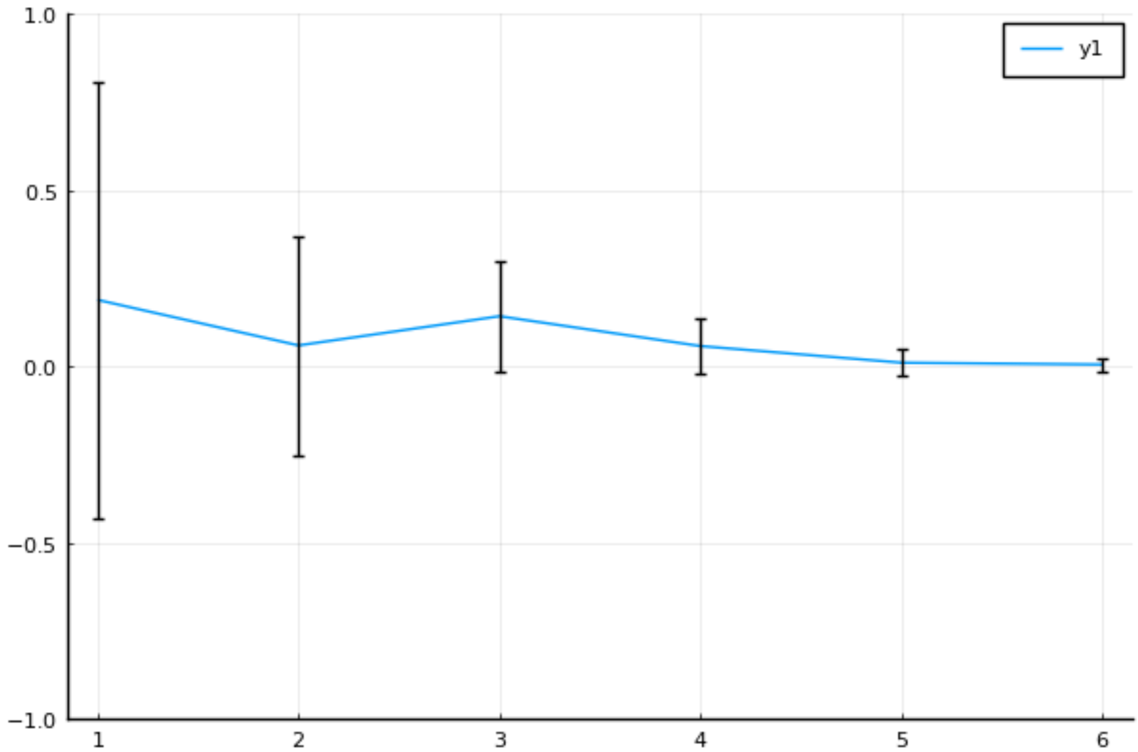
In [74]:
```
plot(y,
ylims = (-1,1),
)
```

Out[74]:



In [75]:
```
plot(y,
ylims = (-1,1),
err = errs
)
```

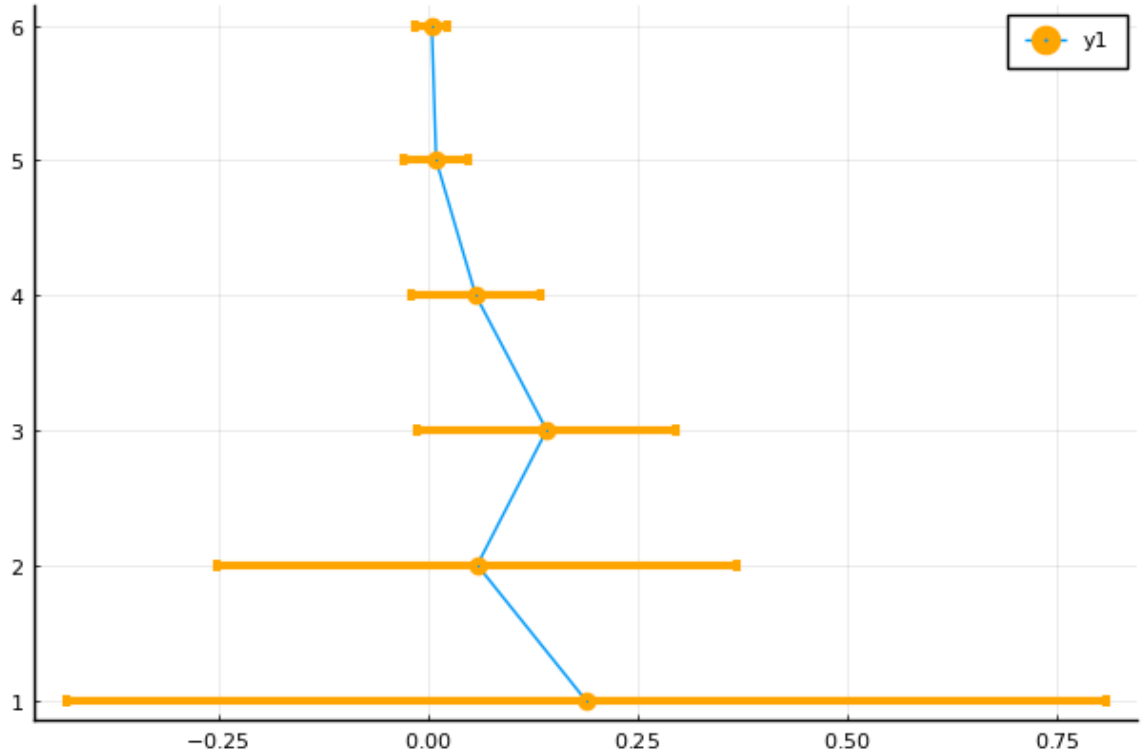
Out[75]:




```
sys:1: UserWarning: You passed a edgecolor/edgecolors ((0.0, 0.0,
0.0, 1.0)) for an unfilled marker ('_').  Matplotlib is ignoring th
e edgecolor in favor of the facecolor.  This behavior may change in
the future.
sys:1: UserWarning: You passed a edgecolor/edgecolors ((0.0, 0.0,
0.0, 1.0)) for an unfilled marker ('_').  Matplotlib is ignoring th
e edgecolor in favor of the facecolor.  This behavior may change in
the future.
sys:1: UserWarning: You passed a edgecolor/edgecolors ((0.0, 0.0,
0.0, 1.0)) for an unfilled marker ('_').  Matplotlib is ignoring th
e edgecolor in favor of the facecolor.  This behavior may change in
the future.
sys:1: UserWarning: You passed a edgecolor/edgecolors ((0.0, 0.0,
0.0, 1.0)) for an unfilled marker ('_').  Matplotlib is ignoring th
e edgecolor in favor of the facecolor.  This behavior may change in
the future.
```

In [76]:
```
plot(y, 1:length(y),
xerr = errs,
marker = stroke(3,:orange)
)
```

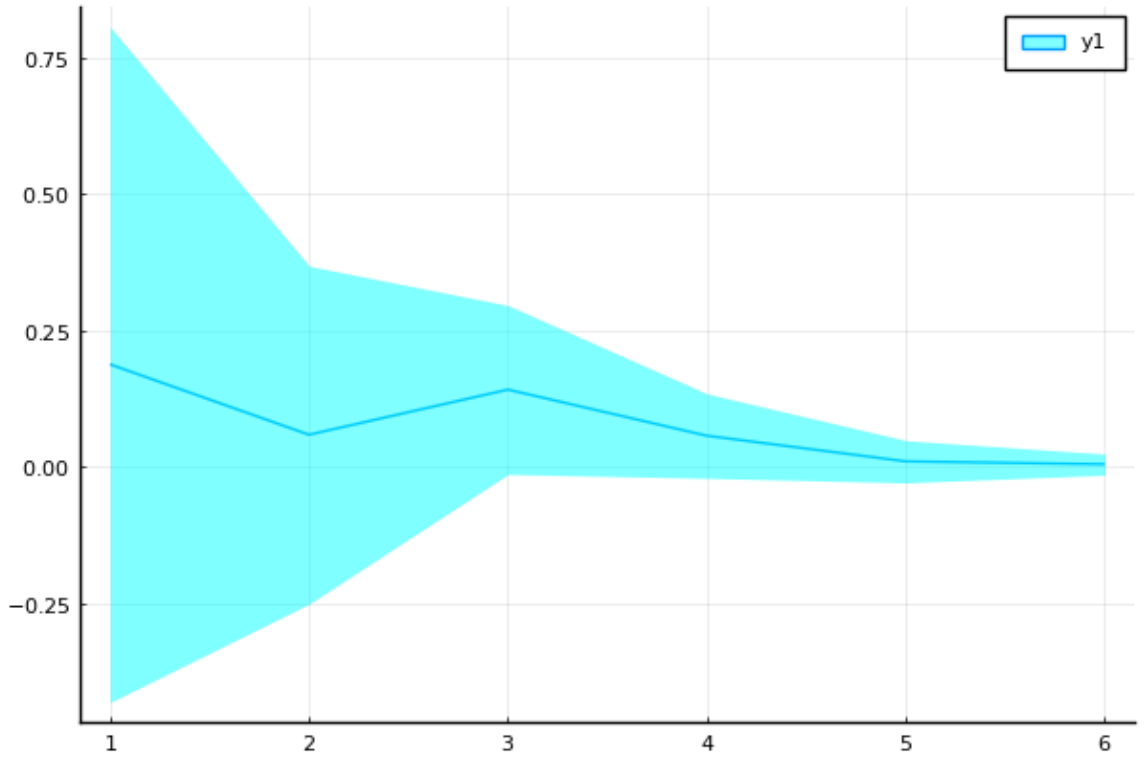
Out[76]:




```
sys:1: UserWarning: You passed a edgecolor/edgecolors ((1.0, 0.6470588235294118, 0.0, 1.0)) for an unfilled marker ('|'). Matplotlib is ignoring the edgecolor in favor of the facecolor. This behavior may change in the future.
sys:1: UserWarning: You passed a edgecolor/edgecolors ((1.0, 0.6470588235294118, 0.0, 1.0)) for an unfilled marker ('|'). Matplotlib is ignoring the edgecolor in favor of the facecolor. This behavior may change in the future.
sys:1: UserWarning: You passed a edgecolor/edgecolors ((1.0, 0.6470588235294118, 0.0, 1.0)) for an unfilled marker ('|'). Matplotlib is ignoring the edgecolor in favor of the facecolor. This behavior may change in the future.
sys:1: UserWarning: You passed a edgecolor/edgecolors ((1.0, 0.6470588235294118, 0.0, 1.0)) for an unfilled marker ('|'). Matplotlib is ignoring the edgecolor in favor of the facecolor. This behavior may change in the future.
```

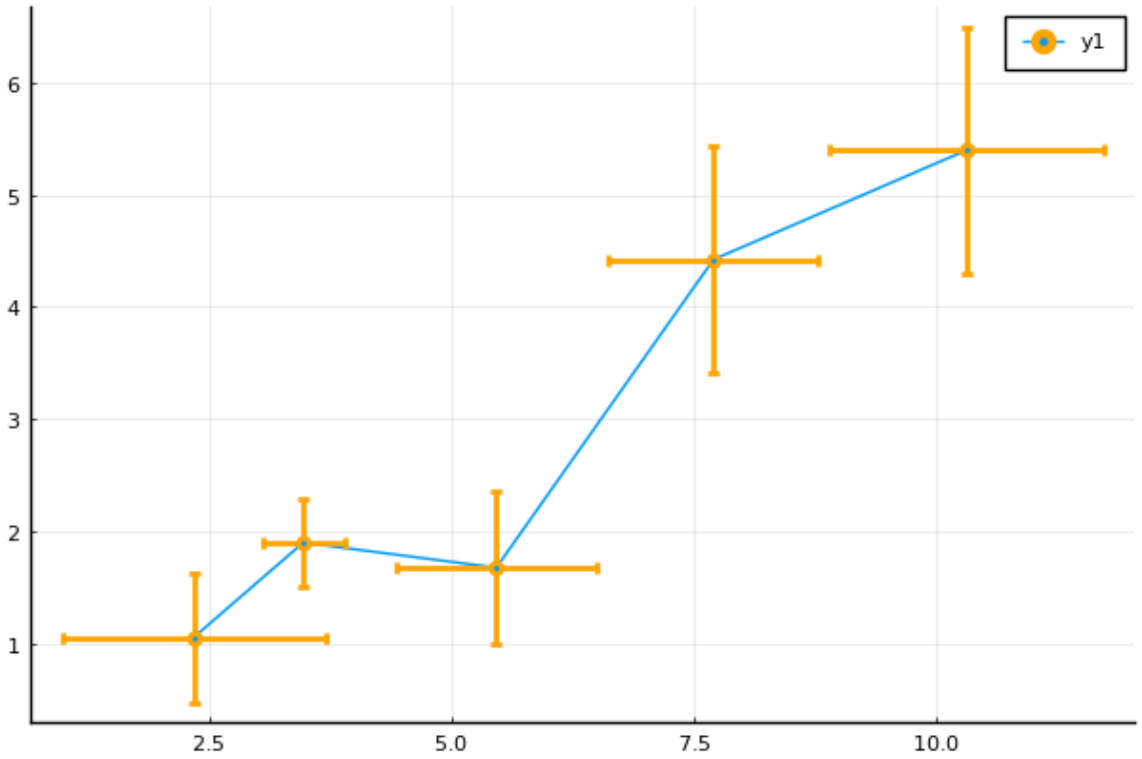
In [77]:
```
plot(y,
ribbon=errs,
fill=:cyan
)
```

Out[77]:



In [78]:
```
n = 10
x = [(rand()+1) .* randn(n) .+ 2i for i in 1:5]
y = [(rand()+1) .* randn(n) .+ i for i in 1:5]
f(v) = 1.96std(v) / sqrt(n)
xerr = map(f, x)
yerr = map(f, y)
x = map(mean, x)
y = map(mean, y)
plot(x, y,
xerr = xerr,
yerr = yerr,
marker = stroke(2, :orange)
)
```

Out[78]:

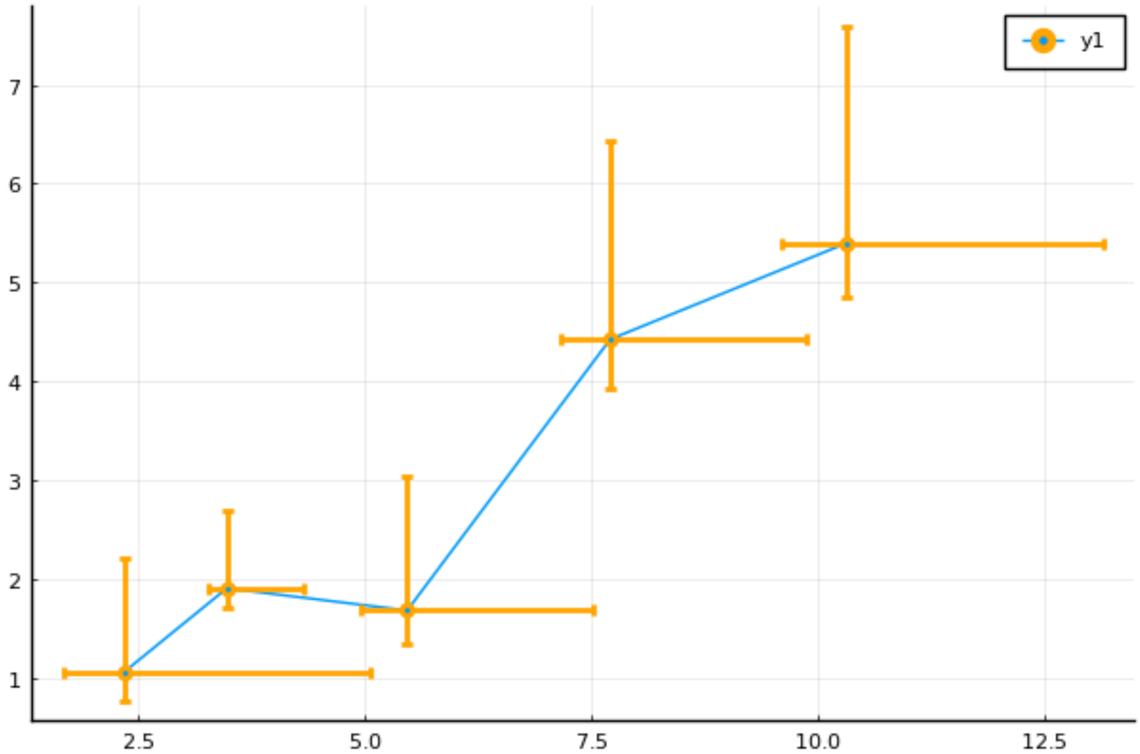



```
sys:1: UserWarning: You passed a edgecolor/edgecolors ((1.0, 0.6470
588235294118, 0.0, 1.0)) for an unfilled marker ('|').  Matplotlib
is ignoring the edgecolor in favor of the facecolor.  This behavior
may change in the future.
sys:1: UserWarning: You passed a edgecolor/edgecolors ((1.0, 0.6470
588235294118, 0.0, 1.0)) for an unfilled marker ('_').  Matplotlib
is ignoring the edgecolor in favor of the facecolor.  This behavior
may change in the future.
sys:1: UserWarning: You passed a edgecolor/edgecolors ((1.0, 0.6470
588235294118, 0.0, 1.0)) for an unfilled marker ('|').  Matplotlib
is ignoring the edgecolor in favor of the facecolor.  This behavior
may change in the future.
sys:1: UserWarning: You passed a edgecolor/edgecolors ((1.0, 0.6470
588235294118, 0.0, 1.0)) for an unfilled marker ('_').  Matplotlib
is ignoring the edgecolor in favor of the facecolor.  This behavior
may change in the future.
sys:1: UserWarning: You passed a edgecolor/edgecolors ((1.0, 0.6470
588235294118, 0.0, 1.0)) for an unfilled marker ('|').  Matplotlib
is ignoring the edgecolor in favor of the facecolor.  This behavior
may change in the future.
sys:1: UserWarning: You passed a edgecolor/edgecolors ((1.0, 0.6470
588235294118, 0.0, 1.0)) for an unfilled marker ('_').  Matplotlib
```

```
is ignoring the edgecolor in favor of the facecolor.  This behavior
may change in the future.
sys:1: UserWarning: You passed a edgecolor/edgecolors ((1.0, 0.6470
588235294118, 0.0, 1.0)) for an unfilled marker ('|').  Matplotlib
is ignoring the edgecolor in favor of the facecolor.  This behavior
may change in the future.
sys:1: UserWarning: You passed a edgecolor/edgecolors ((1.0, 0.6470
588235294118, 0.0, 1.0)) for an unfilled marker ('_').  Matplotlib
is ignoring the edgecolor in favor of the facecolor.  This behavior
may change in the future
```

In [79]:
```
plot(x, y,
xerr = (0.5xerr,2xerr),
yerr = (0.5yerr,2yerr),
marker = stroke(2, :orange)
)
```

Out[79]:




```
sys:1: UserWarning: You passed a edgecolor/edgecolors ((1.0, 0.6470
588235294118, 0.0, 1.0)) for an unfilled marker ('|').  Matplotlib
is ignoring the edgecolor in favor of the facecolor.  This behavior
may change in the future.
sys:1: UserWarning: You passed a edgecolor/edgecolors ((1.0, 0.6470
588235294118, 0.0, 1.0)) for an unfilled marker ('_').  Matplotlib
is ignoring the edgecolor in favor of the facecolor.  This behavior
may change in the future.
sys:1: UserWarning: You passed a edgecolor/edgecolors ((1.0, 0.6470
588235294118, 0.0, 1.0)) for an unfilled marker ('|').  Matplotlib
is ignoring the edgecolor in favor of the facecolor.  This behavior
may change in the future.
sys:1: UserWarning: You passed a edgecolor/edgecolors ((1.0, 0.6470
588235294118, 0.0, 1.0)) for an unfilled marker ('_').  Matplotlib
is ignoring the edgecolor in favor of the facecolor.  This behavior
may change in the future.
sys:1: UserWarning: You passed a edgecolor/edgecolors ((1.0, 0.6470
588235294118, 0.0, 1.0)) for an unfilled marker ('|').  Matplotlib
is ignoring the edgecolor in favor of the facecolor.  This behavior
may change in the future.
sys:1: UserWarning: You passed a edgecolor/edgecolors ((1.0, 0.6470
588235294118, 0.0, 1.0)) for an unfilled marker ('_').  Matplotlib
is ignoring the edgecolor in favor of the facecolor.  This behavior
may change in the future.
sys:1: UserWarning: You passed a edgecolor/edgecolors ((1.0, 0.6470
588235294118, 0.0, 1.0)) for an unfilled marker ('|').  Matplotlib
is ignoring the edgecolor in favor of the facecolor.  This behavior
may change in the future.
sys:1: UserWarning: You passed a edgecolor/edgecolors ((1.0, 0.6470
588235294118, 0.0, 1.0)) for an unfilled marker ('_').  Matplotlib
```

```
is ignoring the edgecolor in favor of the facecolor.  This behavior
may change in the future
```

5.2.13. Использование пакета Distributions

In [81]:
```
import Pkg
Pkg.add("Distributions")
using Distributions
```

```
   Resolving package versions...
   Installed Rmath_jll ──────────────── v0.4.0+0
   Installed OpenSpecFun_jll ────────── v0.5.5+0
   Installed Calculus ───────────────── v0.5.1
   Installed PDMats ─────────────────── v0.11.31
   Installed DualNumbers ────────────── v0.6.8
   Installed StatsFuns ──────────────── v1.3.0
   Installed HypergeometricFunctions ─ v0.3.23
   Installed SpecialFunctions ────────── v2.3.1
   Installed Rmath ──────────────────── v0.7.1
   Installed FillArrays ─────────────── v1.9.2
   Installed QuadGK ─────────────────── v2.9.1
   Installed Distributions ──────────── v0.25.104
    Updating `~/.julia/environments/v1.9/Project.toml`
  [31c24e10] + Distributions v0.25.104
    Updating `~/.julia/environments/v1.9/Manifest.toml`
  [49dc2e85] + Calculus v0.5.1
  [31c24e10] + Distributions v0.25.104
  [fa6b7ba4] + DualNumbers v0.6.8
  [1a297f60] + FillArrays v1.9.2
  [34004b35] + HypergeometricFunctions v0.3.23
  [90014a1f] + PDMats v0.11.31
  [1fd47b50] + QuadGK v2.9.1
  [79098fc4] + Rmath v0.7.1
  [276daf66] + SpecialFunctions v2.3.1
  [4c63d2b9] + StatsFuns v1.3.0
  [efe28fd5] + OpenSpecFun_jll v0.5.5+0
  [f50d1b31] + Rmath_jll v0.4.0+0
  [4607b0f0] + SuiteSparse
Precompiling project...
  ✓ PDMats
  ✓ Calculus
  ✓ FillArrays
  ✓ Rmath_jll
  ✓ OpenSpecFun_jll
  ✓ QuadGK
  ✓ FillArrays → FillArraysSparseArraysExt
  ✓ FillArrays → FillArraysPDMatsExt
  ✓ FillArrays → FillArraysStatisticsExt
  ✓ Rmath
  ✓ SpecialFunctions
  ✓ ColorVectorSpace → SpecialFunctionsExt
  ✓ DualNumbers
  ✓ HypergeometricFunctions
  ✓ StatsFuns
  ✓ ColorSchemes
  ✓ Distributions
  ✓ PlotlyBase
  ✓ Distributions → DistributionsTestExt
  ✓ PlotlyJS
  ✓ PlotUtils
  ✓ Plotly
  ✓ PlotThemes
  ✓ RecipesPipeline
  ✓ UnicodePlots
```

```
  ✓ UnicodePlots → UnitfulExt
  ✓ Plots
  ✓ Plots → IJuliaExt
  ✓ Plots → UnitfulExt
  29 dependencies successfully precompiled in 580 seconds. 171 alre
ady precompiled.
  6 dependencies precompiled but different versions are currently l
oaded. Restart julia to access the new versions
```

In [82]:
```
pyplot()
```

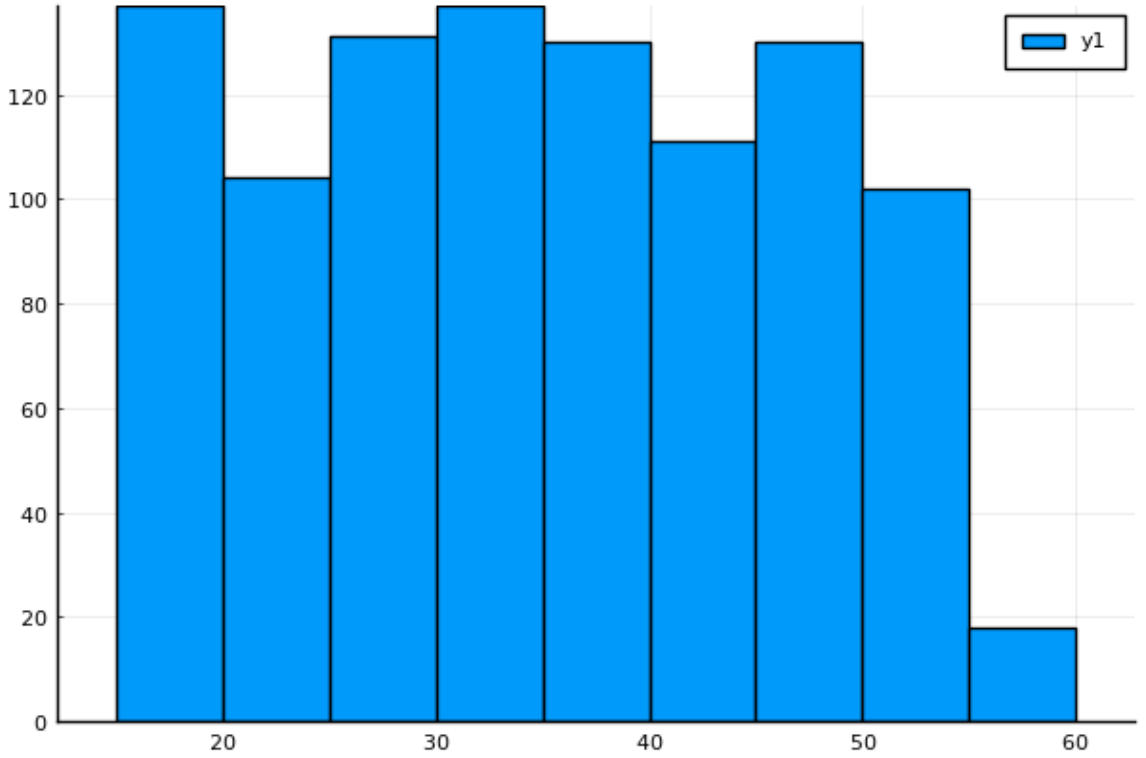
Out[82]:
```
Plots.PyPlotBackend()
```

In [83]:
```
ages = rand(15:55,1000)
```

Out[83]:
```
1000-element Vector{Int64}:
 31
 50
 41
 40
 29
 46
 36
 17
 26
 34
 47
 27
 23
  ⋮
 27
 25
 54
 19
 47
 28
 37
 54
 16
 19
 42
 53
```

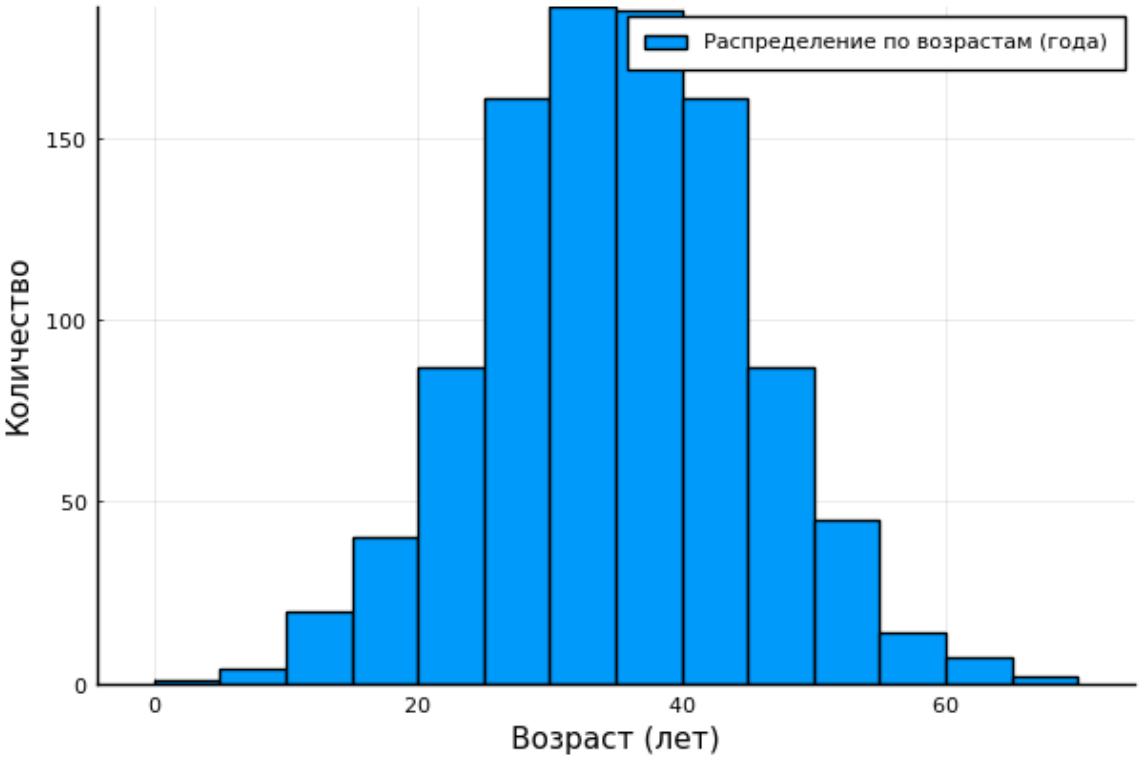
In [84]:
```
histogram(ages)
```

Out[84]:



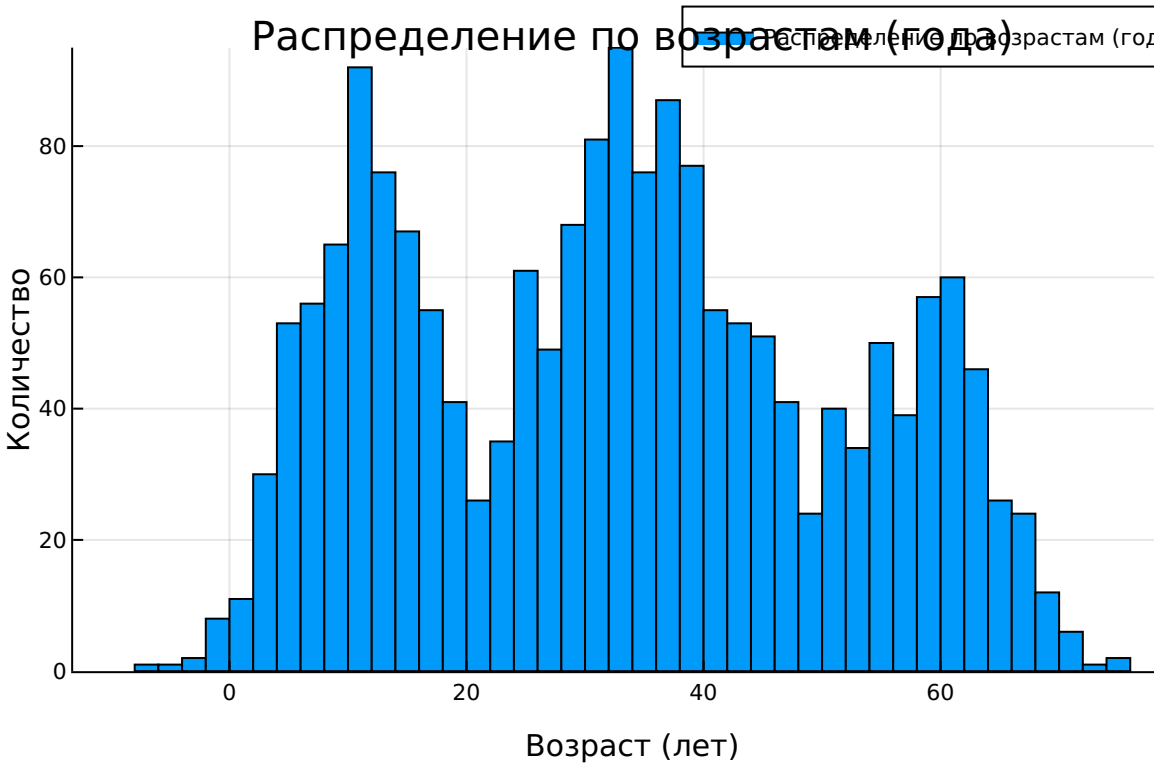
In [85]:
```
d=Normal(35.0,10.0)
ages = rand(d,1000)
histogram(
ages,
label="Распределение по возрастам (года)",
xlabel = "Возраст (лет)",
ylabel= "Количество"
)
```

Out[85]:



In [86]:
```
plotly()
d1=Normal(10.0,5.0);
d2=Normal(35.0,10.0);
d3=Normal(60.0,5.0);
N=1000;
ages = (Float64)[];
ages = append!(ages,rand(d1,Int64(ceil(N/2))));
ages = append!(ages,rand(d2,N));
ages = append!(ages,rand(d3,Int64(ceil(N/3))));
histogram(
ages,
bins=50,
label="Распределение по возрастам (года)",
xlabel = "Возраст (лет)",
ylabel= "Количество",
title = "Распределение по возрастам (года)"
)
```

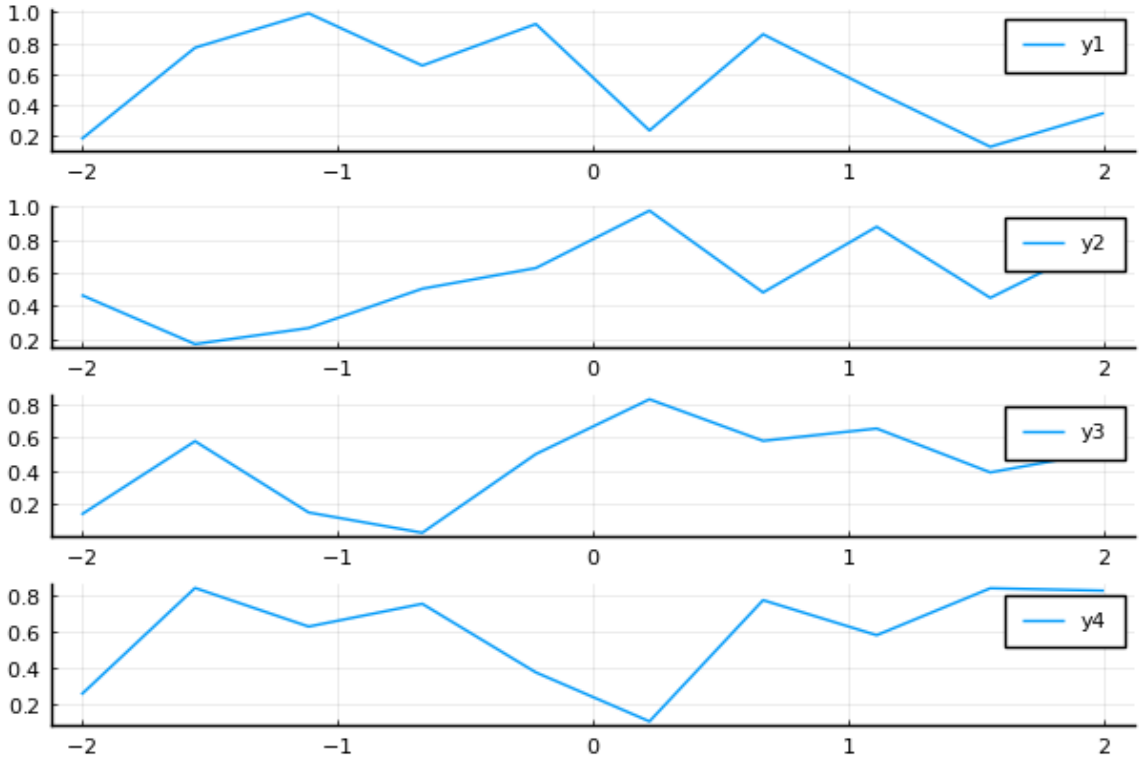
Out[86]:



5.2.14. Подграфики

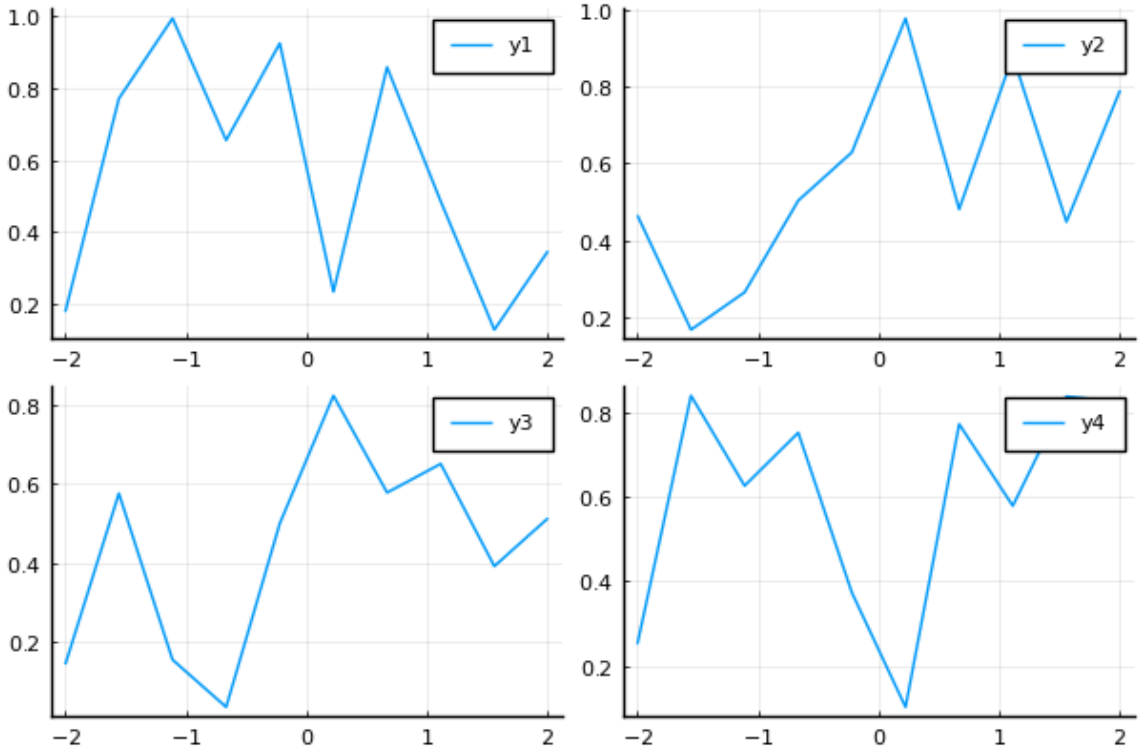
In [87]:
```
# подгружаем pyplot():
pyplot()
# построение серии графиков:
x=range(-2,2,length=10)
y = rand(10,4)
plot(x,y,
layout=(4,1)
)
```

Out[87]:



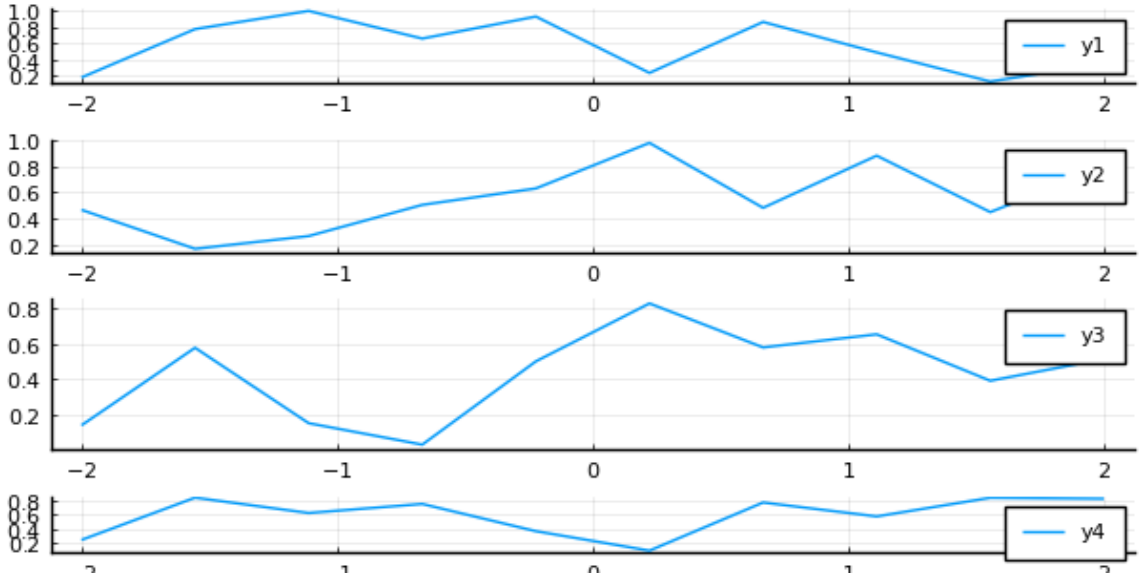
In [88]:
```
plot(x,y,
layout=4
)
```

Out[88]:



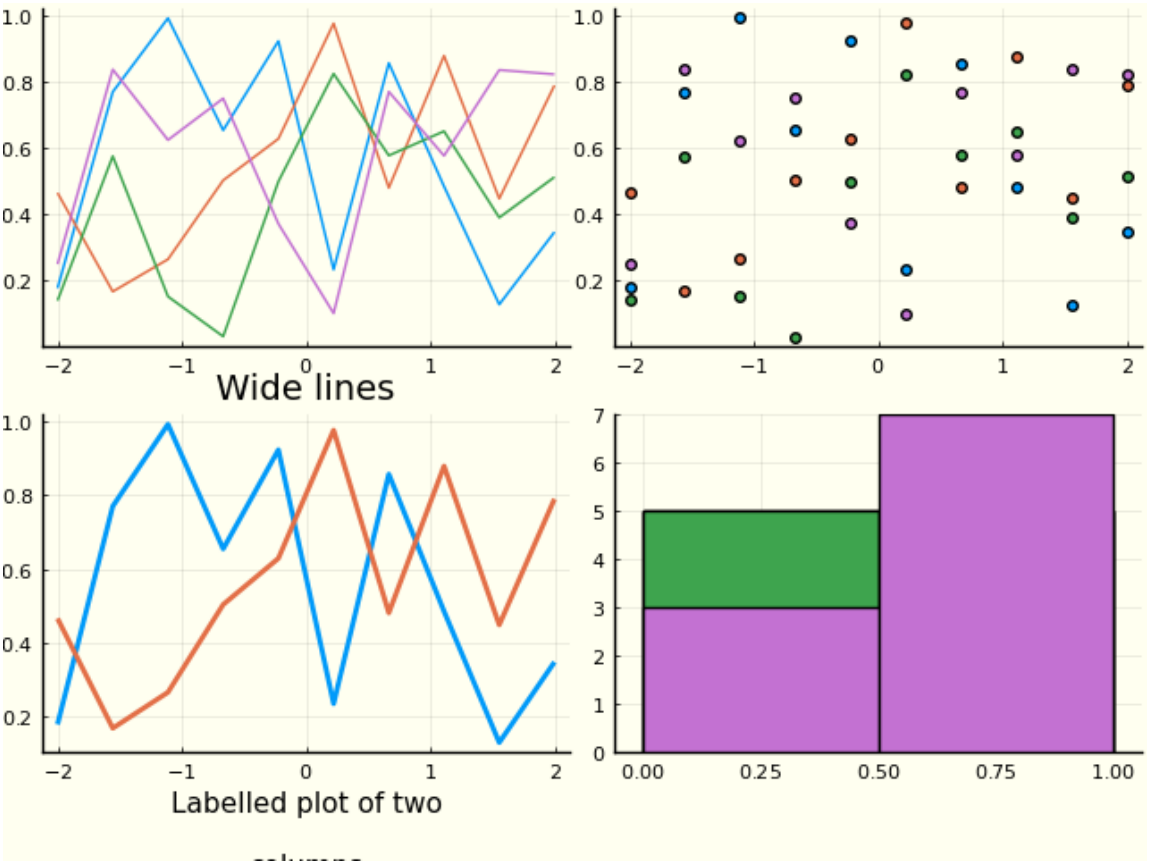
In [89]:
```
plot(x,y,
size=(600,300),
layout = grid(4,1,heights=[0.2,0.3,0.4,0.15])
)
```

Out[89]:



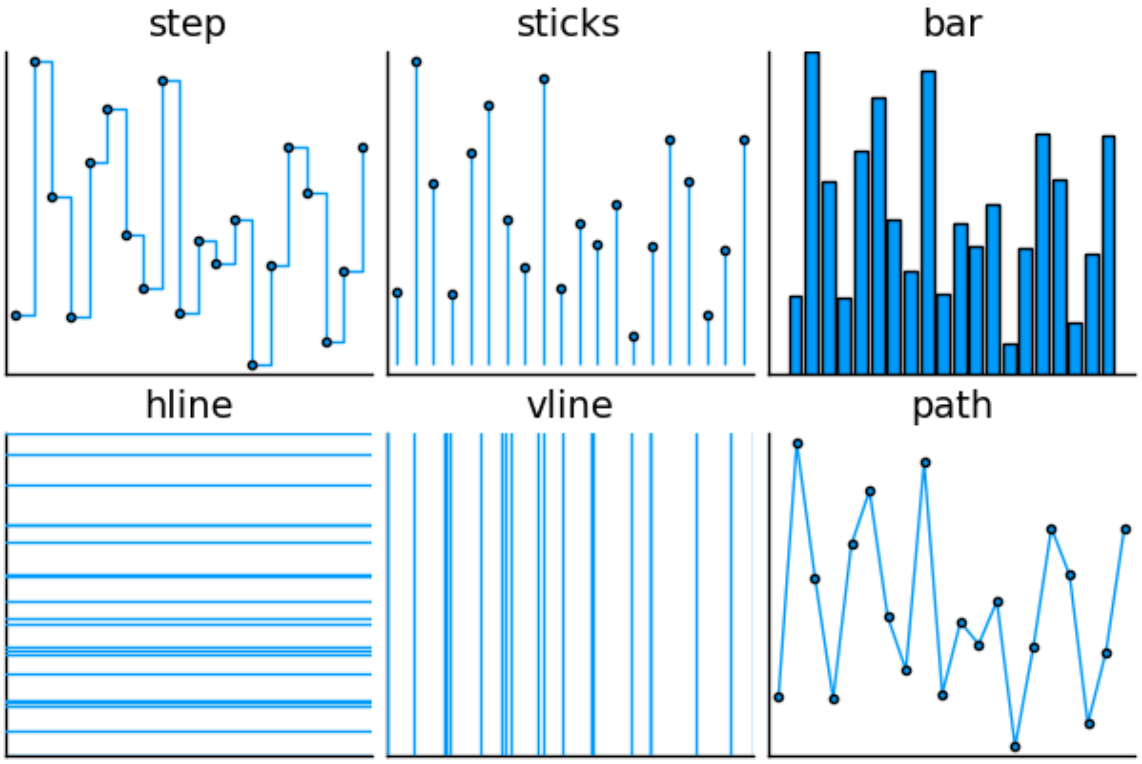
In [90]:
```
# график в виде линий:
p1 = plot(x,y)
# график в виде точек:
p2 = scatter(x,y)
# график в виде линий с оформлением:
p3 = plot(x,y[:,1:2],xlabel="Labelled plot of two

columns",lw=2,title="Wide lines")
# 4 гистограммы:
p4 = histogram(x,y)
plot(
p1,p2,p3,p4,
layout=(2,2),
legend=false,
size=(800,600),
background_color = :ivory
)
```

Out[90]:



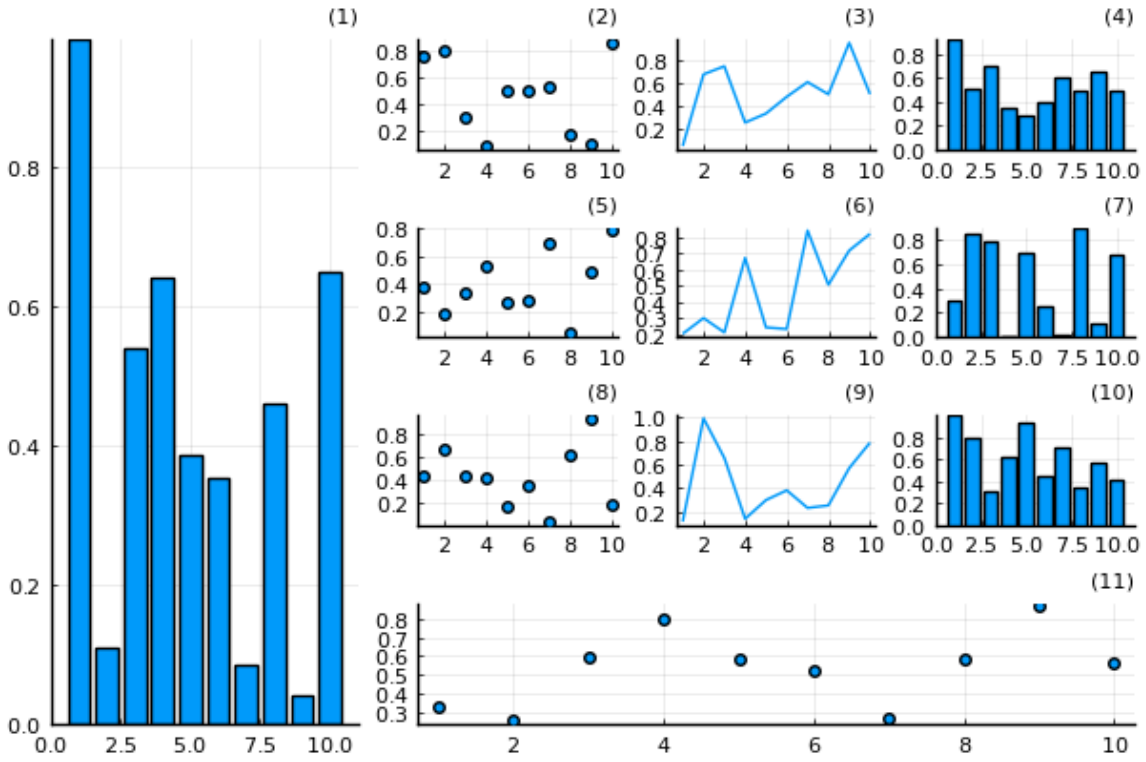
In [91]:
```
seriestypes = [:step, :sticks, :bar, :hline, :vline, :path]
titles =["step" "sticks" "bar" "hline" "vline" "path"]
plot(rand(20,1), st = seriestypes,
layout = (2,3),
ticks=nothing,
legend=false,
title=titles,
m=3
)
```

Out[91]:



In [92]:
```
l = @layout [
a{0.3w} [grid(3,3)
b{0.2h} ]]
plot(
rand(10,11),
layout = l, legend = false, seriestype = [:bar :scatter :path],
title = ["($i)" for j = 1:1, i=1:11], titleloc = :right, titlefont

= font(8)
)
```

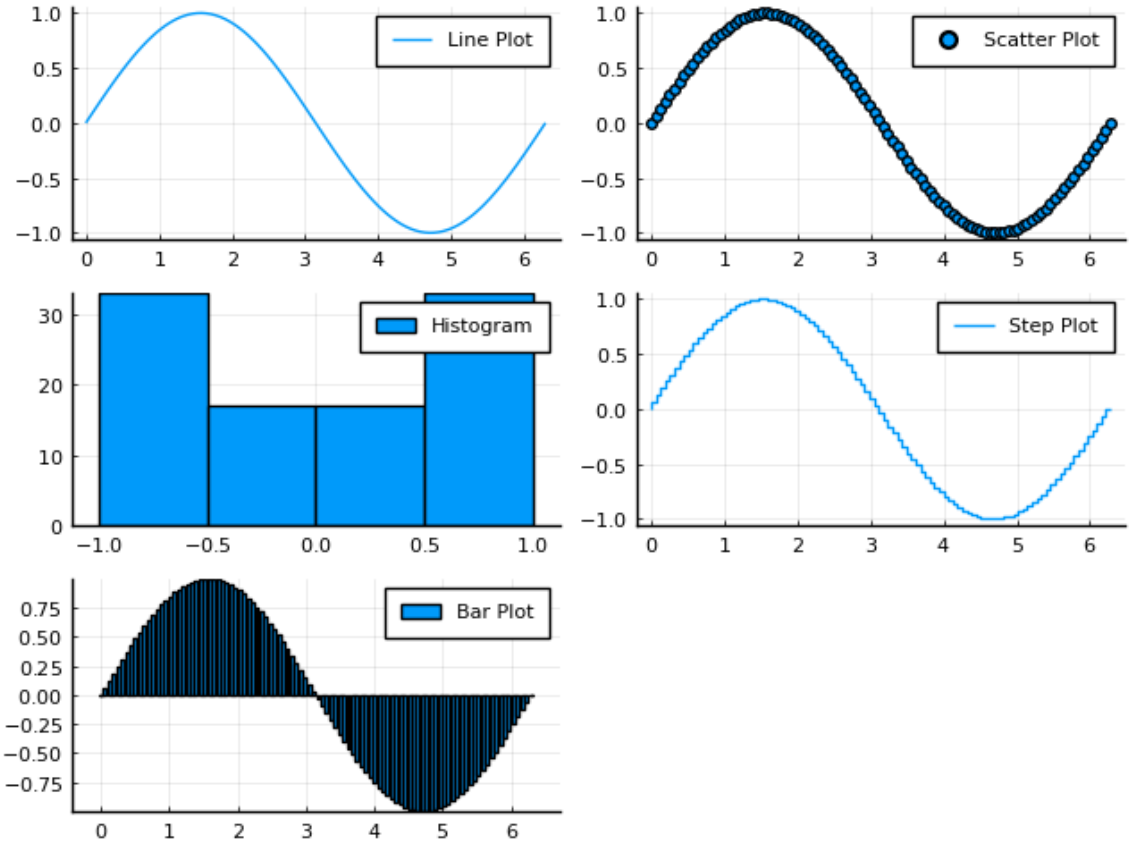
Out[92]:



5.4. Задания для самостоятельного выполнения

In [95]:
```
 using Plots
x_vals = range(0, stop=2π, length=100)
y_vals = sin.(x_vals)
p1 = plot(x_vals, y_vals, label="Line Plot")
p2 = scatter(x_vals, y_vals, label="Scatter Plot")
p3 = plot(x_vals, y_vals, st=:histogram, label="Histogram")
p4 = plot(x_vals, y_vals, st=:steppre, label="Step Plot")
p5 = plot(x_vals, y_vals, st=:bar, label="Bar Plot")
plot(p1, p2, p3, p4, p5, layout=(3, 2), legend=:topright)
```

Out[95]:

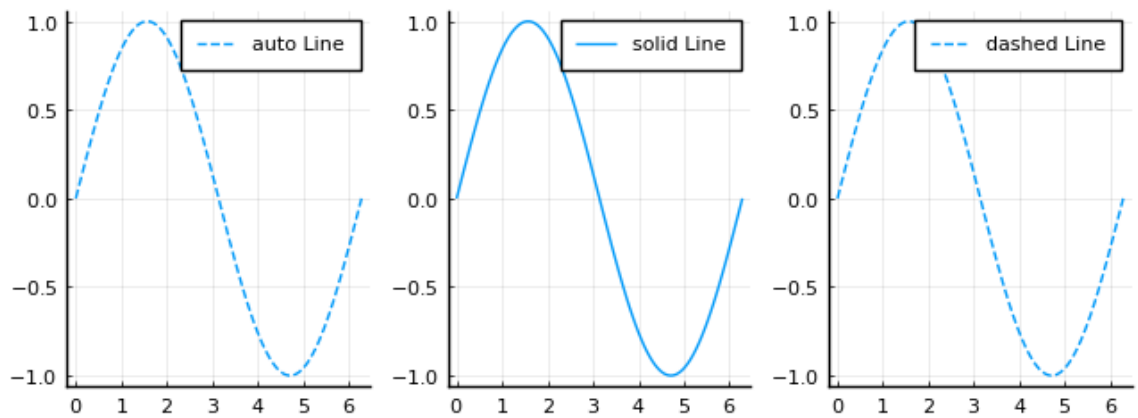


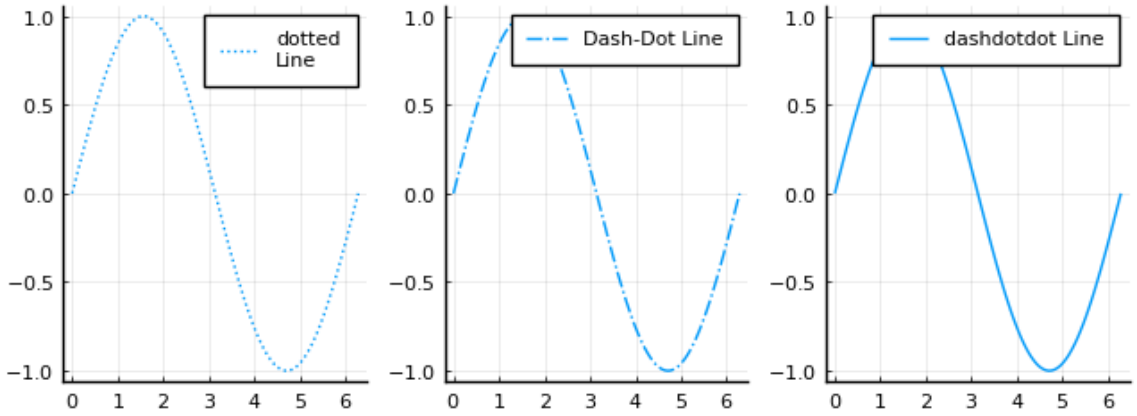
2.

In [97]:
```
using Plots
x_vals = range(0, stop=2π, length=100)
y_vals = sin.(x_vals)
plot_titles = ["auto Line", "solid Line", "dashed Line", "dotted
Line", "Dash-Dot Line", "dashdotdot Line"]
plot_styles = [
:auto,
   :solid,
:dash,
:dot,
:dashdot,
:dashdotdot
]
plot_arr = []
for (i, style) in enumerate(plot_styles)
pl = plot(x_vals, y_vals, label=plot_titles[i], linestyle=style)
push!(plot_arr, pl)
end
plot(plot_arr..., layout=(2, 3), legend=:topright)
```

```
┌ Warning: linestyle dashdotdot is unsupported with Plots.PyPlotBac
kend(). Choose from: [:auto, :dash, :dashdot, :dot, :solid]
└ @ Plots ~/.julia/packages/Plots/sxUvK/src/args.jl:1573
┌ Warning: Unknown linestyle dashdotdot
└ @ Plots ~/.julia/packages/Plots/sxUvK/src/backends/deprecated/pyp
lot.jl:106
┌ Warning: Unknown linestyle dashdotdot
└ @ Plots ~/.julia/packages/Plots/sxUvK/src/backends/deprecated/pyp
lot.jl:106
┌ Warning: Unknown linestyle dashdotdot
└ @ Plots ~/.julia/packages/Plots/sxUvK/src/backends/deprecated/pyp
lot.jl:106
┌ Warning: Unknown linestyle dashdotdot
└ @ Plots ~/.julia/packages/Plots/sxUvK/src/backends/deprecated/pyp
lot.jl:106
┌ Warning: Unknown linestyle dashdotdot
└ @ Plots ~/.julia/packages/Plots/sxUvK/src/backends/deprecated/pyp
lot.jl:106
┌ Warning: Unknown linestyle dashdotdot
└ @ Plots ~/.julia/packages/Plots/sxUvK/src/backends/deprecated/pyp
lot.jl:106
┌ Warning: Unknown linestyle dashdotdot
└ @ Plots ~/.julia/packages/Plots/sxUvK/src/backends/deprecated/pyp
lot.jl:106
┌ Warning: Unknown linestyle dashdotdot
└ @ Plots ~/.julia/packages/Plots/sxUvK/src/backends/deprecated/pyp
lot.jl:106
```

Out[97]:



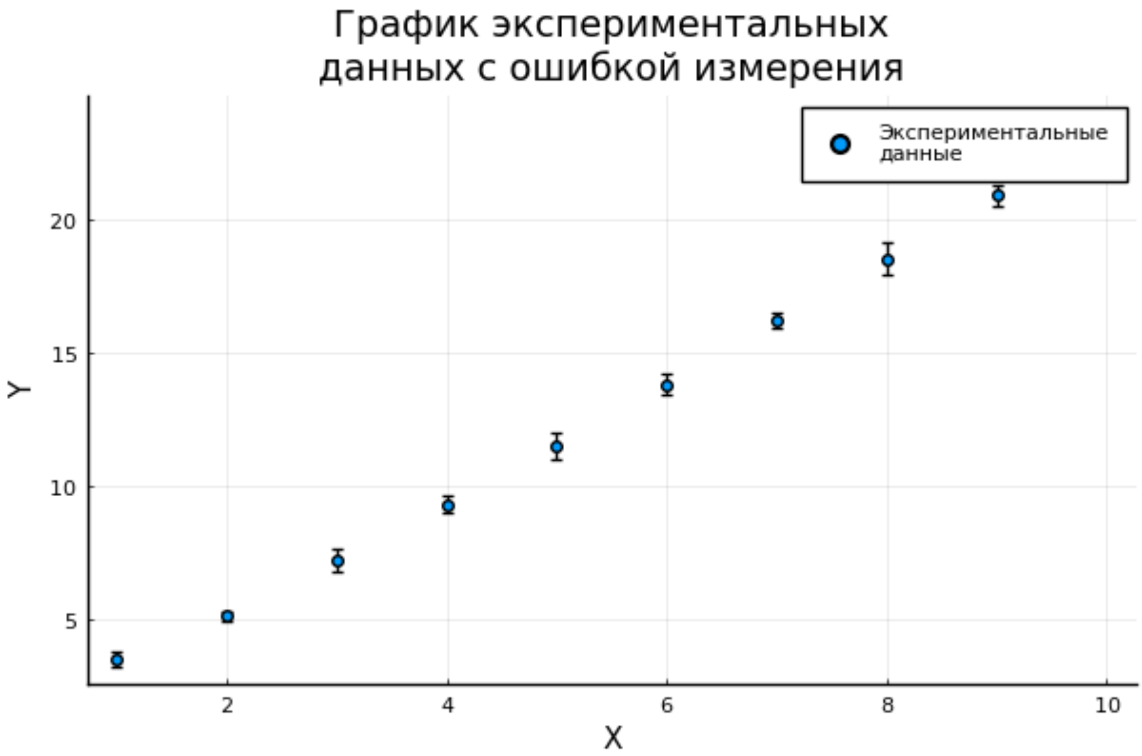


In [118]:
```
using Plots

# Задаем вектор x
x = [-2, -1, 0, 1, 2]

# Задаем функцию y(x) = x^3 - 3x
y(x) = x^3 - 3x

# Создаем графическое окно с 2x2 подокнами
plot_grid = plot(layout=(2,2), legend=false)

# Подокно 1: Точки
scatter!(plot_grid[1], x, y.(x), markersize=8, markercolor=:blue, lab

# Подокно 2: Линии
plot!(plot_grid[2], x, y.(x), line=:line, linewidth=2, color=:green,

# Подокно 3: Линии и точки
plot!(plot_grid[3], x, y.(x), line=:scatter, markersize=6, markercolo

# Подокно 4: Кривая
plot!(plot_grid[4], x, y.(x), line=:auto, linewidth=2, color=:purple,

# Настройка графика
title!(plot_grid, "Графики функции y(x) = x^3 - 3x")
xlabel!(plot_grid, "x")
ylabel!(plot_grid, "y")

# Сохраняем изображение в файл
savefig("figure_familiya.png")

# Отображаем график
display(plot_grid)
```

```
cannot define function y; it already has a value

Stacktrace:
 [1] top-level scope
   @ none:0
 [2] top-level scope
   @ In[118]:7
```

In [114]:
```
using Plots
# Придуманные данные для иллюстрации
x_data = 1:10
y_data = [3.5, 5.1, 7.2, 9.3, 11.5, 13.8, 16.2, 18.5, 20.9, 23.5]
# Создание ошибки для данных
error = [0.3, 0.2, 0.4, 0.3, 0.5, 0.4, 0.3, 0.6, 0.4, 0.5]
# Построение графика с ошибкой измерения
scatter(x_data, y_data, yerror=error, label="Экспериментальные
данные", xlabel="X", ylabel="Y", title="График экспериментальных
данных с ошибкой измерения")
```

Out[114]:



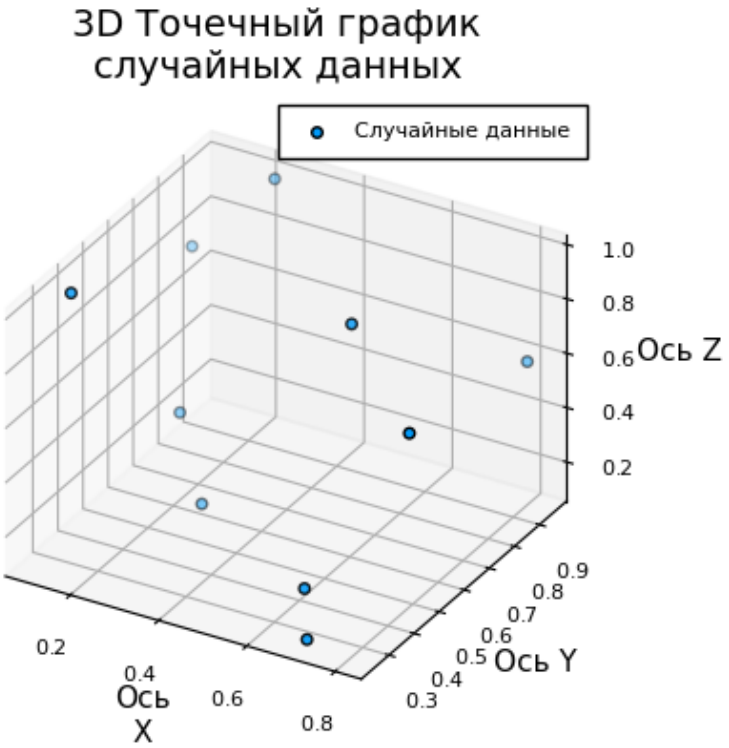

```
sys:1: UserWarning: You passed a edgecolor/edgecolors ((0.0, 0.0,
0.0, 1.0)) for an unfilled marker ('_').  Matplotlib is ignoring th
e edgecolor in favor of the facecolor.  This behavior may change in
the future.
sys:1: UserWarning: You passed a edgecolor/edgecolors ((0.0, 0.0,
0.0, 1.0)) for an unfilled marker ('_').  Matplotlib is ignoring th
e edgecolor in favor of the facecolor.  This behavior may change in
the future.
sys:1: UserWarning: You passed a edgecolor/edgecolors ((0.0, 0.0,
0.0, 1.0)) for an unfilled marker ('_').  Matplotlib is ignoring th
e edgecolor in favor of the facecolor.  This behavior may change in
the future.
sys:1: UserWarning: You passed a edgecolor/edgecolors ((0.0, 0.0,
0.0, 1.0)) for an unfilled marker ('_').  Matplotlib is ignoring th
e edgecolor in favor of the facecolor.  This behavior may change in
the future.
```

In [115]:
```
using Plots
# Генерация случайных данных для x и y
x_data = rand(10)
y_data = rand(10)
# Построение точечного графика
scatter(x_data, y_data, label="Случайные данные", xlabel="Ось X",
ylabel="Ось Y", title="Точечный график случайных данных")
```

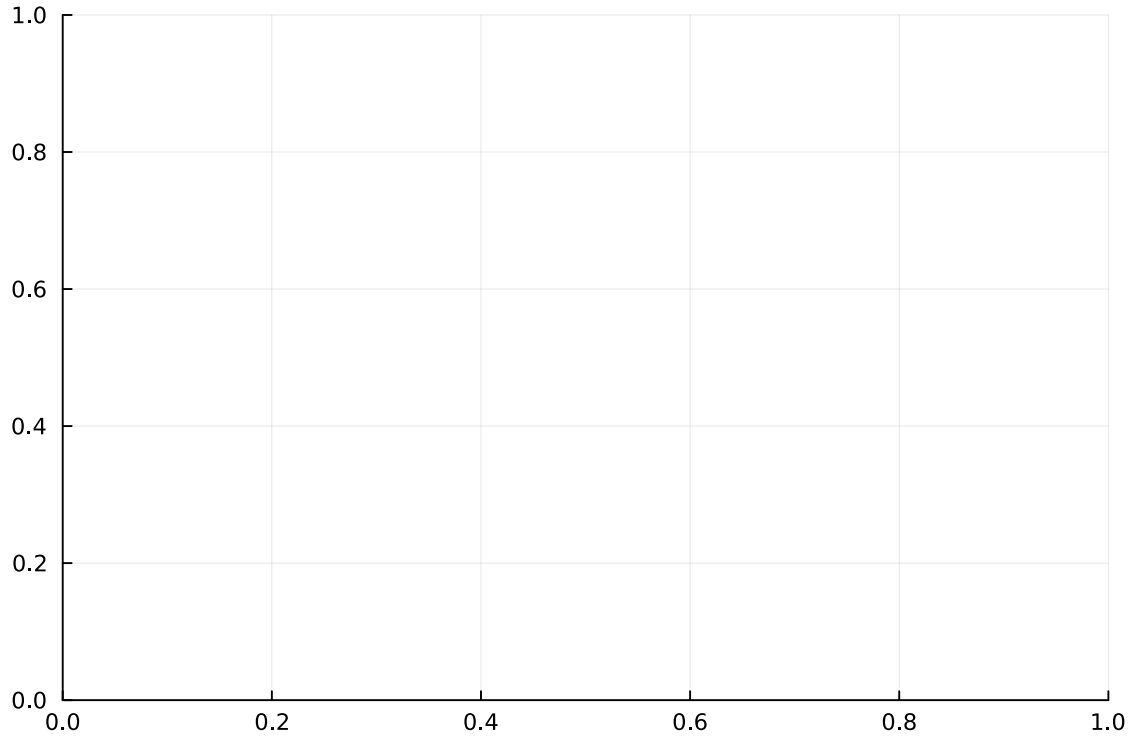
Out[115]:

In [116]:
```
using Plots
# Генерация случайных данных для x, y, z
x_data = rand(10)
y_data = rand(10)
z_data = rand(10)
# Построение трехмерного точечного графика
scatter(x_data, y_data, z_data, label="Случайные данные", xlabel="Ось
X", ylabel="Ось Y", zlabel="Ось Z", title="3D Точечный график
случайных данных")
```

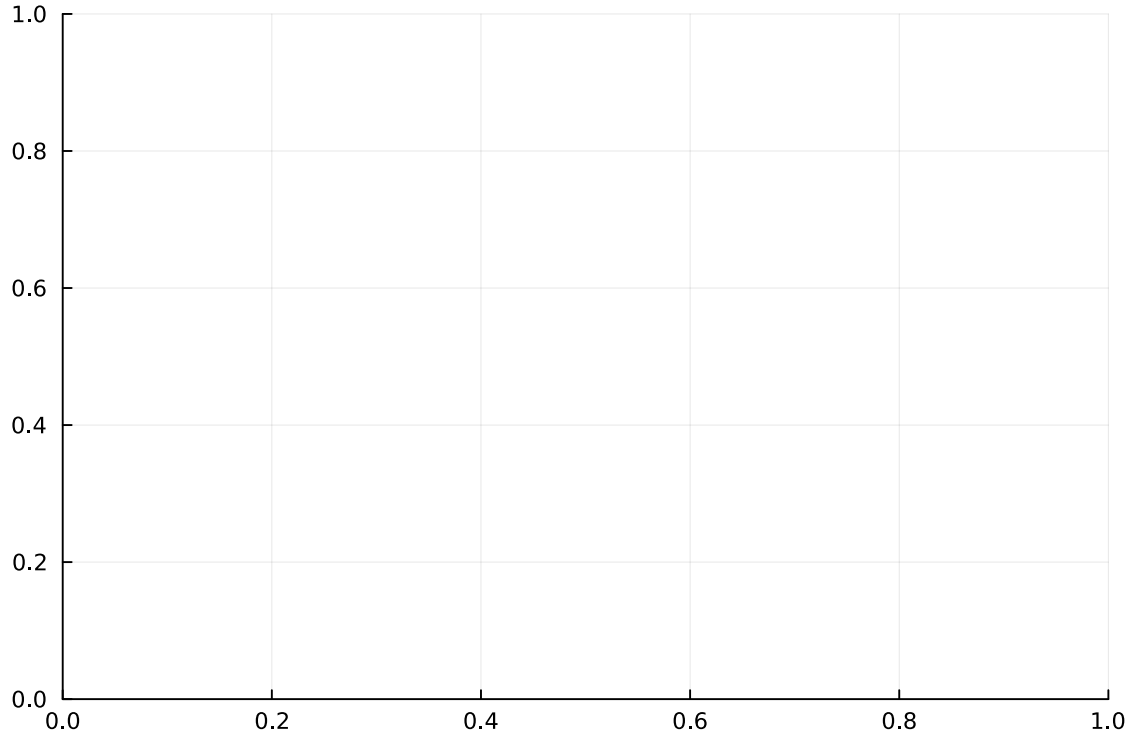
Out[116]:



In [127]:
```
using Plots
# Создаем случайные данные для трехмерного графика
x_data_3d = rand(1:10, 20)
y_data_3d = rand(1:10, 20)
z_data_3d = rand(1:10, 20)
# Построение трехмерного точечного графика
scatter3d(x_data_3d, y_data_3d, z_data_3d, label="Случайные данные",
Y", zlabel="Ось Z", title="Трехмерный точечный график случайных данны
# Отображение графика
display(Plots.plot())
```



In [123]:
```
using Plots
# Создаем случайные данные
x_data = rand(1:10, 20)
y_data = rand(1:10, 20)
# Построение точечного графика
scatter(x_data, y_data, label="Случайные данные", xlabel="Ось X", yla
график случайных данных", legend=:topright)
# Отображение графика
display(Plots.plot())
```



In [122]:
```
using Plots
# Создаем функцию для синусоиды
f(x) = sin(x)
# Создаем вектор значений x от 0 до 2π с шагом 0.1
x_values = 0:0.1:2π
# Создаем анимацию
anim = @animate for i in 1:length(x_values)
x_current = x_values[1:i]
y_current = f.(x_current)
plot(x_current, y_current, label="Синусоида", xlabel="x", ylabel="sin
построения синусоиды", legend=:topright)
end
# Отображение анимации
gif(anim, "sin_animation.gif", fps = 10)
```

```
[ Info: Saved animation to /home/marc/Desktop/sin_animation.gif
```

Out[122]: