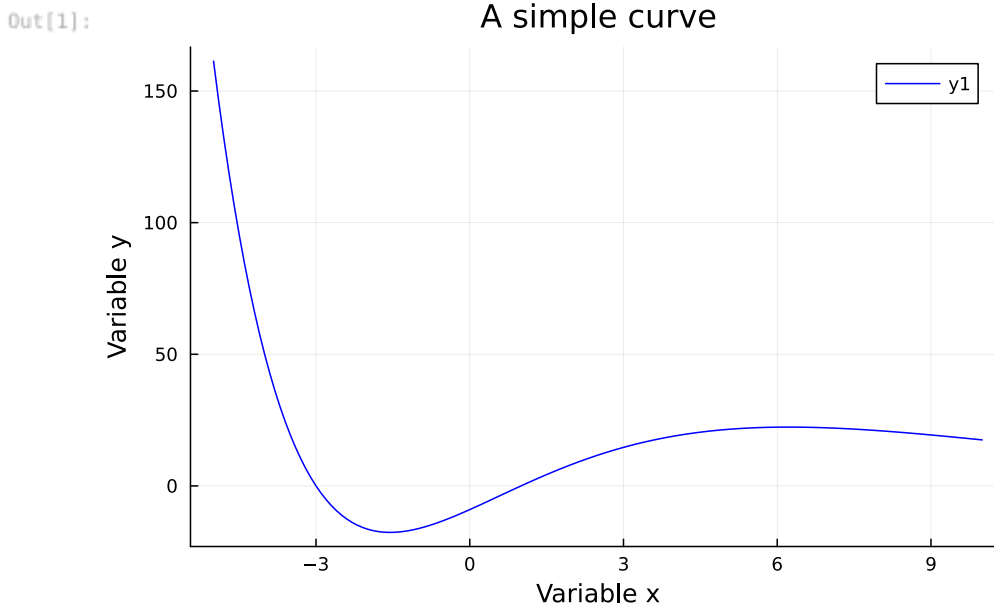


Повторение примеров

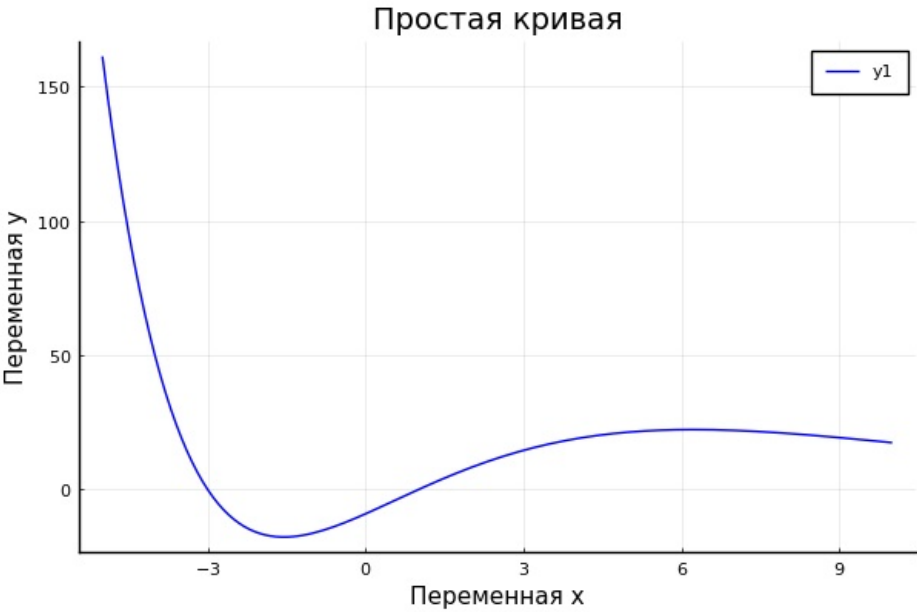
Основные пакеты для работы с графиками в Julia

```
In [1]: # подключаем для использования Plots:
using Plots
# задание функции:
f(x) = (3x.^2 + 6x .- 9).*exp.(-0.3x)
# генерирование массива значений x в диапазоне от -5 до 10 с шагом 0,1
# (шаг задан через указание длины массива):
x = collect(range(-5,10,length=151))
# генерирование массива значений y:
y = f(x)
# указывается, что для построения графика используется gr():
gr()
# задание опций при построении графика
# (название кривой, подписи по осям, цвет графика):
plot(x,y,
      title="A simple curve",
      xlabel="Variable x",
      ylabel="Variable y",
      color="blue")
```



```
In [2]: # указывается, что для построения графика используется pyplot():
pyplot()
# 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")
```

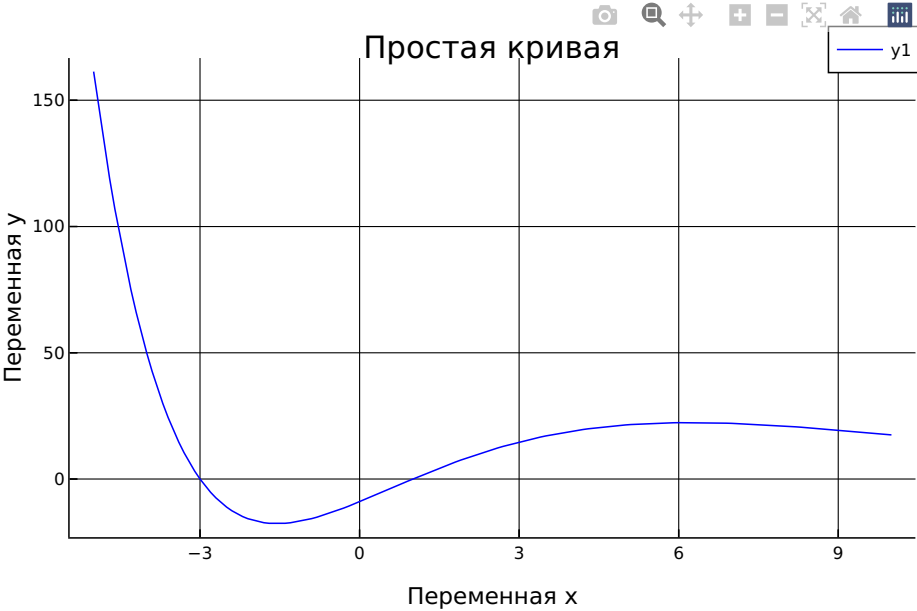
Out[2]:



In [3]:

```
plotly()
plot(x,y,
      title="Простая кривая",
      xlabel="Переменная x",
      ylabel="Переменная y",
      color="blue")
```

Out[3]:



In [4]:

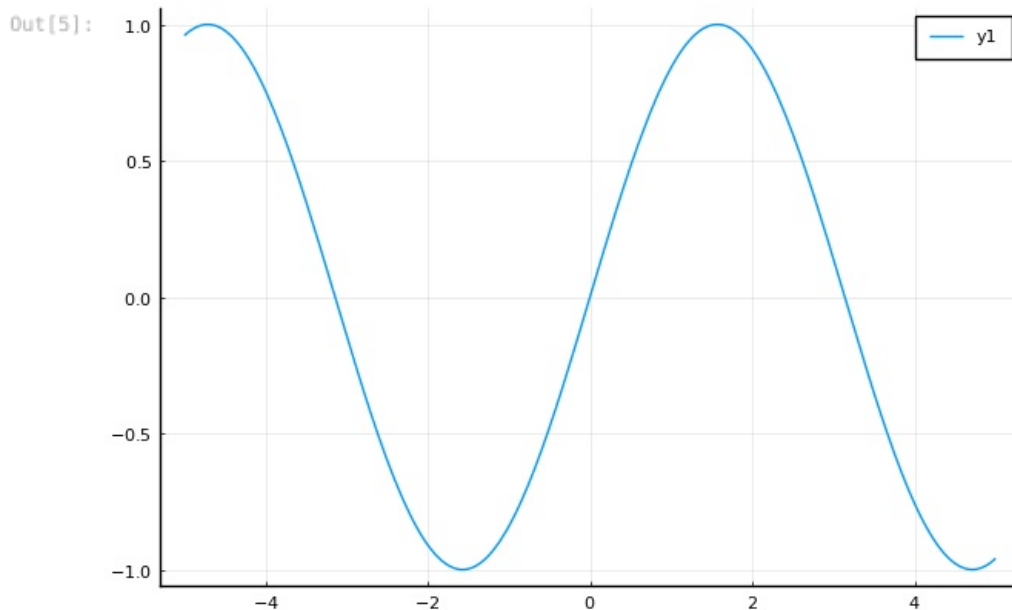
```
unicodeplots()
plot(x,y,
      title="Простая кривая",
      xlabel="Переменная x",
      ylabel="Переменная y",
      color="blue")
```

Out[4]:

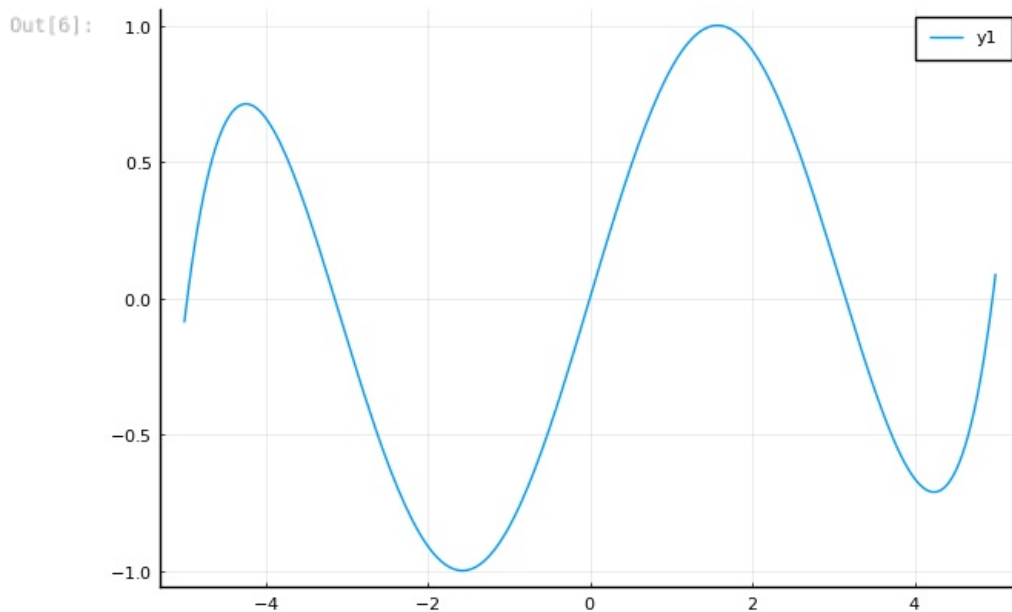


Опции при построении графика

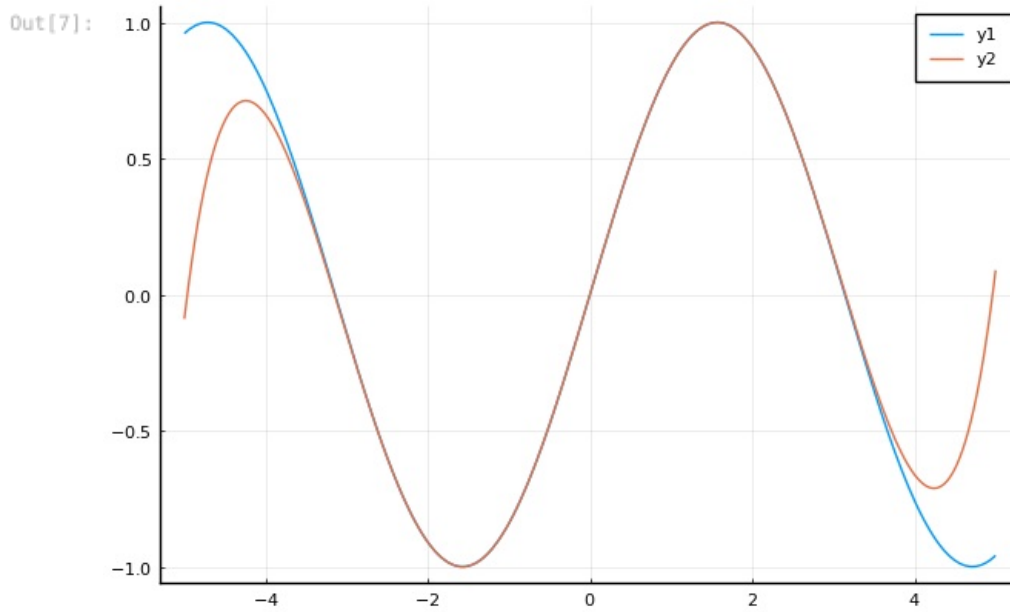
```
In [5]: # указывается, что для построения графика используется pyplot():
pyplot()
# задание функции sin(x):
sin_theor(x) = sin(x)
# построение графика функции sin(x):
plot(sin_theor)
```



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

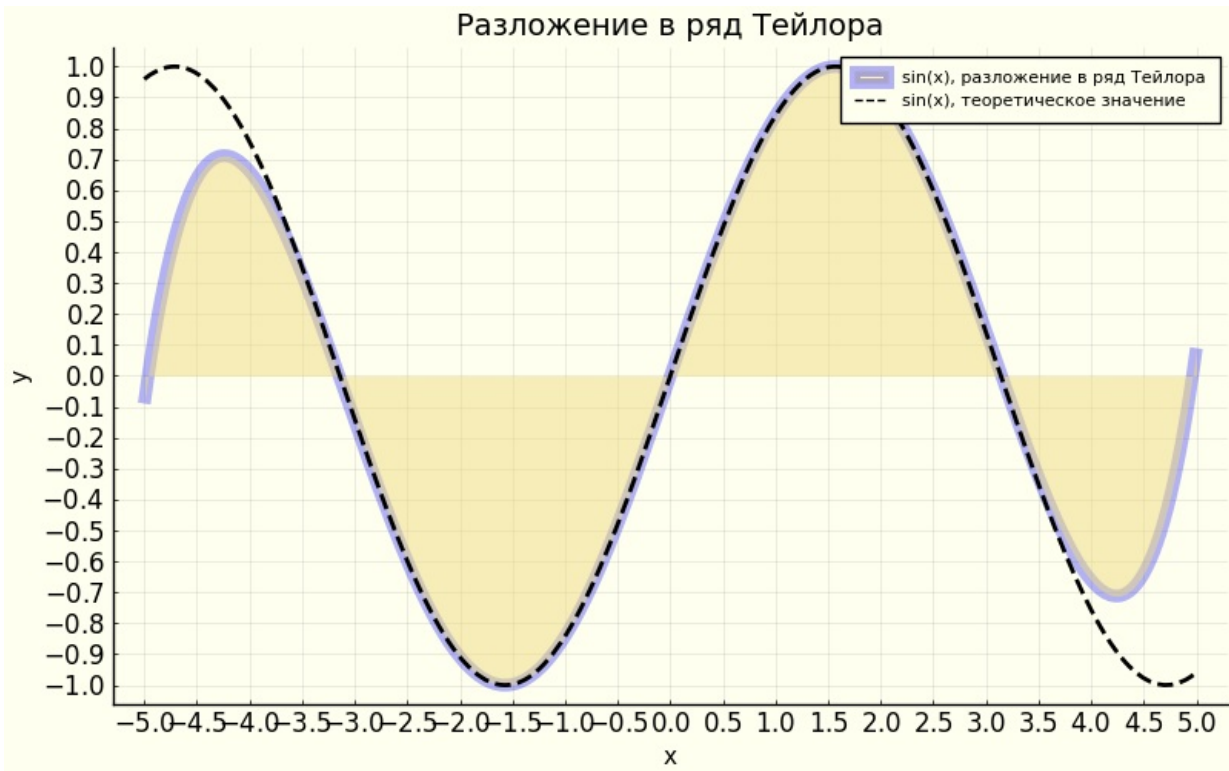


```
In [7]: # построение двух функций на одном графике:
plot(sin_theor)
plot!(sin_taylor)
```



```
In [8]: 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[8]:



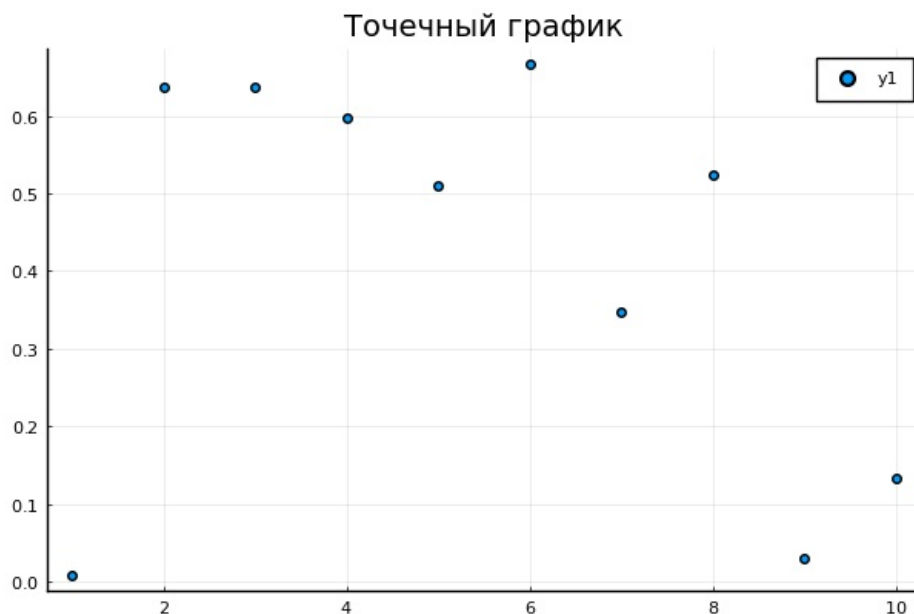
```
In [9]: savefig("taylor.pdf")
savefig("taylor.png")
```

Out[9]: "D:\\Education\\КомпПрактикумПоСтатМоделированию\\labs\\gitrepo\\lab5\\taylor.png"

Точечный график

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

Out[10]:

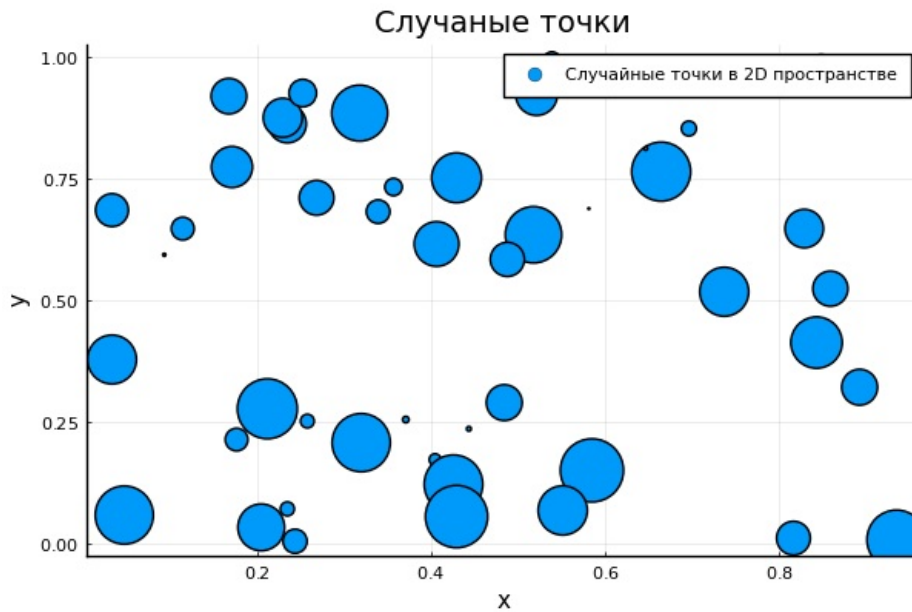


sys:1: UserWarning: No data for colormapping provided via 'c'. Parameters 'vmin', 'vmax' will be ignored

```
In [11]: # параметры распределения точек на плоскости:
n = 50
x = rand(n)
y = rand(n)
ms = rand(n) * 30
# параметры построения графика:
scatter(x, y,
      label = "Случайные точки в 2D пространстве",
      leg=:topright,
      ylabel = "y",
```

```
xlabel = "x",
title = "Случайные точки",
markersize=ms)
```

Out[11]:



In [12]: # параметры распределения точек в пространстве:

```
n = 50
x = rand(n)
y = rand(n)
z = rand(n)
ms = rand(n) * 30
# параметры построения графика:
scatter(x, y, z,
        label = "Случайные точки в 3D пространстве",
        leg=:topleft,
        ylabel = "y",
        xlabel = "x",
        zlabel = "z",
        title = "Случайные точки",
        markersize=ms)
```

Out[12]:

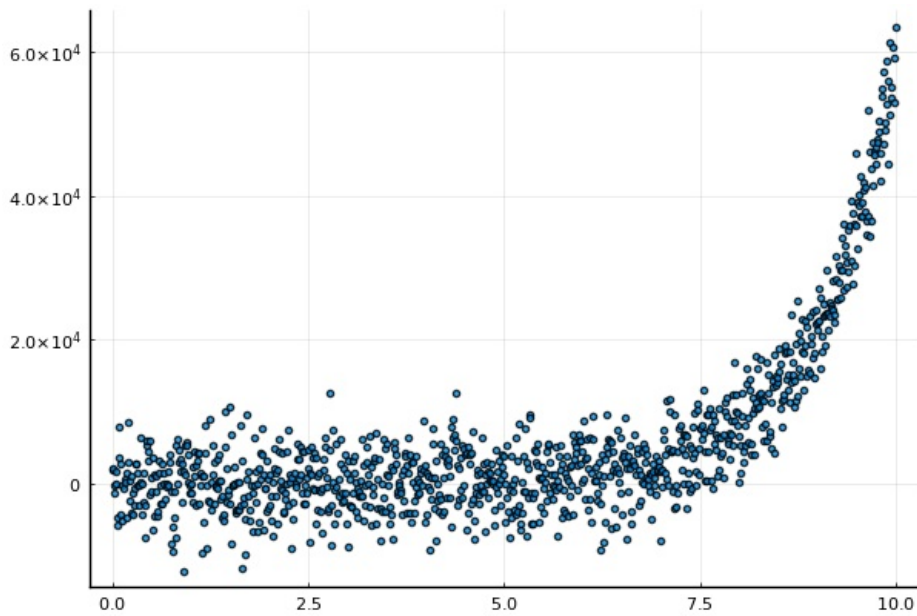


Аппроксимация данных

In [13]: # 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[13]:



In [14]: # определение массива для нахождения коэффициентов полинома:

```
A = [ones(1000) x x.^2 x.^3 x.^4 x.^5]
```

решение матричного уравнения:

```
c = A\y
```

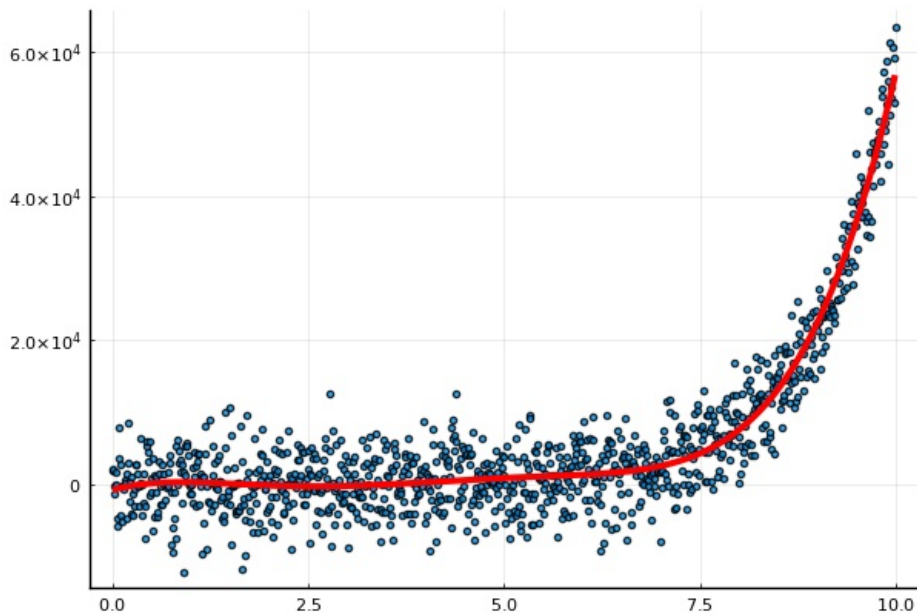
построение полинома:

```
f1 = c[1]*ones(1000) + c[2]*x + c[3]*x.^2 + c[4]*x.^3 + c[5]*x.^4 + c[6]*x.^5
```

построение графика аппроксимирующей функции:

```
plot(x,f1,linewidth=3, color=:red)
```

Out[14]:



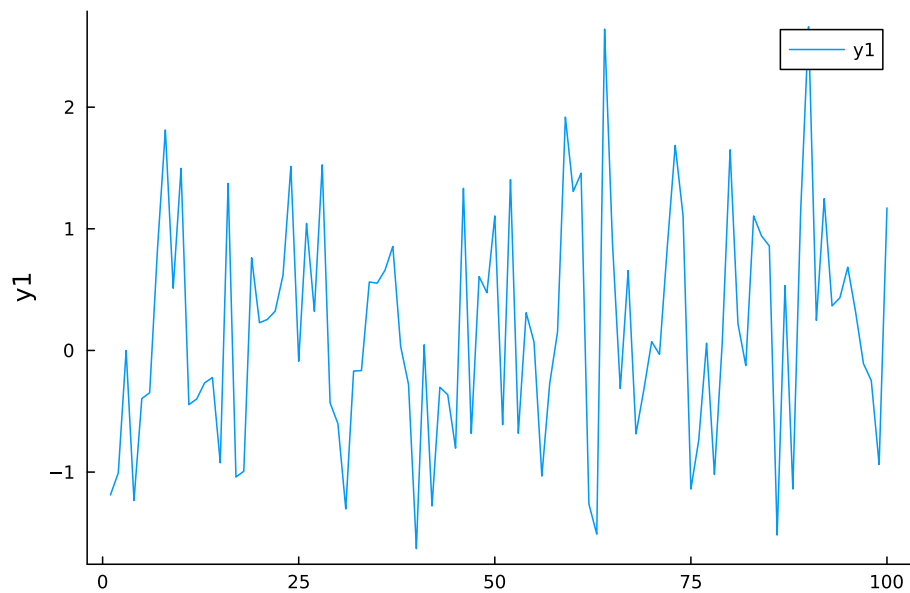
Две оси ординат

In [120.. # пример случайной траектории

(заданы обозначение траектории, легенда вверху справа, без сетки)

```
plot(randn(100),  
     ylabel="y1",  
     leg=:topright,  
     grid = :off,  
     )
```

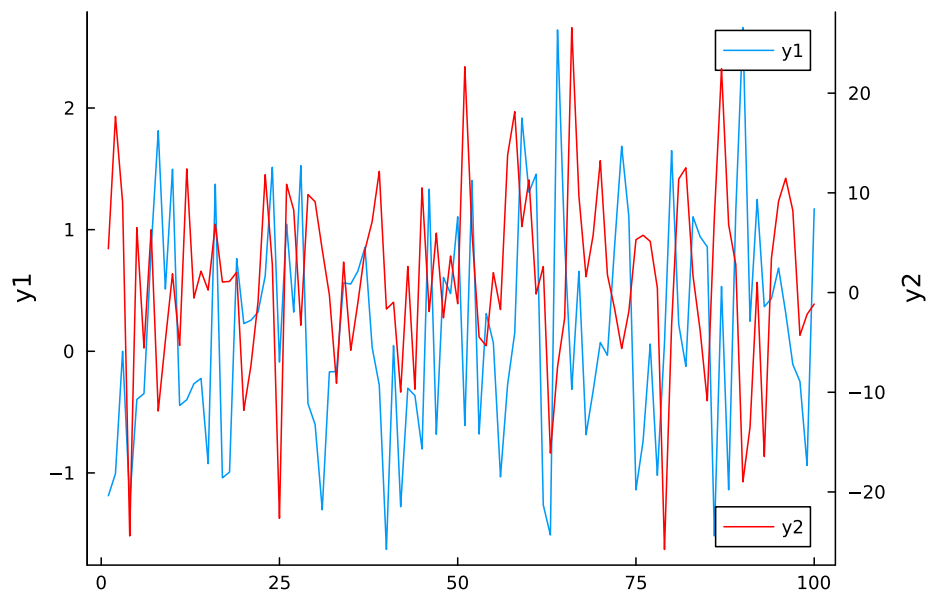
Out[120...



In [121...

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

Out[121...

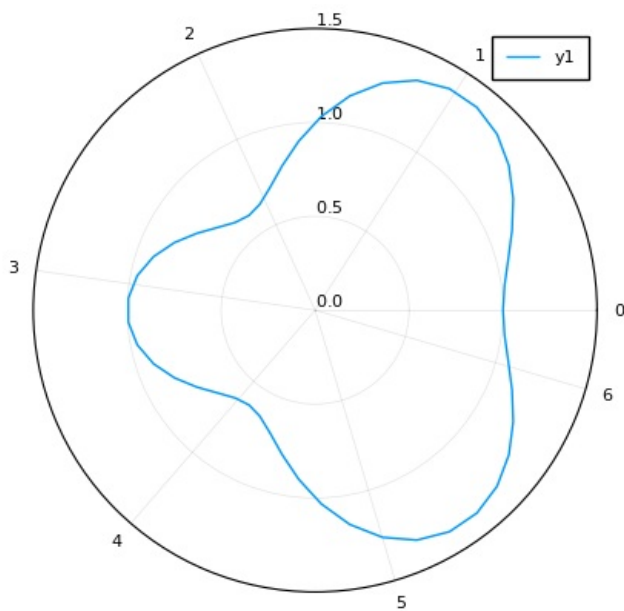


Полярные координаты

In [17]:

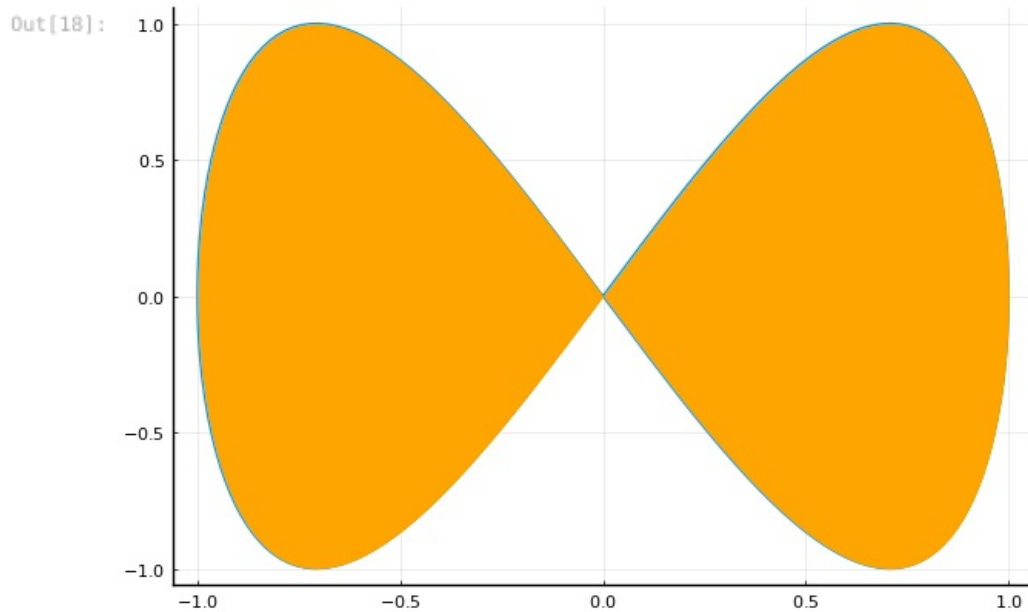
```
# функция в полярных координатах:
r(θ) = 1 + cos(θ) * sin(θ)^2
# полярная система координат:
θ = range(0, stop=2π, length=50)
# график функции, заданной в полярных координатах:
plot(θ, r.(θ),
     proj=:polar,
     lims=(0,1.5))
```


Out[17]:



Параметрический график

```
In [18]: # параметрическое уравнение:
x1(t) = sin(t)
y1(t) = sin(2t)
# построение графика:
plot(x1, y1, 0, 2π, leg=false, fill=(0,:orange))
```



```
In [19]: # параметрическое уравнение
t = range(0, stop=10, length=1000)
x = cos.(t)
y = sin.(t)
z = sin.(5t)
# построение графика:
plot(x, y, z)
```

Out[19]:

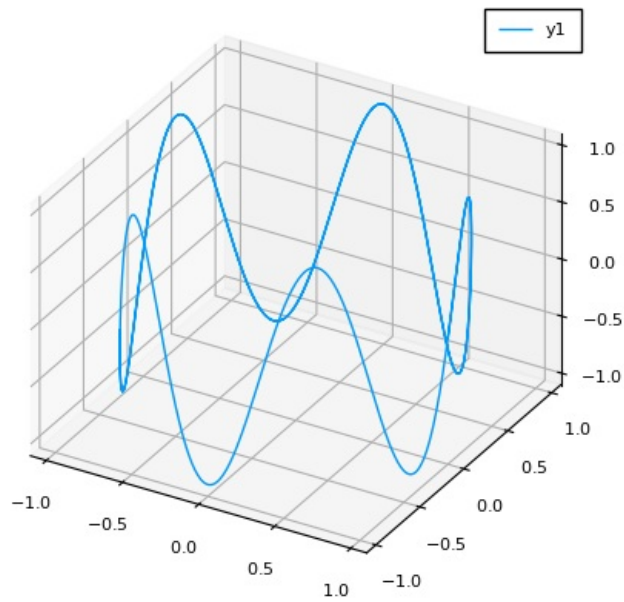
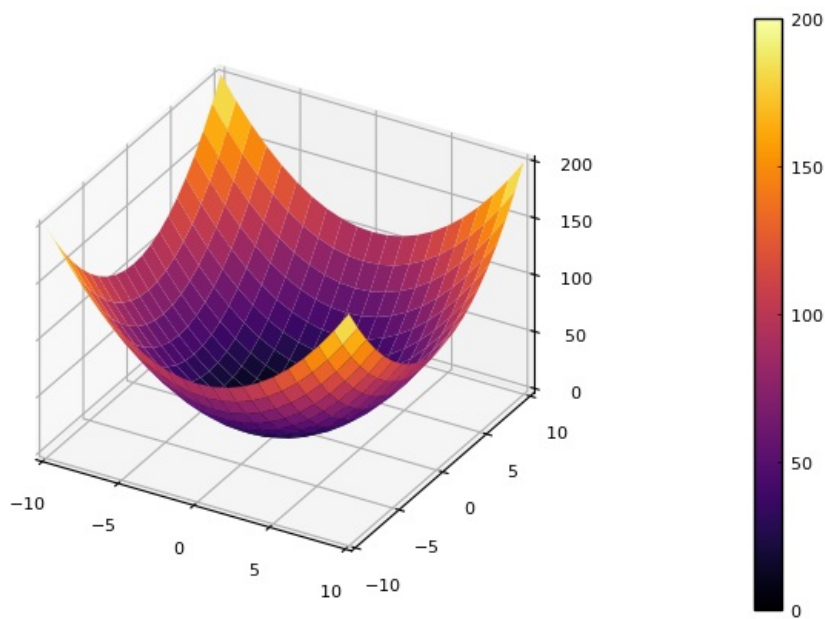


График поверхности

In [20]: # построение графика поверхности:

```
f(x,y) = x^2 + y^2  
x = -10:10  
y = x  
surface(x, y, f)
```

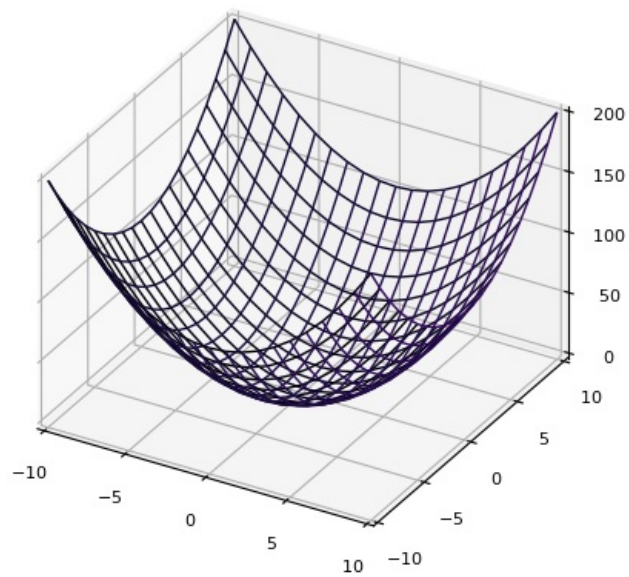
Out[20]:



In [21]: # построение графика поверхности:

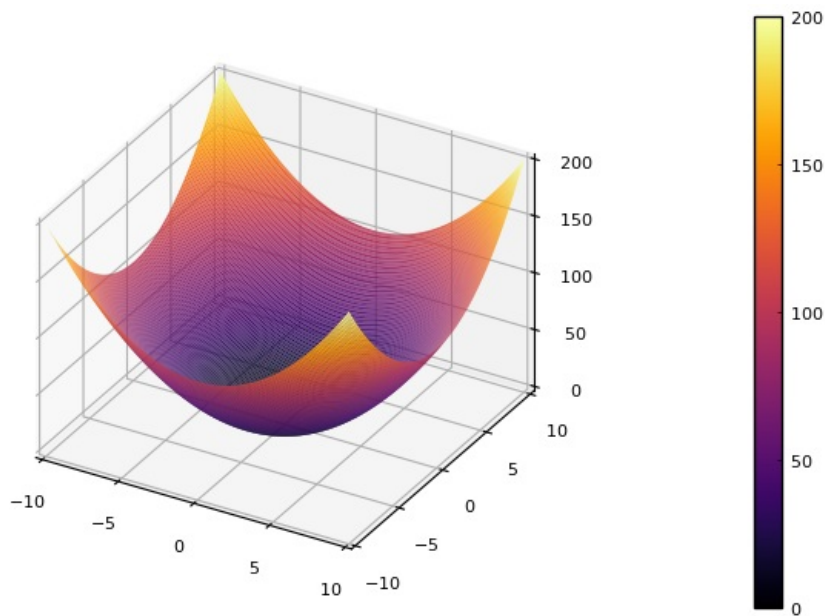
```
f(x,y) = x^2 + y^2  
x = -10:10  
y = x  
plot(x, y, f,  
      linestyle=:wireframe)
```

Out[21]:



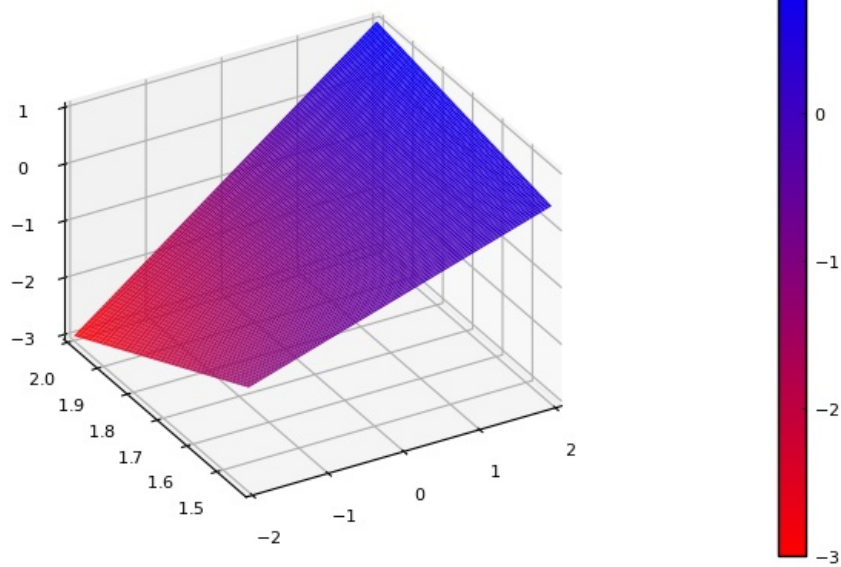
```
In [22]: f(x,y) = x^2 + y^2
x = -10:0.1:10
y = x
plot(x, y, f,
      linestyle = :surface)
```

Out[22]:



```
In [23]: 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,
      linestyle = :surface,
      c=cgrad([:red,:blue]),
      camera=(-30,30),
)
```

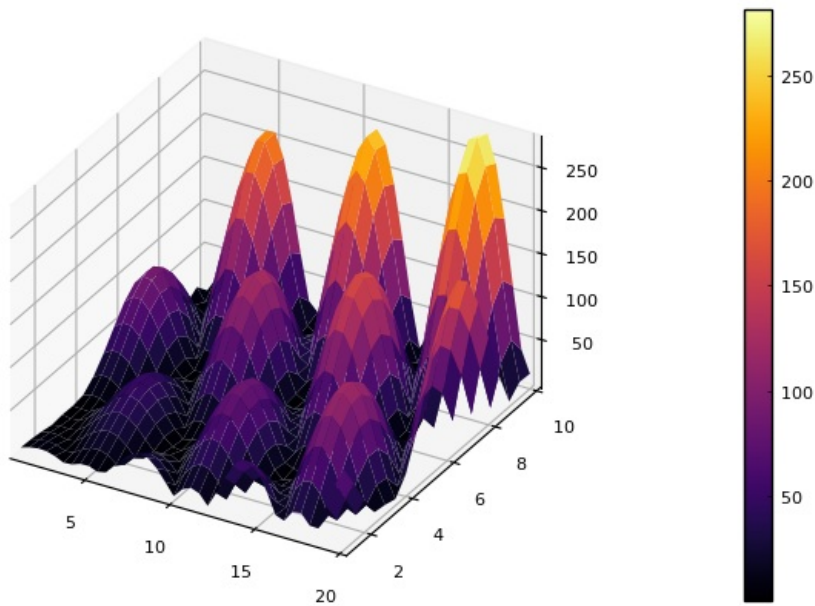
Out[23]:



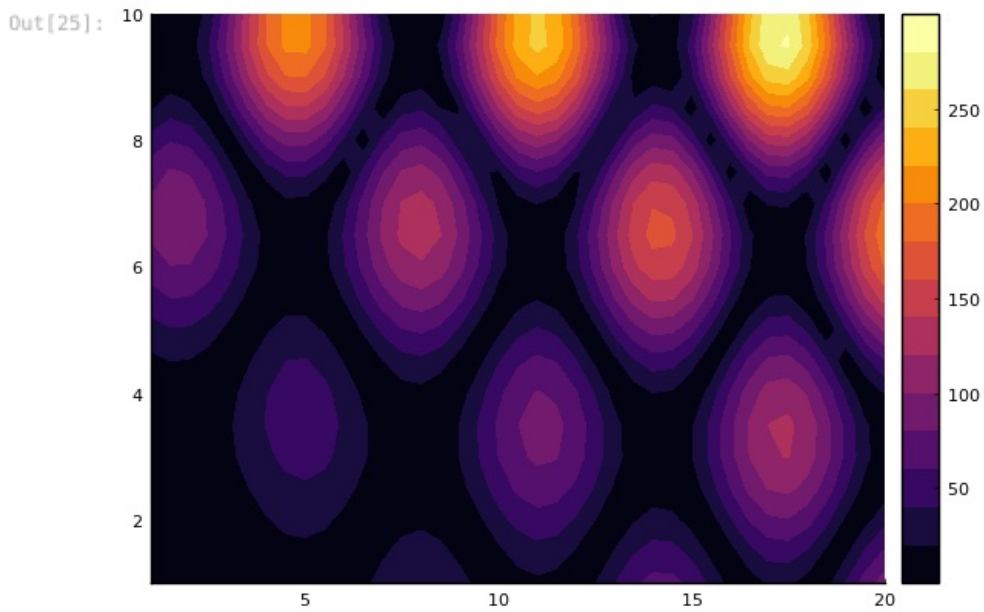
Линии уровня

```
In [24]: 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,  
      linestyle = :surface,  
      )
```

Out[24]:



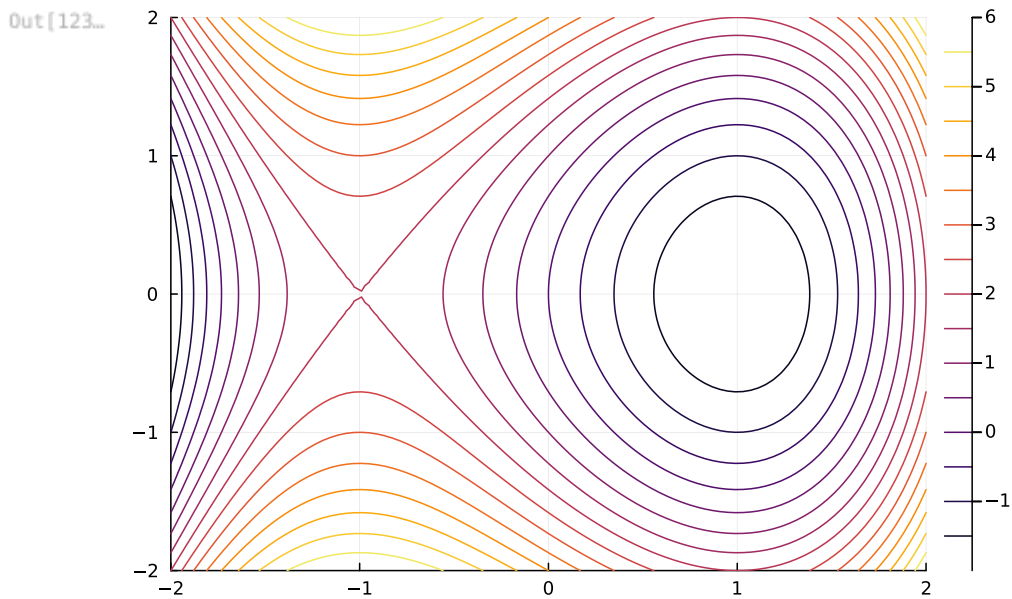
```
In [25]: p = contour(x, y, g, fill=true)  
plot(p)
```



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

Векторные поля

```
In [123... # определение переменных:
X = range(-2, stop=2, length=100)
Y = range(-2, stop=2, length=100)
# определение функции:
h(x, y) = x^3 - 3x + y^2
# построение поверхности:
plot(X,Y,h,linetype = :surface)
# построение линий уровня:
contour(X, Y, h)
```



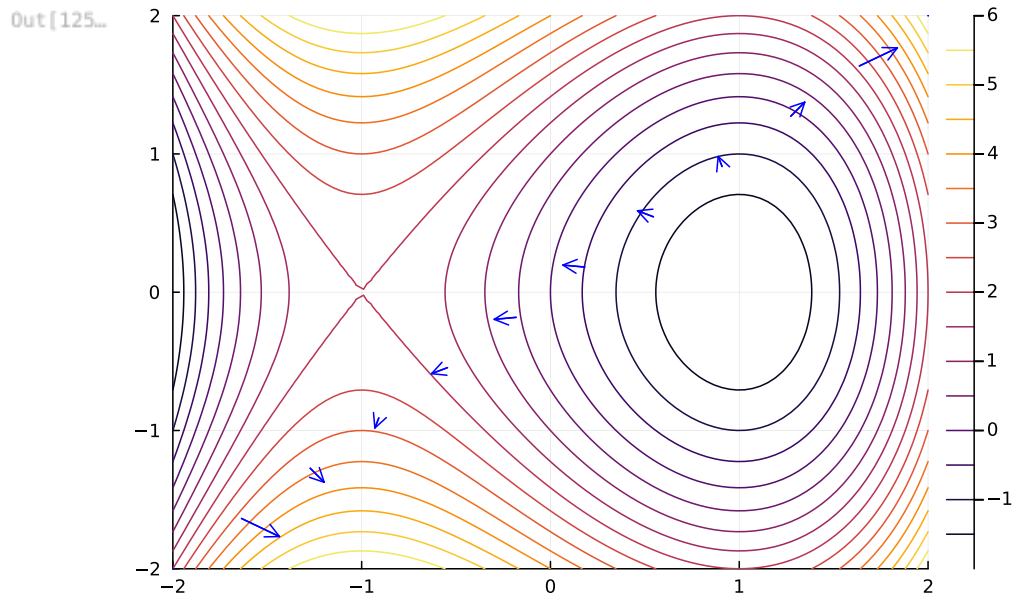
```
In [124... # градиент:
x = range(-2, stop=2, length=12)
y = range(-2, stop=2, length=12)
# производная от исходной функции:
dh(x, y) = [3x^2 - 3; 2y] / 25
# построение векторного поля:
quiver!(x, y', quiver=dh, c=:blue)
# коррекция области видимости графика:
xlims!(-2, 2)
ylims!(-2, 2)
```

Expects 12 elements in each col of y, found 1.

Stacktrace:

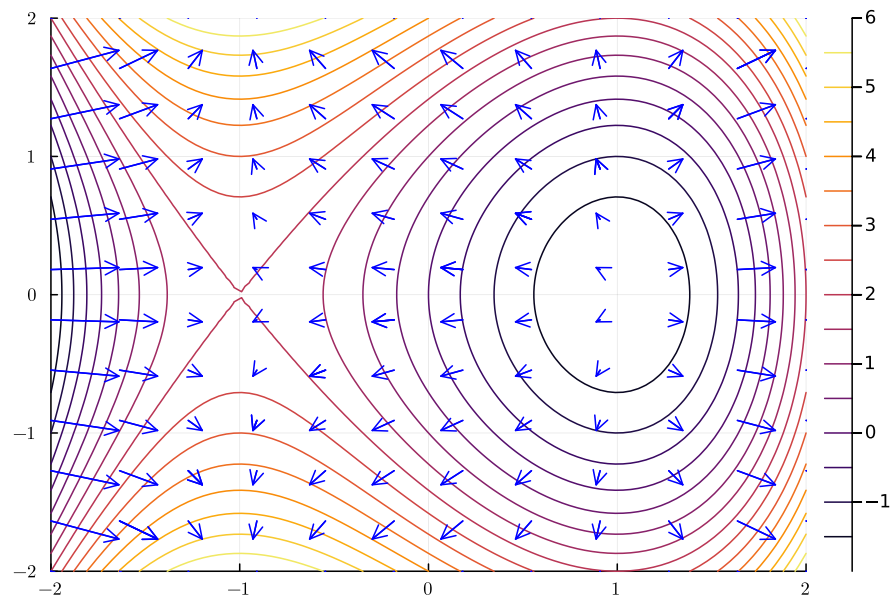
```
[1] error(s::String)
@ Base .\error.jl:35
[2] _compute_xyz(x::StepRangeLen{Float64, Base.TwicePrecision{Float64}, Base.TwicePrecision{Float64}}, Int64,
y::Vector{Float64}, z::Nothing, nice_error::Bool)
@ RecipesPipeline C:\Users\vemanaeva\.julia\packages\RecipesPipeline\BGM3l\src\series.jl:90
[3] macro expansion
@ C:\Users\vemanaeva\.julia\packages\RecipesPipeline\BGM3l\src\series.jl:140 [inlined]
[4] apply_recipe(plotattributes::AbstractDict{Symbol, Any}, #unused#::Type{RecipesPipeline.SliceIt}, x::Any, y
::Any, z::Any)
@ RecipesPipeline C:\Users\vemanaeva\.julia\packages\RecipesBase\BRe07\src\RecipesBase.jl:300
[5] _process_userrecipes!(plt::Any, plotattributes::Any, args::Any)
@ RecipesPipeline C:\Users\vemanaeva\.julia\packages\RecipesPipeline\BGM3l\src\user_recipe.jl:38
[6] recipe_pipeline!(plt::Any, plotattributes::Any, args::Any)
@ RecipesPipeline C:\Users\vemanaeva\.julia\packages\RecipesPipeline\BGM3l\src\RecipesPipeline.jl:72
[7] _plot!(plt::Plots.Plot, plotattributes::Any, args::Any)
@ Plots C:\Users\vemanaeva\.julia\packages\Plots\sxUvK\src\plot.jl:223
[8] plot!{::Plots.Plot, ::Any, ::Vararg{Any}; kw::Base.Pairs{Symbol, V, Tuple{Vararg{Symbol, N}}, NamedTuple{names, T}} where {V, N, names, T}<:Tuple{Vararg{Any, N}}}})
@ Plots C:\Users\vemanaeva\.julia\packages\Plots\sxUvK\src\plot.jl:213
[9] plot!{::Any, ::Vararg{Any}; kw::Base.Pairs{Symbol, V, Tuple{Vararg{Symbol, N}}, NamedTuple{names, T}} where {V, N, names, T}<:Tuple{Vararg{Any, N}}}})
@ Plots C:\Users\vemanaeva\.julia\packages\Plots\sxUvK\src\plot.jl:202
[10] quiver!{::Any, ::Vararg{Any}; kw::Base.Pairs{Symbol, V, Tuple{Vararg{Symbol, N}}, NamedTuple{names, T}} where {V, N, names, T}<:Tuple{Vararg{Any, N}}}})
@ Plots C:\Users\vemanaeva\.julia\packages\RecipesBase\BRe07\src\RecipesBase.jl:429
[11] top-level scope
@ In[124]:7
```

```
In [125.. # градиент:
x = range(-2, stop=2, length=12)
y = range(-2, stop=2, length=12)
# производная от исходной функции:
dh(x, y) = [3x^2 - 3; 2y] / 25
# построение векторного поля:
quiver!(x, y, quiver=dh, c=:blue)
# коррекция области видимости графика:
xlims!(-2, 2)
ylims!(-2, 2)
```



```
In [126.. # градиент:
pyplot()
xv = collect(range(-2, stop=2, length=12))
yv = collect(range(-2, stop=2, length=12))
# производная от исходной функции:
dh(x,y) = [3*x^2 - 3; 2y] / 25
x = vcat([xv for i in 1:12]...)
y = reshape(hcat([yv for i in 1:12]...)', :, 1)
# построение векторного поля:
quiver!(x, y, quiver=dh, c=:blue)
# коррекция области видимости графика:
xlims!(-2, 2)
ylims!(-2, 2)
```

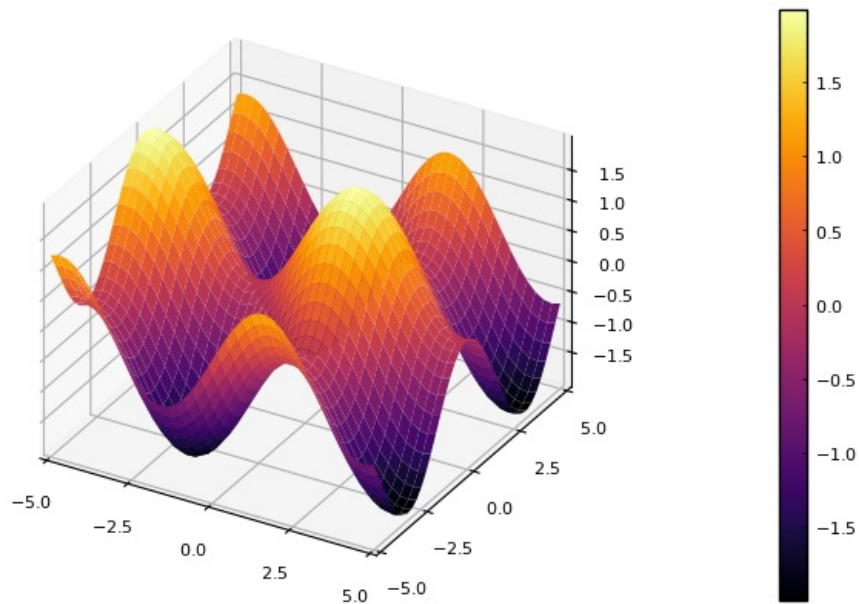

Out[126..



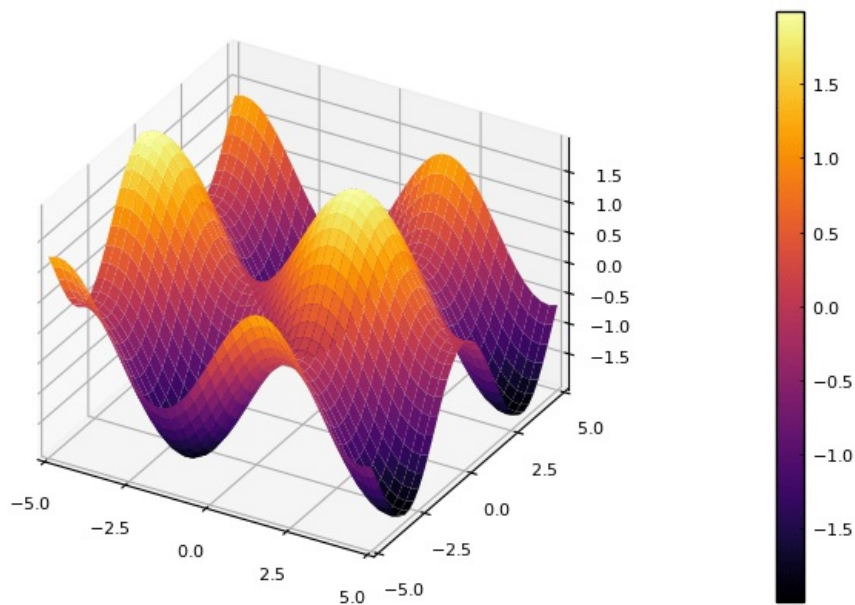
Анимация

```
In [29]: # построение поверхности:
i = 0
X = Y = range(-5, stop=5, length=40)
surface(X, Y, (x, y) -> sin(x+10sin(i))+cos(y))
# анимация:
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 D:\Education\КомпПрактикумПоСтатМоделированию\labs\gitrepo\lab5\tmp.gif



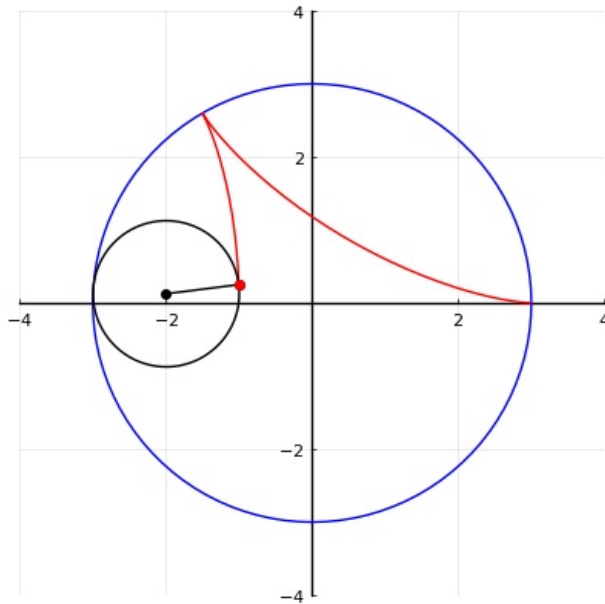
Out [29]:



Гипоциклоида

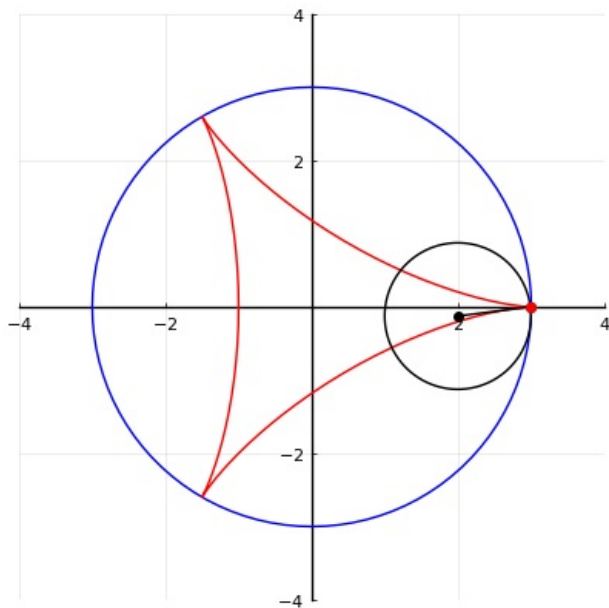
```
In [30]: # радиус малой окружности:
r1 = 1
# коэффициент для построения большой окружности:
k = 3
# число отсчётов:
n = 100
# массив значений угла  $\theta$ :
#  $\theta$  from 0 to  $2\pi$  ( + a little extra)
 $\theta = \text{collect}(0:2*\pi/100:2*\pi+2*\pi/100)$ 
# массивы значений координат:
X = r1*k*cos.( $\theta$ )
Y = r1*k*sin.( $\theta$ )
# задаём оси координат:
plt=plot(5,xlim=(-4,4),ylim=(-4,4), c=:red, aspect_ratio=1, legend=false, framestyle=:origin)
# большая окружность:
plot!(plt, X,Y, c=:blue, legend=false)
i = 50
t =  $\theta[1:i]$ 
# гипоциклоида:
x = r1*(k-1)*cos.(t) + r1*cos.((k-1)*t)
y = r1*(k-1)*sin.(t) - r1*sin.((k-1)*t)
plot!(x,y, c=:red)
# малая окружность:
xc = r1*(k-1)*cos(t[end]) .+ r1*cos.( $\theta$ )
yc = r1*(k-1)*sin(t[end]) .+ r1*sin.( $\theta$ )
plot!(xc,yc,c=:black)
# радиус малой окружности:
xl = transpose([r1*(k-1)*cos(t[end]) x[end]])
yl = transpose([r1*(k-1)*sin(t[end]) y[end]])
plot!(xl,yl,markershape=:circle,markersize=4,c=:black)
scatter!([x[end]], [y[end]], c=:red, markerstrokecolor=:red)
```


Out[30]:

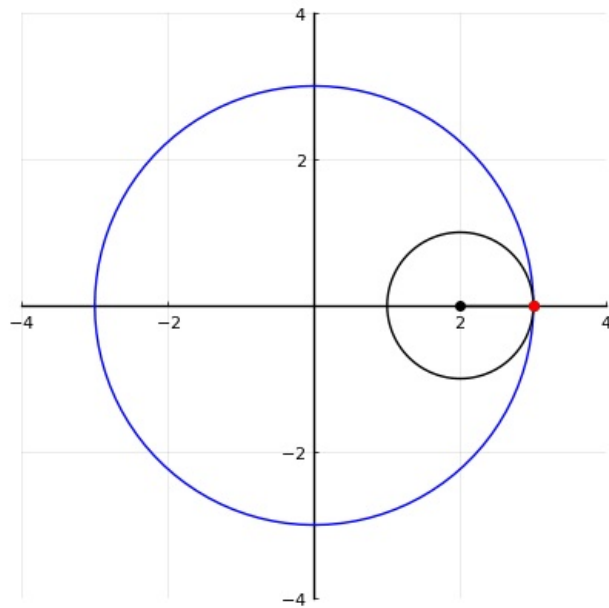


```
In [31]: anim = @animate for i in 1:n
# задаём оси координат:
plt=plot(5,xlim=(-4,4),ylim=(-4,4), c=:red, aspect_ratio=1,legend=false, framestyle=:origin)
# большая окружность:
plot!(plt, X,Y, c=:blue, legend=false)
t = 0[1:i]
# гипоциклоида:
x = r1*(k-1)*cos.(t) + r1*cos.((k-1)*t)
y = r1*(k-1)*sin.(t) - r1*sin.((k-1)*t)
plot!(x,y, c=:red)
# малая окружность:
xc = r1*(k-1)*cos(t[end]) .+ r1*cos.(θ)
yc = r1*(k-1)*sin(t[end]) .+ r1*sin.(θ)
plot!(xc,yc,c=:black)
# радиус малой окружности:
xl = transpose([r1*(k-1)*cos(t[end]) x[end]])
yl = transpose([r1*(k-1)*sin(t[end]) y[end]])
plot!(xl,yl,markershape=:circle,markersize=4,c=:black)
scatter!([x[end]], [y[end]], c=:red, markerstrokecolor=:red)
end
gif(anim, "hypocycloid.gif")
```

[Info: Saved animation to D:\Education\КомпПрактикумПоСтатМоделированию\labs\gitrepo\lab5\hypocycloid.gif



Out[31]:

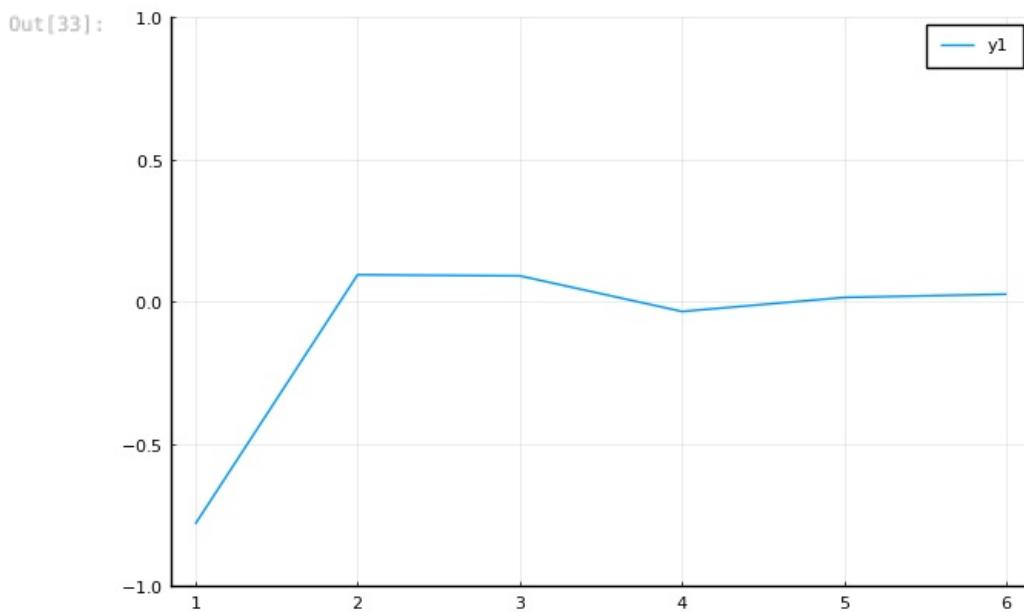


Errorbars

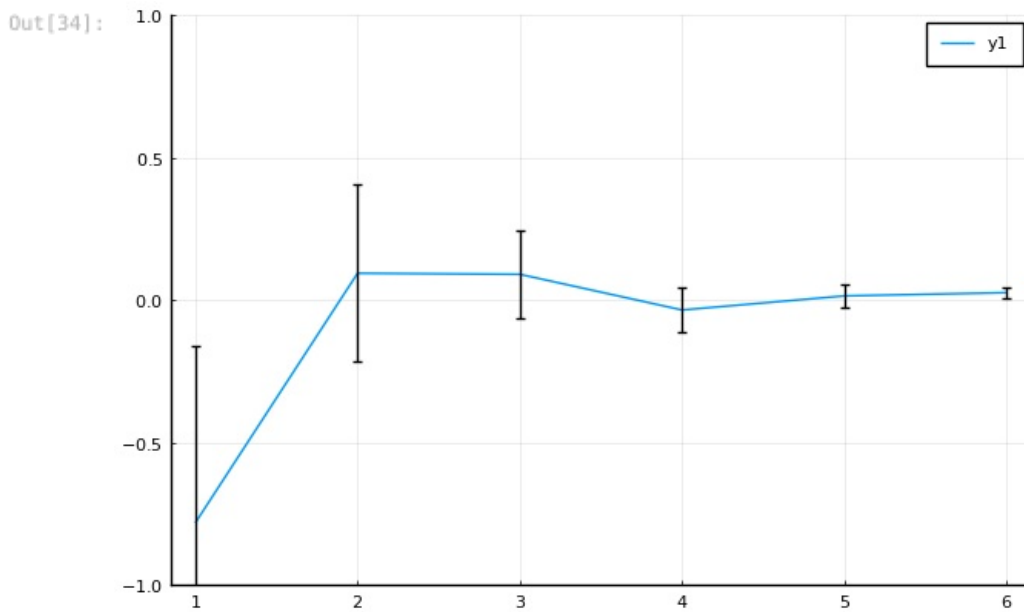
```
In [32]: # подключение пакета Statistics:
using Statistics
sds = [1, 1/2, 1/4, 1/8, 1/16, 1/32]
n = 10
y = [mean(sd*randn(n)) for sd in sds]
errs = 1.96 * sds / sqrt(n)
```

```
Out[32]: 6-element Vector{Float64}:
 0.6198064213930023
 0.3099032106965012
 0.1549516053482506
 0.0774758026741253
 0.03873790133706265
 0.019368950668531323
```

```
In [33]: plot(y,
ylims = (-1,1),
)
```

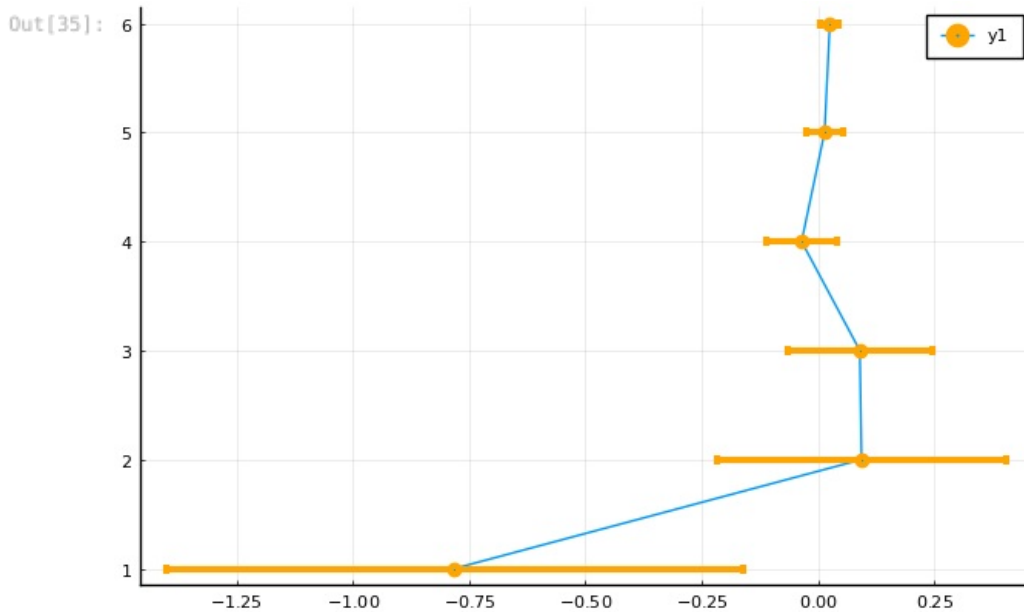


```
In [34]: plot(y,
ylims = (-1,1),
err = errs
)
```



sys:1: UserWarning: You passed a edgecolor/edgecolors ((0.0, 0.0, 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.

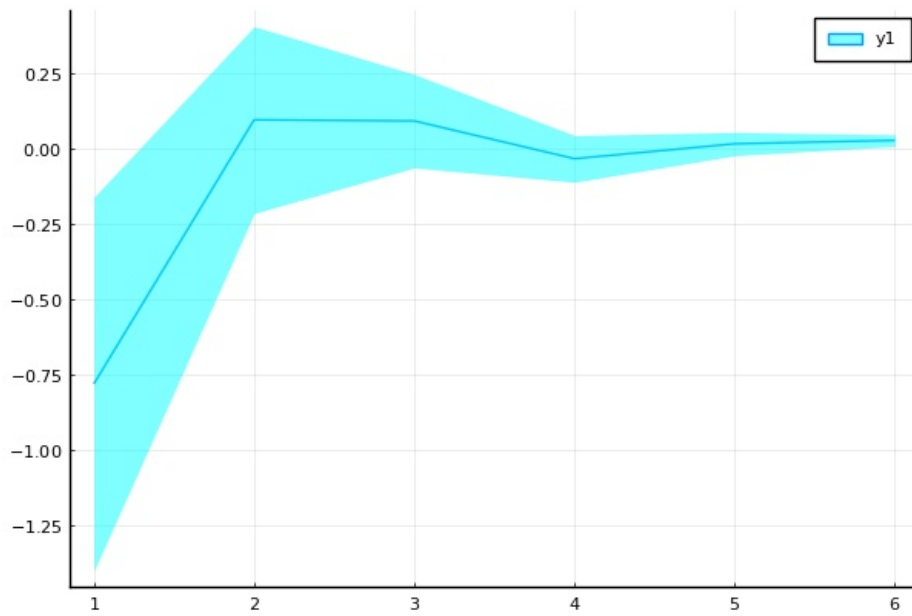
```
In [35]: plot(y, 1:length(y),
xerr = errs,
marker = stroke(3,:orange)
)
```



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

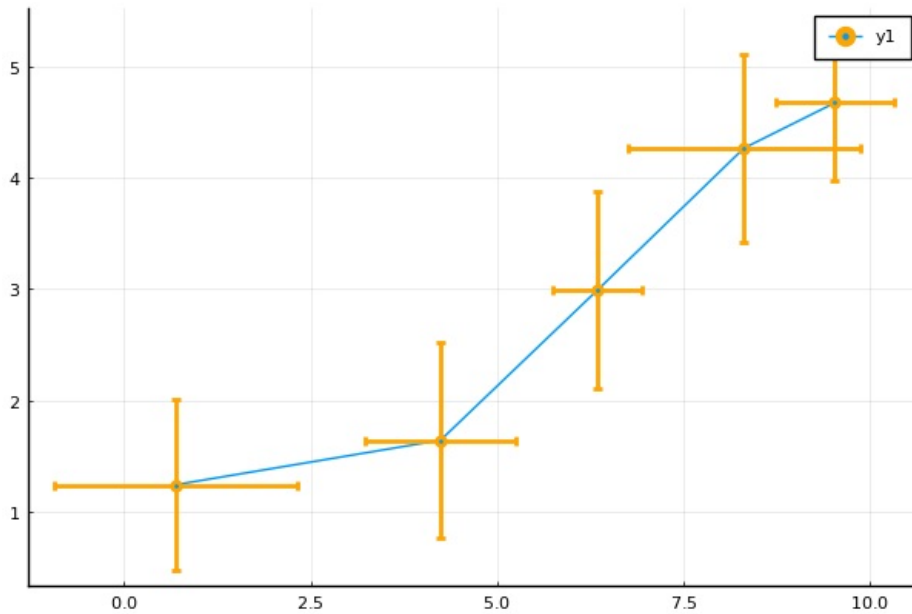
Out[36]:



In [37]:

```
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)
)
```

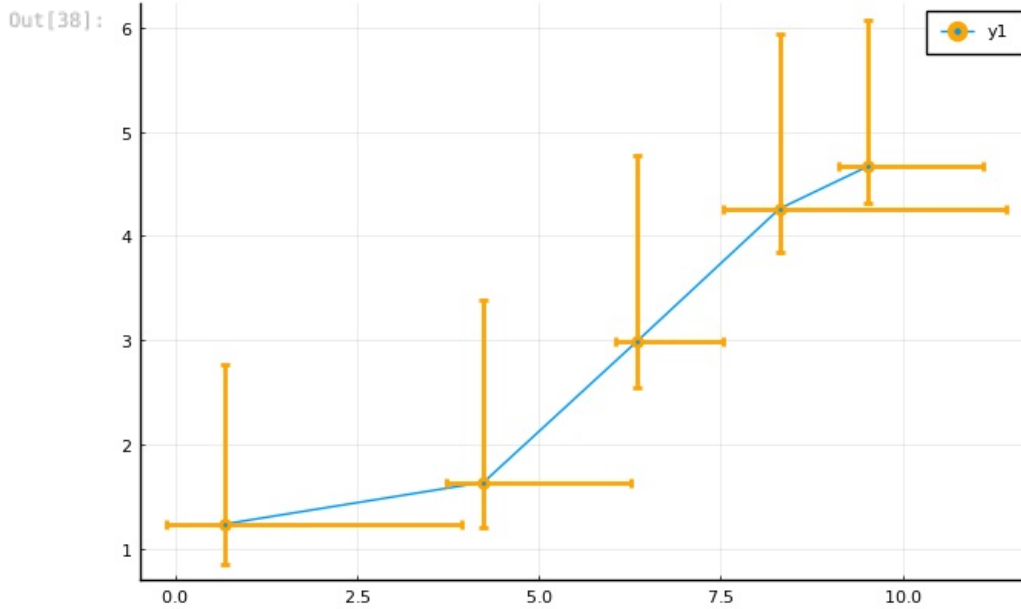
Out[37]:



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.

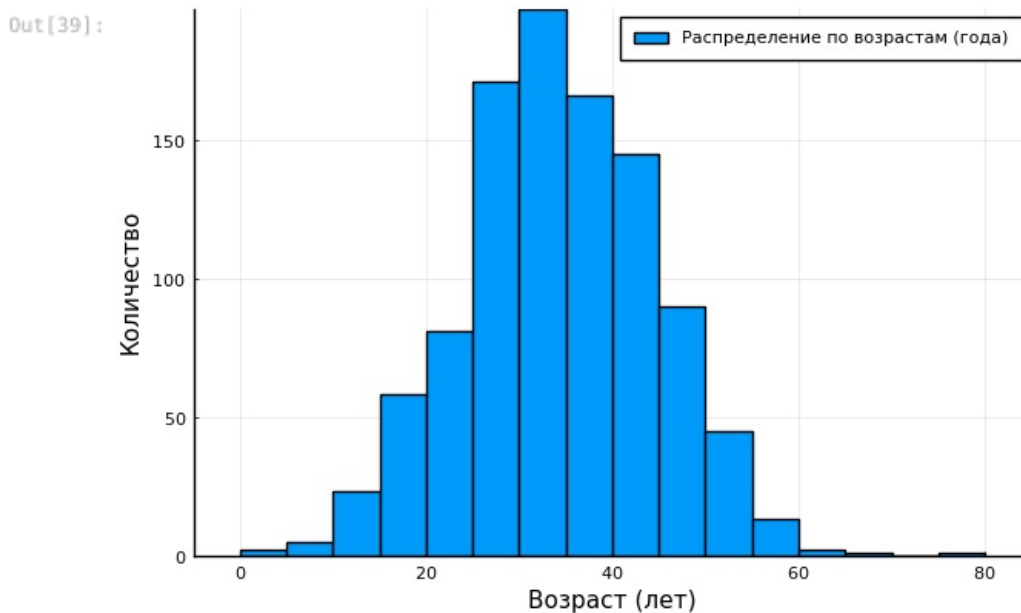
In [38]:

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



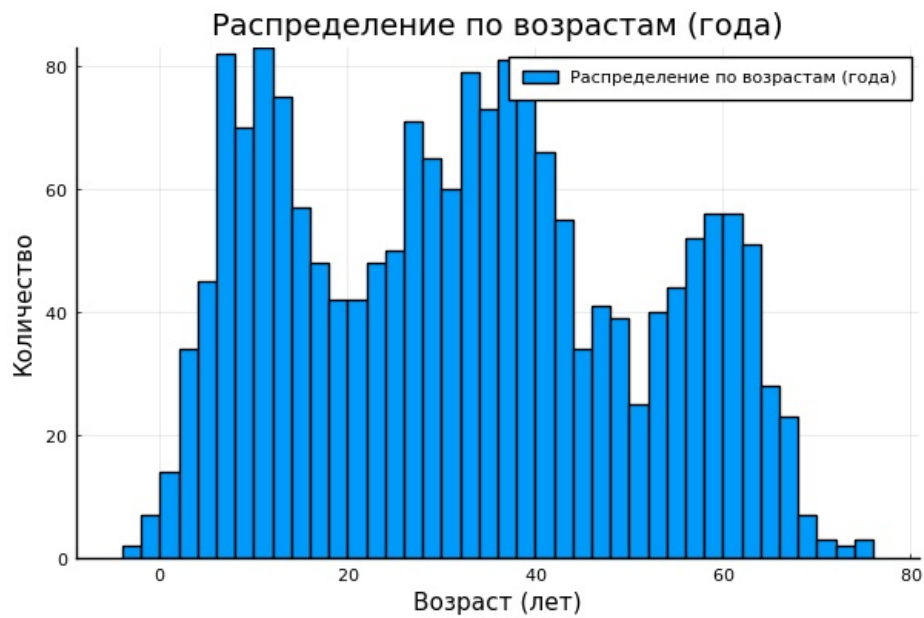
Использование пакета Distributions

```
In [39]: using Distributions
pyplot()
ages = rand(15:55,1000)
histogram(ages)
d=Normal(35.0,10.0)
ages = rand(d,1000)
histogram(
    ages,
    label="Распределение по возрастам (года)",
    xlabel = "Возраст (лет)",
    ylabel= "Количество"
)
```



```
In [40]: pyplot()
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)"
)
```

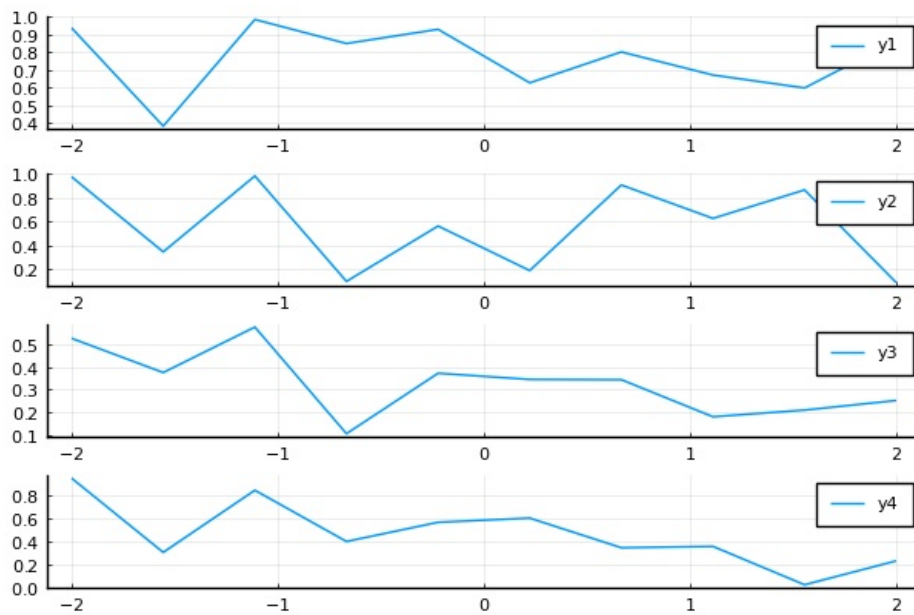
Out[40]:



Подграфики

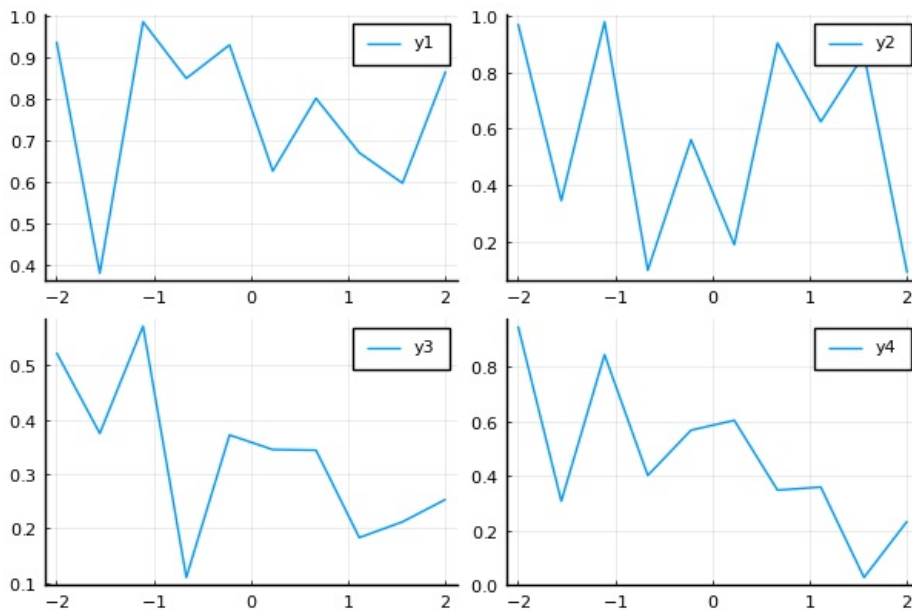
```
In [41]: # подгружаем pyplot():
pyplot()
# построение серии графиков:
x=range(-2,2,length=10)
y = rand(10,4)
plot(x,y,
      layout=(4,1)
)
```

Out[41]:



```
In [42]: plot(x,y,
              layout=4
            )
```

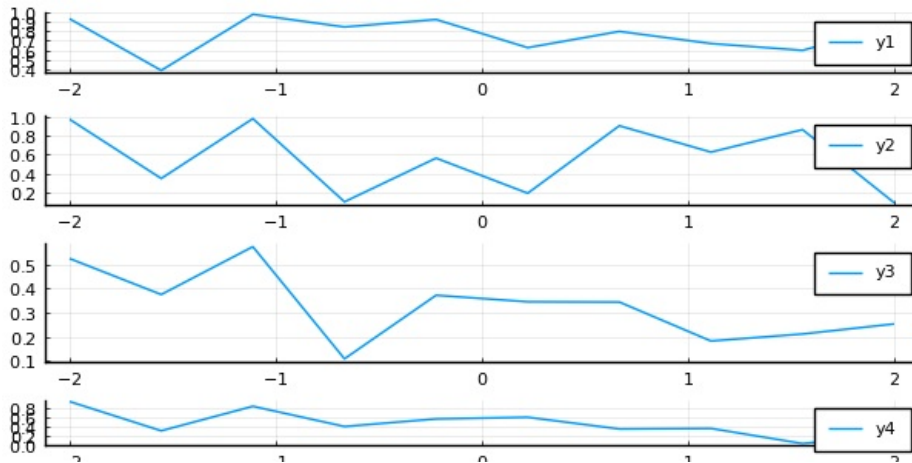
Out [42]:



In [43]:

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

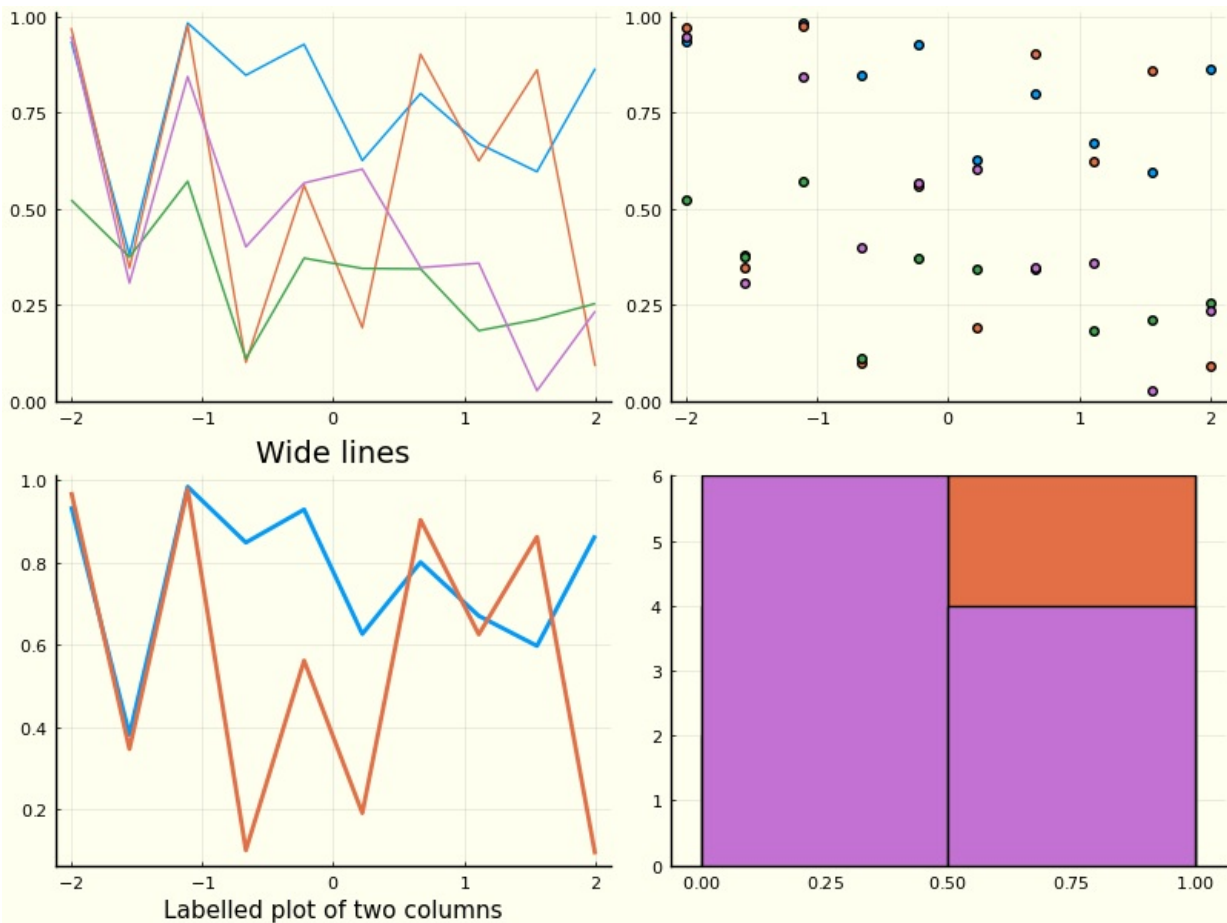
Out [43]:



In [44]:

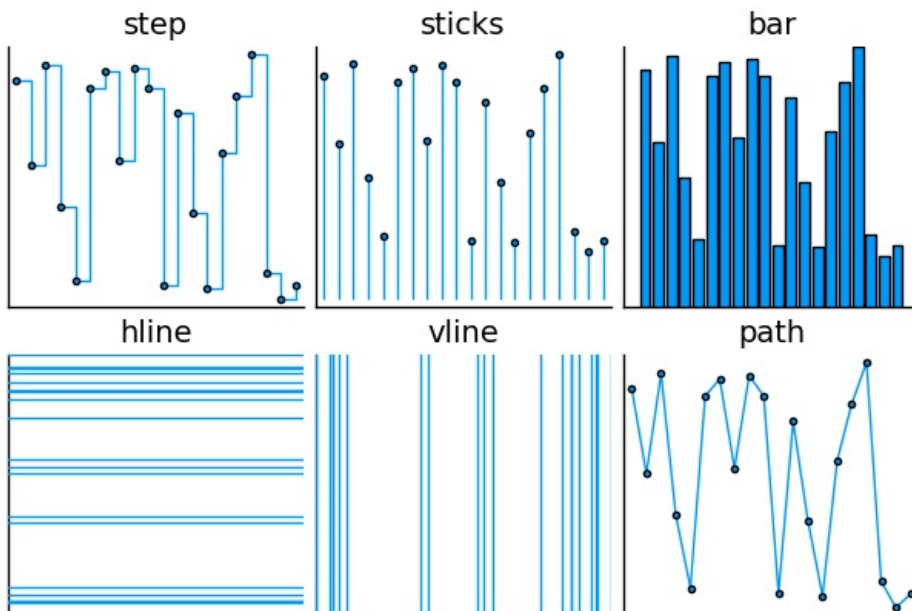
```
# график в виде линий:
p1 = plot(x,y)
# график в виде точек:
p2 = scatter(x,y)
# график в виде линий с оформлением:
p3 = plot(x,y[:,1:2],xlabel="Labelled plot of two columns",lw=2,title="Wide lines")
# 4 гистограммы:
p4 = histogram(x,y)
plot(p1,p2,p3,p4,
      layout=(2,2),
      legend=false,
      size=(800,600),
      background_color = :ivory
)
```

Out[44]:



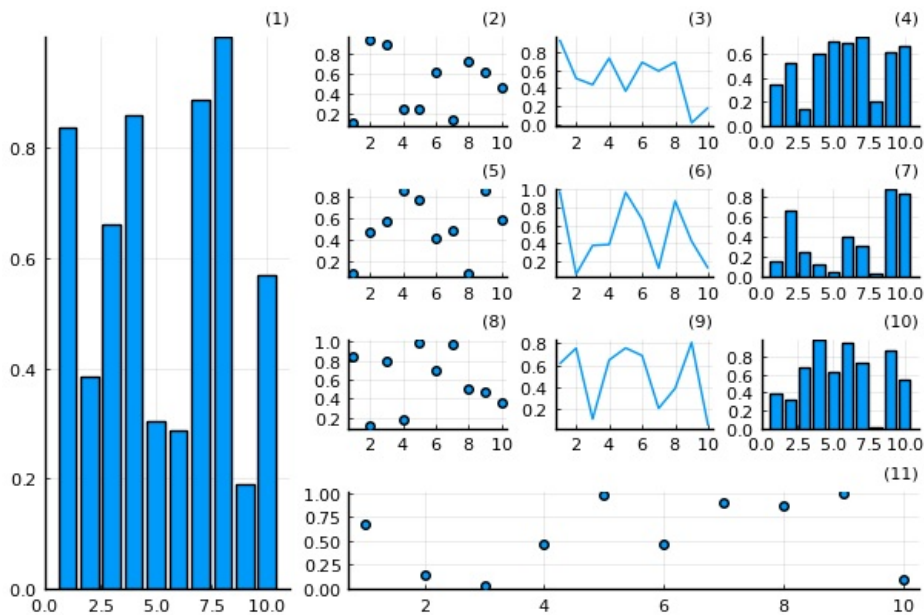
```
In [45]: seriestypes = [:step, :sticks, :bar, :hline, :vline, :path]
titles = ["step" "sticks" "bar" "hline" "vline" "path"]
plot(rand(20,1), st = seriestypes,
      layout = (2,3),
      ticks=nothing,
      legend=false,
      title=titles,
      m=3
)
```

Out[45]:



```
In [46]: 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 [46]:

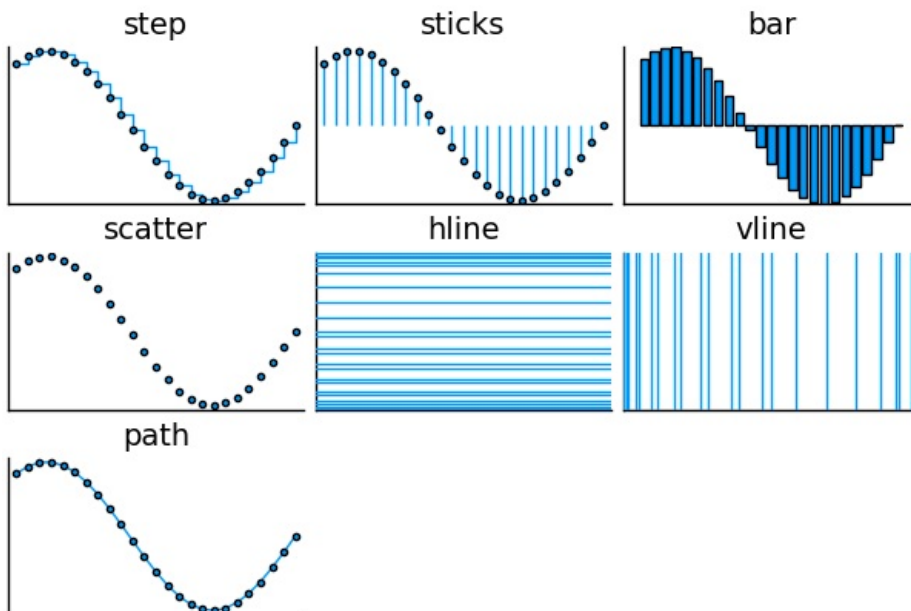


Самостоятельная работа

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

```
In [47]: seriestypes = [:step, :sticks, :bar, :scatter, :hline, :vline, :path]
titles = ["step" "sticks" "bar" "scatter" "hline" "vline" "path"]
x_values = collect(range(1, stop=2*pi, length=25))
y2(x) = sin(x)
plot(x_values, y2,
     st = seriestypes,
     layout = 7,
     ticks=nothing,
     legend=false,
     title=titles,
     m=3
)
```

Out [47]:



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

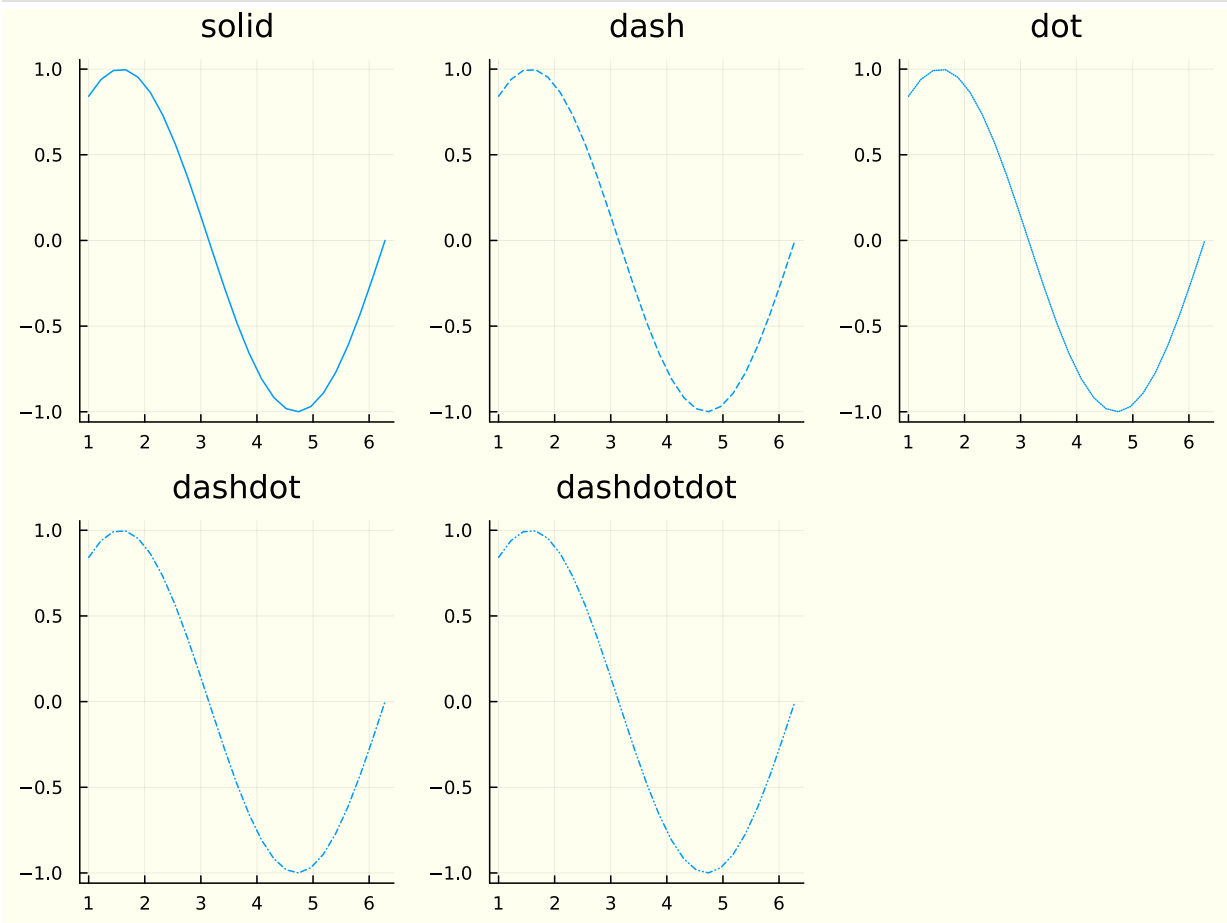
```
In [56]: lss = [:solid, :dash, :dot, :dashdot, :dashdotdot]
titles = ["solid" "dash" "dot" "dashdot" "dashdotdot"]
gr()
p1 = plot(x_values, y2,
         ls = lss[1],
         title = titles[1],)
p2 = plot(x_values, y2,
```

```

ls = lss[2],
title = titles[2],)
p3 = plot(x_values, y2,
ls = lss[3],
title = titles[3],)
p4 = plot(x_values, y2,
ls = lss[4],
title = titles[4],)
p5 = plot(x_values, y2,
ls = lss[5],
title = titles[5],)
plot(p1,p2,p3,p4,p5,
layout=5,
legend=false,
size=(800,600),
background_color = :ivory
)

```

Out[56]:



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

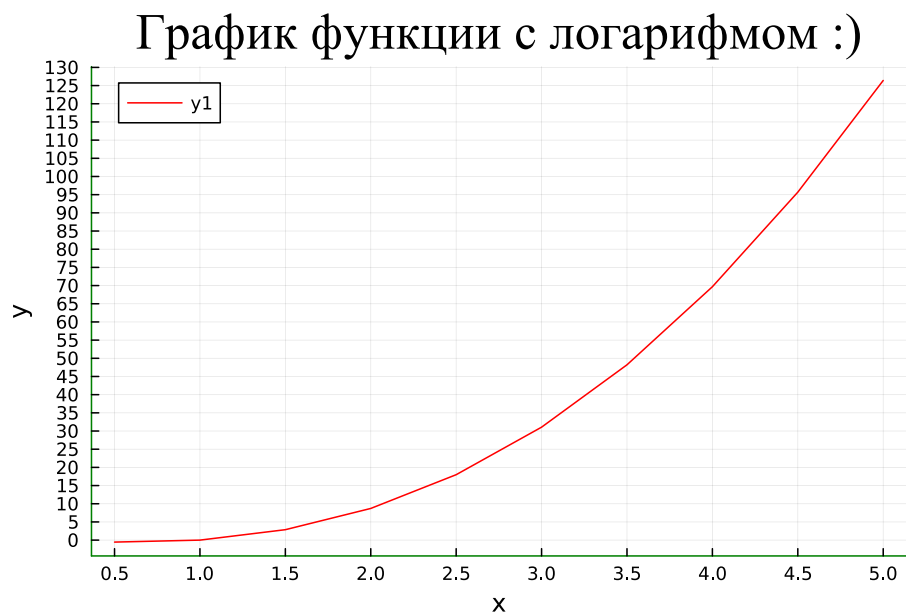
In [60]:

```

y3(x) = pi*x^2*log(x)
x_values = collect(0.5:0.5:5)
plot(x_values, y3,
foreground_color_border = :green,
ylabel = "y",
xlabel = "x",
titlefont = (20, "times"),
title = "График функции с логарифмом :)",
color=:red,
xticks = (0:0.5:5),
yticks = (0:5:130),
)

```

Out[60]:



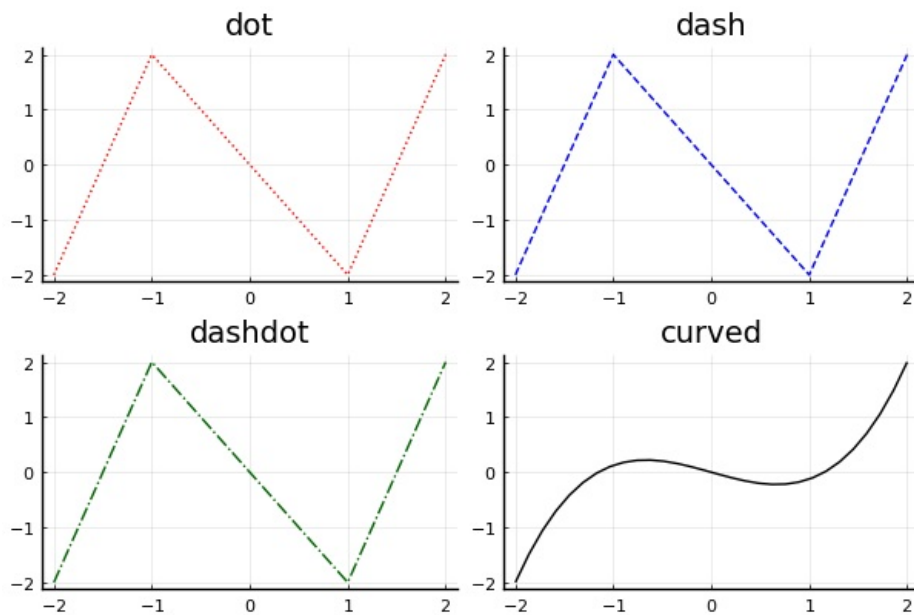
4. Задайте вектор $x = (-2, -1, 0, 1, 2)$. В одном графическом окне (в 4-х подокнах) изобразите графически по точкам x значения функции $y(x) = x^3 - 3x$ в виде:

- точек,
- линий,
- линий и точек,
- кривой.

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

```
In [127... x_values = [-2, -1, 0, 1, 2] # x_values = collect(-2:2)
y4(x) = x^3 - 3*x
titles = ["curved" "dash" "dot" "dashdot"]
p1 = curves(x_values, y4,
            title = titles[1],
            color = :black)
p2 = plot(x_values, y4,
          ls = :dash,
          title = titles[2],
          color = :blue)
p3 = plot(x_values, y4,
          ls = :dot,
          title = titles[3],
          color = :red)
p4 = plot(x_values, y4,
          ls = :dashdot,
          title = titles[4],
          color = :darkgreen)
plot(p3, p2, p4, p1,
     layout=4,
     legend=false,
)
```

Out[127...

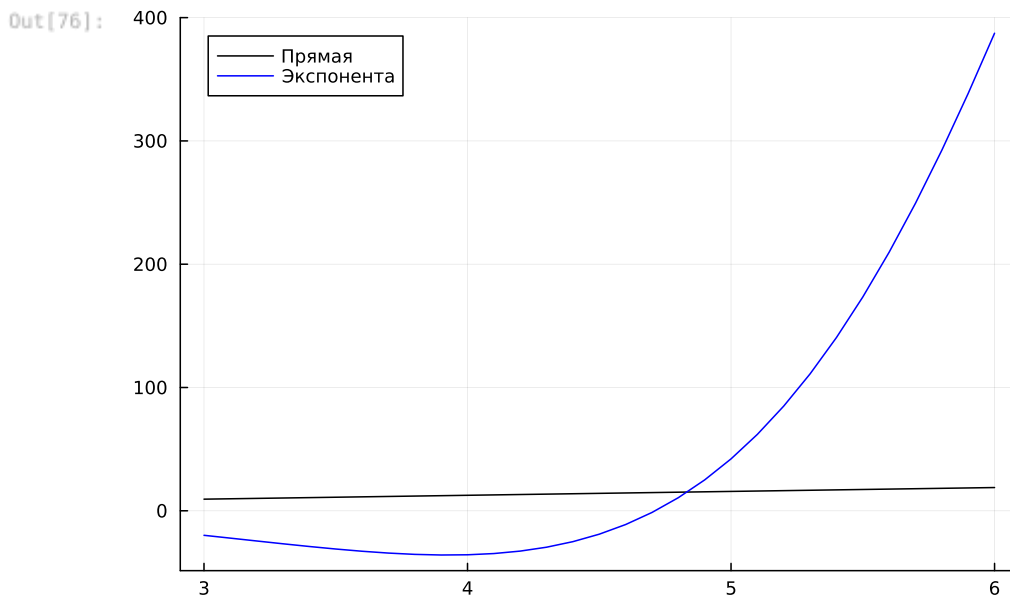


In [75]: `savefig("figure_Manaveva.png")`

Out[75]: "D:\\Education\\КомпПрактикумПоСтатМоделированию\\labs\\gitrepo\\lab5\\figure_Manaveva.png"

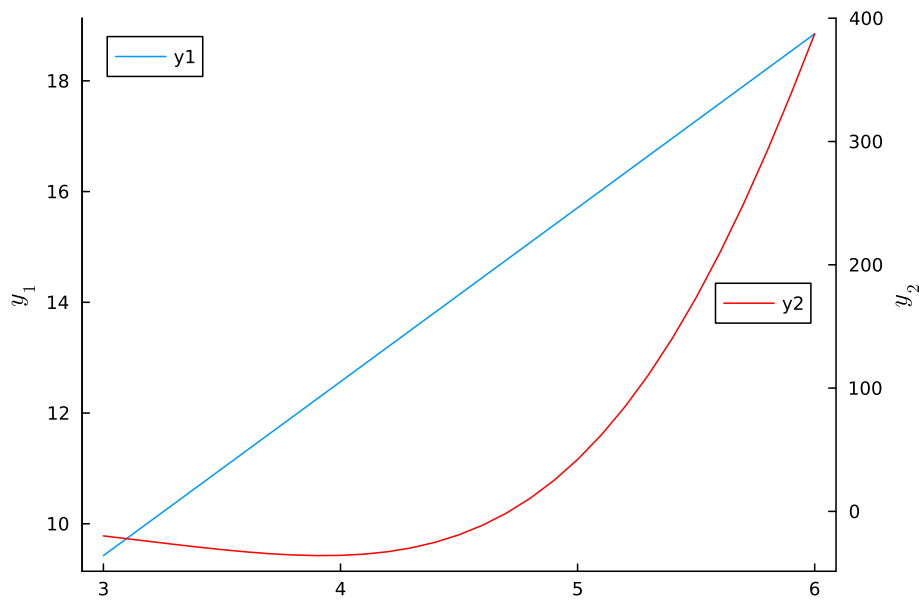
5. Задайте вектор $x = (3, 3.1, 3.2, \dots, 6)$. Постройте графики функций $y_1(x) = \pi x$ и $y_2(x) = e^x \cos(x)$ в указанном диапазоне значений аргумента x двумя разными способами

```
In [76]: x_v = collect(3:0.1:6)
y_1(x) = pi*x
y_2(x) = exp(x)*cos(x)
p1 = plot(x_v, y_1,
          label = "Прямая",
          color = :black)
p2 = plot!(x_v, y_2,
          label = "Экспонента",
          color = :blue)
```



```
In [81]: plot(x_v, y_1,
              ylabel="\$y_1\$",
              leg=:topleft,
              grid = :off,
              )
plot!(twinx(), x_v, y_2,
      c=:red,
      ylabel="\$y_2\$",
      leg=:right,
      grid = :off,
      box = :on,
      )
```

Out[81]:

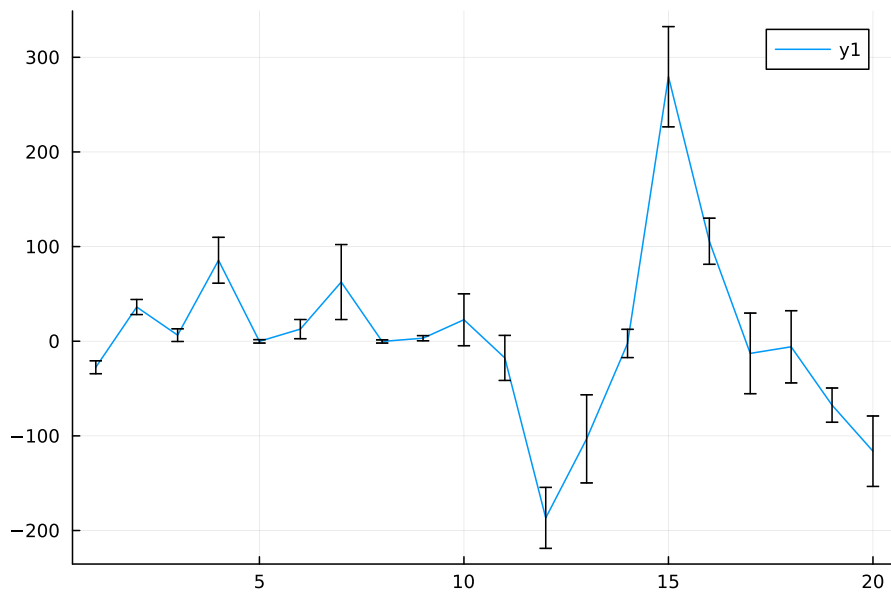


6. Постройте график некоторых экспериментальных данных (придумайте сами), учитывая ошибку измерения.

In [87]:

```
sds = collect(rand(1:500, 20))
n = 10
y = [mean(sd*randn(n)) for sd in sds]
errs = 0.36 * sds / sqrt(n)
plot(y,
err = errs
)
```

Out[87]:



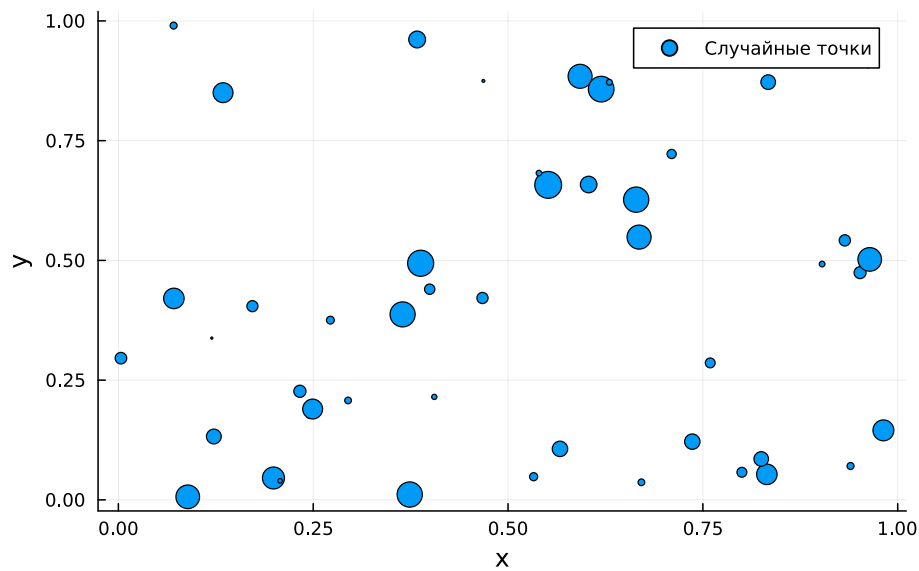
7. Постройте точечный график случайных данных. Подпишите оси, легенду, название графика.

In [88]:

```
n = 50
x = rand(n)
y = rand(n)
ms = rand(n) * 10
# параметры построения графика:
scatter(x, y,
label = "Случайные точки",
leg=:topright,
ylabel = "y",
xlabel = "x",
title = "Случайные точки в 2D пространстве",
markersize=ms)
```

Out[88]:

Случайные точки в 2D пространстве



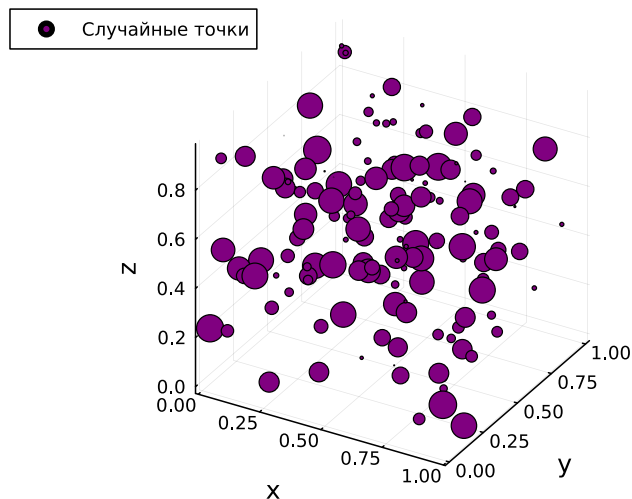
8. Постройте 3-мерный точечный график случайных данных. Подпишите оси, легенду, название графика.

In [93]:

```
n = 150
x = rand(n)
y = rand(n)
z = rand(n)
ms = rand(n) * 10
# параметры построения графика:
scatter(x, y, z,
        label = "Случайные точки",
        leg=:topleft,
        ylabel = "y",
        xlabel = "x",
        zlabel = "z",
        title = "Случайные точки в 3D пространстве",
        markersize=ms,
        mc = :purple
)
```

Out[93]:

Случайные точки в 3D пространстве



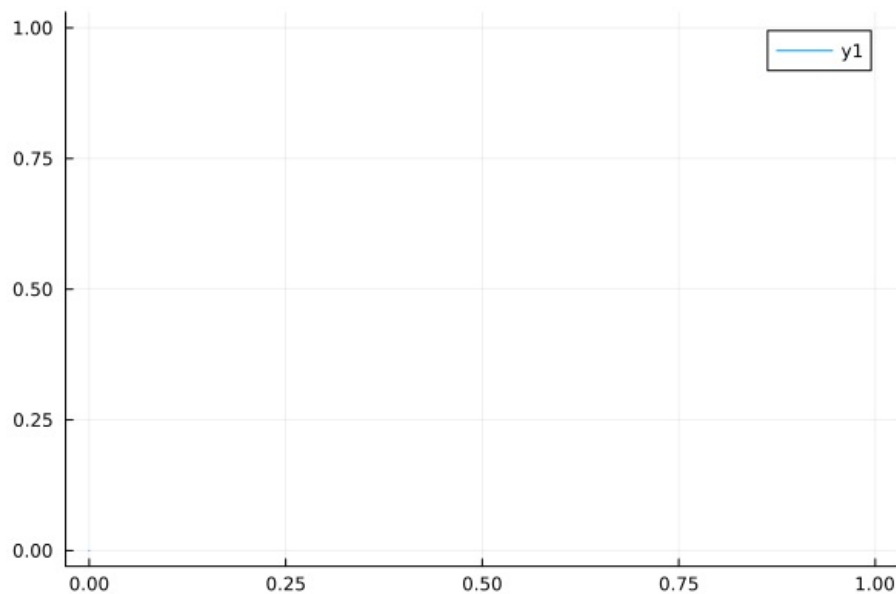
9. Создайте анимацию с построением синусоиды. То есть, постройте последовательность графиков синусоиды, постепенно увеличивая значение аргумент, пПосл чегое соединит их в анимацию.

In [100]:

```
@gif for i in range(0, stop=10π, length=300)
    x_v = range(0, stop=i, length=200)
    y_(x) = sin(x)
    plot(x_v, y_)
end
```

[Info: Saved animation to D:\Education\КомпПрактикумПоСтатМоделированию\labs\gitrepo\lab5\tmp.gif

Out[100...



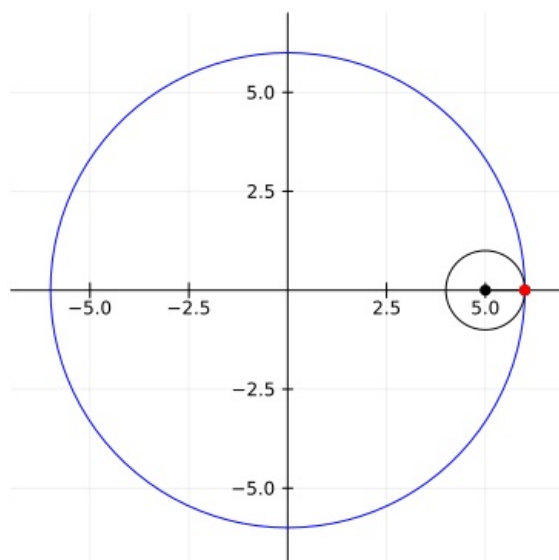
10. Постройте анимированную гипоциклоиду для 2 целых значений модуля k и 2 рациональных значений модуля k .

In [104...

```
r1 = 1
k = 6
n = 100
θ = collect(0:2*π/100:2*π+2*π/100)
X = r1*k*cos.(θ)
Y = r1*k*sin.(θ)
anim = @animate for i in 1:n
    # задаём оси координат:
    plt=plot(5,xlim=(-k-1,k+1),ylim=(-k-1,k+1), c=:red, aspect_ratio=1,legend=false, framestyle=:origin)
    # большая окружность:
    plot!(plt, X,Y, c=:blue, legend=false)
    t = θ[1:i]
    # гипоциклоида:
    x = r1*(k-1)*cos.(t) + r1*cos.((k-1)*t)
    y = r1*(k-1)*sin.(t) - r1*sin.((k-1)*t)
    plot!(x,y, c=:red)
    # малая окружность:
    xc = r1*(k-1)*cos(t[end]) .+ r1*cos.(θ)
    yc = r1*(k-1)*sin(t[end]) .+ r1*sin.(θ)
    plot!(xc,yc,c=:black)
    # радиус малой окружности:
    xl = transpose([r1*(k-1)*cos(t[end]) x[end]])
    yl = transpose([r1*(k-1)*sin(t[end]) y[end]])
    plot!(xl,yl,markershape=:circle,markersize=4,c=:black)
    scatter!([x[end]], [y[end]], c=:red, markerstrokecolor=:red)
end
gif(anim, "hypocycloid_1.gif")
```

[Info: Saved animation to D:\Education\КомпПрактикумПоСтатМоделированию\labs\gitrepo\lab5\hypocycloid.gif

Out[104...



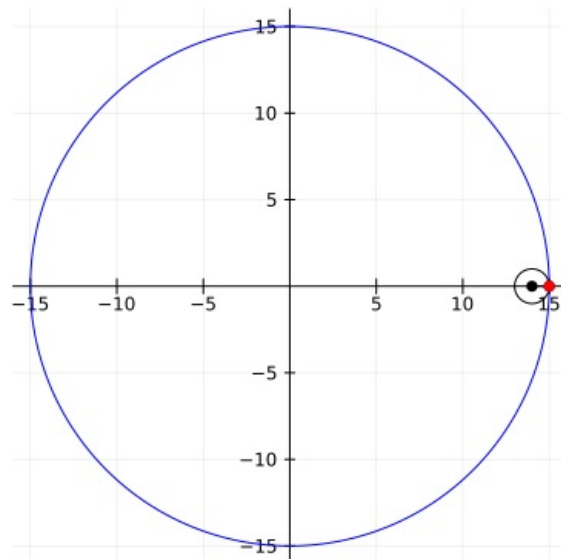
```

In [107... r1 = 1
k = 15
n = 200
θ = collect(0:2*π/n:2*π+2*π/n)
X = r1*k*cos.(θ)
Y = r1*k*sin.(θ)
anim = @animate for i in 1:n
    # задаём оси координат:
    plt=plot(5,xlim=(-k-1,k+1),ylim=(-k-1,k+1), c=:red, aspect_ratio=1,legend=false, framestyle=:origin)
    # большая окружность:
    plot!(plt, X,Y, c=:blue, legend=false)
    t = θ[1:i]
    # гипоциклоида:
    x = r1*(k-1)*cos.(t) + r1*cos.((k-1)*t)
    y = r1*(k-1)*sin.(t) - r1*sin.((k-1)*t)
    plot!(x,y, c=:red)
    # малая окружность:
    xc = r1*(k-1)*cos(t[end]) .+ r1*cos.(θ)
    yc = r1*(k-1)*sin(t[end]) .+ r1*sin.(θ)
    plot!(xc,yc,c=:black)
    # радиус малой окружности:
    xl = transpose([r1*(k-1)*cos(t[end]) x[end]])
    yl = transpose([r1*(k-1)*sin(t[end]) y[end]])
    plot!(xl,yl,markershape=:circle,markersize=4,c=:black)
    scatter!([x[end]], [y[end]], c=:red, markerstrokecolor=:red)
end
gif(anim, "hypocycloid_2.gif")

```

[Info: Saved animation to D:\Education\КомпПрактикумПоСтатМоделированию\labs\gitrepo\lab5\hypocycloid_1.gif

Out[107...



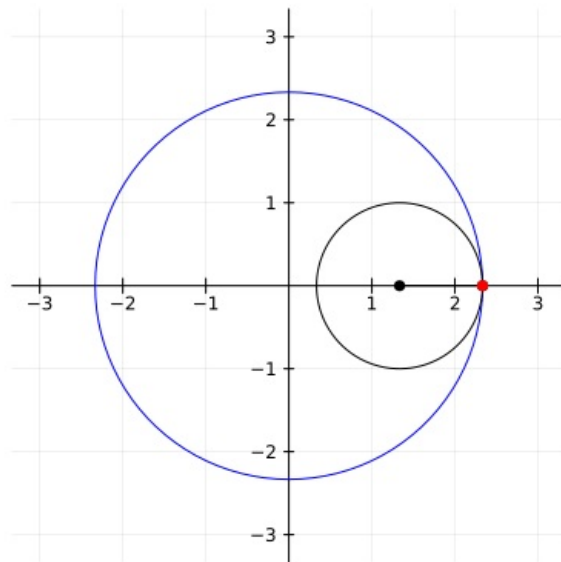
```

In [112... r1 = 1
k = 7/3
n = 100
θ = collect(0:2*π/n:20*π+20*π/n)
X = r1*k*cos.(θ)
Y = r1*k*sin.(θ)
anim = @animate for i in 1:n
    # задаём оси координат:
    plt=plot(5,xlim=(-k-1,k+1),ylim=(-k-1,k+1), c=:red, aspect_ratio=1,legend=false, framestyle=:origin)
    # большая окружность:
    plot!(plt, X,Y, c=:blue, legend=false)
    t = θ[1:i]
    # гипоциклоида:
    x = r1*(k-1)*cos.(t) + r1*cos.((k-1)*t)
    y = r1*(k-1)*sin.(t) - r1*sin.((k-1)*t)
    plot!(x,y, c=:red)
    # малая окружность:
    xc = r1*(k-1)*cos(t[end]) .+ r1*cos.(θ)
    yc = r1*(k-1)*sin(t[end]) .+ r1*sin.(θ)
    plot!(xc,yc,c=:black)
    # радиус малой окружности:
    xl = transpose([r1*(k-1)*cos(t[end]) x[end]])
    yl = transpose([r1*(k-1)*sin(t[end]) y[end]])
    plot!(xl,yl,markershape=:circle,markersize=4,c=:black)
    scatter!([x[end]], [y[end]], c=:red, markerstrokecolor=:red)
end
gif(anim, "hypocycloid_3.gif")

```

[Info: Saved animation to D:\Education\КомпПрактикумПоСтатМоделированию\labs\gitrepo\lab5\hypocycloid_3.gif

Out[112...



In [114...

```
r1 = 1
k = 3/7
n = 100
θ = collect(0:2*π/n:20*π+20*π/n)
X = r1*k*cos.(θ)
Y = r1*k*sin.(θ)
anim = @animate for i in 1:n
    # задаём оси координат:
    plt=plot(5,xlim=(-k-1,k+1),ylim=(-k-1,k+1), c=:red, aspect_ratio=1,legend=false, framestyle=:origin)
    # большая окружность:
    plot!(plt, X,Y, c=:blue, legend=false)
    t = θ[1:i]
    # гипоциклоида:
    x = r1*(k-1)*cos.(t) + r1*cos.((k-1)*t)
    y = r1*(k-1)*sin.(t) - r1*sin.((k-1)*t)
    plot!(x,y, c=:red)
    # малая окружность:
    xc = r1*(k-1)*cos(t[end]) .+ r1*cos.(θ)
    yc = r1*(k-1)*sin(t[end]) .+ r1*sin.(θ)
    plot!(xc,yc,c=:black)
    # радиус малой окружности:
    xl = transpose([r1*(k-1)*cos(t[end]) x[end]])
    yl = transpose([r1*(k-1)*sin(t[end]) y[end]])
    plot!(xl,yl,markershape=:circle,markersize=4,c=:black)
    scatter!([x[end]], [y[end]], c=:red, markerstrokecolor=:red)
end
gif(anim, "hypocycloid_4.gif")
```

[Info: Saved animation to D:\Education\КомпПрактикумПоСтатМоделированию\labs\gitrepo\lab5\hypocycloid_4.gif

Out[114...

