# 1 Plots und einfache Datenvisualisierung in Julia

- Frage "Wie plottet man in Julia?" inzwischen genauso sinnvoll wie "Wie plottet man in C++?";
- es gibt zahlreiche Pakete, die unterschiedlich geeignet sind je nach
  - Ausgabemedium: Grafikkarte, PDF, SVG, PNG, WebApp,....
  - Interaktivität
  - 2D- und 3D-Fähigkeiten
  - Plottypen
  - Umgang mit großen Datenmengen ...

## 1.1 Einbindung anderer Grafikbibliotheken

### 1.1.1 JavaScript: Interaktivität im Browser

- PlotlyJS.jl Interface zur PlotlyJS-Grafikbibliothek
- Bokeh.jl Interface zur Bokeh-Grafikbibliothek
- VegaLite.jl Interface zu VegaLite, vor allem statistische Plots

### 1.1.2 Cairo: 2D Vektorgrafik, für Grafikkarte (screen), PDF, PNG, SVG,...

- Luxor.jl für Vektorgrafik
- Javis.jl für animierte Vektorgrafik

### 1.1.3 Matplotlib (Python)

- PyPlot.jl
  - weitgehende 1:1-Übertragung der Python-API, deswegen wird auch auf die Matplotlib-Dokumentation verwiesen
  - Beispiele mit Gegenüberstellung Python/Julia: `https://gist.github.com/gizmaa/7214002`

## 1.2 Pure Julia: Makie.jl

`Makie` bezeichnet sich selbst als *"data visualization ecosystem for Julia"*

Es ist vollständig in Julia geschrieben und bietet als *backends* `Cairo` (Vektorgrafik), `OpenGL` und `WebGL` an.

- Makie.jl
- Beautiful Makie eine Seite mit vielen Beispielen

# 2 Plots.jl

- Plots.jl ist konzipiert als ein einheitliches Interface zu verschiedenen *backends* (Grafikbibliotheken). > Man kann zu einem anderen *backend* wechseln und dieselben Plot-Kommandos und -Attribute verwenden.
- Einige *backends*:
  - GR
  - PyPlot (d.h., Matplotlib)
  - Plotly(JS)

Plots.jl schien mir am besten geeignet für den Workflow

`Jupyter notebooks` ⇒ `Konvertierung nach HTML/LaTeX/PDF`

**Im Rest dieses Notebooks wird Plots.jl vorgestellt.**

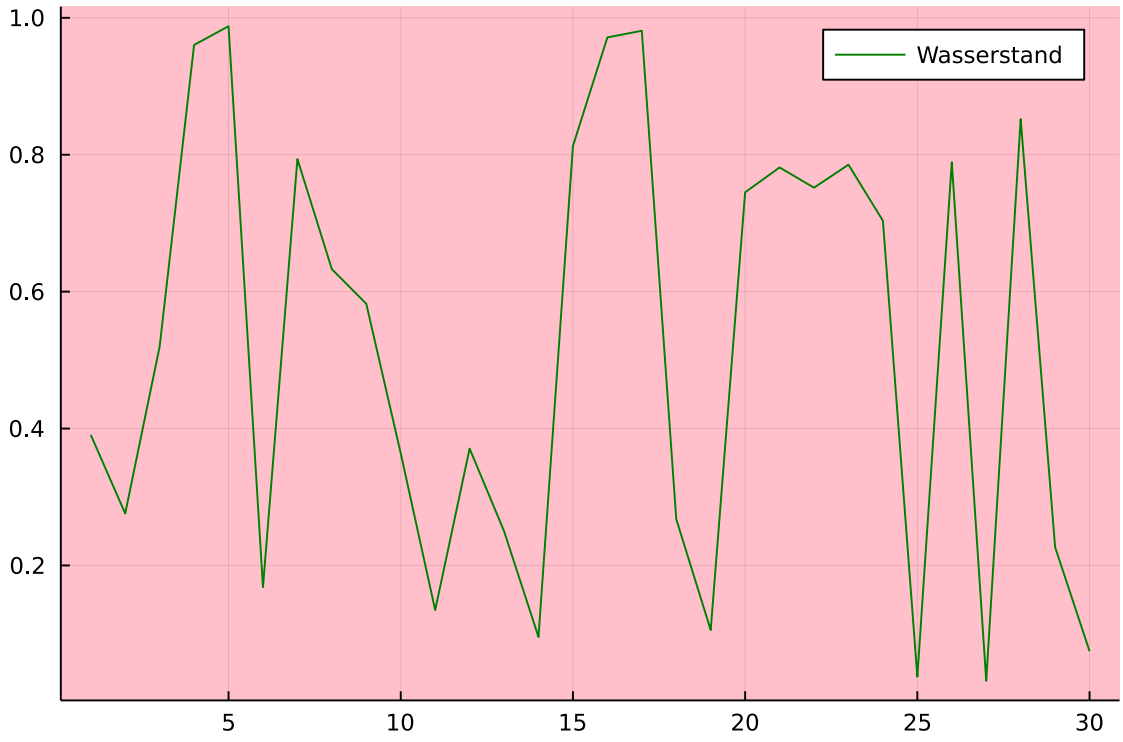
## 2.1 einige *backends*

[1]:
```
using Plots
backend()      # Anzeige des gewählten backends, GR ist der default
```

[1]:
```
Plots.GRBackend()
```

[2]:
```
x = 1:30
y = rand(30)

plot(x, y, linecolor =:green, bg_inside =:pink, line =:solid, label = "Wasserstand")
```

[2]:




[3]:
```
# wir wechseln das backend zu
# plotly/js


plotly()
```

```
┌ Info: For saving to png with the Plotly backend PlotlyBase has to be
installed.
└ @ Plots /home/hellmund/.julia/packages/Plots/MzlNY/src/backends.jl:318
```

[3]:
```
Plots.PlotlyBackend()
```

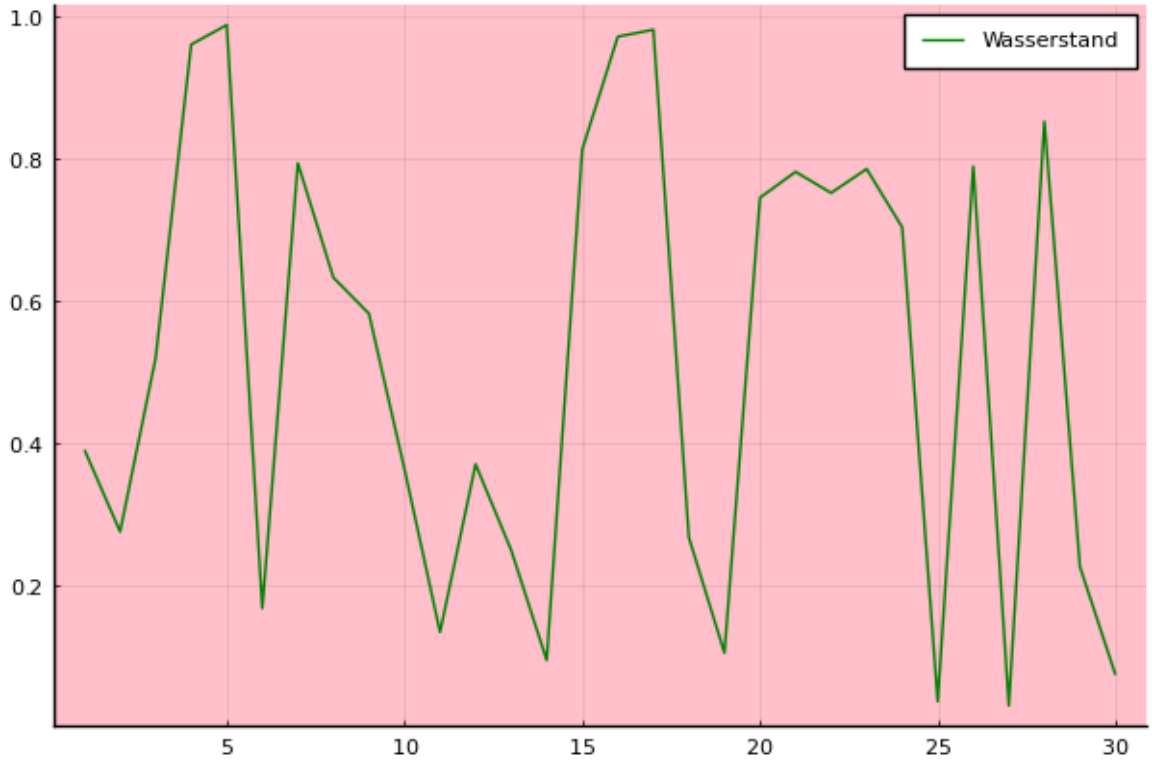
[4]:
```
# dasselbe Plot-Kommando

# das ist interaktiv (zoom in/out, pan),
# 'überlebt' aber leider die PDF-Konvertierung des notebooks nicht

plot(x, y, linecolor =:green, bg_inside =:pink, line =:solid, label = "Wasserstand")
```

[5]:
```
# und noch ein backend

pyplot()
```

[5]:
```
Plots.PyPlotBackend()
```

[6]:
```
plot(x, y, linecolor =:green, bg_inside =:pink, line =:solid, label = "Wasserstand")
```

[6]:




[7]:
```
# zurück zu GR als backend

gr()
```

[7]:
```
Plots.GRBackend()
```

## 2.2 Plots.jl und recipes

Andere Pakete können die Möglichkeiten von `Plots.jl` erweitern, indem sie *recipes* für spezielle Plots und Datenstrukturen definieren, siehe https://docs.juliaplots.org/latest/ecosystem/, z.B.:

- `StatsPlots.jl` direktes Plotten von Dataframes, spezielle statistische Plots,...
- `GraphRecipes.jl` Plotten von Graphstrukturen
- ...

## 2.3 Einige Verschönerungen

[8]:
```
using Plots.PlotMeasures   # für Angaben in mm, cm,...
using LaTeXStrings         # für LaTeX-Konstrukte in Plot-Beschriftungen
using PlotThemes           # vorgefertigte Themen
```

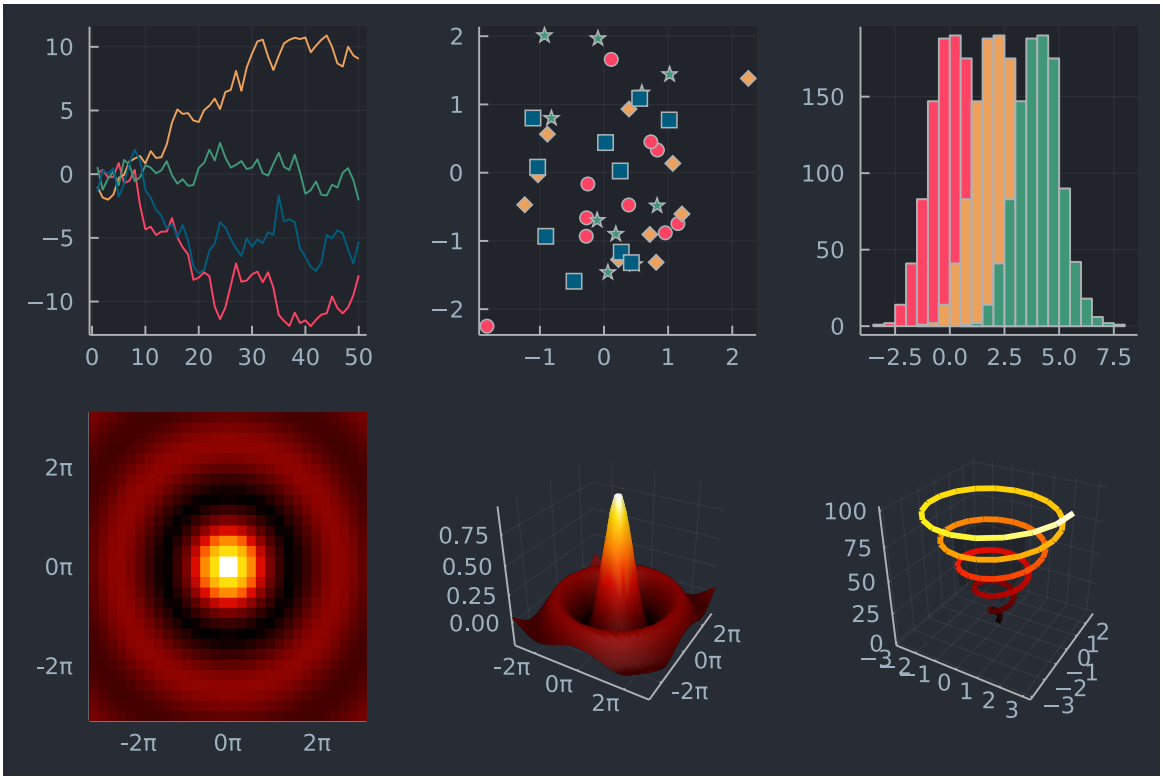
[9]:
```
# Liste der Themen
keys(PlotThemes._themes)
```

[9]:
```
KeySet for a Dict{Symbol, PlotThemes.PlotTheme} with 18 entries. Keys:
  :juno
  :default
  :dao
  :ggplot2
  :gruvbox_dark
  :dark
  :gruvbox_light
  :solarized
  :wong
```

```
:dracula
:solarized_light
:mute
:wong2
:sand
:lime
:bright
:vibrant
:orange
```

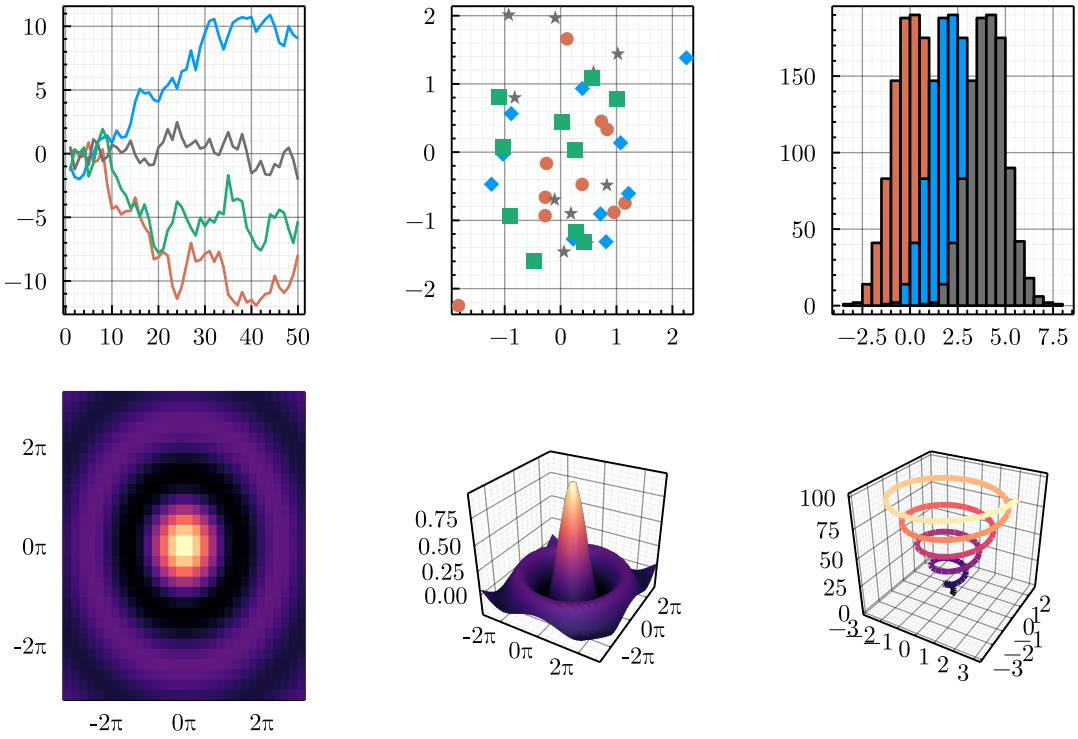
[10]:
```
Plots.showtheme(:juno)
```

[10]:




[11]:
```
Plots.showtheme(:dao)
```

[11]:




[12]:
```
# so legt man ein Thema für die folgenden Plots fest:

theme(:dao)

# Wir wollen es wieder langweilig haben...

theme(:default)
```

## 2.4 Funktionen in Plots.jl

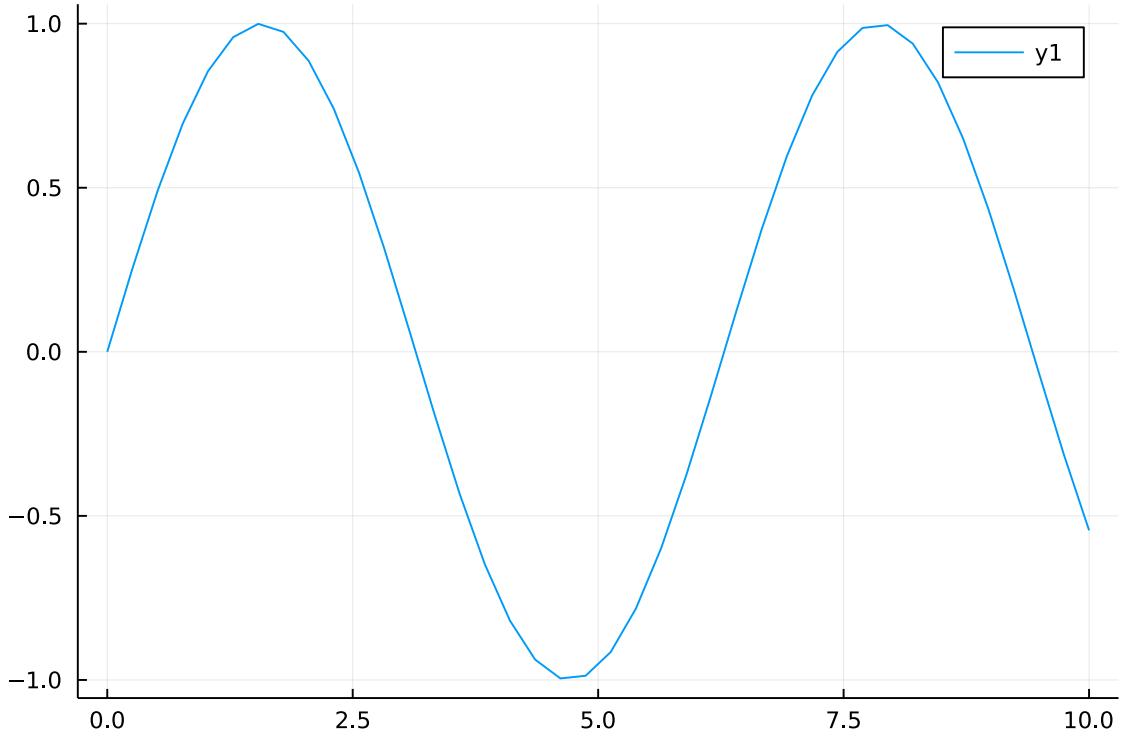
```
plot()
scatter()
contour()
heatmap()
histogram()
bar()
plot3d()
... und weitere
```

Diese Funktionen erzeugen ein neues `Plot`-Objekt.

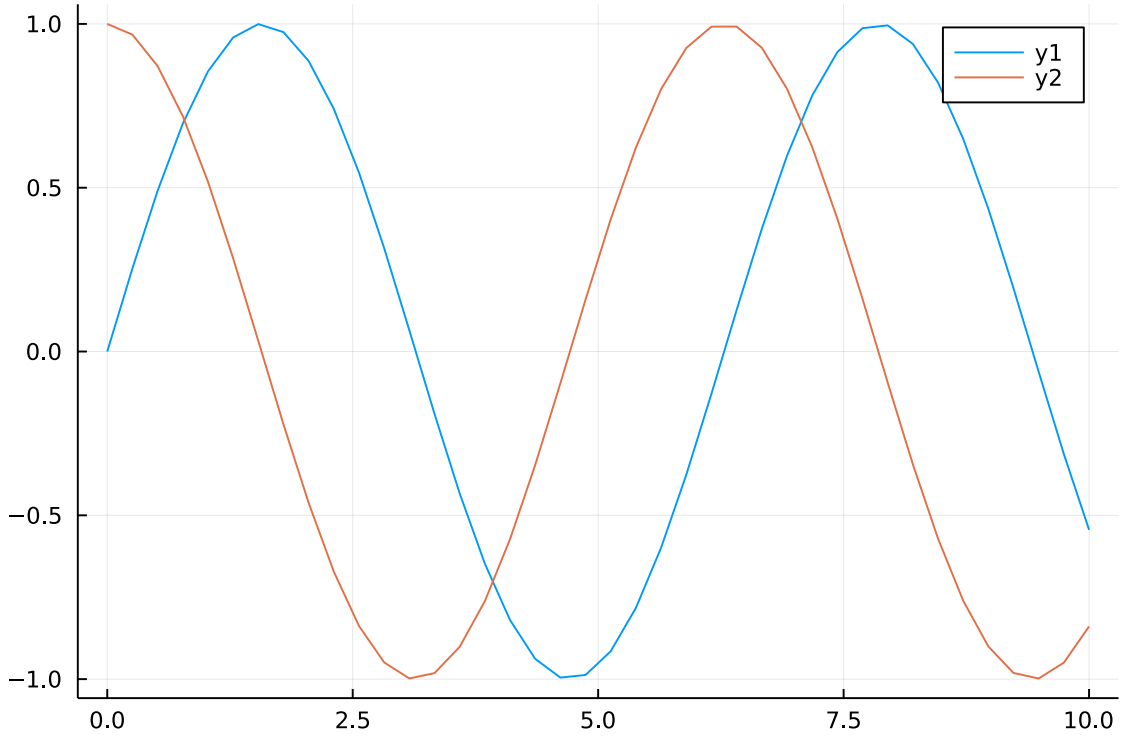
Die Varianten mit Ausrufezeichen `plot!(), scatter!(),...` modifizieren das letzte `Plot`-Objekt oder das `Plot`-Objekt, das ihnen als 1. Argument übergeben wird:

[13]:
```
x = range(0,10, length=40)   # 40 x-Werte von 0 bis 10

pl1 = plot(x, sin.(x))
```

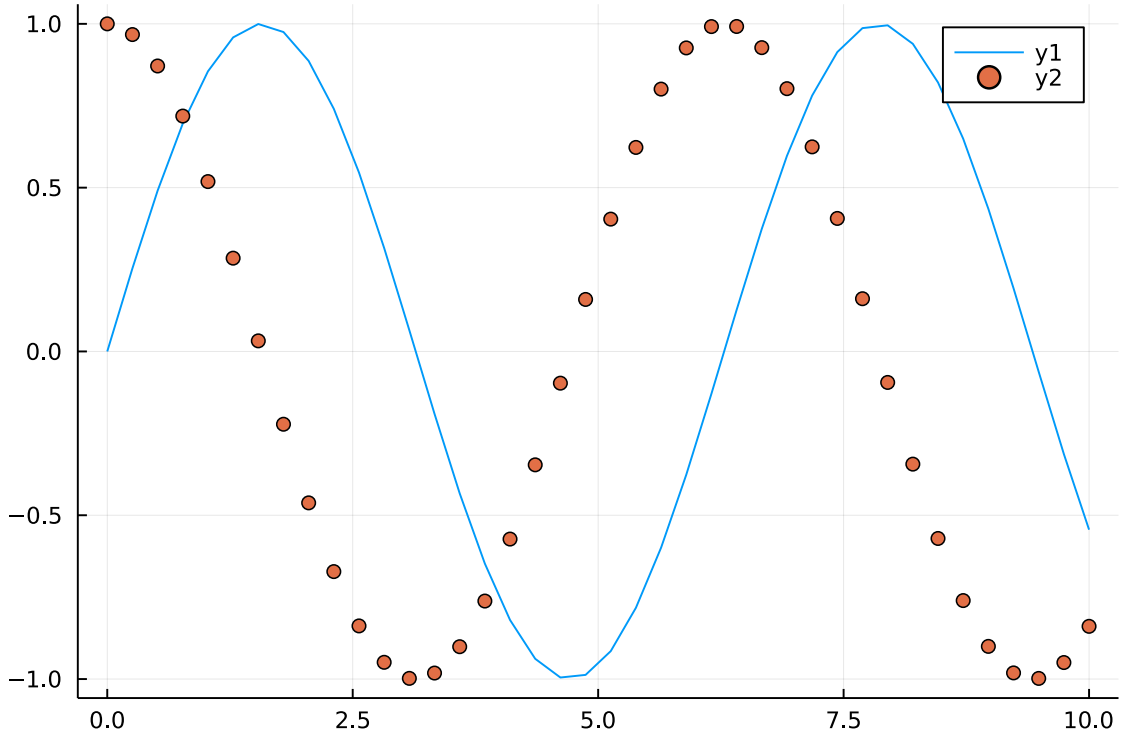
[13]:




[14]:
```
pl1a = deepcopy(pl1)  # unmodifizierte copy aufheben

pl2 = plot!(x, cos.(x))    # modifiziert pl1
```
[14]:




[15]:
```
pl3 = scatter!(pl1a, x, cos.(x))   # add to (copy of) original Plot
```
[15]:




Plot-Objekte kann man als Grafikdateien (PDF, SVG, PNG,...) abspeichern:

[16]:
```
savefig(pl2, "pl2.png")
```

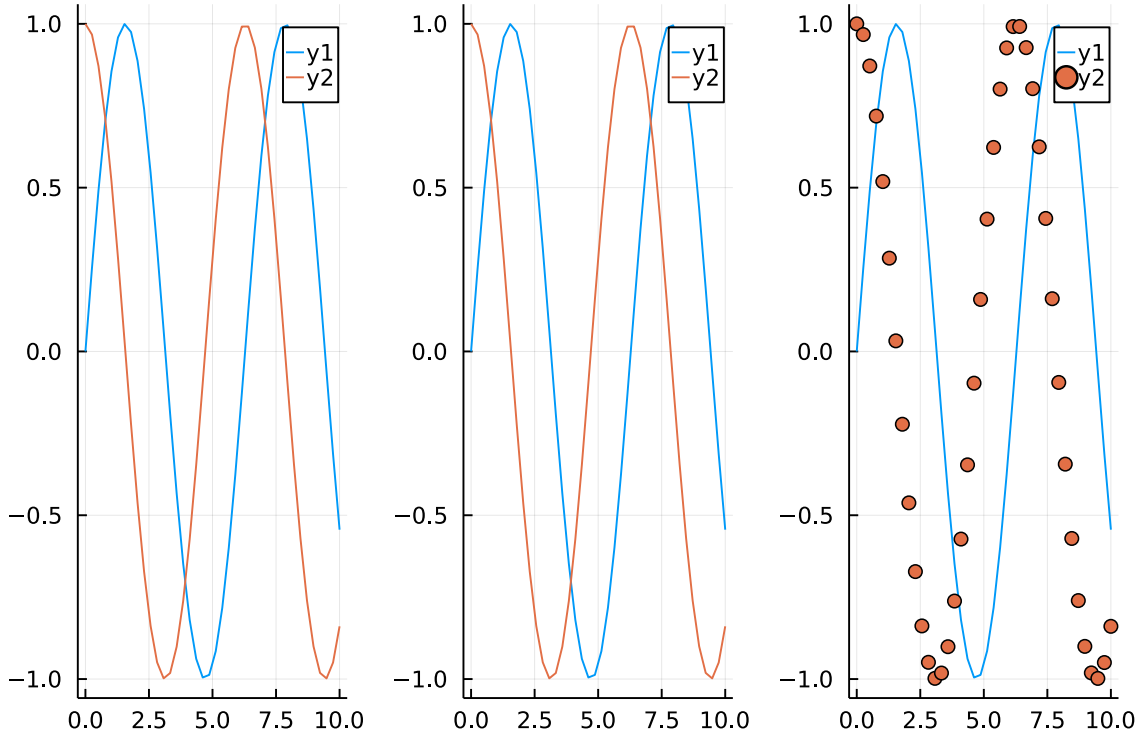
[17]:
```
;ls -l pl2.png
```

```
-rw-r--r-- 1 hellmund hellmund 30634 Jun 28 13:09 pl2.png
```

... oder zB als Sub-Plots mit einem layout-Parameter zusammenfügen:

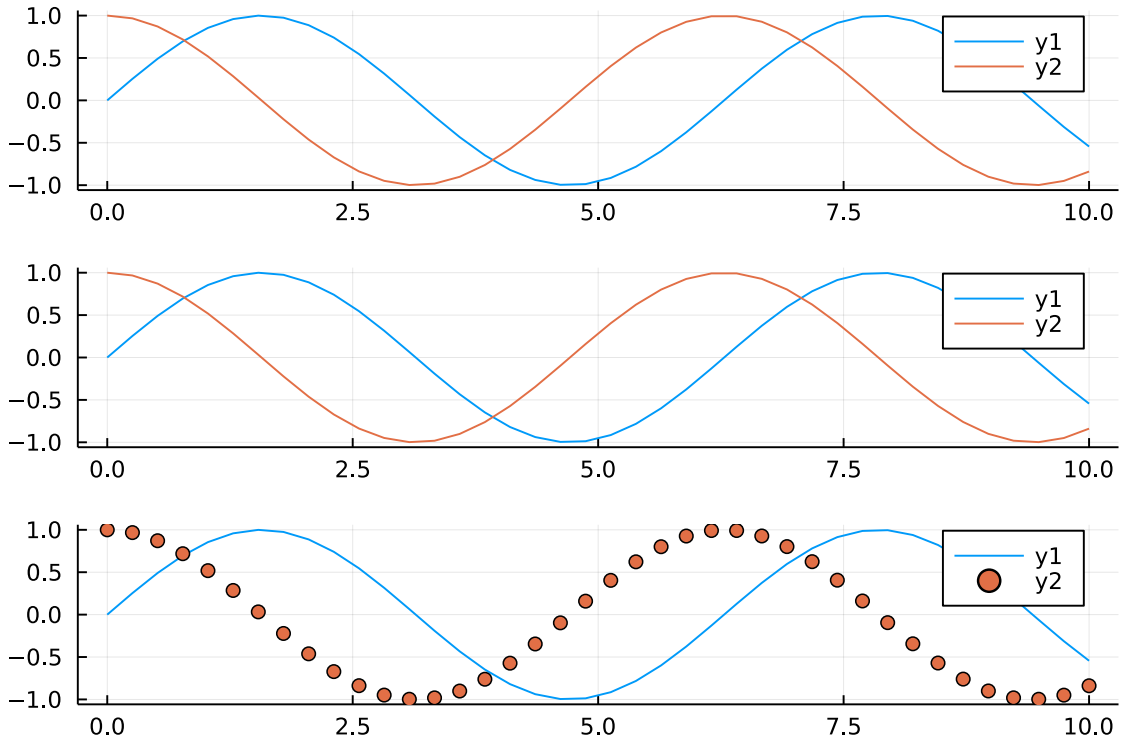
[18]:
```
plot(pl1, pl2, pl3, layout = (1,3))
```

[18]:




[19]:
```
plot(pl1, pl2, pl3, layout = (3,1))
```

[19]:

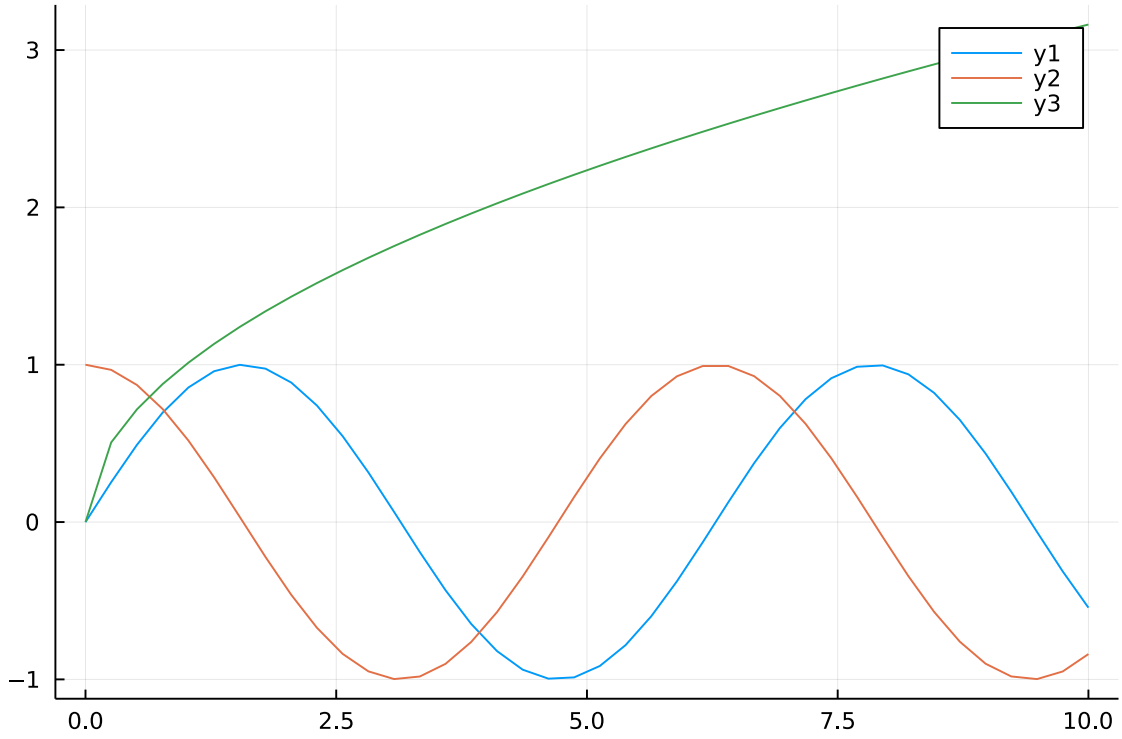



## 2.5 Input-Daten

- im einfachsten Fall ein Vektor von $m$ `x`-Werten und ein gleichlanger Vektor von $m$ `y`-Werten
- falls $y$ eine $m \times n$-Matrix ist, wird jeder Spaltenvektor als eine `Series` angesehen und es werden $n$ Kurven geplottet:

[20]:
```
plot(x, [sin.(x) cos.(x) sqrt.(x)])
```

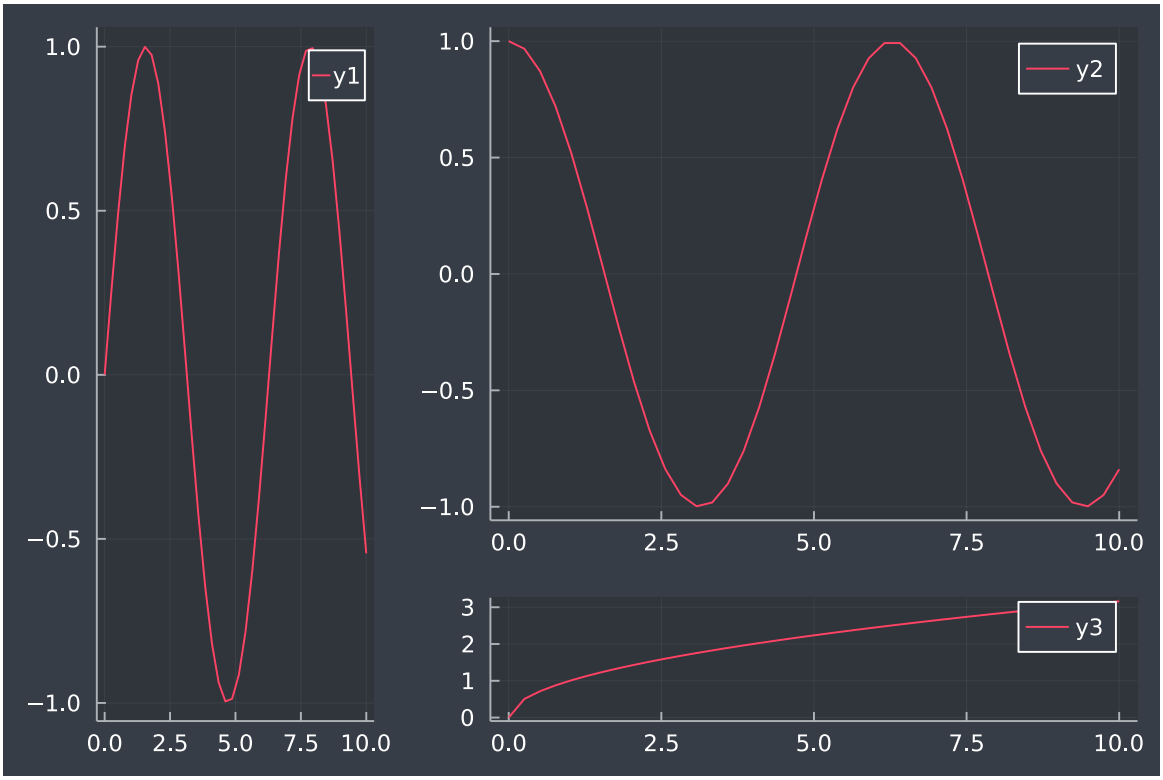
[20]:




- Durch eine `layout`-Angabe kann man die einzelnen `series` auch in einzelnen Subplots unterbringen.
- Man kann `layouts` auch schachteln und explizite Breiten/Höhenangaben verwenden.

[21]:
```
theme(:dark)

la1 = @layout [
        a{0.3w} [ b
                  c{0.2h}  ]
    ]

plot(x, [sin.(x) cos.(x) sqrt.(x)] , layout = la1)
```

[21]:




## 2.6 Plot-Attribute

`Plots.jl` teilt die Attribute in 4 Gruppen ein:

[22]:
```
plotattr(:Plot)  # Attribute für den Gesamtplot
```

```
Defined Plot attributes are:
background_color, background_color_outside, display_type, dpi, extra_kwargs,
extra_plot_kwargs, fontfamily, foreground_color, html_output_format,
inset_subplots, layout, link, overwrite_figure, plot_title, plot_titlefontcolor,
plot_titlefontfamily, plot_titlefonthalign, plot_titlefontrotation,
plot_titlefontsize, plot_titlefontvalign, plot_titleindex, plot_titlelocation,
plot_titlevspan, pos, show, size, tex_output_standalone, thickness_scaling,
warn_on_unsupported, window_title
```

[23]:
```
plotattr(:Subplot) # Attribute für einen Teilplot
```

```
Defined Subplot attributes are:
annotationcolor, annotationfontfamily, annotationfontsize, annotationhalign,
annotationrotation, annotations, annotationvalign, aspect_ratio,
background_color_inside, background_color_subplot, bottom_margin, camera, clims,
color_palette, colorbar, colorbar_continuous_values, colorbar_discrete_values,
colorbar_fontfamily, colorbar_formatter, colorbar_scale, colorbar_tickfontcolor,
colorbar_tickfontfamily, colorbar_tickfonthalign, colorbar_tickfontrotation,
colorbar_tickfontsize, colorbar_tickfontvalign, colorbar_ticks, colorbar_title,
colorbar_title_location, colorbar_titlefontcolor, colorbar_titlefontfamily,
colorbar_titlefonthalign, colorbar_titlefontrotation, colorbar_titlefontsize,
colorbar_titlefontvalign, extra_kwargs, fontfamily_subplot,
foreground_color_subplot, foreground_color_title, framestyle, left_margin,
legend_background_color, legend_column, legend_font, legend_font_color,
legend_font_family, legend_font_halign, legend_font_pointsize,
legend_font_rotation, legend_font_valign, legend_foreground_color,
legend_position, legend_title, legend_title_font, legend_title_font_color,
legend_title_font_family, legend_title_font_halign, legend_title_font_pointsize,
legend_title_font_rotation, legend_title_font_valign, margin, projection,
```

```
right_margin, subplot_index, title, titlefontcolor, titlefontfamily,
titlefonthalign, titlefontrotation, titlefontsize, titlefontvalign,
titlelocation, top_margin
```

[24]:
```
plotattr(:Axis)  # Attribute für eine Achse
```

```
Defined Axis attributes are:
discrete_values, draw_arrow, flip, foreground_color_axis,
foreground_color_border, foreground_color_grid, foreground_color_guide,
foreground_color_minor_grid, foreground_color_text, formatter, grid, gridalpha,
gridlinewidth, gridstyle, guide, guide_position, guidefontcolor,
guidefontfamily, guidefonthalign, guidefontrotation, guidefontsize,
guidefontvalign, lims, link, minorgrid, minorgridalpha, minorgridlinewidth,
minorgridstyle, minorticks, mirror, rotation, scale, showaxis, tick_direction,
tickfontcolor, tickfontfamily, tickfonthalign, tickfontrotation, tickfontsize,
tickfontvalign, ticks, widen
```

[25]:
```
plotattr(:Series) # Attribute für eine Serie, also zB ein Linienzug im Plot
```

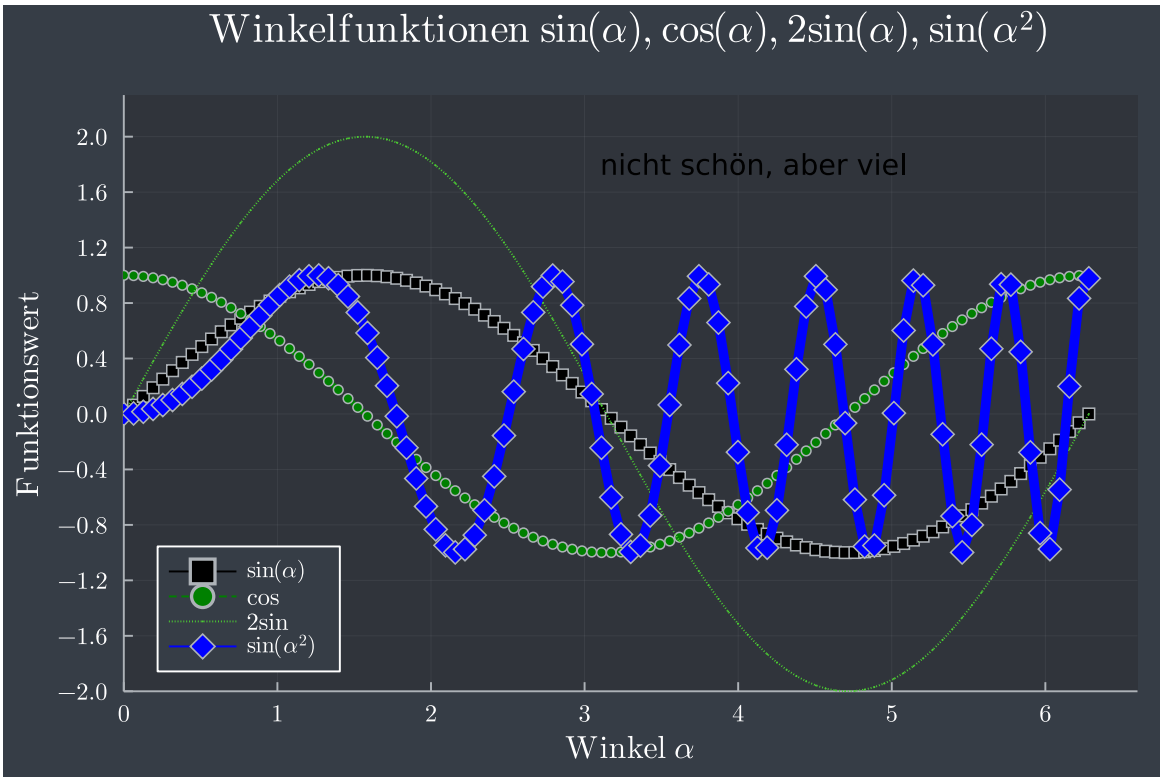
```
Defined Series attributes are:
arrow, bar_edges, bar_position, bar_width, bins, colorbar_entry, connections,
contour_labels, contours, extra_kwargs, fill_z, fillalpha, fillcolor, fillrange,
fillstyle, group, hover, label, levels, line_z, linealpha, linecolor, linestyle,
linewidth, marker_z, markeralpha, markercolor, markershape, markersize,
markerstrokealpha, markerstrokecolor, markerstrokestyle, markerstrokewidth,
normalize, orientation, permute, primary, quiver, ribbon, series_annotations,
seriesalpha, seriescolor, seriestype, show_empty_bins, smooth, stride, subplot,
weights, x, xerror, y, yerror, z, z_order, zerror
```

[26]:
```
# Zur Erinnerung nochmal:

using Plots
using Plots.PlotMeasures   # für Angaben in mm, cm,...
using LaTeXStrings         # für LaTeX-Konstrukte in Plot-Beschriftungen
using PlotThemes           # vorgefertigte Themen
```

[27]:
```
xs = range(0, 2π, length = 100)

data = [sin.(xs) cos.(xs) 2sin.(xs) (x->sin(x^2)).(xs)]

pl10 = plot(xs, data,
           fontfamily="Computer Modern",

           # LaTeX-String L""  ist im Math-mode
           title = L"\textrm{Winkelfunktionen}\ \sin(\alpha), \cos(\alpha), 2\sin(\alpha),␣
 ↪\sin(\alpha^2)",
           xlabel = L"\textrm{ Winkel\ } \alpha",
           ylabel = L"\textrm{Funktionswert}",

           # 1x4-Matrizen mit Farben, Marker,...  für die 4 'Series'
           color=[:black :green RGB(0.3, 0.8, 0.2) :blue ],
           markers = [:rect :circle :utriangle :diamond],
           markersize = [3 3 0 6],
           linewidth = [1 3 1 5],
           linestyle = [:solid :dash :dot :solid ],

           # Achsen
           xlim = (0, 6.6),
           ylim = (-2, 2.3),
           yticks = -2:.4:2.3,

           legend = :bottomleft,
```

```
            label = [ L"\sin(\alpha)" "cos" "2sin" L"\sin(\alpha^2)"],

            top_margin = 5mm,
)

# Zusatzelement

annotate!(pl10, 4.1, 1.8, text("nicht schön, aber viel",10))
```

[27]:




### 2.6.1 Säulendiagramm

[28]:
```
using RDatasets
```

Inhalt: über 700 freie (*"public domain"*) Datensätze, darunter z.B:

- Passagierliste der *Titanic*
- Verbrauchsdaten amerikanischer Autos aus den 70ern
- historische Währungskurse

[29]:
```
RDatasets.datasets()
```

[29]:

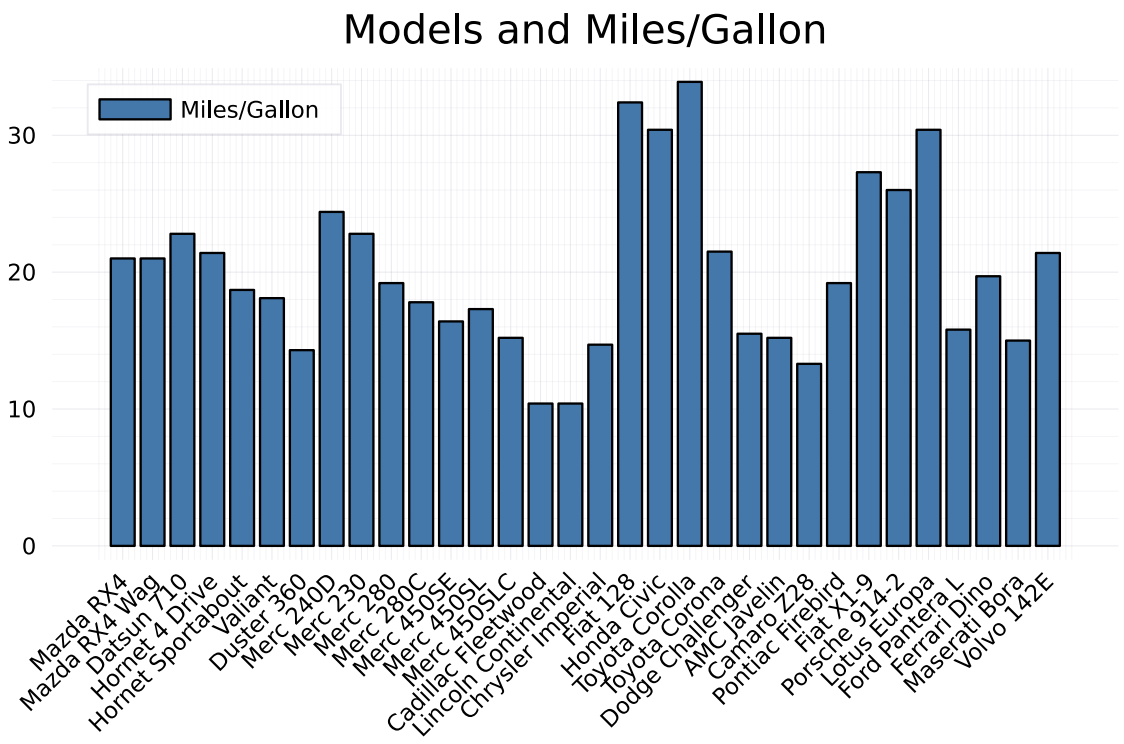
| | Package | Dataset | Title | Rows | Columns |
|---|---|---|---|---|---|
| | String15 | String31 | String | Int64 | Int64 |
| 1 | COUNT | affairs | affairs | 601 | 18 |
| 2 | COUNT | azdrg112 | azdrg112 | 1798 | 4 |
| 3 | COUNT | azpro | azpro | 3589 | 6 |
| 4 | COUNT | badhealth | badhealth | 1127 | 3 |
| 5 | COUNT | fasttrakg | fasttrakg | 15 | 9 |
| 6 | COUNT | lbw | lbw | 189 | 10 |
| 7 | COUNT | lbwgrp | lbwgrp | 6 | 7 |
| 8 | COUNT | loomis | loomis | 410 | 11 |
| 9 | COUNT | mdvis | mdvis | 2227 | 13 |
| 10 | COUNT | medpar | medpar | 1495 | 10 |
| 11 | COUNT | rwm | rwm | 27326 | 4 |
| 12 | COUNT | rwm5yr | rwm5yr | 19609 | 17 |
| 13 | COUNT | ships | ships | 40 | 7 |
| 14 | COUNT | titanic | titanic | 1316 | 4 |
| 15 | COUNT | titanicgrp | titanicgrp | 12 | 5 |
| 16 | Ecdat | Accident | Ship Accidents | 40 | 5 |
| 17 | Ecdat | Airline | Cost for U.S. Airlines | 90 | 6 |
| 18 | Ecdat | Airq | Air Quality for Californian Metropolitan Areas | 30 | 6 |
| 19 | Ecdat | Benefits | Unemployement of Blue Collar Workers | 4877 | 18 |
| 20 | Ecdat | Bids | Bids Received By U.S. Firms | 126 | 12 |
| 21 | Ecdat | BudgetFood | Budget Share of Food for Spanish Households | 23972 | 6 |
| 22 | Ecdat | BudgetItaly | Budget Shares for Italian Households | 1729 | 11 |
| 23 | Ecdat | BudgetUK | Budget Shares of British Households | 1519 | 10 |
| 24 | Ecdat | Bwages | Wages in Belgium | 1472 | 4 |
| 25 | Ecdat | CPSch3 | Earnings from the Current Population Survey | 11130 | 3 |
| 26 | Ecdat | Capm | Stock Market Data | 516 | 5 |
| 27 | Ecdat | Car | Stated Preferences for Car Choice | 4654 | 70 |
| 28 | Ecdat | Caschool | The California Test Score Data Set | 420 | 17 |
| 29 | Ecdat | Catsup | Choice of Brand for Catsup | 2798 | 14 |
| 30 | Ecdat | Cigar | Cigarette Consumption | 1380 | 9 |
| ... | ... | ... | ... | ... | ... |

[30]:
```
cars = dataset("datasets", "mtcars")
```
[30]:

| | Model | MPG | Cyl | Disp | HP | DRat | WT | QSec | |
|---|---|---|---|---|---|---|---|---|---|
| | String31 | Float64 | Int64 | Float64 | Int64 | Float64 | Float64 | Float64 | |
| 1 | Mazda RX4 | 21.0 | 6 | 160.0 | 110 | 3.9 | 2.62 | 16.46 | ... |
| 2 | Mazda RX4 Wag | 21.0 | 6 | 160.0 | 110 | 3.9 | 2.875 | 17.02 | ... |
| 3 | Datsun 710 | 22.8 | 4 | 108.0 | 93 | 3.85 | 2.32 | 18.61 | ... |
| 4 | Hornet 4 Drive | 21.4 | 6 | 258.0 | 110 | 3.08 | 3.215 | 19.44 | ... |
| 5 | Hornet Sportabout | 18.7 | 8 | 360.0 | 175 | 3.15 | 3.44 | 17.02 | ... |
| 6 | Valiant | 18.1 | 6 | 225.0 | 105 | 2.76 | 3.46 | 20.22 | ... |
| 7 | Duster 360 | 14.3 | 8 | 360.0 | 245 | 3.21 | 3.57 | 15.84 | ... |
| 8 | Merc 240D | 24.4 | 4 | 146.7 | 62 | 3.69 | 3.19 | 20.0 | ... |
| 9 | Merc 230 | 22.8 | 4 | 140.8 | 95 | 3.92 | 3.15 | 22.9 | ... |
| 10 | Merc 280 | 19.2 | 6 | 167.6 | 123 | 3.92 | 3.44 | 18.3 | ... |
| 11 | Merc 280C | 17.8 | 6 | 167.6 | 123 | 3.92 | 3.44 | 18.9 | ... |
| 12 | Merc 450SE | 16.4 | 8 | 275.8 | 180 | 3.07 | 4.07 | 17.4 | ... |
| 13 | Merc 450SL | 17.3 | 8 | 275.8 | 180 | 3.07 | 3.73 | 17.6 | ... |
| 14 | Merc 450SLC | 15.2 | 8 | 275.8 | 180 | 3.07 | 3.78 | 18.0 | ... |
| 15 | Cadillac Fleetwood | 10.4 | 8 | 472.0 | 205 | 2.93 | 5.25 | 17.98 | ... |
| 16 | Lincoln Continental | 10.4 | 8 | 460.0 | 215 | 3.0 | 5.424 | 17.82 | ... |
| 17 | Chrysler Imperial | 14.7 | 8 | 440.0 | 230 | 3.23 | 5.345 | 17.42 | ... |
| 18 | Fiat 128 | 32.4 | 4 | 78.7 | 66 | 4.08 | 2.2 | 19.47 | ... |
| 19 | Honda Civic | 30.4 | 4 | 75.7 | 52 | 4.93 | 1.615 | 18.52 | ... |
| 20 | Toyota Corolla | 33.9 | 4 | 71.1 | 65 | 4.22 | 1.835 | 19.9 | ... |
| 21 | Toyota Corona | 21.5 | 4 | 120.1 | 97 | 3.7 | 2.465 | 20.01 | ... |
| 22 | Dodge Challenger | 15.5 | 8 | 318.0 | 150 | 2.76 | 3.52 | 16.87 | ... |
| 23 | AMC Javelin | 15.2 | 8 | 304.0 | 150 | 3.15 | 3.435 | 17.3 | ... |
| 24 | Camaro Z28 | 13.3 | 8 | 350.0 | 245 | 3.73 | 3.84 | 15.41 | ... |
| 25 | Pontiac Firebird | 19.2 | 8 | 400.0 | 175 | 3.08 | 3.845 | 17.05 | ... |
| 26 | Fiat X1-9 | 27.3 | 4 | 79.0 | 66 | 4.08 | 1.935 | 18.9 | ... |
| 27 | Porsche 914-2 | 26.0 | 4 | 120.3 | 91 | 4.43 | 2.14 | 16.7 | ... |
| 28 | Lotus Europa | 30.4 | 4 | 95.1 | 113 | 3.77 | 1.513 | 16.9 | ... |
| 29 | Ford Pantera L | 15.8 | 8 | 351.0 | 264 | 4.22 | 3.17 | 14.5 | ... |
| 30 | Ferrari Dino | 19.7 | 6 | 145.0 | 175 | 3.62 | 2.77 | 15.5 | ... |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | |

[31]:
```
theme(:bright)

bar(cars.Model, cars.MPG,
           label = "Miles/Gallon",
           title = "Models and Miles/Gallon",
           xticks =:all,
           xrotation = 45,
           size = [600, 400],
           legend =:topleft,
           bottom_margin = 10mm
   )
```

[31]:


Models and Miles/Gallon
Miles/Gallon
30
20
10
0
Mazda RX4
Mazda RX4 Wag
Datsun 710
Hornet 4 Drive
Hornet Sportabout
Valiant
Duster 360
Merc 240D
Merc 230
Merc 280
Merc 280C
Merc 450SE
Merc 450SL
Merc 450SLC
Cadillac Fleetwood
Lincoln Continental
Chrysler Imperial
Fiat 128
Honda Civic
Toyota Corolla
Toyota Corona
Dodge Challenger
AMC Javelin
Camaro Z28
Pontiac Firebird
Fiat X1-9
Porsche 914-2
Lotus Europa
Ford Pantera L
Ferrari Dino
Maserati Bora
Volvo 142E

[ ]: