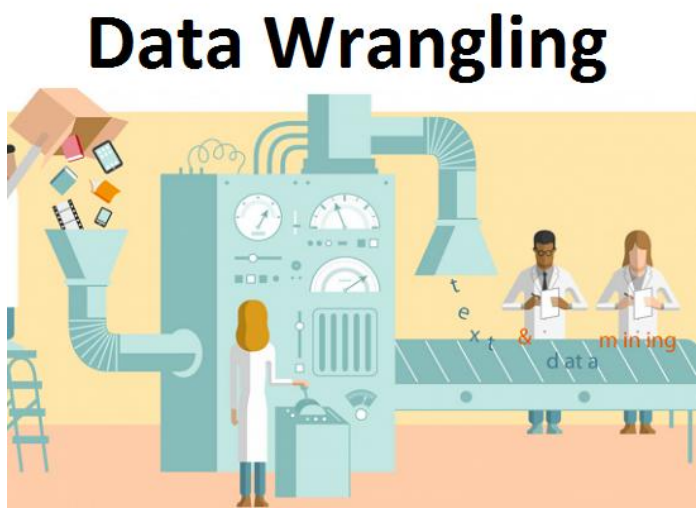
Algorytmy w inżynierii danych

Wykład 05 - Interaktywna analiza danych z wykorzystaniem Julia?

Bartosz Chaber, Robert Szmurło

e-mail: bartosz.chaber@ee.pw.edu.pl, robert.szmurlo@ee.pw.edu.pl 2020L

Plan



Motywacja - w analizie danych oprócz algorytmów matematycznych często wykorzystuje inne formy wspomagające, np.: wykresy do prezentacji danych.

Na tym wykładzie przyjrzymy się narzędziom do obróbki danych oraz różnym rodzajom wykresów oraz metodom ich generacji w środowisku Julia.

Przy okazji zapoznamy się z kilkoma specyficznymi cechami języka Julia.

Plan szczegółowy:

- Definicja badań eksploracyjnych (ang. EDA - Exploratory Data Analysis)
- Metody wykorzystywane w eksploracji danych - wykresy
- Interaktywna praca z Julia i Plots w Jupyter
- Wprowadzenie do DataFrames (takie 'Pandas' ale dla Julia)
- Wprowadzenie do wizualizacji danych statystycznych

Badania eksploracyjne

Badania obejmujące opis, wizualizację, wstępne przekształcanie zebranych danych bez potrzeby zakładania z góry hipotez. Może korzystać z modelowania statystycznego pewnych hipotez statystycznych (np. zależności między cechami danych), ale do ich znajdowania a nie potwierdzania.

Wykorzystuje się obserwację i analizę danych graficznych i statystycznych do postawienia hipotez, znalezienia brakujących odpowiedzi.

W etapie tym wykonuje się wstępne 'czyszczenie danych' - polegające na uzupełnieniu brakujących danych, przekształceniu ich do postaci, która może być wykorzystana w modelowaniu statystycznym.

Na etapie tym identyfikuje się również potencjalne błędy danych wejściowych (tzw. wartości odstające, ang. outliers).

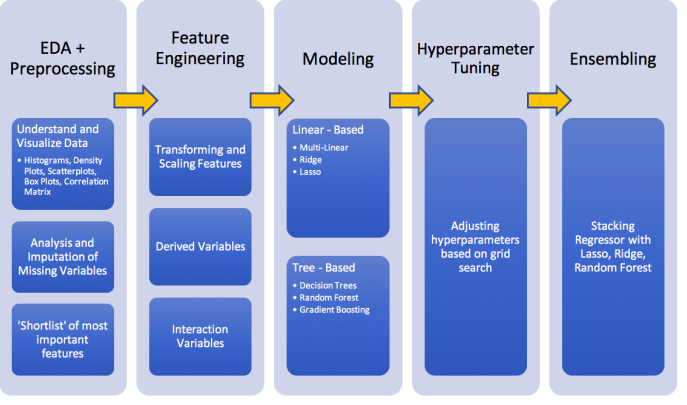
Badania eksploracyjne obejmują również wstępną klasyfikację, grupowanie.

> Przykładem może być odkrycie w danych z supermarketu zależności polegającej na tym że klient, który kupuje szampana i kwiaty, kupuje zwykle również czekoladki.

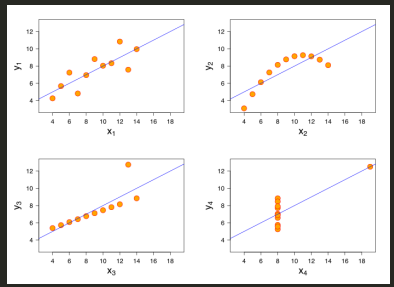
Metody eksploracji danych

http://wazniak.mimuw.edu.pl/index.php?title=Eksploracja_danych
https://nycdatascience.com/blog/student-works/machine-learning/kaggle-competition-house-pricing-in-ames-iowa/

- klasyfikacja / regresja (następny wykład)
- grupowanie (np.: K-Means)
- odkrywanie sekwencji (http://www.philippe-fournier-viger.com/spmf/index.php?link=algorithms.php)
- odkrywanie charakterystyk
- analiza przebiegów czasowych
- odkrywanie asocjacji
- wykrywanie zmian i odchyleń
- eksploracja WWW
- eksploracja tekstów



Wizualizacja danych
jako metoda eksploracji danych

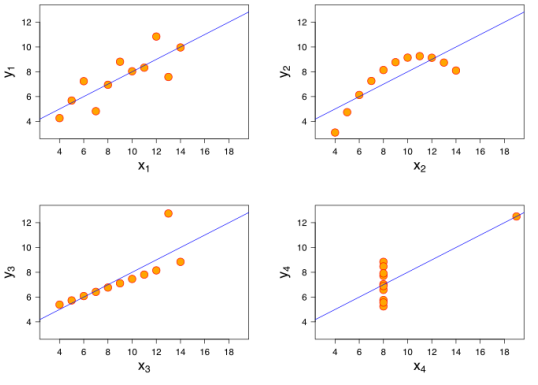


Kwartet Anscombe'a

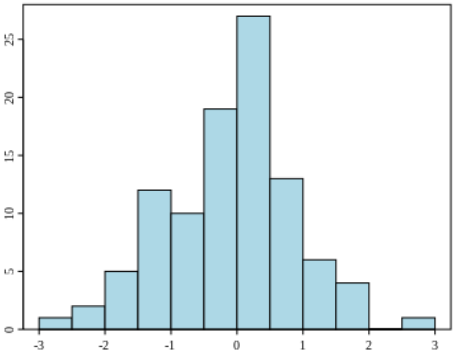
Cecha	Wartość
Średnia arytmetyczna zmiennej x	9
Wariancja zmiennej x	11
Średnia arytmetyczna zmiennej y	7.50 (identyczna do dwóch cyfr po przecinku)
Wariancja zmiennej y	4.122 lub 4.127 (identyczna do trzech cyfr po przecinku)
Współczynnik korelacji pomiędzy zmiennymi	0.816 (identyczny do trzech cyfr po przecinku)
Równanie regresji liniowej	$y = 3.00 + 0.500x$ (identyczne do kolejno: dwóch i trzech miejsc po przecinku)

Kwartet Anscombe'a

I		II		III		IV	
x	y	x	y	x	y	x	y
10.0	8.04	10.0	9.14	10.0	7.46	8.0	6.58
8.0	6.95	8.0	8.14	8.0	6.77	8.0	5.76
13.0	7.58	13.0	8.74	13.0	12.74	8.0	7.71
9.0	8.81	9.0	8.77	9.0	7.11	8.0	8.84
11.0	8.33	11.0	9.26	11.0	7.81	8.0	8.47
14.0	9.96	14.0	8.10	14.0	8.84	8.0	7.04
6.0	7.24	6.0	6.13	6.0	6.08	8.0	5.25
4.0	4.26	4.0	3.10	4.0	5.39	19.0	12.50
12.0	10.84	12.0	9.13	12.0	8.15	8.0	5.56
7.0	4.82	7.0	7.26	7.0	6.42	8.0	7.91
5.0	5.68	5.0	4.74	5.0	5.73	8.0	6.89



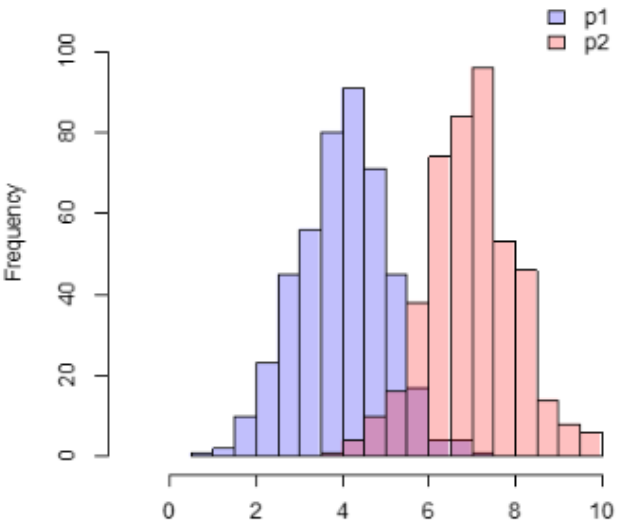
Badania eksploracyjne - wizualizacja



- Histogram

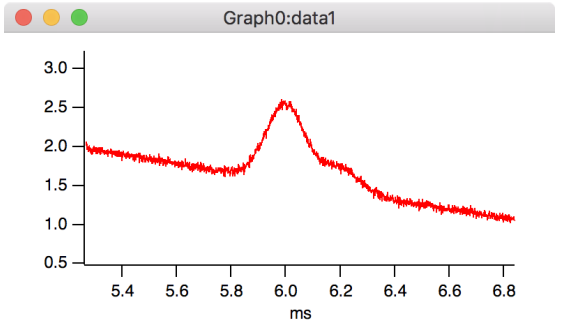
  Histogramy dla wybranych zmiennych predykcyjnych są tym lepsze im słupki są bardziej zróżnicowane. W sytuacji, gdy się na siebie nakładają, praktycznie nie ma możliwości wyznaczenia do jakiej klasy należą.

  Histogram wskazuje na wartość środkową (najczęściej występującą) Skośność (wskazuje czy dane mają układ centralny czy skośny - rosnący / malejący) Wartości odstające (identyfikujemy czy jest ich dużo)

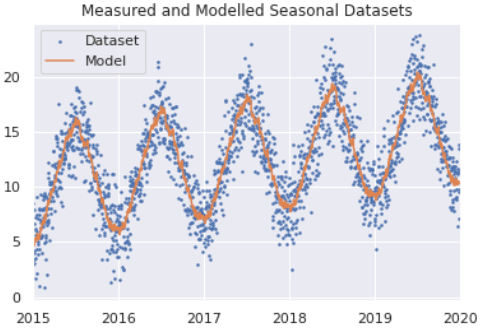


Badania eksploracyjne - wykres liniowy

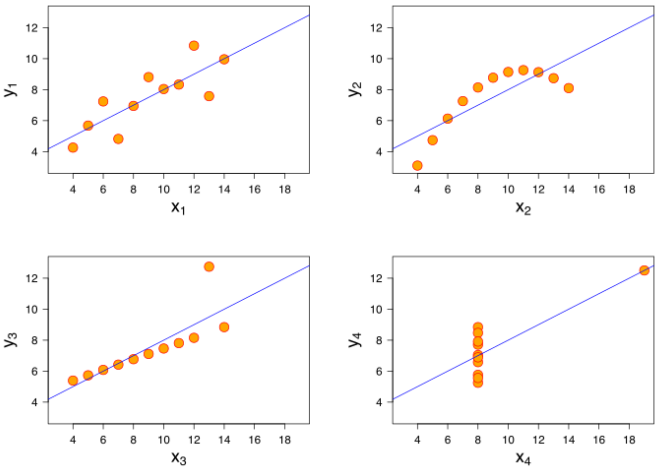
- Używany do przebiegów czasowych
- Pokazuje trend zmian zmiennej



- Pozwala zaobserwować cykliczność
- Pozwala porównać wiele zmiennych

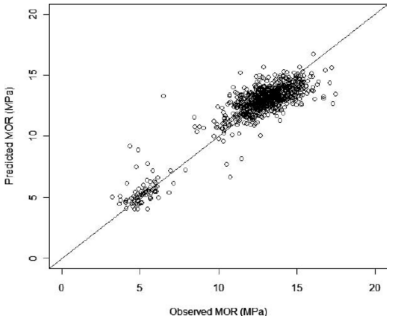


Wykres punktowy (scatter plot)



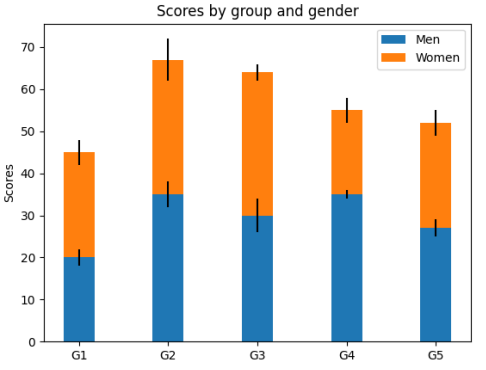
- Idealny do zależności między dwoma zmiennymi
- Wskazuje jak zmienia się jedna zmienna względem drugiej
- Dodatnia korelacja - jak jedno zwiększymy to drugie też
- Negative correlation - jak jedno się zwiększa to drugie zmniejsza
- Nieliniowa korelacja - change will not always correspond
- Brak korelacji - randomly placed dots

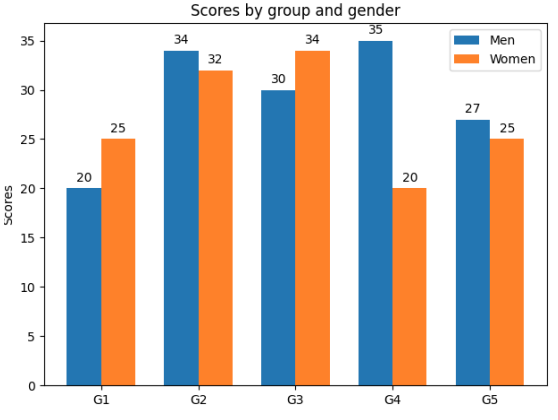
Idealny do prezentacji zgodności wyników z wartościami obserwowanymi w modelowaniu danych - oczekujemy zależności ściśle liniowej. Im dalej od linii tym mamy większe niezgodności.



Wykres słupkowy (ang. bar plot)

- Używany do danych kategorycznych (np. liczbę występujących danych)
- Pozwala porównać dane dane w dwóch różnych kategoriach:
  - Grouped bar chart - wykres słupkowy (kolumnowy) grupowany
  - Stacked bar chart - wykres słupkowy (kolumnowy) skumulowany



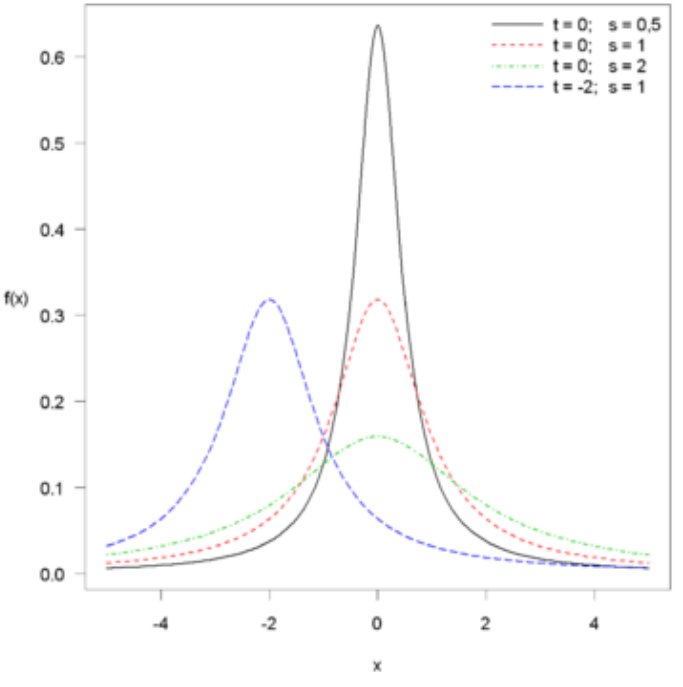


Badania eksploracyjne - gęstość prawdopodobieństwa

- Gęstość

  Wykresy rozkładów empirycznych dla wybranych zmiennych predykcyjnych są tym lepsze im linie obrazujące poszczególne klasy są względem siebie znacząco przesunięte. W sytuacji, gdy się na siebie nakładają, praktycznie nie ma możliwości wyznaczenia do jakiej klasy należą.

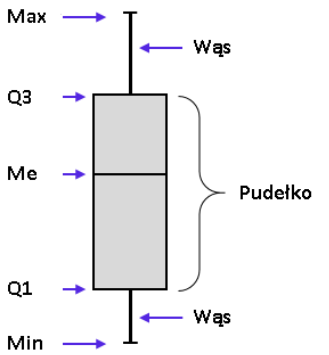
- jak się okaże wykresy gęstości prawdopodobieństwa możemy również uzyskać dla danych dyskretnych za pomocą wykresu KDE (kernel Density Estimator - estymator gęstości jądrowej



[https://pl.wikipedia.org/wiki/Estymator_j%C4%85drowy_g%C4%99sto%C5%9Bci]

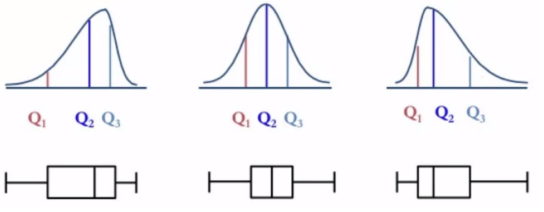
Badania eksploracyjne - Wykres pudełkowy

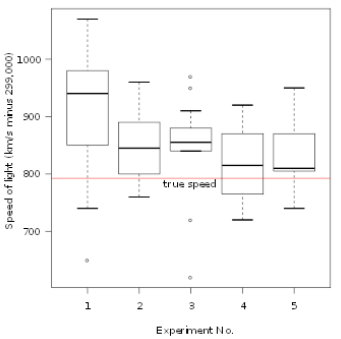
Celem obrazowania właściwości poszczególnych cech na wykresach pudełkowych jest wyłonienie takich zmiennych, które charakteryzują się największymi przesunięciami względem siebie kwantyli, wartości maksymalnych, minimalnych oraz median.



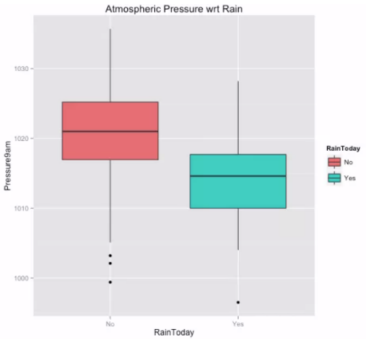
Pozwala odczytać:

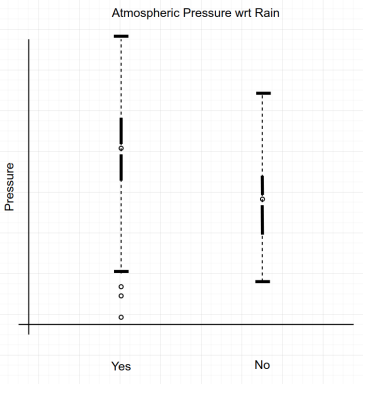
- zakres wartości
- wartości odstające
- skośność rozkładu danych





Wizualizacja danych - Edward Tufte





Współczynnik Atrament do Danych (ang. Ink to data ratio)

Definicje:

- atrament zużyty w celu prezentacji danych (stanowiących nieusuwalny rdzeń grafiki) / cały zużyty atrament
- udział atramentu poświęconego na prezentację danych istotnych
- udział atramentu, który może zostać usunięty bez straty informacji

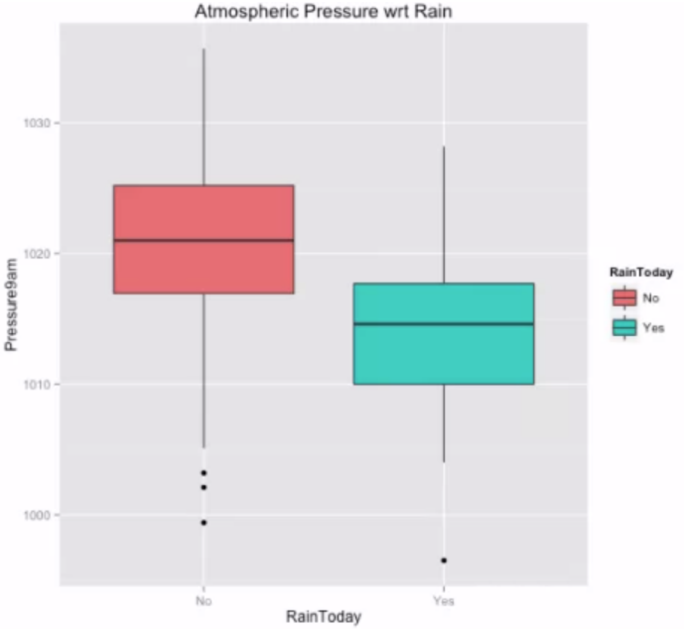
Maksymalizuj Współczynnik Dane/Atrament w granicach zdrowego rozsądku. Każda kropla atramentu wymaga uzasadnienia. I prawie zawsze uzasadnieniem powinno być to, że dodatkowy atrament prezentuje nowe informacje istotne dla odbiorcy.

Wizualizacja danych - zasady Edwarda Tuftego

http://www.sealthreinhold.com/school/tuftes-rules/rule_one.php

Edward Tufte (ur. 1942)

Tufte, Edward R (2001) [1983], The Visual Display of Quantitative Information (2nd ed.), Cheshire, CT: Graphics Press,



1. Pokazuj dane, które chcesz przekazać
2. Używaj grafiki (a nie opisów)
3. Unikaj chartjunk (np.: niepotrzebnych, zwracających uwagę ozdobników)
4. Wykorzystuj w miarę możliwiości tylko data-ink (atrament związany bezpośrednio z danymi)
5. Używaj etykiet (adnotacji stanowiących przewodnik)
6. Używaj mikro / makro (prezentuj dane na różnych poziomach)
7. Rozróżniaj poziomy (np. kolorami, wytłuszeniem)
8. Wykorzystuj wielokrotnie (ten sam układ danych, ale inne dane, pozwlają je lepiej porównać)
9. Wykorzystuj kolory zgodnie z intuicją (niebieski - morze, brązowy - ziemia)
10. Zrozum narrację (wykres z opowieścią w 'czasie' i 'przestrzeni' powinien być spójny)

Co można wyczytać z wykresów?

	Gęstość	Pudełkowy	Histogram
Kwantyl	✗	✓	✗
Mediana	✗	✓	✗
Minimum	✓	✓	✓
Maksimum	✓	✓	✓
Wartość cechy	✓	✓	✓
Liczebność	✓	✗	✓
Częstość	✓	✗	✓
Wzajemna korelacja zmiennych	✓	✗	✓

Pytania sprawdzające

1. Na czym polega zasada E-Tuftego dotycząca minimalizacji atramentu.
2. Jakie informacje związane z analizą danych można odczytać z wykresu pudełkowego?
3. Scharakteryzuj elementy wykresu pudełkowego.
4. Na czym polega metoda grupowania / klastryzacji w analizie danych (podaj przykład algorytmu)?
5. Jakie elementy analizy danych możemy wykonać za pomocą wykresu słupkowego?
6. Omów kwartet Anscombe'a w kontekście metod stosowanych do analizy danych.
7. Czy można narysować ciągły wykres gęstości prawdopodobieństwa dla danych dyskretnych. Odpowiedź uzasadnij. s

Wykresy oraz interaktywność w REPL w Julia

Wykresy - Julia

If you require...	... then use...
features	PyPlot, Plotly(JS), GR
speed	GR, InspectDR
interactivity	Plotly(JS), PyPlot, InspectDR
beauty	Plotly(JS), PGFPlots/ PGFPlotsX
REPL Plotting	UnicodePlots
3D plots	PyPlot, GR, Plotly(JS)
a GUI Window	GR, PyPlot, PlotlyJS, InspectDR
a small footprint	UnicodePlots, Plotly
plot+data -> `.hdf5` file	HDF5

Konfiguracja

Będziemy używać wielu pakietów...

```
julia> Pkg.status()
Status `~/.julia/environments/v1.4/Project.toml`
  [336ed68f] CSV v0.6.1
  [5d742f6a] CSVFiles v1.0.0
  [5ae59095] Colors v0.12.0
  [a93c6f00] DataFrames v0.20.2
  [31c24e10] Distributions v0.23.2
  [7073ff75] IJulia v1.21.2
  [6218d12a] ImageMagick v1.1.4
  [916415d5] Images v0.22.2
  [c601a237] Interact v0.10.3
  [5ab0869b] KernelDensity v0.5.1
  [3b7a836e] PGFPlots v3.2.1
  [91a5bcdd] Plots v1.0.12
  [f3b207a7] StatsPlots v0.14.5
  [0f1e0344] WebIO v0.8.13
  [10745b16] Statistics
  ...
julia>
```

Komenda: `Pkg.status()`

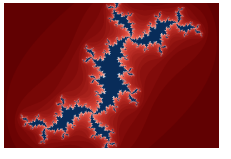
Jupyter notebook


```
const maxiter = 100

"""Compute the 'escape time' of the Julia set ar (z,c)"""

function julia(z, c)
    for n = 1:maxiter
        abs2(z) > 4 && return n-1
        z = z*z + c
    end
    return maxiter
end

julia(0.4 + 0.5im, 0.6 + 0.7im)

[julia(r + i*im, -0.06 + 0.665im) for i=-0.5:.002:0.5, r=-1:.002:1]

using Colors

const cmap = colormap("RdBu", 100) # typeof(cmap)=Array{RGB{Float64},1}

plot([cmap[julia(r + i*im, -0.05+0.85*im)]
         for i=-1:.005:1, r=-1.5:.005:1.5
     ])
```

- Prosta funkcja testowa w bazie liczb zespolonych: `julia(z,c)`.
- Wywołujemy jawnie
- Używamy 'list comprehension' do wygenerowania tablicy danych
- Użyjemy pakietu Colors, który pozwala mapować wartości liczbowe na kolory RGB.
- Zbudujemy tablicę wartości mapujących (colormap - jest obiektem funcyjnym)

`-0.05+0.85*im`

`-0.05+0.8*im`

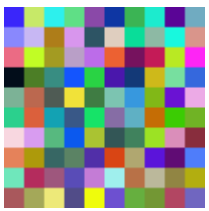


`-0.05+0.75*im`

Moduł colors

Moduł `Colors` udostępnia nam między innymi typ danych RGB

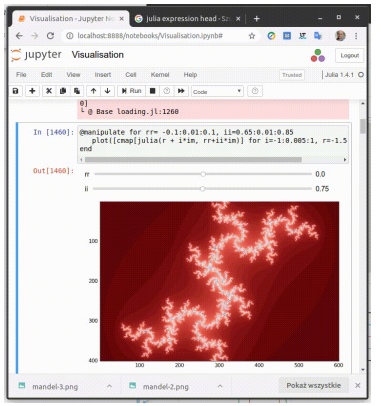
```
fieldnames(Colors.RGB)
(:r, :g, :b)

dump(RGB)

UnionAll
  var: TypeVar
    name: Symbol T
    lb: Union{}
    ub: Union{AbstractFloat, FixedPoint}
  body: RGB{T<:Union{AbstractFloat, FixedPoint}} <: AbstractRGB{T<:Union{AbstractFloat, FixedPoint}}
    r::T
    g::T
    b::T

img = rand(RGB, 10,10)
```



```
dump(RGB(0.5,0.5,0))

RGB{Float64}
  r: Float64 0.5
  g: Float64 0.5
  b: Float64 0.0
```

Jupyter - interaktywność

Moduł `Interact` pozwala tworzyć proste interfejsy interaktywne bazujące na technologii webowej. Te mini-aplikacje mogą być osadzane w notatnikach Jupyter, w oknie aplikacji Electron (moduł `Blink`) lub bezpośrednio w oknie przeglądarki. Tak naprawdę Interact to 'metapakiet', który wykorzystuje między innymi WebIO, i inne.




```
using Interact
@manipulate for rr= -0.1:0.01:0.1,
                ii=0.65:0.01:0.85
   plot([cmap[julia(r + i*im, rr+ii*im)]
         for i=-1:0.005:1, r=-1.5:0.005:1.5])
end
```

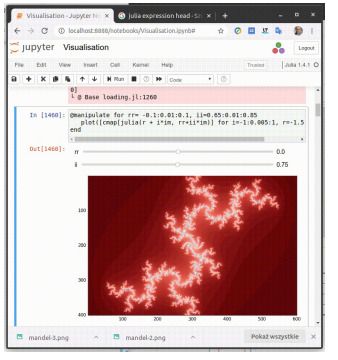
`@manipulate` - makro zdefiniowane w pakiecie Widget (który współpracuje z Interact), które generuje zbiór Widgetów do interakcji. (Źródło: manipulate.jl)

Generowanie bezpośrednio kodu HTML:

```
@manipulate for r = 0:.05:1,
                g = 0:.05:1,
                b = 0:.05:1
    HTML(string("<div style='color:#",
         hex(RGB(r,g,b)),
         "'>Color me</div>"))
```

Efektyność implementacji - Prealokacja danych

```
let I= -1:0.005:1, R= -1.5:0.005:1.5
    data = Array{RGB{Float64}}(undef, length(I), length(R))
    @manipulate for rr = -0.1:0.01:0.1, ii= 0.65:0.01:0.85
        for (j,i) in enumerate(I),
            (k,r) in enumerate(R)
            data[j,k] = cmap[julia(r + i*im, rr + ii*im)]
        end
        plot(data)
    end
end
```



Instrukcja `let` - tworzy blok w którego zasięgu będą dostępne zmienne (z punktu widzenia programowania funkcyjnego jest to wykorzystywane do tworzenia domknięć (ang. closures))

Zmienna `data` - będzie również widoczna w bloku i jest zadeklarowana jako pusta macierz dwuwymiarowa

Inny przykład licznika wywołań funkcji:

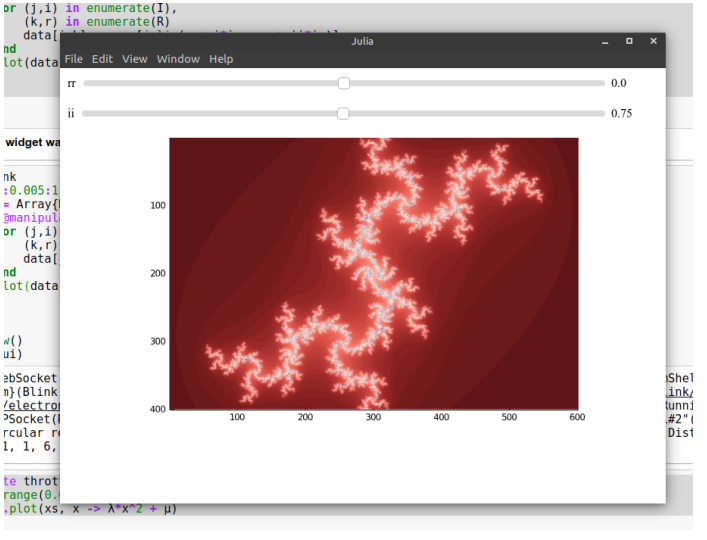
```
let c = 0
    global myfunc() = (c = c + 1)
end

myfunc()
1
myfunc()
2
myfunc()
3
```

Interakcja - Blink

https://juliagizmos.github.io/Blink.jl/latest/

Warto rozważyć użycie osobnego okna opartego na Electronie:

```
using Blink
let I= -1:0.005:1, R= -1.5:0.005:1.5
    data = Array{RGB{Float64}}(undef,
              length(I), length(R))
    ui = @manipulate for rr = -0.1:0.01:0.1,
                         ii= 0.65:0.01:0.85
        for (j,i) in enumerate(I),
            (k,r) in enumerate(R)
            data[j,k] = cmap[julia(r + i*im,
                                   rr + ii*im)]
        end
        plot(data)
    end

end

w = Window()
body!(w, ui)
```

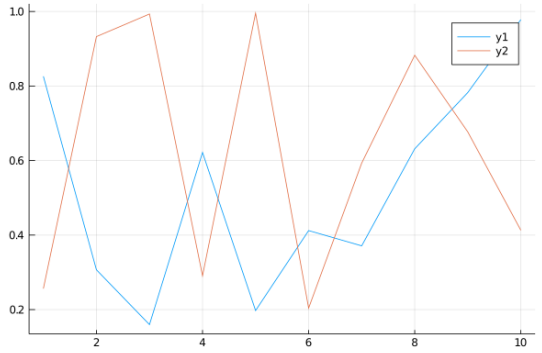


albo

```
w = Window()
loadurl(w, "https://www.ee.pw.edu.pl/")
```

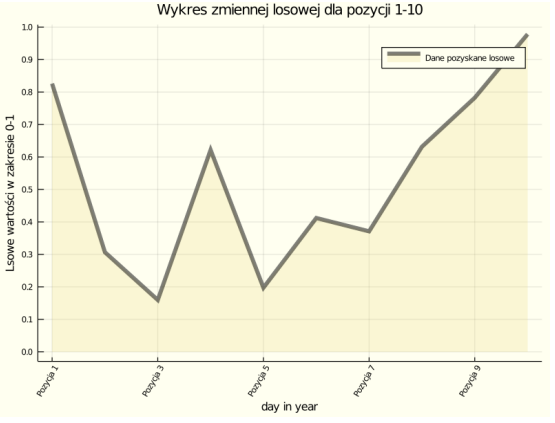
Plots - w Julia

```
using Plots
x = 1:10; y = rand(10); # These are the plotting da
plot(x, y)
plot!(x, rand(10))

gr(format="png")
```



Atrybuty wykresów:

```
xlabels = ["Pozycja $i" for i in 1:10]
plot(
    y,
    label  = "Dane pozyskane losowe",
    line=(:black, 0.5, 6, :solid),

    size=(800, 600),

    xticks = (1:2:10, xlabels[1:2:10]),
    yticks = 0:0.1:1,

    ylabel = "Lsowe wartości w zakresie 0-1",
    xlabel = "day in year",

    title  = "Wykres zmiennej losowej dla pozycji 1
    xrotation = rad2deg(pi/3),

    fillrange = 0,
    fillalpha = 0.25,
    fillcolor = :lightgoldenrod,

    background_color = :ivory
    )
```



Eksploracja możliwości modułu Plots

```
methods(Plots.supported_seriestypes) # pobierz liste wszystkich wariantów metody

Plots.supported_seriestypes() # wyświetl listę wspieranych rodzajów wykresów a bieżącym backendzie

Plots.supported_attrs() # wyświetl listę wspieranych atrybutów - niestety nie ma wszystkich :-()

plotattr(:Axis)
plotattr(:Series)
plotattr(:Plot)

plotattr("markersize")

#markersize {Number or AbstractVector}
#markersizes, ms, msize

#Size (radius pixels) of the markers.
#Series attribute,  default: 4

println.(collect(p.subplots[1].attr[:xaxis].plotattributes));
```

Przykład `xrotation`

Można go znaleźć przeszukując dokumentację:

> The common approach is to go to http://docs.juliaplots.org/latest/ and do a page search for the word, e.g. "legend". I'll fully admit that isn't the most intuitive procedure though.

ale za nic nie znajdziemy go w funkcji zwracającej atrybuty:

```
filter(x -> occursin("rota", string(x)),  Plots.supported_attrs() ) # zwraca: Set{Symbol} with 0 elements
```

to samo gdy użyjemy funkcji `plotattr(...)`

Magiczne argumenty i aliasy

Aliasy

```
plot(y, color = :blue)

# to praktycznie to samo co:

plot(y, seriescolor = :blue)
```

Moduł Plots wykorzystuje sprawdzanie typów oraz multimetody (wielometody, ?polimorfizm dynamiczny - czasu wykonania - ang. *multiple dispatch*)

```
plot(y, xaxis = ("my label", (0,10), 0:0.5:10, :log, :flip, font(20, "Courier")))

plot(y,
    xlabel = "my label",
    xlims = (0,10),
    xticks = 0:0.5:10,
    xscale = :log,
    xflip = true,
    xtickfont = font(20, "Courier")
)
```

Plots - receptury

https://docs.juliaplots.org/latest/recipes/

Ciekawą właściwością Julia jesy wykorzystanie metaprogramowania w Julia - receptury (`@recipe`), które pozwalają rozszewrzyć funcjonalność modułu bez ingerencji w jego kod źródłowy.

```
mutable struct MyVecWrapper
  v::Vector{Float64}
end
mv = MyVecWrapper(rand(10))

@recipe function f(mv::MyVecWrapper)
    markershape --> :circle
    markersize  --> 8
    seriestype  :=  :path
    mv.v
end

plot(
    plot(mv.v),
    plot(mv)
)
```

`-->` - opertaor zdefiniowany w makrze `@recipe`, który modyfikuje atrybuty obrazu tylko jeżeli nie istnieją (czyli operator np.: `linecolor --> :blue` zastąpi `get!(plotattributes, :linecolor, :blue)`.

`:=` - operator wymusi wstawienie nowej wartości to słownika atrybutów

Własne wykresy

Własny typ wykresu (series type):

```
@recipe function f(::Type{Val{:awid}}, x,y,z)
    seriestype := :line
    linecolor := :red
#    x := x
#   y := y

    x := [1, 2]
    y := [1, -1]
end
```

Własny typ wykresu:

```
@userplot awid # makro tworzące typ danych, oraz metody awid(...) oraz awid!()

@recipe function f(h::awid)
    seriestype --> :bar
    linecolor --> :red
    (h.args)
end

awid([1,2,3], [5,5,7], linecolor=:blue)
```

Dziękuję za uwagę