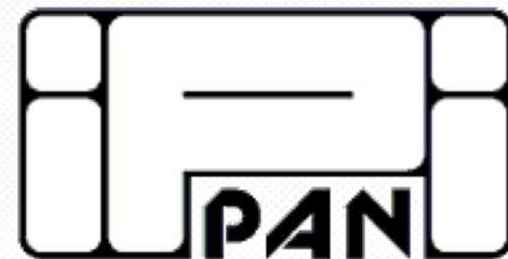




Zaawansowane Metody Analizy Danych w Biologii Molekularnej

Semest Letni 2018

Prowadzący



Zespół Biologii Obliczeniowej IPI PAN

- dr Magdalena Mozolewska (5 zajęć)
- dr Michał J. Dąbrowski (5 zajęć)
- dr inż Michał Dramiński (5 zajęć)

<http://zmadbm.ipipan.waw.pl/>



Zaliczenie przedmiotu

- Wykład 90 minut (egzamin ustny w sesji)
- Ćwiczenia 90 minut (15 x 10 pkt = 150 pkt)
 - 76 - 90 pkt (> 50%) ocena 3
 - 91 - 105 pkt (> 60%) ocena 3.5
 - 106 - 120 pkt (> 70%) ocena 4
 - 121 - 135 pkt (> 80%) ocena 4.5
 - 136 - 150 pkt (> 90%) ocena 5
- Dopuszcza się nieobecność na ćwiczeniach i wysłanie mailem rozwiązań, lecz wtedy maksymalna liczba punktów za ćwiczenia wynosi 5 (50%).
- Czas na wysłanie mailem rozwiązań do zadań to 1 tydzień (północ kolejnej środy).
- Oceny 4.5 i 5 z ćwiczeń uprawniają do zwolnienia z egzaminu z przepisaną oceną za przedmiot. Wyższą ocenę można uzyskać po przystąpieniu do egzaminu.

Wstępny harmonogram zajęć

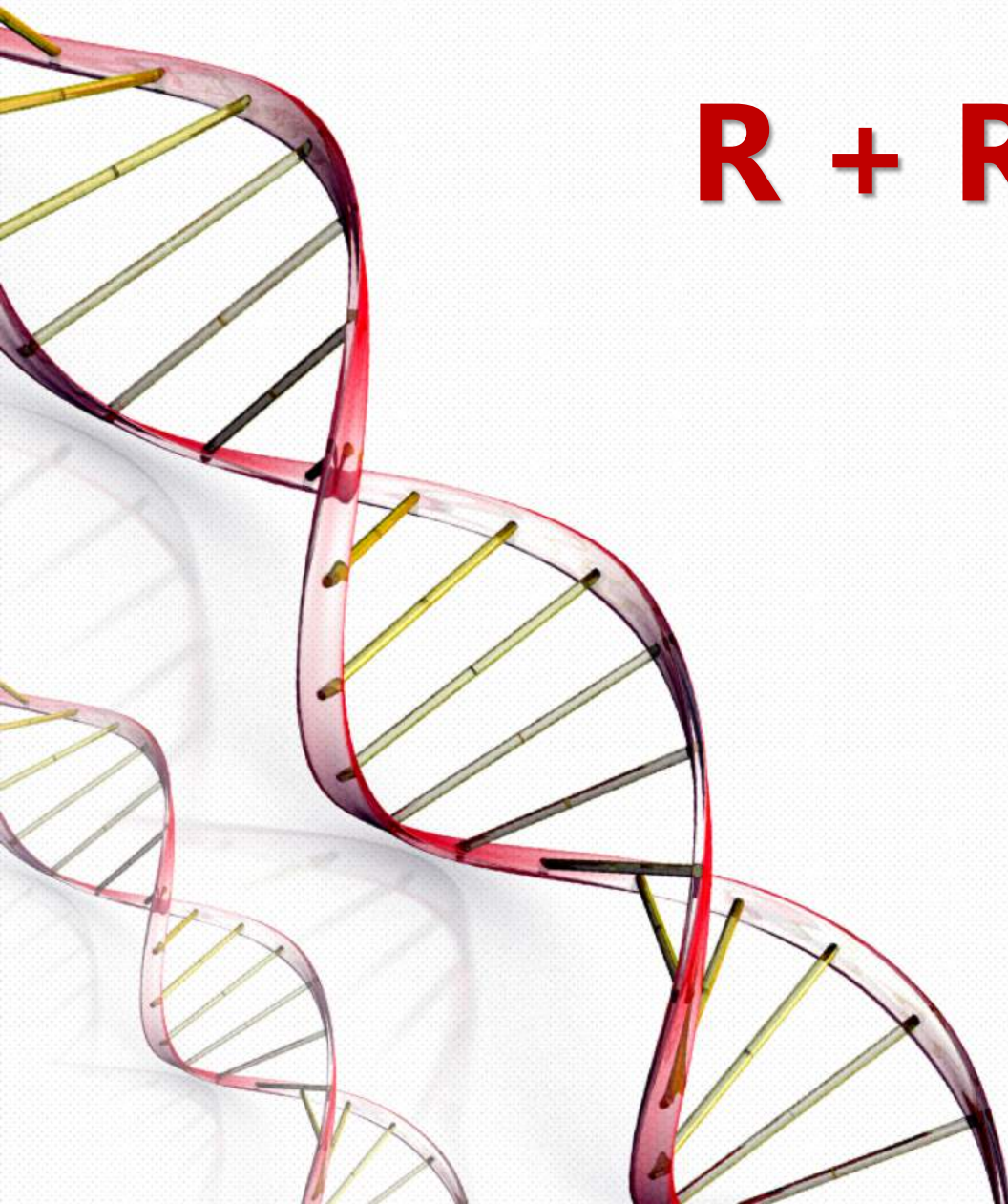
#	Data	Prowadzący/a	Temat Wykładu	Temat Ćwiczeń	Uwagi
1	21.02.2018	MDR	Wprowadzenie do przedmiotu oraz do R i RStudio.	Pakiety, petle, listy, warunki, data.frame, data.table, join, dplyr, ggplot. Zadanie: napisać generator losowych zbiorów o zadanej korelacji z decyzją.	
2	28.02.2018	MJD	Podstawy informacji genetycznej	Informacja genetyczna, mutacje, zasady dziedziczenia, od genu do białka, poznanie baz danych	
3	07.03.2018	MDR	Wstępna analiza danych. Wprowadzenie do statystyki.	Histogramy, boxploty, badanie rozkładów, wykresy, anova dplyr ggplot2 data.frame	
4	14.03.2018	MZO	Analiza baz sekwencyjnych biomakromolekuł	Wyszukiwanie białek w bazie pdb, analiza sekwencji, psiblast, homologi, uniprot	
5	21.03.2018	MJD	Regulacja ekspresji genów	Formaty zapisu informacji genetycznej, wyszukiwanie obszarów regulatorowych, wyszukiwanie obszarów kodujących	
6	28.03.2018	MDR	Modelowanie danych i analiza wyników klasyfikacji	Modelowanie i metody oceny wyników TrainTest CV macierz błędów	
7	04.04.2018	Wielkanoc	brak zajęć	brak zajęć	W zasadzie to już nie jest dzień wolny na UKSW
	11.04.2018	MJD	Specyficzność tkankowa	Różnicowe analizy pomiędzy różnymi liniami komórkowymi	
8	18.04.2018	MDR	Grupowanie obiektów oraz selekcja cech	Grupowanie (kmeans, hierarchiczne grupowanie), filtry, wrappery, metody wbudowane, wymiar WC, PCA.	
9	25.04.2018	MZO	Analiza strukturalna białek	Białko, struktury I, II, III, IV rzędowe, białko wyszukać pobrać obejrzeć, zaznaczyć reszty hydrofilowe, hydrofobowe, mostki solne, disulfidowe, modelowanie homologiczne wstęp, Analiza jakości i prawdopodobieństwa modeli - przewidywanie a rozwiązywanie struktur białek	
	02.05.2018	Święto 3 maja	brak zajęć	brak zajęć	
10	09.05.2018	MJD	Podłoże genetyczne stanów chorobowych	Analiza wybranych fragmentów genomu w celu wykrycia zaburzeń chorobotwórczych	
11	16.05.2018		Samodzielna analiza danych molekularnych	Mini projekt z prezentacją na koniec	Prawdopodobnie brak zajęć
12	23.05.2018	MZO	Analiza modelowania homologicznego, dokowanie molekularne i analiza oddziaływań	Analiza oddziaływań w kompleksach białko-białko, białko-ligand	
13	30.05.2018	MDR	Pułapki w analizie danych o dużym rozmiarze	Prawidłowa cross walidacja, korekty statystyczne, nadmierne dopasowanie, shadowing	
14	06.06.2018	MZO	Analiza trajektorii dynamiki molekularnej	VMD jako przykładowe narzędzie do analizowania trajektorii dynamiki molekularnej	
15	13.06.2018	MJD	Prezentacja	Prezentacja wyników analiz z mini projektu	
	20.06.2018		EGZAMIN USTNY		

Literatura

- "Statystyczne systemy uczące się" Jacek Koronacki, Jan Ćwik, Akademicka Oficyna Wydawnicza EXIT, 2015.
- "Systemy uczące się", Paweł Cichosz, WNT 2007.
- "Data Mining Algorithms: Explained Using R", Paweł Cichosz, Wiley 2015.
- "Przewodnik po pakiecie R", Przemysław Biecek, Oficyna wydawnicza GiS.
- <https://cran.r-project.org/doc/contrib/Komsta-Wprowadzenie.pdf>



R + R Studio



Agenda zajęć z R + Analiza danych

1. Wprowadzenie do przedmiotu. Wprowadzenie do R i RStudio.

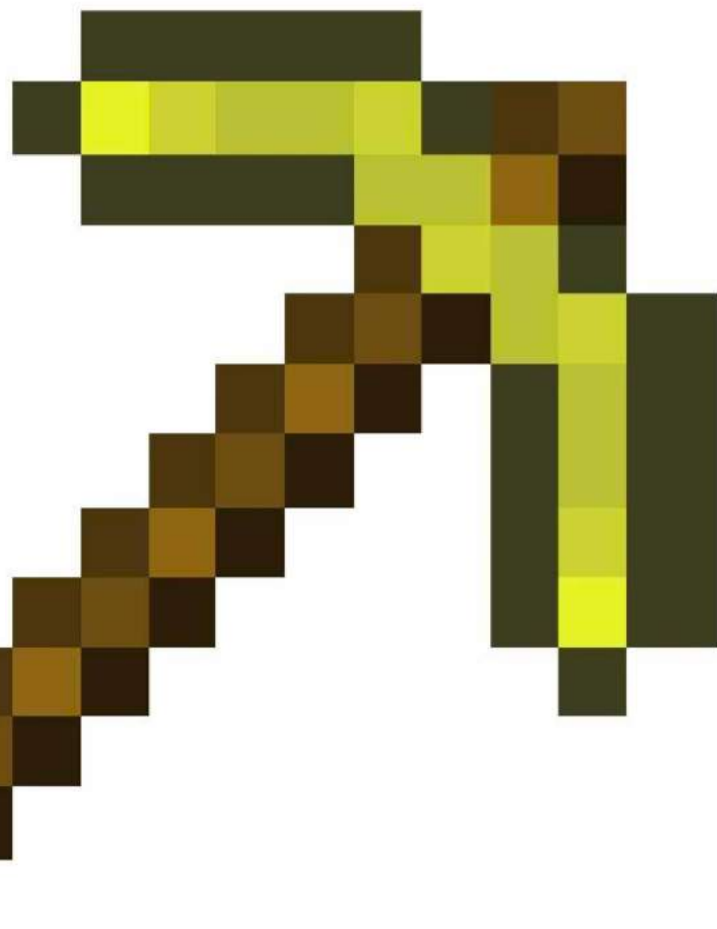
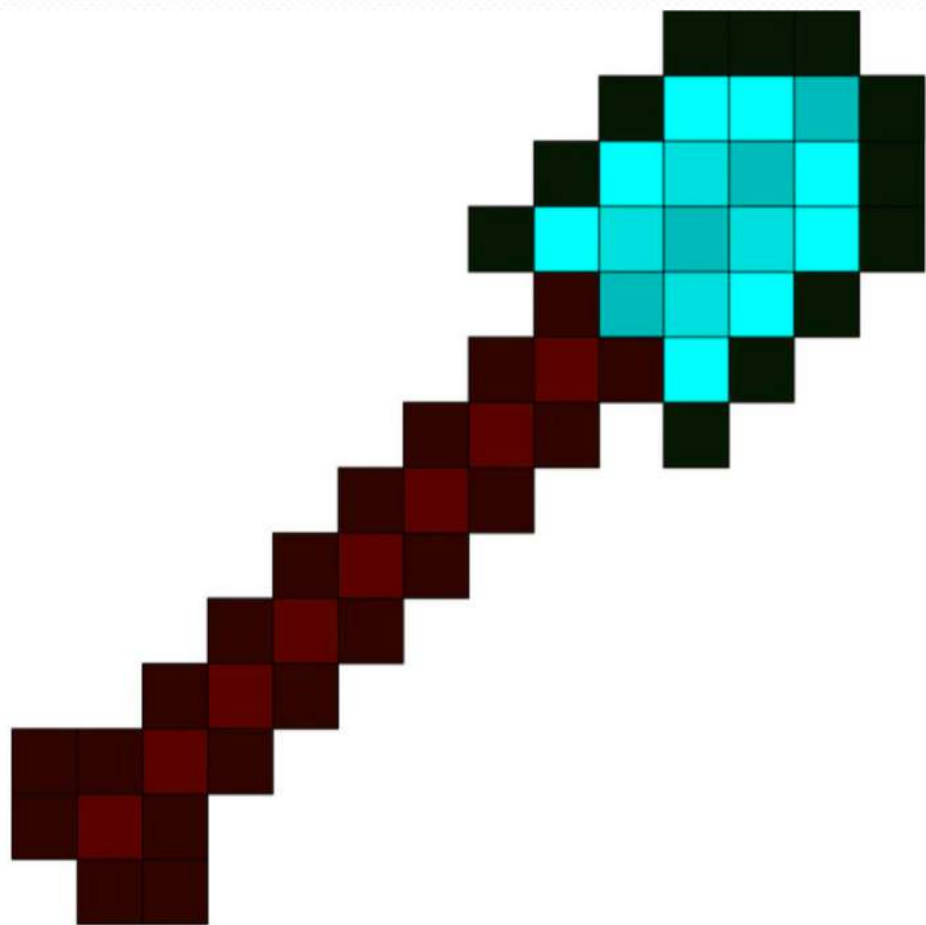
3. Wstępna analiza danych. Wprowadzenie do statystyki.

6. Selekcja oraz budowanie nowych cech.

8. Modelowanie danych i analiza wyników selekcji cech.

13. Pułapki w analizie danych o dużym rozmiarze.





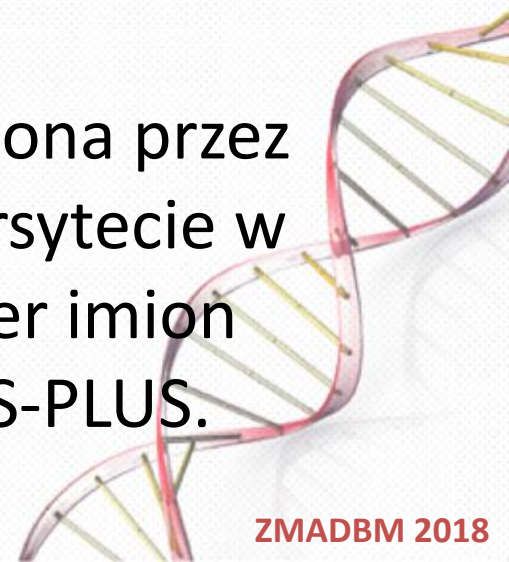
Co to ten R?



Wikipedia:

GNU R to interpretowany język programowania oraz środowisko do obliczeń statystycznych i wizualizacji wyników. Jest to projekt GNU na licencji GPL podobny do języka i środowiska S stworzonego w Bell Laboratories (kiedyś AT&T teraz Lucent Technologies) przez Johna Chambersa i jego współpracowników.

R jako implementacja języka S została stworzona przez Roberta Gentlemana i Rossa Ihakę na uniwersytecie w Auckland. Nazwa pochodzi od pierwszych liter imion twórców oraz jest nawiązaniem do języka S/S-PLUS.



Co to ten R?



R jest podstawowym językiem programowania w **bioinformatyce**, spopularyzowanym głównie dzięki stworzonemu przez **Roberta Gentlemana** repozytorium **Bioconductor**.

Artykuły Gentlemana o R i Bioconductorze należą do najczęściej cytowanych w bioinformatyce (ponad 4000 cytowań według Google Scholar).



Popularność R

R statistics, SAS statistics, python statistics, and SPSS statistics Job Trends

R statistics ×

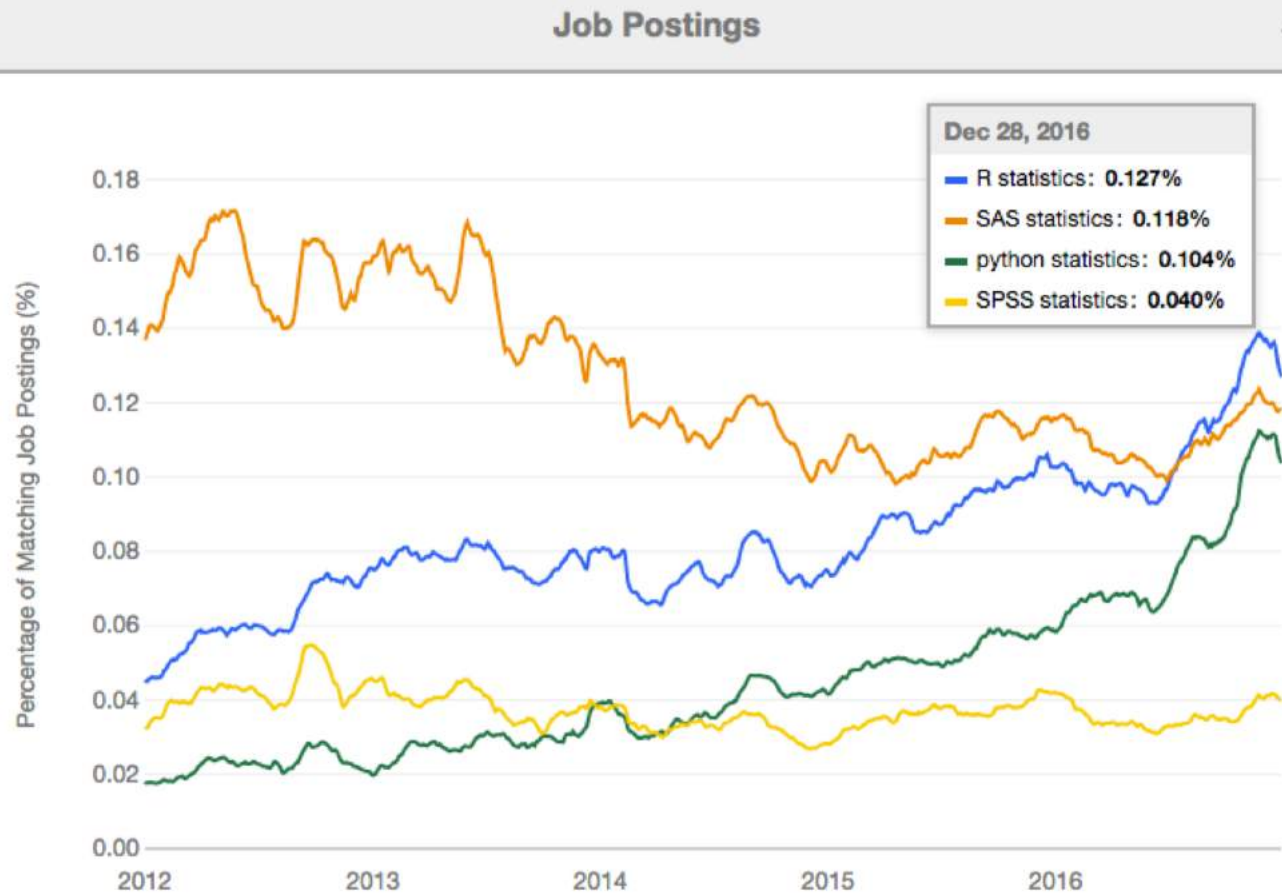
SAS statistics ×

python statistics ×

SPSS statistics ×

+ Add Term

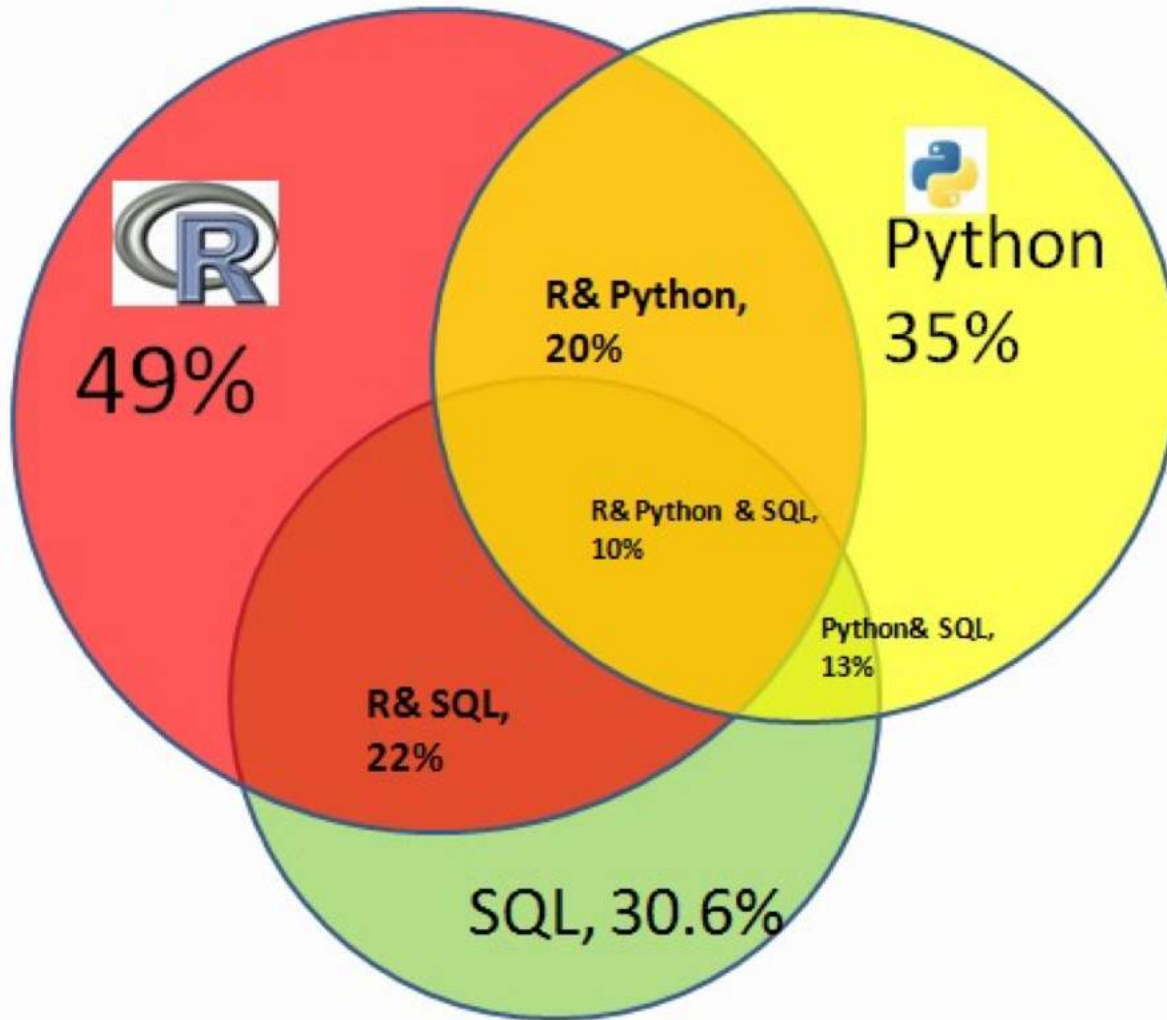
Find Trends



<https://www.indeed.com/jobtrends/>

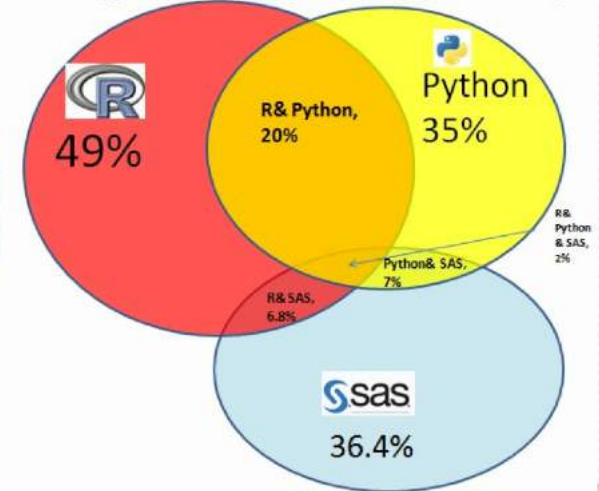
Popularność R

KDnuggets 2014 Poll: Languages used for Analytics/Data Mining



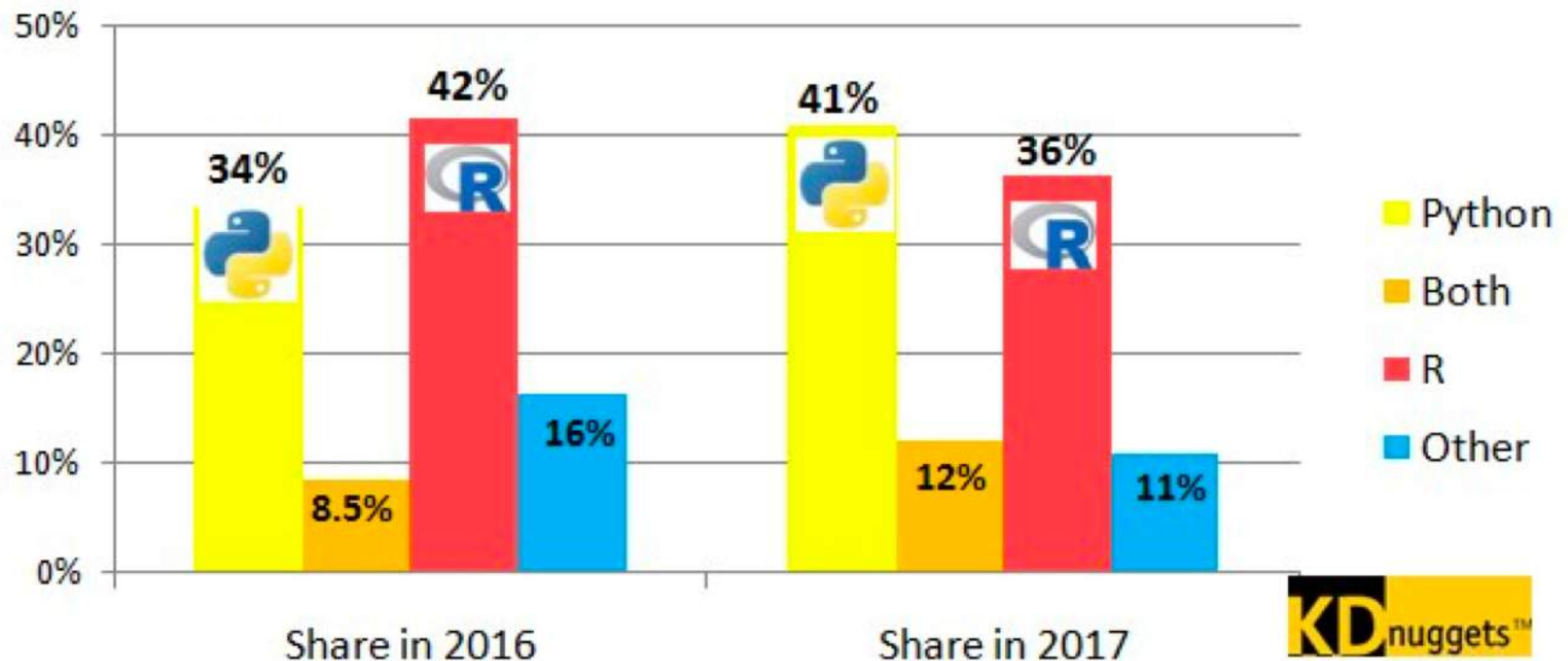
R + SAS + Python + SQL: 91%

KDnuggets 2014 Poll: Languages used for Analytics/Data Mining, 2



Popularność R

Python, R, Both, or Other platforms for Analytics, Data Science, Machine Learning

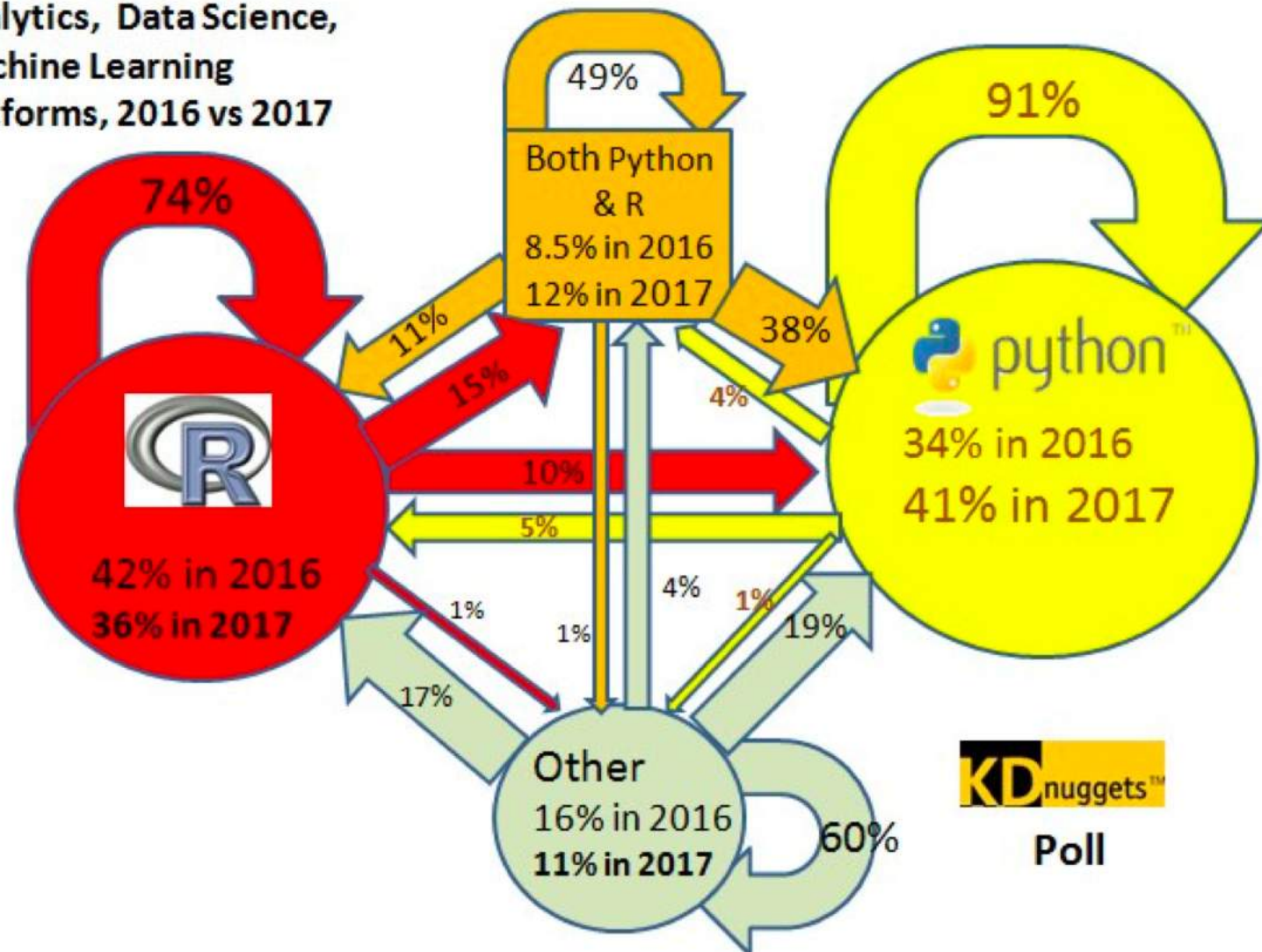


KDnuggets™

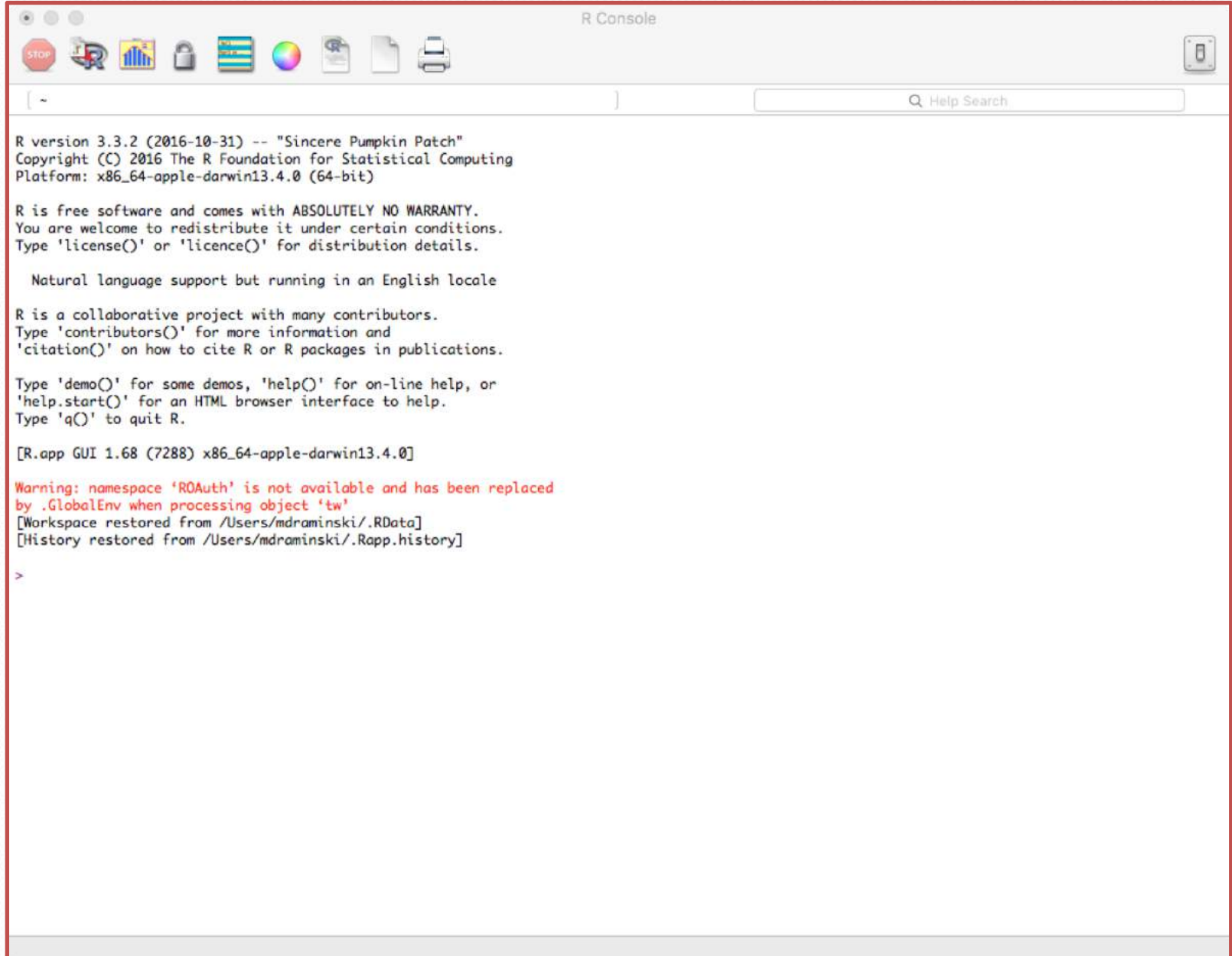
Poll 2017

Popularność R

Analytics, Data Science,
Machine Learning
Platforms, 2016 vs 2017



<https://cran.r-project.org/>



The screenshot shows the R Console window with a standard macOS title bar. The window contains the following text:

```
R version 3.3.2 (2016-10-31) -- "Sincere Pumpkin Patch"
Copyright (C) 2016 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin13.4.0 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

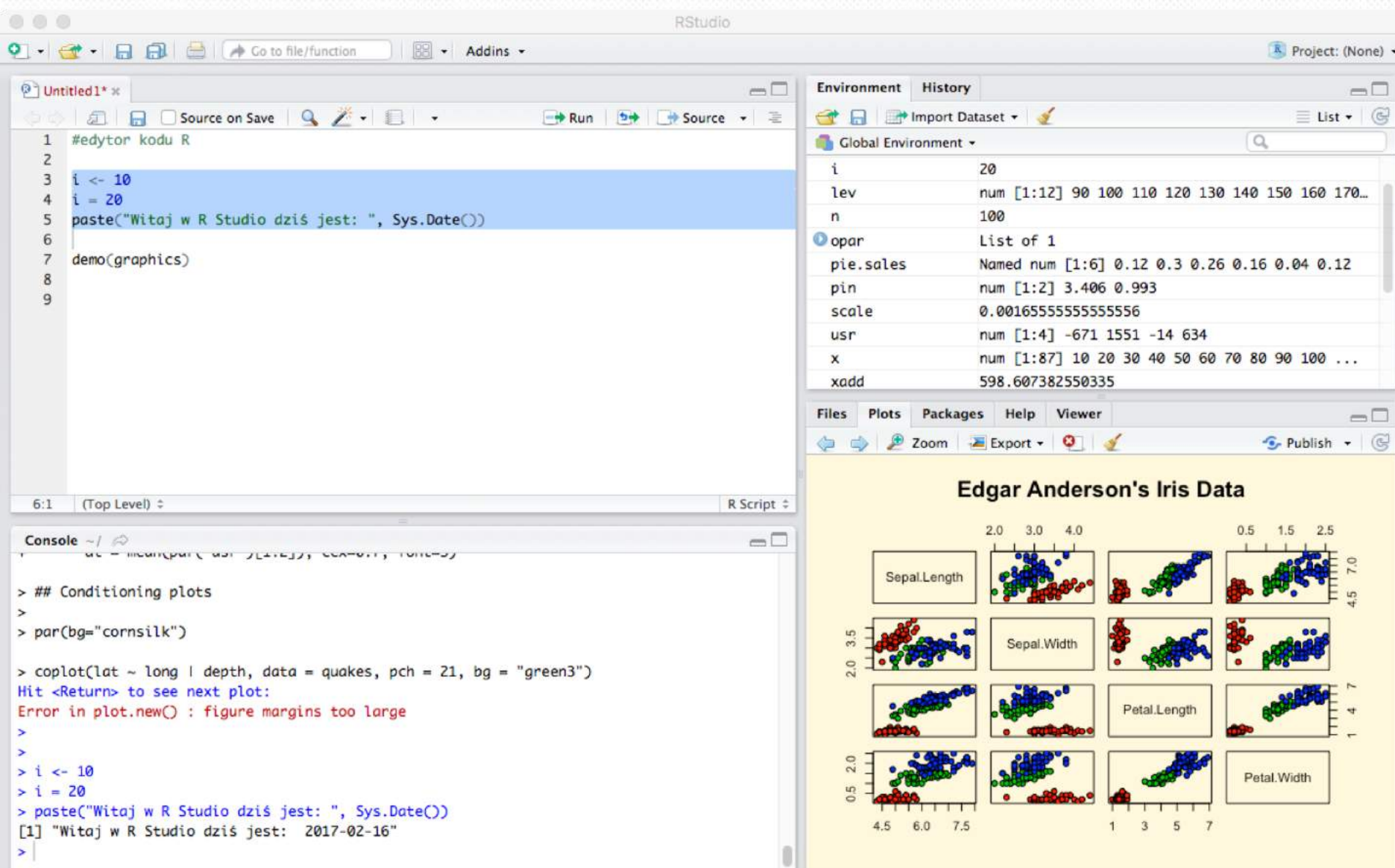
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[R.app GUI 1.68 (7288) x86_64-apple-darwin13.4.0]

Warning: namespace 'ROAuth' is not available and has been replaced
by .GlobalEnv when processing object 'tw'
[Workspace restored from /Users/mdraminski/.RData]
[History restored from /Users/mdraminski/.Rapp.history]

>
```

Studio <https://www.rstudio.com/> Integrated development environment (IDE)



The screenshot displays the RStudio Integrated Development Environment (IDE) interface. The top toolbar includes icons for file operations and a search bar. The main source editor shows an R script with the following code:

```
1 #edytor kodu R
2
3 i <- 10
4 i = 20
5 paste("Witaj w R Studio dziś jest: ", Sys.Date())
6
7 demo(graphics)
8
9
```

The Environment pane on the right shows the Global Environment with variables: i (20), lev (num [1:12] 90 100 110 120 130 140 150 160 170...), n (100), opar (List of 1), pie.sales (Named num [1:6] 0.12 0.3 0.26 0.16 0.04 0.12), pin (num [1:2] 3.406 0.993), scale (0.001655555555555556), usr (num [1:4] -671 1551 -14 634), x (num [1:87] 10 20 30 40 50 60 70 80 90 100 ...), and xadd (598.607382550335).

The Console pane at the bottom shows the execution of the script, including the output of the paste function and the error message: "Error in plot.new() : figure margins too large".

The Plots pane on the right displays a scatter plot titled "Edgar Anderson's Iris Data". The plot shows the relationship between Sepal.Length, Sepal.Width, Petal.Length, and Petal.Width, with data points colored by species (red, green, blue).

Pakiety w R

- Pakiet to biblioteka funkcji
 - 6 stycznia 2017 - 10000 pakietów na CRAN
 - 20 lutego 2018 - 12161 pakietów na CRAN
- Instalacja
 - > `install.packages("stringr")`
- Ładowanie
 - > `library("stringr")`
- Przykładowe przydatne pakiety
 - ggplot2 – pakiet do budowania wykresów
 - stringr – manipulacja tekstami
 - stringi – manipulacja tekstami
 - reshape2 – do przetwarzania tabel danych (tabele przestawne)
 - plyr – do przetwarzania tabel danych
 - dplyr - do szybkiego przetwarzania tabel danych (grupowanie, join itp)
 - data.table – do szybkiego przetwarzania dużych tabel danych
 - shiny – pakiet do budowania dynamicznego GUI dostępnego przez WWW

Gdzie szukać pakietów i pomocy?

INTERNET!

- <https://cran.r-project.org/>
- <https://www.bioconductor.org/>
- <https://www.r-bloggers.com/>



CRAN
Mirror
What's new?
Task Views
Search
About R
R Homepage
The R Journal
Software
R Source
R Binaries
Packages
Other
Documentation
Manuals
FAQs
Contributed

The Comprehensive R Archive Network

Download and Install R

Precompiled binary distributions of the base system and contributed packages. Windows and Mac users most likely want one of these versions of R:

- Download R for Linux
- Download R for Mac OS X
- Download R for Windows

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

Source Code for all Platforms

Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The source code has to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (Monday 2016-10-31, Sincere Pumpkin Patch) R-3.2.2 [page](#), read [what's new](#) in the latest version.
- Sources of R alpha and beta releases (daily snapshots, created only in time periods before a planned release).
- Sources of R alpha and beta releases (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are available [here](#). Please read about [new features](#) and [bug fixes](#) before filing corresponding feature requests or bug reports.
- Source code of older versions of R is [available here](#).
- Contributed extension packages

Questions About R

- If you have questions about R like how to download and install the software, or what the license terms are, please read our [answers to frequently asked questions](#) before you send an email.

What are R and CRAN?

R is "GNU S", a freely available language and environment for statistical computing and graphics which provides a wide variety of statistical and graphical techniques: linear and nonlinear modeling, statistical tests, time series analysis, classification, clustering, etc. Please consult the [R project homepage](#) for further information.

CRAN is a network of ftp and web servers around the world that store identical, up-to-date, versions of code and documentation for R. Please use the CRAN [mirror](#).



R news and tutorials contributed by (750) R bloggers

Home About RSS add your blog! Learn R R jobs Contact us

WELCOME!

Here you will find daily news and tutorials about R, contributed by over 750 bloggers.

There are many ways to follow us -

By e-mail:

Your e-mail here

Subscribe

On Facebook:

If you are an R blogger yourself you are invited to add your own R content feed to this site (Non-English R bloggers should add themselves here)

coauthorship and citation networks

February 20, 2017

By xi'an

As I discovered (!) the Annals of Applied Statistics in my mailbox just prior to taking the local train to Dauphine for the first time in 2017 (!), I started reading it on the way, but did not get any further than the first discussion paper by Pengsheng Ji and Jianhui Jin on coauthorship and

Read more »

RECENT POPULAR POSTS

stroke - structure your code better

PCA - hierarchical tree - partition: Why do we need to choose for visualizing data?

Pacoeextra R Package: Easy Multivariate Data Analysis and Elegant Visualization

Predicting food preferences with sparklyr (machine learning)

MOST VISITED ARTICLES OF THE WEEK

1. Catterplots: Plots with cats
2. How to write the first for loop in R
3. Installing R packages
4. Tutorials for learning R
5. Using apply, supply, apply in R
6. Should I Learn R or Python? ... It Doesn't Matter
7. How to Make a Histogram with Basic R
8. Predicting food preferences with sparklyr (machine learning)
9. How to perform PCA with R?

JOB FOR R-USERS

Research Analyst; National Parks & Public Lands

Senior Data Scientist - Predance

Full-stack developer with expertise in strachange R package, Github and

R Weekly

February 20, 2017

By readatweb

rxNeuralNet vs. xgBoost vs. H2O

February 20, 2017

By tomasz

9665407401

3124722111

Bioconductor

OPEN SOURCE SOFTWARE FOR BIOINFORMATICS

Search

Home Install Help Developers About

About Bioconductor

Bioconductor provides tools for the analysis and comprehension of high-throughput genomic data. Bioconductor uses the R statistical programming language, and is open source and open development. It has two releases each year, 1.200 software packages, and an active user community. Bioconductor is also available as an AWS (Amazon Machine Image) and a series of Docker images.

News

- Bioconductor 3.4 is available.
- Bioconductor 1.600 Research Channel launched.
- Orchestrating high-throughput genomic analysis with Bioconductor (abstract) and other recent literature.
- Read our latest newsletter and course material.
- Use the support site to get help installing, learning and using Bioconductor.

Install

Get started with Bioconductor

- Install Bioconductor
- Explore packages
- Get support
- Latest newsletter
- Follow us on Twitter
- Install R

Learn

Master Bioconductor tools

- Courses
- Support site
- Package vignettes
- Literature citations
- Carthagen work flows
- FAQ
- Community resources
- Videos

Use

Create bioinformatic solutions with Bioconductor

- Software Annotation and Expectation packages
- Amazon Machine Image
- Latest release announcements
- Support site

Develop

Contribute to Bioconductor

- Developer resources
- Use Rsrc Server
- Open Source Software Annotation and Expectation packages
- Package submission
- New package submission
- Build reports

About 66,400,000 results (0.45 seconds)

R: File Manipulation

<https://stat.ethz.ch/R-manual/R-devel/library/base/html/files.html> ▼

file.remove attempts to **remove** the **files** named in its argument. ... The **R** subscript recycling rule is used to align names given in vectors of different lengths.

You've visited this page many times. Last visit: 1/28/17

R: Delete Files and Directories

<https://stat.ethz.ch/R-manual/R-patched/library/base/html/unlink.html> ▼

x. a character vector with the names of the **file(s)** or directories to be **deleted**. Wildcards (normally '*' and '?') are allowed. recursive. logical. Should directories be ...

how to delete a file with R? - Stack Overflow

stackoverflow.com/questions/14219887/how-to-delete-a-file-with-r ▼

Jan 8, 2013 - I would like to know if there is a way in **R** to check up if a **file** is in my ... Type ?system at the console prompt ... Thanks @Arun I will try that solution ...

Automatically Delete Files/Folders in R - Stack Overflow

stackoverflow.com/questions/9296377/automatically-delete-files-folders-in-r ▼

Feb 15, 2012 - Maybe you're just looking for a combination of **file.remove** and **list.files** ? Maybe something like: `do.call(file.remove, list(list.files("C:/Temp", full.names ...`

R: Delete Files and Directories - Mit

<https://stuff.mit.edu/afs/sipb/project/r-project/lib/R/library/base/html/unlink.html> ▼

Details. If recursive = FALSE directories are not **deleted**, not even empty ones. **file.remove** can only **remove files**, but gives more detailed error information.

R help - R function to delete a csv file from disk - Nabble R

Pakiety

The screenshot displays the R Studio environment. A blue speech bubble with the word "Pakiety" (Packages) points to the "Packages" tab in the right-hand pane. The code editor on the left contains R code for installing and loading a package, and a console window at the bottom shows the execution of a plot function and subsequent help commands.

Code Editor (example.R):

```
1 #edytor kodu R
2
3 i <- 10
4 i = 20
5 paste("Witaj w R Studio dziś jest: ", Sys.Date())
6
7 demo(graphics)
8
9 install.packages("moj_pakiet")
10 library("moj_pakiet")
11
12 ?read.csv
```

Console:

```
> cplot(lat ~ long | depth, data = quakes, pch = 21, bg = "green3")
Hit <Return> to see next plot:
Error in plot.new() : figure margins too large
>
>
> i <- 10
> i = 20
> paste("Witaj w R Studio dziś jest: ", Sys.Date())
[1] "Witaj w R Studio dziś jest: 2017-02-16"
> help(read.csv)
> help("read.csv")
> help('read.csv')
> ?('read.csv')
> ?read.csv
>
```

Environment Pane:

Variable	Value
i	20
lev	num [1:12] 90 100 110 120 130 140 150 160 170...
n	100
opar	List of 1
ie.sales	Named num [1:6] 0.12 0.3 0.26 0.16 0.04 0.12
scd	num [1:2] 3.406 0.993
usr	0.001655555555555556
x	num [1:4] -671 1551 -14 634
xadd	num [1:87] 10 20 30 40 50 60 70 80 90 100 ...

Packages Pane:

Name	Description	Version
<input type="checkbox"/> acepack	ACE and AVAS for Selecting Multiple Regression Transformations	1.4.1
<input type="checkbox"/> assertthat	Easy pre and post assertions.	0.1
<input type="checkbox"/> base64enc	Tools for base64 encoding	0.1-3
<input type="checkbox"/> BH	Boost C++ Header Files	1.60.0-2
<input type="checkbox"/> bitops	Bitwise Operations	1.0-6
<input type="checkbox"/> boot	Bootstrap Functions (Originally by Angelo Canty for S)	1.3-18
<input type="checkbox"/> brew	Templating Framework for Report Generation	1.0-6
<input type="checkbox"/> caTools	Tools: moving window statistics, GIF, Base64, ROC AUC, etc.	1.17.1
<input type="checkbox"/> chron	Chronological Objects which can Handle Dates and Times	2.3-47
<input type="checkbox"/> class	Functions for Classification	7.3-14
<input type="checkbox"/> cluster	"Finding Groups in Data": Cluster Analysis Extended Rousseeuw et al.	2.0.5

Help, komentarze

```
# w R to jest komentarz  
# w R nie ma komentarzy blokowych  
# każdy komentarz musi zaczynać się  
# od znaku #
```

```
> help("read.csv")  
> help(`read.csv`)  
> help(read.csv)  
> ?read.csv
```



Help

The screenshot shows the R Studio interface. A blue speech bubble with the word "Help" points to the `read.csv` function in the Environment pane. The Source pane shows an R script with the following code:

```
1 #edytor kodu R
2
3 i <- 10
4 i = 20
5 paste("Witaj w R Studio dziś jest: ", Sys.Date())
6
7 demo(graphics)
8
9 install.packages("moj_pakiet")
10 library("moj_pakiet")
11
12 ?read.csv
```

The Environment pane shows the following objects:

Object	Value
i	20
lev	num [1:12] 90 100 110 120 130 140 150 160 170... 100
pl	List of 1
pin	Named num [1:6] 0.12 0.3 0.26 0.16 0.04 0.12
scale	num [1:2] 3.406 0.993
usr	0.001655555555555556
x	num [1:4] -671 1551 -14 634
xadd	num [1:87] 10 20 30 40 50 60 70 80 90 100 ... 598.607382550335

The Console shows the following output:

```
> coplot(lat ~ long | depth, data = quakes, pch = 21, bg = "green3")
Hit <Return> to see next plot:
Error in plot.new() : figure margins too large
>
>
> i <- 10
> i = 20
> paste("Witaj w R Studio dziś jest: ", Sys.Date())
[1] "Witaj w R Studio dziś jest: 2017-02-16"
> help(read.csv)
> help("read.csv")
> help('read.csv')
> ?('read.csv')
> ?read.csv
>
```

The Files pane shows the following files:

- read.table {utils}
- R Documentation

The Plots pane shows the following plots:

- Data Input

The Packages pane shows the following packages:

- read.table {utils}

The Help pane shows the following help text:

Data Input

Description

Reads a file in table format and creates a data frame from it, with cases corresponding to lines and variables to fields in the file.

Usage

```
read.table(file, header = FALSE, sep = "", quote = "\"'",
  dec = ".", numerals = c("allow.loss", "warn.loss",
    row.names, col.names, as.is = !stringsAsFactors,
    na.strings = "NA", colClasses = NA, nrows = -1,
    skip = 0, check.names = TRUE, fill = !blank.lines.skip,
    strip.white = FALSE, blank.lines.skip = TRUE,
    comment.char = "#",
```


Proste operacje arytmetyczne

```
> sin(0.5)
> 2^7 # potęgowanie
> log(10);2+2; # dwie operacje w 1 linii
> 10%%3 # operacja modulo

> moja_zmienna <- 2 * pi # przypisanie
> moja_zmienna = 2 * pi # j.w.
> 2 * pi -> moja_zmienna #zły styl!
> rm(moja_zmienna) #usuwamy zmienną
> remove(moja_zmienna) # j.w.
```

Zmienne

The screenshot displays the R Studio environment with the following components:

- Source Editor:** Contains R code for variable assignment and package management.
- Environment:** Lists variables in the Global Environment.
- Console:** Shows the execution of the code, including an error message and the output of the `paste` function.
- Documentation:** Displays the help page for `read.table`.

```
1 #edytor kodu R
2
3 i <- 10
4 i = 20
5 paste("Witaj w R Studio dziś jest: ", Sys.Date())
6
7 demo(graphics)
8
9 install.packages("moj_pakiet")
10 library("moj_pakiet")
11
12 ?read.csv
```

Environment:

Variable	Value
i	20
lev	num [1:12] 90 100 110 120 130 140 150 160 170...
n	100
opar	List of 1
pie.sales	Named num [1:6] 0.12 0.3 0.26 0.16 0.04 0.12
pin	num [1:2] 3.406 0.993
scale	0.0016555555555555556
usr	num [1:4] -671 1551 -14 634
x	num [1:87] 10 20 30 40 50 60 70 80 90 100 ...
xadd	598.607382550335

Console:

```
> coplot(lat ~ long | depth, data = quakes, pch = 21, bg = "green3")
Hit <Return> to see next plot:
Error in plot.new() : figure margins too large
>
>
> i <- 10
> i = 20
> paste("Witaj w R Studio dziś jest: ", Sys.Date())
[1] "Witaj w R Studio dziś jest: 2017-02-16"
> help(read.csv)
> help("read.csv")
> help('read.csv')
> ?('read.csv')
> ?read.csv
>
```

Documentation: Data Input

Description

Reads a file in table format and creates a data frame from it, with cases corresponding to lines and variables to fields in the file.

Usage

```
read.table(file, header = FALSE, sep = "", quote = "\"'",
  dec = ".", numerals = c("allow.loss", "warn.loss",
    row.names, col.names, as.is = !stringsAsFactors,
    na.strings = "NA", colClasses = NA, nrows = -1,
    skip = 0, check.names = TRUE, fill = !blank.lines.skip,
    strip.white = FALSE, blank.lines.skip = TRUE,
    comment.char = "#",
```


Wektory, sekwencje

```
> moj_wektor <- c(1, 2, 3)
> moj_wektor <- c(moj_wektor, 4, c(5))

> moj_wektor <- 1:15
> moj_wektor <- seq(from = 1, to = 15,
by = 1)
> moj_wektor <- seq(-5, 5)
> moj_wektor <- seq(5, -5, length=10)
> moj_wektor^2
> rep(1:3, 3)
```



Indeksowanie wektorów

```
> wektor <- 1:10  
> wektor[1] #pierwszy element  
> wektor[c(1,3)] #pierwszy i trzeci  
> wektor[1,3] #źle bo wektor ma 1 wymiar  
> wektor[1:3] #od pierwszego do trzeciego  
> wektor>5  
> wektor[moj_wektor>5]  
> wektor[-2] #wszystkie poza drugim  
  
> letters[1]  
> LETTERS[1:3]
```



Operacje na wektorach

```
> a <- 1:20
```

```
> b <- seq(5, 25, 5)
```

```
> a + b
```

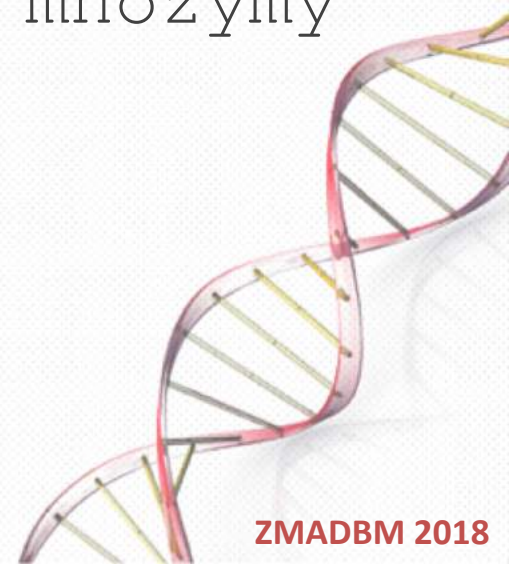
```
> [1]  6 12 18 24 30 11 17 23 29 35 16  
22 28 34 40 21 27 33 39 45
```

```
> a * c(1,2) #co drugi element mnożymy  
przez 2
```

```
> a[c(3,4)] <- 55
```

```
> a[3] <- b[3]
```

```
> a[1] <- NA
```



Operacje na wektorach - funkcje

- > `length(a)` #długość wektora
- > `summary(a)` #podsumowanie/statystyki
- > `sum(a)` #suma elementów
- > `min(a)` #minimalna wartość
- > `max(a)` #maksymalna wartość
- > `mean(a)` #średnia
- > `sort(a)` #zwraca posortowany wektor
- > `rev(a)` #odwraca wektor
- > `which(a==3)` #indeks elementu wartości 3
- > `diff(a)` #różnica kolejnych elementów



Typy danych i rzutowanie

```
> a <- 4  
> class(a) #typ numeric  
> class("tekst") #typ character  
> class(1:10) #ten wektor to typ integer  
> class(as.character(1:8)) #rzutujemy typ  
> class(as.numeric("1")) #rzutujemy typ  
> format(pi, digits = 3) #formatujemy  
> is.numeric(10)  
> format(as.Date("01/10/2017",  
    "%m/%d/%Y"), "%a, %d %b %Y")  
> class(T) # To samo co TRUE
```

Logiczne operacje na wektorach

- > `c(3,4,5,6) %in% c(3, 5)` #które elementy
- > `c(3, 5) %in% c(3,4,5,6)` #są takie same
- > `all(c(rep(T,5)))`
- > `all(c(rep(T,5),F))` #czy wszystkie T?
- > `any(c(rep(T,5),F))` #czy jakikolwiek T?
- > `c(T,F,T) & c(F,T,F)`
- > `c(T,F,T) | c(F,T,F)`
- > `xor(c(T,F,T), c(F,T,F))`



Operacje na tekstach

```
> moj_tekst <- "czesc" #przypisz tekst
> print(moj_tekst) #wyswietl na ekranie
> print(paste("dzis jest", Sys.Date()))
> cat("Czesc\n")
> gsub("a", "_", "abc") #zamiana ciągu
> substr("abc", 1, 2)
> grep("ma", c("ala", "ma", "kota"))

> library(stringr)
> str_trim('      te kst ') #usuń spacje
> str_replace_all("abc", "a",
"_) #zamiana
```

`rm(list = ls())`

`load.image`

`save.image`

The screenshot shows the RStudio interface with the following components:

- Source Editor:** Contains R code for a script named `example.R`. The code includes comments, variable assignments, date printing, a graphics demo, package installation, and a call to `?read.csv`.
- Environment Pane:** Displays the global environment with variables like `lev`, `n`, `opar`, `pie.sales`, `pin`, `scale`, `usr`, `x`, and `xadd`. A blue callout bubble labeled `load.image` points to the top of this pane. Another blue callout bubble labeled `save.image` points to the `Global Environment` dropdown menu.
- Console:** Shows the execution of the code from the source editor, including an error message: `Error in plot.new() : figure margins too large`.
- Files, Plots, Packages, Help, Viewer:** The bottom right pane shows the `R: Data Input` section with the `read.table` function documentation.

Console Output:

```
> coplot(lat ~ long | depth, data = quakes, pch = 21, bg = "green3")
Hit <Return> to see next plot:
Error in plot.new() : figure margins too large
>
>
> i <- 10
> i = 20
> paste("Witaj w R Studio dziś jest: ", Sys.Date())
[1] "Witaj w R Studio dziś jest: 2017-02-16"
> help(read.csv)
> help("read.csv")
> help('read.csv')
> ?('read.csv')
> ?read.csv
>
```

Environment Pane Variables:

Variable	Value
<code>lev</code>	num [1:12] 90 100 110 120 130 140 150 160 170...
<code>n</code>	100
<code>opar</code>	List of 1
<code>pie.sales</code>	Named num [1:6] 0.12 0.3 0.26 0.16 0.04 0.12
<code>pin</code>	num [1:2] 3.406 0.993
<code>scale</code>	0.0016555555555555556
<code>usr</code>	num [1:4] -671 1551 -14 634
<code>x</code>	num [1:87] 10 20 30 40 50 60 70 80 90 100 ...
<code>xadd</code>	598.607382550335

Files, Plots, Packages, Help, Viewer:

R: Data Input

read.table {utils}

Description

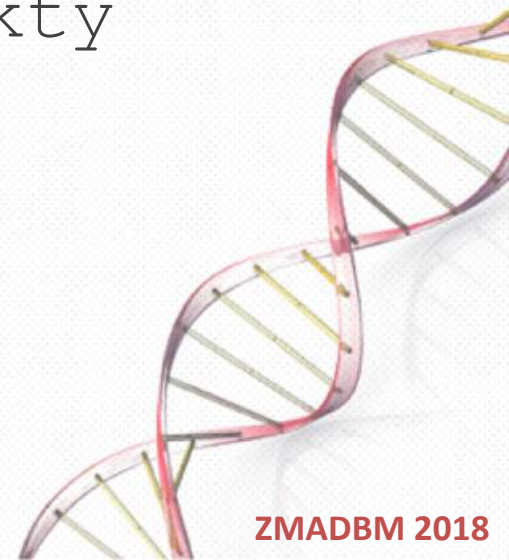
Reads a file in table format and creates a data frame from it, with cases corresponding to lines and variables to fields in the file.

Usage

```
read.table(file, header = FALSE, sep = "", quote = "\"'",
  dec = ".", numerals = c("allow.loss", "warn.loss",
    row.names, col.names, as.is = !stringsAsFactors,
    na.strings = "NA", colClasses = NA, nrows = -1,
    skip = 0, check.names = TRUE, fill = !blank.lines.skip,
    strip.white = FALSE, blank.lines.skip = TRUE,
    comment.char = "#",
```


Zapis i odczyt środowiska (workspace)

```
> getwd()  
> list.files()  
> setwd("C:\\temp")  
> save.image("moj_workspace")  
> load.image("moj_workspace")  
> rm(list = ls()) # usuwa obiekty
```



Import tabel

The screenshot displays the R Studio environment with four main panes:

- Source Editor:** Contains R code for a script named `example.R`. The code includes comments, variable assignments, a date string, a graphics demo, package installation, and a call to `?read.csv`.
- Console:** Shows the execution of the code. It displays a warning about plot margins, followed by the execution of `i <- 10`, `i = 20`, and a `paste` statement that outputs the current date: `[1] "Witaj w R Studio dziś jest: 2017-02-16"`. It also shows help text for `read.csv`.
- Environment:** Lists the objects in the global environment. These include variables `i` (value 20), `lev` (a numeric vector), `n` (value 100), `opar` (a list), `pie.sales` (a named numeric vector), `pin` (a numeric vector), `scale` (a numeric vector), `usr` (a numeric vector), `x` (a numeric vector), and `xadd` (a numeric vector).
- Help Pane:** Displays the documentation for the `read.table` function. It includes a description of the function's purpose (reading a file into a data frame) and its usage, showing the full signature of `read.table` with its arguments and default values.

```
1 #edytor kodu R
2
3 i <- 10
4 i = 20
5 paste("Witaj w R Studio dziś jest: ", Sys.Date())
6
7 demo(graphics)
8
9 install.packages("moj_pakiet")
10 library("moj_pakiet")
11
12 ?read.csv
```

```
> coplot(lat ~ long | depth, data = quakes, pch = 21, bg = "green3")
Hit <Return> to see next plot:
Error in plot.new() : figure margins too large
>
>
> i <- 10
> i = 20
> paste("Witaj w R Studio dziś jest: ", Sys.Date())
[1] "Witaj w R Studio dziś jest: 2017-02-16"
> help(read.csv)
> help("read.csv")
> help('read.csv')
> ?('read.csv')
> ?read.csv
>
```

Environment

Object	Value
i	20
lev	num [1:12] 90 100 110 120 130 140 150 160 170...
n	100
opar	List of 1
pie.sales	Named num [1:6] 0.12 0.3 0.26 0.16 0.04 0.12
pin	num [1:2] 3.406 0.993
scale	0.00165555555555555
usr	num [1:4] -671 1551 -14 634
x	num [1:87] 10 20 30 40 50 60 70 80 90 100 ...
xadd	598.607382550335

Files **Plots** **Packages** **Help** **Viewer**

R: Data Input Find in Topic

read.table {utils}

R Documentation

Data Input

Description

Reads a file in table format and creates a data frame from it, with cases corresponding to lines and variables to fields in the file.

Usage

```
read.table(file, header = FALSE, sep = "", quote = "\"'",
  dec = ".", numerals = c("allow.loss", "warn.loss",
    row.names, col.names, as.is = !stringsAsFactors,
    na.strings = "NA", colClasses = NA, nrows = -1,
    skip = 0, check.names = TRUE, fill = !blank.lines.skip,
    strip.white = FALSE, blank.lines.skip = TRUE,
    comment.char = "#",
```


Ramki danych 1 - tabele (data.frame)

```
> student_mat <- read.csv( file =  
  'student-mat.csv', sep = ';')  
> head(student_mat)  
> tail(student_mat)  
> View(student_mat)  
> student_mat[1,1]  
> student_mat[1:3,]  
> student_mat$age  
> student_mat[student_mat$age==18,]  
> student_mat$ocena <- 5
```



Ramki danych 2 - tabele (data.frame)

```
> ncol(student_mat)
> nrow(student_mat[student_mat$age==18,])
> names(student_mat) #lub colnames
> names(student_mat)[1] <- "szkola"
> student_mat[,2,drop=F]
> ranking <- order(student_mat$G3,
decreasing = T)
> student_mat[ranking,]
> write.csv(student_mat, "studenci.csv")
> saveRDS(student_mat, "studenci.rds")
> studenci <- readRDS("studenci.rds")
```


Ramki danych 3 - tabele (data.frame)

```
> head(student_mat[, c("G1", "G2", "age")])  
> head(student_mat[, names(student_mat)  
  %in% c("G1", "G2", "age")])  
> mtcars #wbudowane zbioru danych
```



View data.frame

The screenshot displays the RStudio environment with the following components:

- Environment Pane:** Shows the `student_mat` data frame with 395 observations and 33 variables.
- Table View:** A preview of the `student_mat` data frame with columns: `school`, `sex`, `age`, `address`, `famsizē`, `Pstatus`, `Medū`, `Fedū`, `Mjob`, `Fjob`, `reason`, `guardian`, `traveltime`, `studytime`, and `failu`.
- Console:** Contains the following R code:

```
trav = col_integer(),
traveltime = col_integer(),
studytime = col_integer(),
failures = col_integer(),
famrel = col_integer(),
freetime = col_integer(),
goout = col_integer(),
Dalc = col_integer(),
Walc = col_integer(),
health = col_integer(),
absences = col_integer(),
G1 = col_integer(),
G2 = col_integer(),
G3 = col_integer()
}
See spec(...) for full column specifications.
> View(student_mat)
> |
```
- Environment Pane (Right):** Shows the `Global Environment` with the following variables:
 - `a`: `int [1:20]` NA 2 3 4 5 6 7 8 9 10 ...
 - `b`: `num [1:5]` 5 10 15 20 25
 - `i`: `10L`
 - `moj_vektor`: `num [1:16]` 0 1 2 3 4 5 6 7 8 9 ...
 - `moja_lista`: `List of 2`
 - `vektor`: `int [1:10]` 1 2 3 4 5 6 7 8 9 10
 - `x`: `9`
- Files Pane:** Shows the `R: Control Flow` section with the following text:

`break` breaks out of a `for`, `while` or `repeat` loop; control is transferred to the first statement outside the inner-most loop. `next` halts the processing of the current iteration and advances the looping index. Both `break` and `next` apply only to the innermost of nested loops.

Note that it is a common mistake to forget to put braces `{ ... }` around your statements, e.g., after `if (...)` or `for (...)`. In particular, you should not have a newline between `}` and `else` to avoid a syntax error in entering a `if ... else` construct at the keyboard or via source. For that reason, one (somewhat extreme) attitude of defensive programming is to always use braces, e.g., for `if` clauses.

The `seq` in a `for` loop is evaluated at the start of the loop; changing it subsequently does not affect the loop. If `seq` has length zero the body of the loop is skipped. Otherwise the variable `var` is assigned in turn the value of each element of `seq`. You can assign to `var` within the body of the loop, but this will not affect the next iteration. When the loop terminates, `var` remains as a variable containing its latest value.

Value

`if` returns the value of the expression evaluated, or `NULL` invisibly if none was (which may happen if there is no `else`).

`for`, `while` and `repeat` return `NULL` invisibly. `for` sets `var` to the last used element of `seq`, or to `NULL` if it was of length zero.

`break` and `next` do not return a value as they transfer control within the loop.

References

Instrukcja warunkowa – if() else

```
> x <- -5  
> if(x > 0) {  
    print("wartosc dodatnia")  
} else if(x == 0) {  
    print("zero")  
} else {  
    print("wartosc ujemna")  
}
```



Pętla for

```
#dodaj do kazdego elementu wektora 3  
> x <- 1:10  
> for(i in 1:length(x)) {  
  x[i] <- x[i] + 3;  
};  
print(x)
```

```
#drugi przyklad  
> for (year in c(2014,2015,2016,2017)){  
  print(paste("The year is", year))  
};
```


Pętla while

```
> x <- 1  
> while(x < 5) {  
  x <- x+1;  
  print(x);  
}
```

next #skok ponownie do warunku
break #przerywa pętlę



Własne funkcje

```
> moja_potega <- function(x, p = 2) {  
  y <- x ^ p  
  return(y)  
};
```

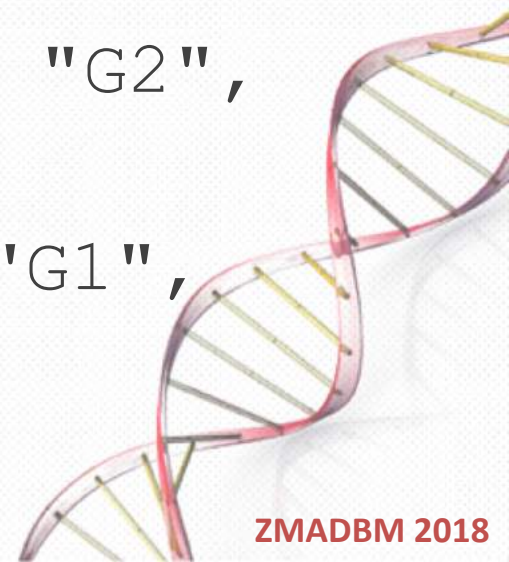
```
> moja_potega(3)
```

```
> moja_potega(1:3, 3) #zwraca wektor
```



Cbind, rbind i apply

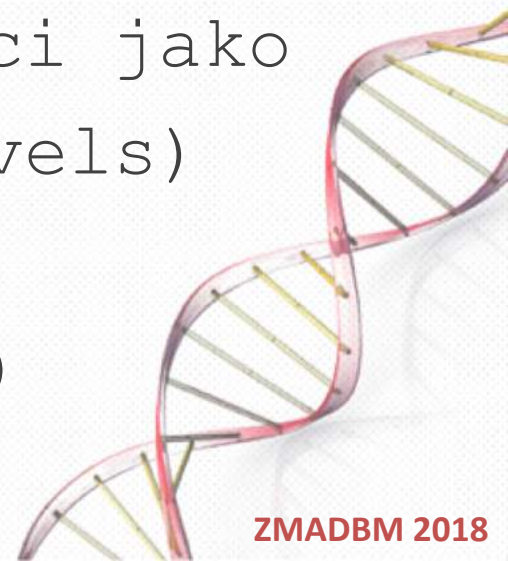
```
> head(cbind(student_mat[,3,drop=F],  
             student_mat[,34,drop=F]))  
  
> rbind(student_mat[1:2,],  
        student_mat[10,])  
  
> student_mat$mean_G <-  
    apply(student_mat[,c("G1", "G2",  
                          "G3")], 1, mean)  
  
> apply(student_mat[,c("age", "G1",  
                        "G2")], 2, mean)
```



Listy i factory

```
> moja_lista <- list()
> moja_lista[[1]] <- seq(1,10,2)
> moja_lista[[2]] <- "drugi element"
> moja_lista[[3]] <- TRUE

> class(student_mat$sex)
#factor czyli nominalne wartości jako
#liczby i słownik poziomów (levels)
> as.integer(student_mat$sex)
> as.character(student_mat$sex)
```





Ćwiczenia 1

michal.draminski@ipipan.waw.pl

Zdjęcia, schematy i rysunki zostały zaczerpnięte
z internetu.

Ćwiczenia 1

Z1. Zbuduj wektor ciągu liczb od 1 do 60 co 1 i przypisz do zmiennej. Zastąp co trzeci wyraz ciągu wartością NA **(1 pkt)**

Z2. Zbuduj wektor postaci *liczba_litera* gdzie liczba jest z zakresu od 1 do 40 a litery to "a" "b" "c" "d" występujące cyklicznie. "1-a", "2-b", "3-c", "4-d" "5-a" "6-b" itd. **(1 pkt)**

Z3. Napisz funkcję która wyświetla kolejne liczby naturalne w dół do zera zaczynając od liczby podanej jako parameter *n* funkcji. Funkcja zwraca resztę z dzielenia przez parameter *m*. Domyślnie $n = 20$ a $m = 2$ **(1 pkt)**

Z4. Dla zbioru *mtcars* wybierz mercedesy ("Merc") i policz średnie wartości kolumn: "mpg", "cyl", "gear". Wynik zapisz jako *data.frame* o jednym wierszu i 3 kolumnach **(2 pkt)**

Ćwiczenia 1 cd

Z5. Napisz funkcję kalkulator dla dwóch parametrów wejściowych m i n i działania arytmetycznego podanego jako typ character "+", "-", "*", "/". Funkcja zwraca data.frame o 3 kolumnach i dwóch wierszach:

- w pierwszym wierszu m , n oraz wynik działania na obu liczbach,
- w drugim wierszu m^2 , n^2 oraz wynik działania na obu liczbach.

Funkcja sprawdza typ danych wejściowych i jeśli m lub n nie są liczbami funkcja zwraca wartość NULL I wyświetla komunikat o braku możliwości przeprowadzenia działania. **(5 pkt)**.

