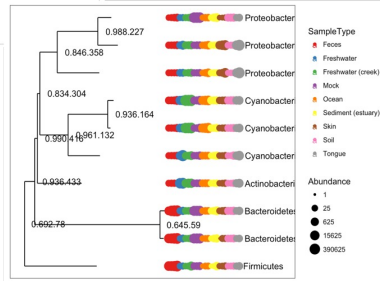


R and Rstudio :Introduction and getting started

```
tax_table(GlobalPatterns)[1:5, 1:4]
```

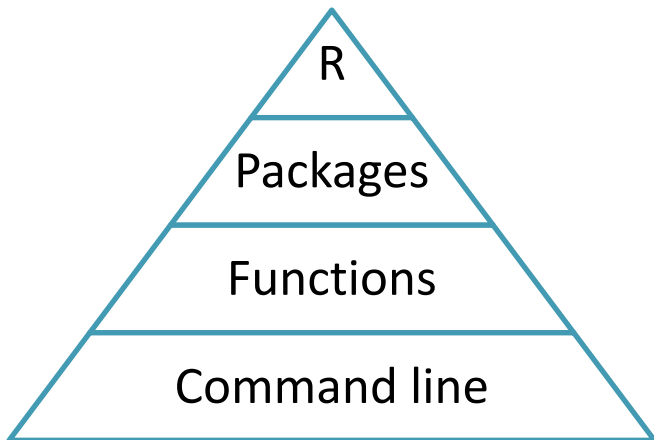
```
## Taxonomy Table: [5 taxa by 4 taxonomic ranks]:  
## Kingdom Phylum Class Order  
## 549322 "Archaea" "Crenarchaeota" "Thermoprotei" NA  
## 522457 "Archaea" "Crenarchaeota" "Thermoprotei" NA  
## 951 "Archaea" "Crenarchaeota" "Thermoprotei" "Sulfolobales"  
## 244423 "Archaea" "Crenarchaeota" "Sd-NA" NA  
## 586076 "Archaea" "Crenarchaeota" "Sd-NA" NA
```



What is cran R?



What is cran R?

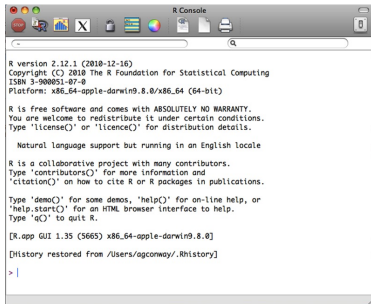


What is cran R?

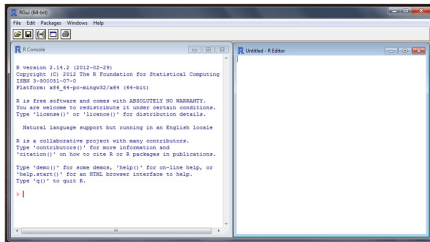
Here are some examples of the core packages that are commonly included with R:

1. **base** - This is the base R package that provides essential functions and data types.
2. **datasets** - This package contains various datasets that are frequently used for examples and testing.
3. **graphics** - Provides functions for creating graphical plots and charts.
4. **stats** - Includes statistical functions and models, such as linear regression and hypothesis testing.
5. ...

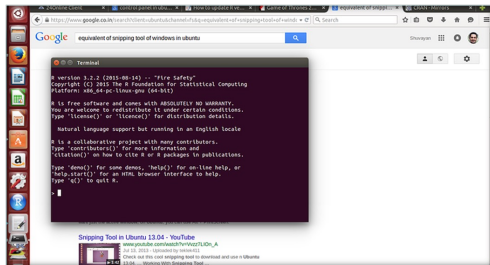
What is cran R?



```
R Console  
R version 2.12.1 (2010-12-16)  
Copyright (C) 2010 The R Foundation for Statistical Computing  
ISBN 3-900051-87-0  
Platform: x86_64-apple-darwin9.8.0/x86_64 (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 1.35 (5665) x86_64-apple-darwin9.8.0]  
[History restored from /Users/gqcomway/.Rhistory]  
> |
```



```
R Console  
R version 2.14.0 (2012-02-29)  
Copyright (C) 2012 The R Foundation for Statistical Computing  
ISBN 3-900051-07-0  
Platform: x86_64-w64-mingw32/x64 (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.  
  
> |
```



```
Terminal  
R version 3.2.2 (2015-06-18) -- "Fire Safety"  
Copyright (C) 2015 The R Foundation for Statistical Computing  
Platform: x86_64-pc-linux-gnu (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.  
  
> |
```



Rstudio is an integrated development environment (IDE)

The screenshot displays the RStudio interface with the following components:

- Source Editor:** Contains R code for loading data, summarizing it, and creating a faceted scatter plot.


```

1 library(ggplot2)
2 source("plots/formatPlot.R")
3
4 View(diamonds)
5 summary(diamonds)
6
7 summary(diamonds$price)
8 aveSize <- round(mean(diamonds$carat), 4)
9 clarity <- levels(diamonds$clarity)
10
11 p <- qplot(carat, price,
12           data=diamonds, color=clarity,
13           xlab="carat", ylab="Price",
14           main="Diamond Pricing")
15

```
- Console:** Shows the output of the executed code, including summary statistics for the 'price' variable and the execution of the plotting commands.


```

Min. x: 0.000 Min. y: 0.000 Min. z: 0.000
1st Qu.: 4.710 1st Qu.: 4.720 1st Qu.: 2.910
Median : 5.700 Median : 5.710 Median : 3.530
Mean : 5.731 Mean : 5.735 Mean : 3.539
3rd Qu.: 6.540 3rd Qu.: 6.540 3rd Qu.: 4.040
Max. :10.740 Max. :58.900 Max. :31.800
> summary(diamonds$price)
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 326    950    2401    3993    5324   18820
> aveSize <- round(mean(diamonds$carat), 4)
> clarity <- levels(diamonds$clarity)
> p <- qplot(carat, price,
+           data=diamonds, color=clarity,
+           xlab="carat", ylab="Price",
+           main="Diamond Pricing")
>
> format.plot(p, size=24)
> |

```
- Workspace:** Lists the loaded data (diamonds) and functions (format.plot, ggplot2).
- Files:** Shows the current project files and options like Zoom and Export.
- Plots:** Displays a faceted scatter plot titled "Diamond Pricing". The x-axis is "Carat" (0.0 to 3.5) and the y-axis is "Price" (0 to 15000). Points are colored by clarity, with a legend on the right showing categories: I1, SI2, SI1, VS2, VS1, VVS2, VVS1, and IF.

An editing window opens where you can write and save a series of instructions in a file: your **script**!

The screenshot displays the RStudio environment. The script editor window, titled 'diamondPricing.R', contains the following R code:

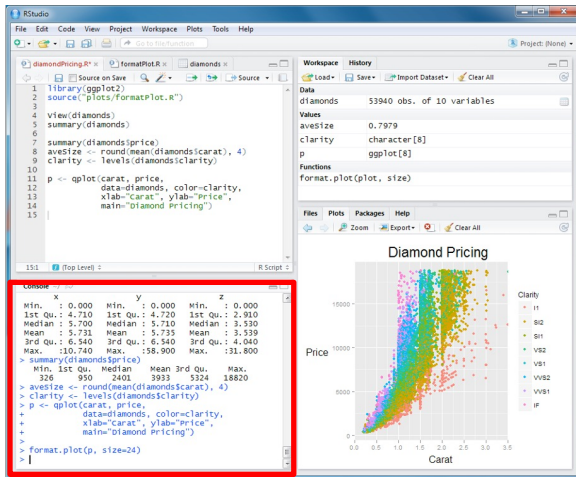
```
1 library(ggplot2)
2 source("plots/formatPlot.R")
3
4 View(diamonds)
5 summary(diamonds)
6
7 summary(diamonds$price)
8 aveSize <- round(mean(diamonds$carat), 4)
9 clarity <- levels(diamonds$clarity)
10
11 p <- qplot(carat, price,
12           data=diamonds, color=clarity,
13           xlab="carat", ylab="Price",
14           main="Diamond Pricing")
15
```

The console window shows the output of the executed code:

```
Min. : 0.000 Min. : 0.000 Min. : 0.000
1st Qu.: 4.710 1st Qu.: 4.720 1st Qu.: 2.910
Median : 5.700 Median : 5.710 Median : 3.530
Mean : 5.731 Mean : 5.735 Mean : 3.539
3rd Qu.: 6.540 3rd Qu.: 6.540 3rd Qu.: 4.040
Max. :10.740 Max. :58.900 Max. :31.800
> summary(diamonds$price)
Min. 1st Qu. Median Mean 3rd Qu. Max.
 326 950 2401 3933 5324 18820
> aveSize <- round(mean(diamonds$carat), 4)
> clarity <- levels(diamonds$clarity)
> p <- qplot(carat, price,
+           data=diamonds, color=clarity,
+           xlab="carat", ylab="Price",
+           main="Diamond Pricing")
> format.plot(p, size=24)
|
```

The workspace pane on the right shows the 'diamonds' data frame with 53940 observations and 10 variables. The 'Data' section lists 'aveSize' (0.7979) and 'clarity' (character [8]). The 'Functions' section lists 'format.plot(plot, size)'. The 'Plots' pane shows a scatter plot titled 'Diamond Pricing' with 'Price' on the y-axis and 'Carat' on the x-axis. The plot is colored by 'Clarity', with a legend on the right showing categories: I1, SI2, SI1, VS2, VSI, VS2, VS1, and IF.

A terminal window giving all the output from the script



The screenshot shows the RStudio interface with a script editor on the left and a console on the right. The script editor contains the following R code:

```
1 library(ggplot2)
2 source("plots/formatPlot.R")
3
4 view(diamonds)
5 summary(diamonds)
6
7 summary(diamonds$price)
8 aveSize <- round(mean(diamonds$carat), 4)
9 clarity <- levels(diamonds$clarity)
10
11 p <- qplot(carat, price,
12            data=diamonds, color=clarity,
13            xlab="carat", ylab="price",
14            main="Diamond Pricing")
15
```

The console window shows the output of the script, which is highlighted with a red box:

```
Console
x      y      z
Min.  :0.000  Min.  :0.000  Min.  :0.000
1st Qu.:4.710  1st Qu.:4.720  1st Qu.:2.910
Median :5.700  Median :5.710  Median :3.530
Mean   :5.731  Mean   :5.735  Mean   :3.539
3rd Qu.:6.540  3rd Qu.:6.540  3rd Qu.:4.040
Max.   :10.740  Max.   :58.900  Max.   :31.800
> summary(diamonds$price)
  Min. 1st Qu. Median Mean 3rd Qu. Max.
  326   950   2401   3933   5324  18820
> aveSize <- round(mean(diamonds$carat), 4)
> clarity <- levels(diamonds$clarity)
> p <- qplot(carat, price,
+           data=diamonds, color=clarity,
+           xlab="carat", ylab="price",
+           main="Diamond Pricing")
> format.plot(p, size=24)
> |
```

The right pane shows a scatter plot titled "Diamond Pricing" with "Price" on the y-axis and "Carat" on the x-axis. The plot is faceted by "Clarity" with a legend on the right showing categories: I1, SI2, SI1, VS2, VS1, VVS2, VVS1, and IF.

A **Workspace** in which all the objects created/generated by the code are displayed (*i.e.* Matrices, Vectors, Functions, Constants...). It includes a **History** tab which returns all the executed instructions.

The screenshot displays the RStudio environment with the following components:

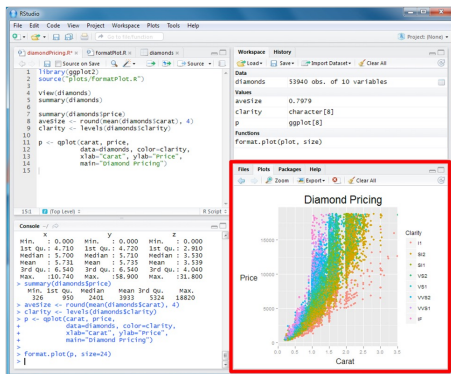
- Source Editor:** Contains R code for loading data, summarizing it, and creating a scatter plot.
- Console:** Shows the output of the executed code, including summary statistics for the 'diamonds' dataset and the execution of the 'qplot' function.
- Workspace/History Panel:** A red box highlights this panel, which lists the objects created in the workspace (diamonds, aveSize, clarity, p) and the functions used (format.plot).
- Plots Panel:** Displays a scatter plot titled 'Diamond Pricing' showing Price vs. Carat, with points colored by Clarity.

```
1 library(ggplot2)
2 source("plots/formatPlot.R")
3
4 view(diamonds)
5 summary(diamonds)
6
7 summary(diamonds$price)
8 aveSize <- round(mean(diamonds$carat), 4)
9 clarity <- levels(diamonds$clarity)
10
11 p <- qplot(carat, price,
12            data=diamonds, color=clarity,
13            xlab="Carat", ylab="Price",
14            main="Diamond Pricing")
15
```

Console Output:

```
Min. : 0.000 Min. : 0.000 Min. : 0.000
1st Qu.: 4.710 1st Qu.: 4.720 1st Qu.: 2.910
Median : 5.700 Median : 5.710 Median : 3.530
Mean : 5.731 Mean : 5.735 Mean : 3.539
3rd Qu.: 6.540 3rd Qu.: 6.540 3rd Qu.: 4.040
Max. :10.740 Max. :18.900 Max. :31.800
> summary(diamonds$price)
Min. 1st Qu. Median Mean 3rd Qu. Max.
326 950 2401 3993 5324 18820
> aveSize <- round(mean(diamonds$carat), 4)
> clarity <- levels(diamonds$clarity)
> p <- qplot(carat, price,
+           data=diamonds, color=clarity,
+           xlab="Carat", ylab="Price",
+           main="Diamond Pricing")
> format.plot(p, size=24)
|
```

- This last window contains 4 tabs :
- Files** (locates your working directory and allows you to navigate),
 - Plots** (returns all graphic outputs),
 - Packages** (package management)
 - Help** (this is R's 112 or 911, contains all of R's help)



The commandes on R:

An R command is either an expression or an assignment :

- ▶ Expression is directly evaluated, and the result is displayed on terminal

Exemple :

```
>2+3  
[1]5  
>sqrt(25)  
[1]5
```

Mathematical calcul

The commandes on R:

An R command is either an expression or an assignment :

- ▶ An assignement is an expression stored in object or variable

Exemple :

```
>a<-5
```

```
>a
```

```
[1]5
```

```
a <- 5
```

Object is named a

Value stored is 5

Sign used for d'affectation

<- ou =

Why store?

To manipulate later

The commandes on R:

An R command is either an expression or an assignment :

- ▶ An assignement is an expression stored in object or variable

Exemple :

```
>a<-5
```

```
>a
```

```
[1]5
```

```
a <- 5
```

```
class(a)
```

```
[1] Num
```

**It is very important to know
what type of object**

The commandes on R:

An R command is either an expression or an assignment :

- ▶ Object can be :

Character : ("A", "B", "C", ..)

Numeric : (1,2, 5.5, 6.7, ...)

Logic : (TRUE, FALSE)

- ▶ But also more complex:

Vector

Matrix

Data frame

List

The commandes on R:

Object Vector

- ▶ Vector contain only element with same mode (ex : numeric or character or logic ..)

```
Age = c(24, 26, 32, 54)
```

C() is the function to build a vector

Object type = Vector

There are the affected elements

The commandes on R:

What is the structure of my object ?

Function `class()` and `str()`

```
Age = c(24, 26, 32, 54)
```

```
str(Age)
```

```
Num[1:4]
```

```
class(Age)
```

```
Numeric
```

During the training we will see, how can we choose element inside a Vector or filter a vector

The commandes on R:

Object Matrix

- ▶ Matrix contains several vectors that build a table with columns and rows. The vectors belong to the same mode.

	[,1]	[,2]
[1,]	1	4
[2,]	2	5
[3,]	3	6

The commandes on R:

Object data frame

- ▶ Data frame contains several vectors that build a table with columns and rows. The vectors can belong to different modes

		colnames			
		Nom	Prenom	Age	Region_naissance
rownames	A	Keïta	Modibo	34	1
	B	Traoré	Moussa	23	2
	C	Konaré	Oumar	21	2
	D	Touré	Toumani	43	4
	E	Keïta	Boubacar	54	3

row.names() : row name (i.e. sample name)

colnames() : header(Nom, Prenom, etc)

The Workflow

Create Project

- Allow to save working environment automatically
- Keep the working environment open (scripts ...)

Check/Install packages

- `install.packages("your packages")`
- `BiocManager::install("dada2")`

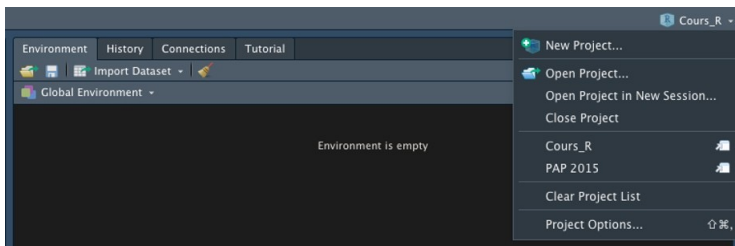
Load library

- `library("dada2")`
- `dada2::function`

Execute function

- `merge.Pairs(dadaFs, derepFS, verbose = TRUE)`




Work with projects with RStudio



Work with projects with RStudio

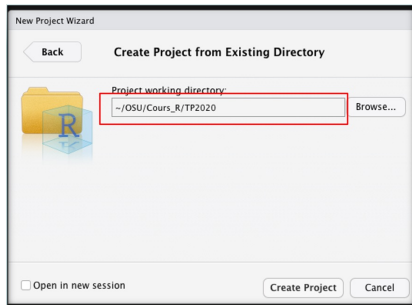
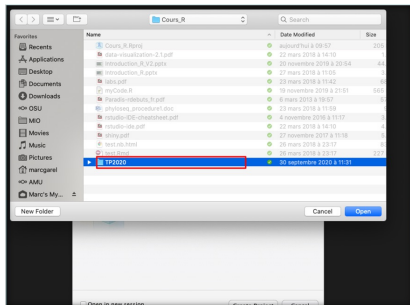
New Project

Create Project

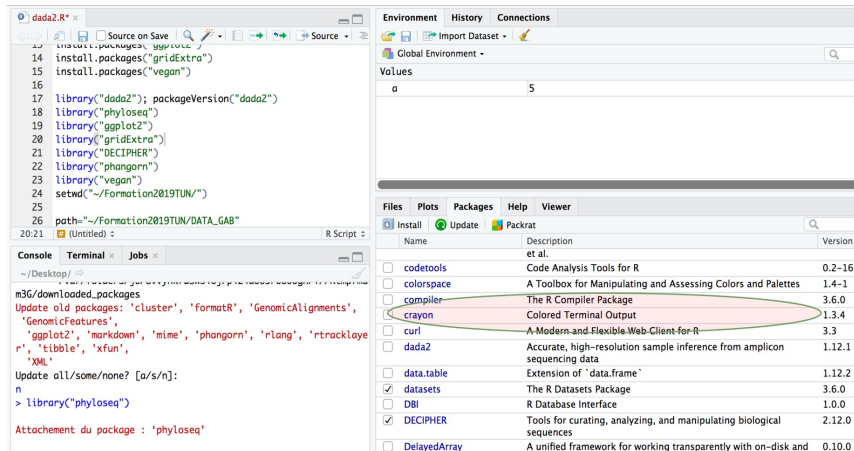
-  **New Directory**
Start a project in a brand new working directory >
-  **Existing Directory**
Associate a project with an existing working directory >
-  **Version Control**
Checkout a project from a version control repository >

Cancel

Work with projects with RStudio



Check install packages



The screenshot displays the RStudio interface. The left pane shows the R script editor with the following code:

```
13 this.dir.packages("ggplot2")
14 install.packages("gridExtra")
15 install.packages("vegan")
16
17 library("dada2"); packageVersion("dada2")
18 library("phyloseq")
19 library("ggplot2")
20 library("gridExtra")
21 library("DECIPHER")
22 library("phangorn")
23 library("vegan")
24 setwd("~/Formation2019TUN/")
25
26 path = "~/Formation2019TUN/DATA_GAB"
```

The bottom-left pane shows the console output:

```
~/Desktop/
m3G/downloaded_packages
Update old packages: 'cluster', 'formatR', 'GenomicAlignments',
'GenomicFeatures',
'ggplot2', 'markdown', 'mime', 'phangorn', 'rlang', 'rtracklaye
r', 'tibble', 'xfun',
'XML'
Update all/some/none? [a/s/n]:
n
> library("phyloseq")
Attachement du package : 'phyloseq'
```

The right pane shows the Environment/History/Connections pane. The 'Packages' tab is active, displaying a list of installed and available packages:

Name	Description	Version
<input type="checkbox"/> codetools	Code Analysis Tools for R	0.2-16
<input type="checkbox"/> colorspace	A Toolbox for Manipulating and Assessing Colors and Palettes	1.4-1
<input type="checkbox"/> compiler	The R Compiler Package	3.6.0
<input type="checkbox"/> crayon	Colored Terminal Output	1.3.4
<input type="checkbox"/> curl	A Modern and Flexible Web Client for R	3.3
<input type="checkbox"/> dada2	Accurate, high-resolution sample inference from amplicon sequencing data	1.12.1
<input type="checkbox"/> data.table	Extension of 'data.frame'	1.12.2
<input checked="" type="checkbox"/> datasets	The R Datasets Package	3.6.0
<input type="checkbox"/> DBI	R Database Interface	1.0.0
<input checked="" type="checkbox"/> DECIPHER	Tools for curating, analyzing, and manipulating biological sequences	2.12.0
<input type="checkbox"/> DelayedArray	A unified framework for working transparently with on-disk and	0.10.0

Check install packages :Bioconductor package

Command line is mandatory

```
19 library("ggplot2")
20 library("gridExtra")
21 library("DECIPHER")
22 library("phangorn")
23 library("vegan")
24 setwd("~/Formation2019TUN/")
25
26 path=~"/Formation2019TUN/DATA_GAB"
20:21 (Untitled) R Script
```

Console Terminal Jobs

```
~/Desktop/ ↵
>
>
> BiocManager::install("dada2", version = "3.9")
Bioconductor version 3.9 (BiocManager 1.30.4), R 3.6.0 (2019-04-26)
Installing package(s) 'dada2'
essai de l'URL 'https://bioconductor.org/packages/3.9/bioc/bin/macosx/el-capitan/contrib/3.6/dada2_1.12.1.tgz'
Content type 'application/x-gzip' length 2576762 bytes (2.5 MB)
=====
downloaded 2.5 MB
```

Files	Plots	Packages	Help
Name	Description		
<input type="checkbox"/> fastmatch	Fast match() fun		
<input type="checkbox"/> foreach	Provides Foreac for R		
<input type="checkbox"/> foreign	Read Data Store 'SAS', 'SPSS', 'Sta 'dBase', ...		
<input type="checkbox"/> formatR	Format R Code		
<input type="checkbox"/> Formula	Extended Mode		
<input type="checkbox"/> futile.logger	A Logging Utilit		
<input type="checkbox"/> futile.options	Futile Options M		
<input type="checkbox"/> gdata	Various R Progr Data Manipulati		
<input type="checkbox"/> GenomeInf...	Utilities for ma chromosome na		

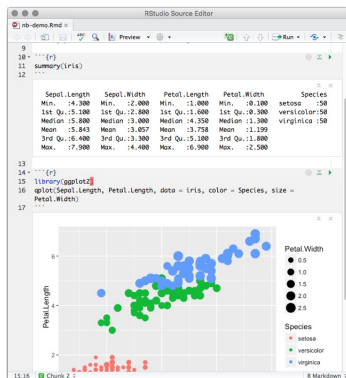
Load library to execute function

```
8 ~/Desktop/
9 installing package(s) from
essai de l'URL 'https://bioconductor.org/packages/3.9/bioc/bin/macosx/el-capitan/contrib/3.6/dada2_1.12.
10 1.tgz'
Content type 'application/x-gzip' length 2576762 bytes (2.5 MB)
=====
11 downloaded 2.5 MB
12
13 The downloaded binary packages are in
   /var/folders/j0/8vvyнк7d3ws46jrp4z4d005r0000gn/T//Rtmp7mum3G/downloaded_packages
14 >
15 >
16 > library("dada2")
Le chargement a nécessité le package : Rcpp
> '|
```

Diapositive 11 sur 16 Français (France) Notes Co

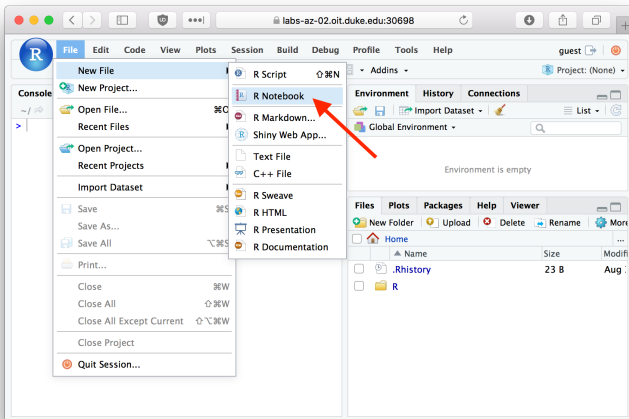
Or use command as : `package::function()`

R Notebook



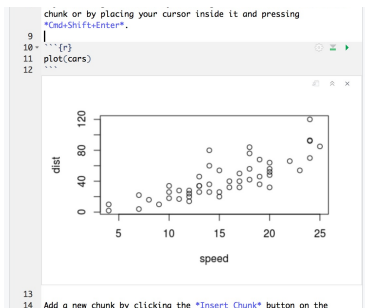
An R Notebook, is a document with chunks that can be executed independently and interactively, with output visible immediately beneath the input

R Notebook



Creating a Notebook instead of the classical script

- ▶ You can create a new notebook in RStudio with the menu command `File` → `New File` → `R Notebook`.
- ▶ Notebook chunks can be inserted quickly using the keyboard shortcut `Ctrl + Alt + I` (macOS: `Cmd + Option + I`), or via the `Insert` menu in the editor toolbar.
- ▶ Between chunks, you can comment all your results.



R Notebook

The screenshot displays the RStudio interface with an R Notebook. The left pane shows the source code, and the right pane shows the rendered notebook content.

```
63  
64 - ## Residuals  
65  
66 To motivate the use of models we're going to start with an  
67 interesting pattern from the NYC flights dataset -- the  
68 number of flights per day.  
69 ```{r}  
70 library(nycflights13)  
71 library(lubridate)  
72 library(dplyr)  
73  
74 daily <- flights %>%  
75   mutate(date = make_datetime(year, month, day)) %>%  
76   group_by(date) %>%  
77   summarise(n = n())  
78  
79 ggplot(daily, aes(date, n)) +  
80   geom_line()  
81  
82
```

The rendered notebook content on the right includes the same code and a plot of the residuals. The plot shows a highly volatile time series of flight counts from December 2012 to December 2013, with a clear daily cycle and a strong day-of-week effect.

Residuals

To motivate the use of models we're going to start with an interesting pattern from the NYC flights dataset – the number of flights per day.

```
library(nycflights13)  
library(lubridate)  
library(dplyr)  
daily <- flights %>%  
  mutate(date = make_datetime(year, month, day)) %>%  
  group_by(date) %>%  
  summarise(n = n())  
ggplot(daily, aes(date, n)) +  
  geom_line()
```

Understand this pattern is challenging because there's a very strong day-of-week effect that dominates the subtler patterns:

A demo before practice...