R workshop

Aurélien Ginolhac

2nd June 2016

R workshop

Day 1 - beginner

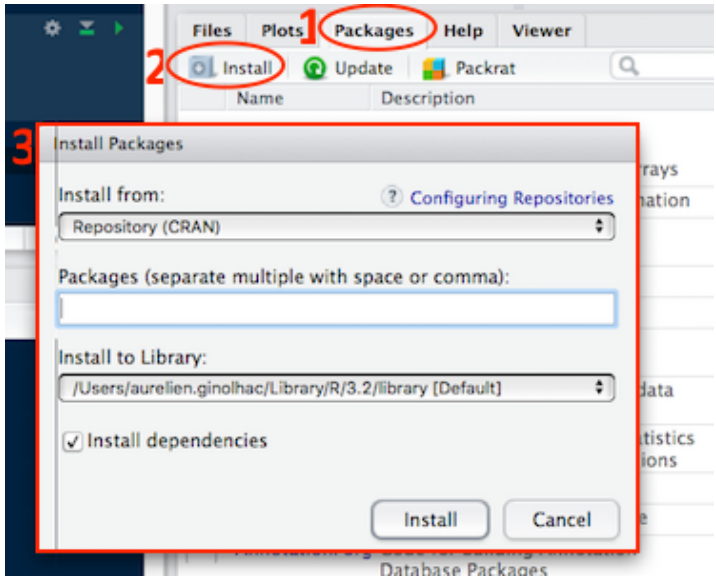
Why learn R?

- Free!
- Packages
- Community

Packages

as easy as `apt` and yes frequent updates

- CRAN, reliable many checks when submitting



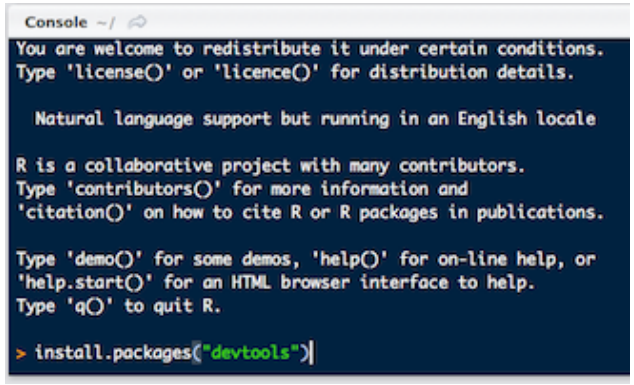
- GitHub using devtools. Check status

```
# install.packages("devtools")
```

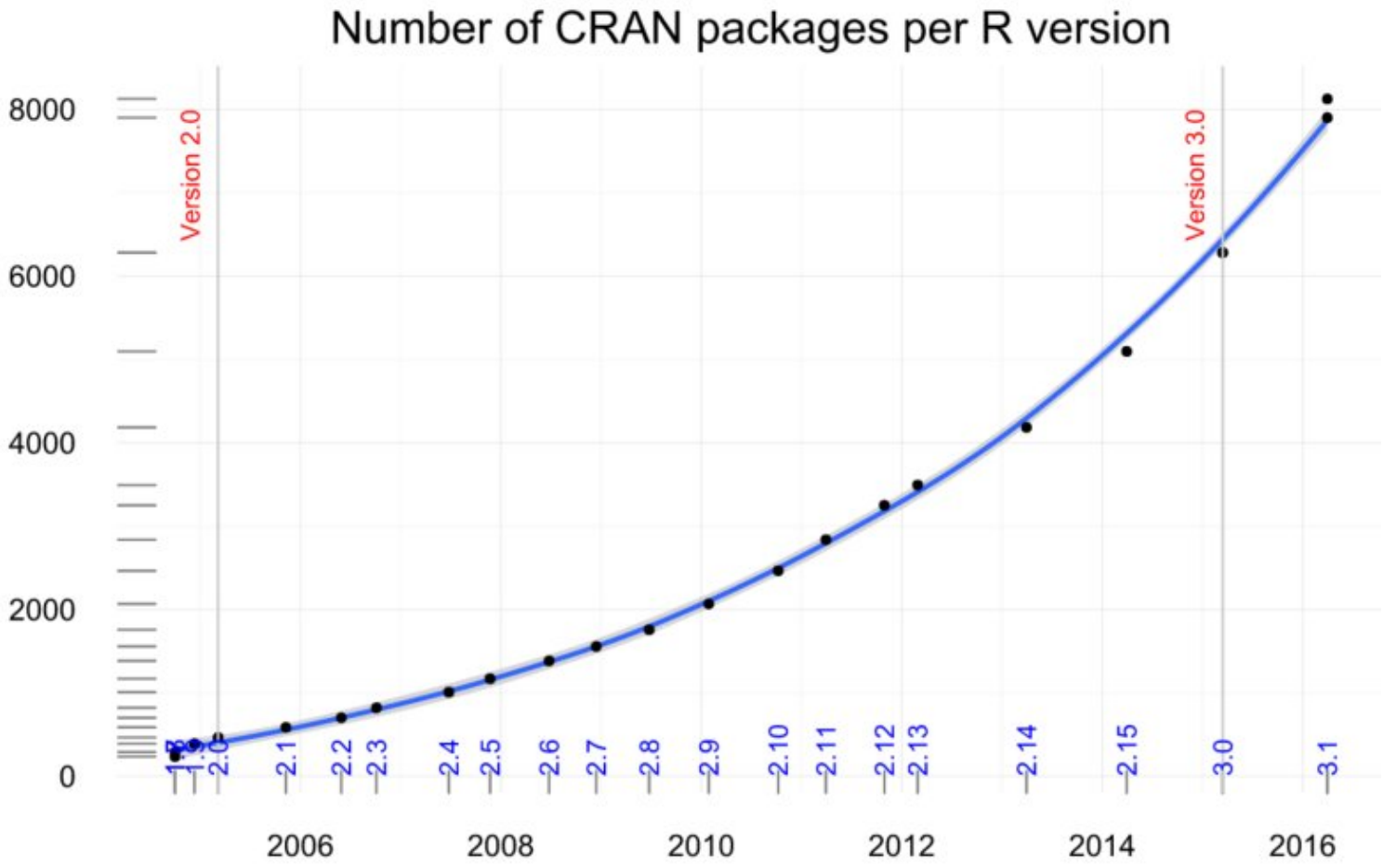
```
devtools::install_github("hadley/readr")
```

- bioconductor. Check status

```
source("https://bioconductor.org/biocLite.R")
biocLite("limma")
```



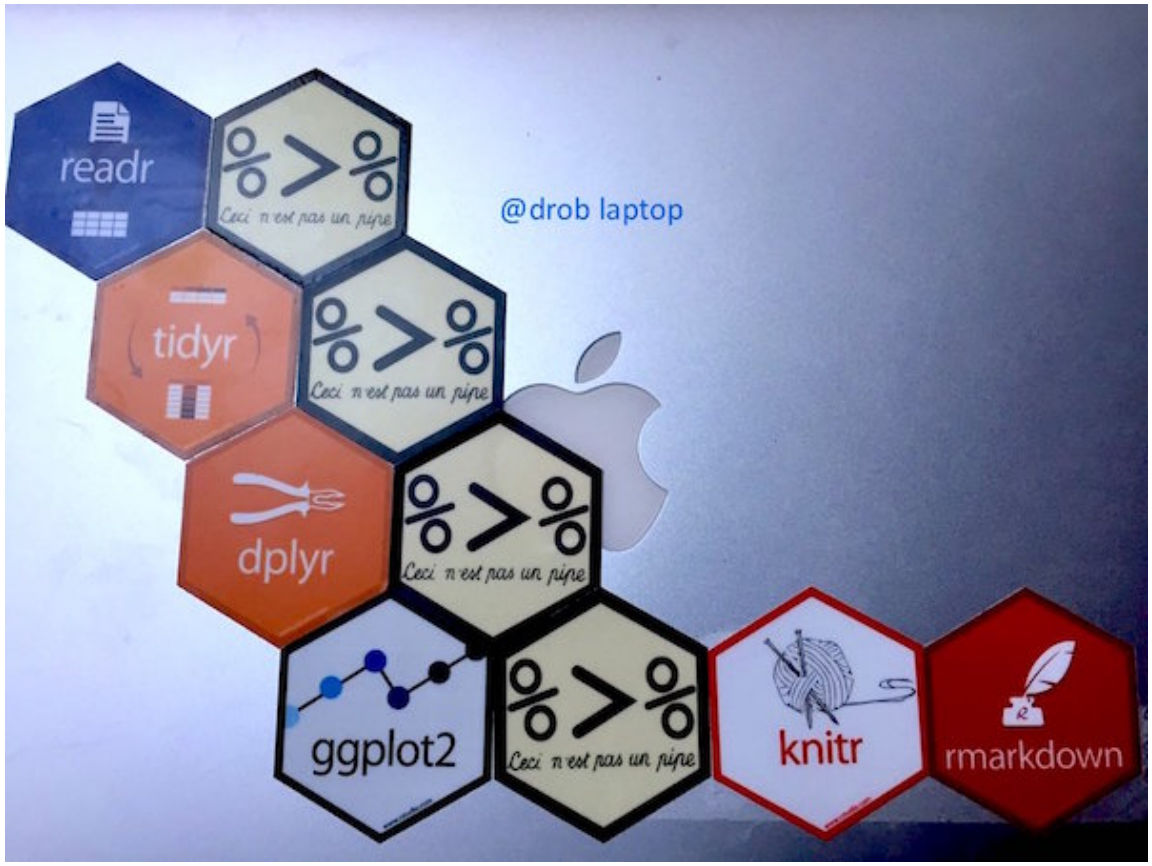
More and more packages



source: Andrie de Vries

Pipeline goal

David Robinson summarized the goal on his laptop



Period of much suckiness

Hadley Wickham's "dplyr" tutorial at useR 2014 (1/2)

Period of much suckiness

> Whenever you're learning a new tool, for a long time you're going to suck…
> But the good news is that it's typical, that's something that happens to everyone, and it's only temporary.
> – Hadley Wickham

R data structures

`vector`, `lists`, `data.frame`, `matrix`, `array` etc

Lists are objects that could contain anything

```
list(a = 1:3, b = c("hello", "bye"), data = head(iris, 2))
```

```
## $a
## [1] 1 2 3
##
## $b
## [1] "hello" "bye"
##
## $data
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
```

Focus on `data.frame` which are `lists` where all columns have **equal** length

```
class(iris)
```

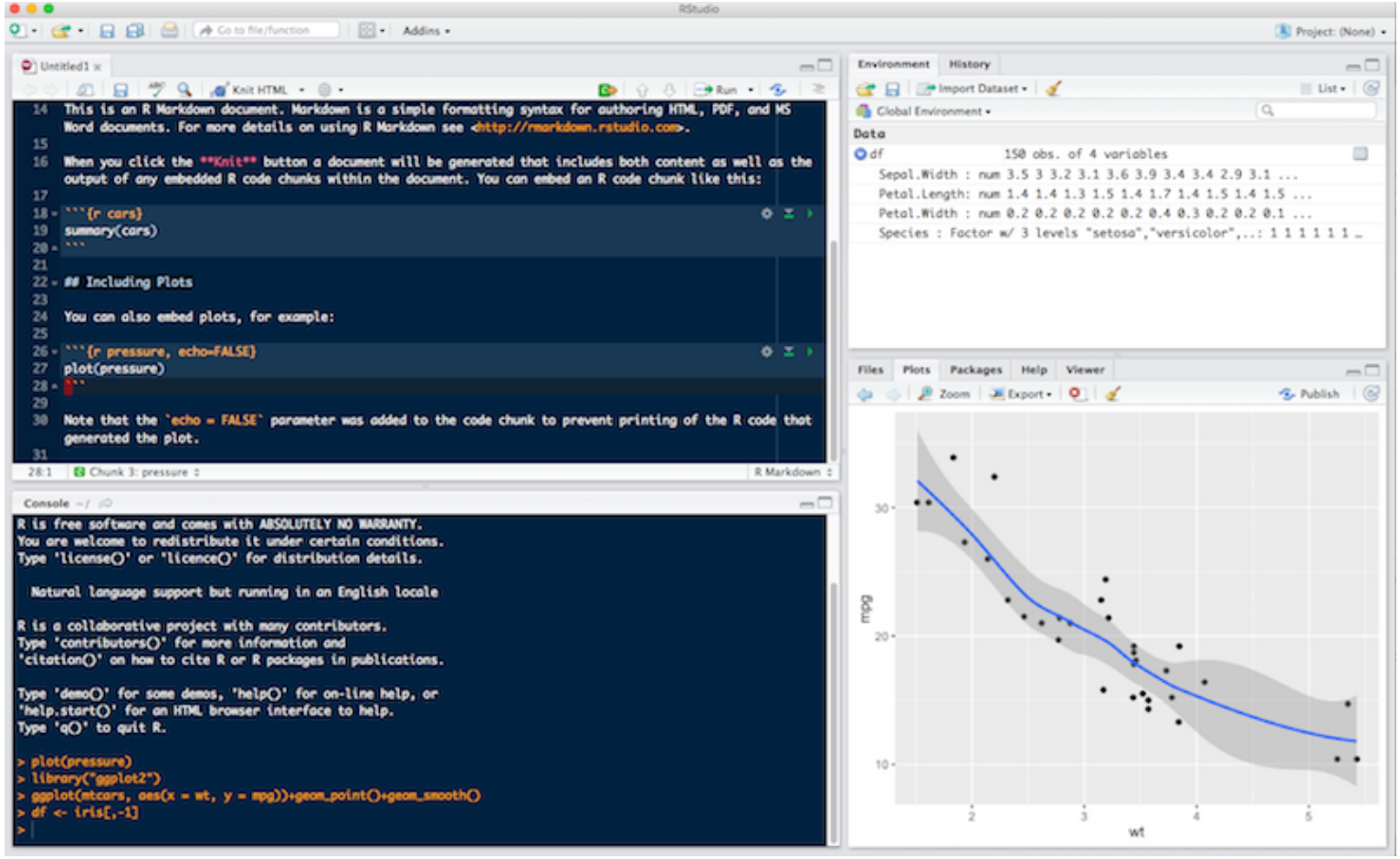
```
## [1] "data.frame"
```

```
class(unclass(iris))
```

Rstudio

Integrated Development Editor

Layout, 4 panels



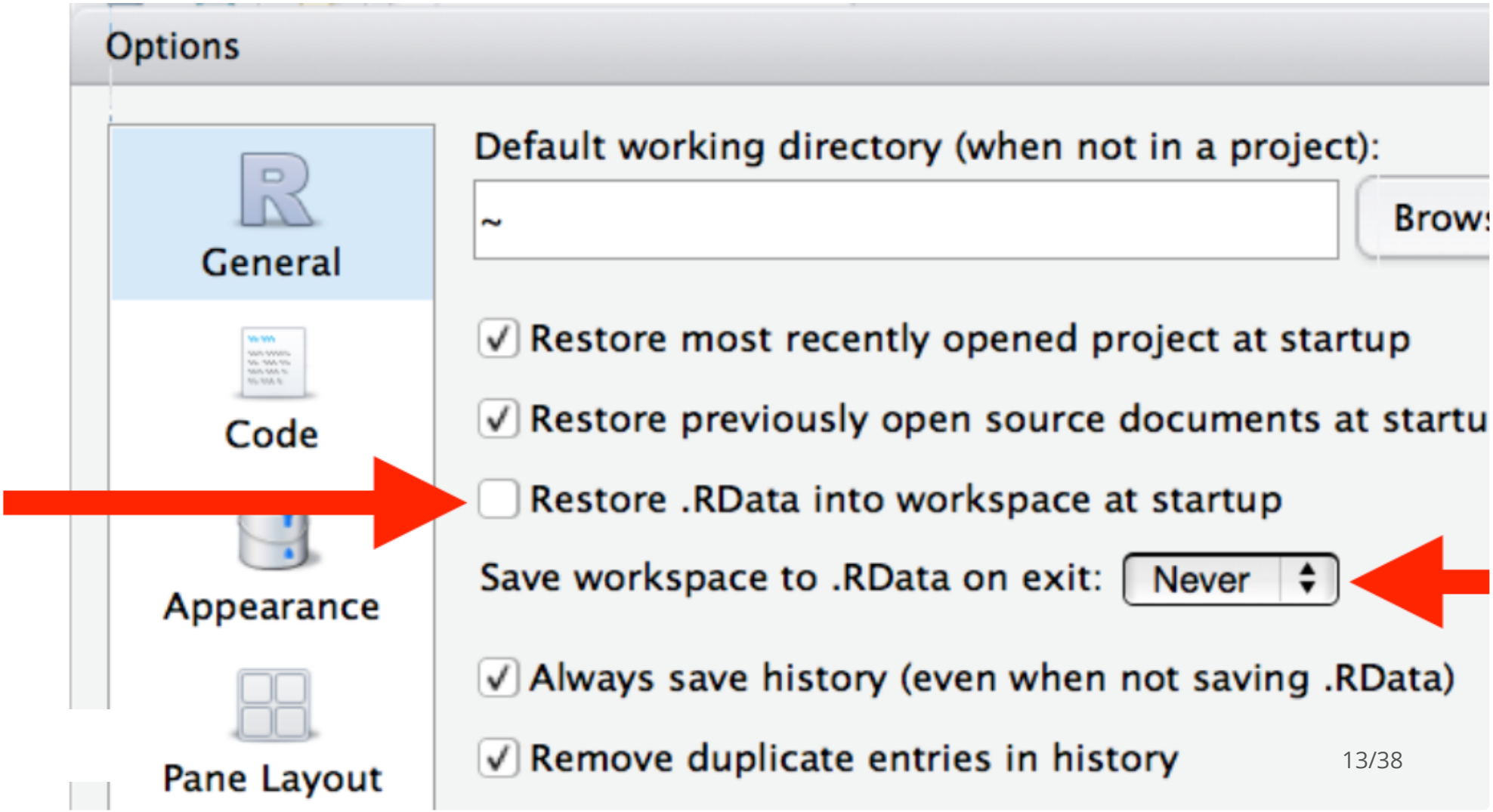
Features

- Package management (also for building)
- Console to run R, with syntax highligther
- Editor to work with scripts / markdown
- autocompletion using TAB
- Cheatsheets
- Keyboard shortcuts

Cmd + Enter: sends line or selection from the editor to the console and runs it. (Ctrl + Enter on a PC) ↑: in the console browse previous commands

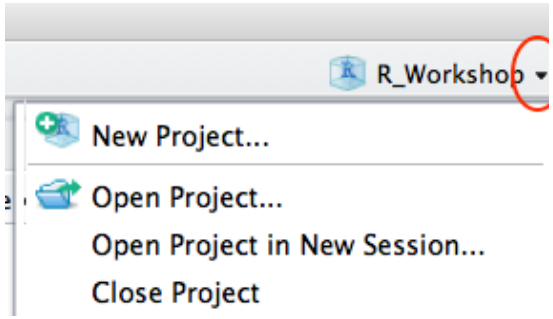
Update options

Recommended in the r4ds To get a clean environment at start-up



Projects

Solve most issues with working directory, get rid of `setwd()`



Chaining

The pipe operator %>%

magrittr by Stefan Milton Bache

Compare

```
set.seed(124)
x <- rnorm(10)
mean(x)
```

```
## [1] 0.2147669
```

```
round(mean(x), 3)
```

```
## [1] 0.215
```

with

```
set.seed(124)
rnorm(10) %>% mean %>% round(3)
```

```
## [1] 0.215
```

natural from left to right.

Even better with **one** instruction per line and **indentation**

```
set.seed(124)
rnorm(10) %>%
  mean %>%
  round(3)
```

```
## [1] 0.215
```

Easier to read

Tidying data

tidyr

Definitions

- **Variable**: A quantity, quality, or property that you can measure.
- **Observation**: A set of values that display the relationship between variables.
- To be an observation, values need to be measured under similar conditions, usually measured on the same observational unit at the same time.
- **Value**: The state of a variable that you observe when you measure it.

source: Garret Grolemund

Rules

1. Each variable is in its own column
2. Each observation is in its own row
3. Each value is in its own cell

Long / wide format

The wide format is generally untidy but found in the majority of datasets

`tidyr` convert from one to another

Wide > Long

Demo with iris

```
head(iris, 3)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
```

gather

```
library("tidyr")
iris_melt <- iris %>%
  tibble::rownames_to_column() %>%
  dplyr::tbl_df() %>%
  gather(flower, measure, contains("al"))
iris_melt
```

```
## Source: local data frame [600 x 4]
##
##    rowname Species       flower measure
##      <chr>  <fctr>        <chr>   <dbl>
## 1        1  setosa Sepal.Length     5.1
## 2        2  setosa Sepal.Length     4.9
## 3        3  setosa Sepal.Length     4.7
## 4        4  setosa Sepal.Length     4.6
## 5        5  setosa Sepal.Length     5.0
## 6        6  setosa Sepal.Length     5.4
```

spread

```
iris_melt %>%
  spread(flower, measure)
```

```
## Source: local data frame [150 x 6]
##
##    rowname    Species Petal.Length Petal.Width Sepal.Length Sepal.Width
##      <chr>     <fctr>        <dbl>       <dbl>        <dbl>       <dbl>
## 1        1     setosa          1.4         0.2          5.1         3.5
## 2       10     setosa          1.5         0.1          4.9         3.1
## 3      100 versicolor          4.1         1.3          5.7         2.8
## 4      101  virginica          6.0         2.5          6.3         3.3
## 5      102  virginica          5.1         1.9          5.8         2.7
## 6      103  virginica          5.9         2.1          7.1         3.0
## 7      104  virginica          5.6         1.8          6.3         2.9
## 8      105  virginica          5.8         2.2          6.5         3.0
## 9      106  virginica          6.6         2.1          7.6         3.0
## 10     107  virginica          4.5         1.7          4.9         2.5
## ..     ...        ...          ...         ...          ...         ...
```

Separate / Unite

unite

```
df %>%
  unite(date, c(year, month, day), sep = "-") -> df_unite
```

separate, use **quotes** since not refering to objects

```
df_unite %>%
  separate(date, c("year", "month", "day"))
```

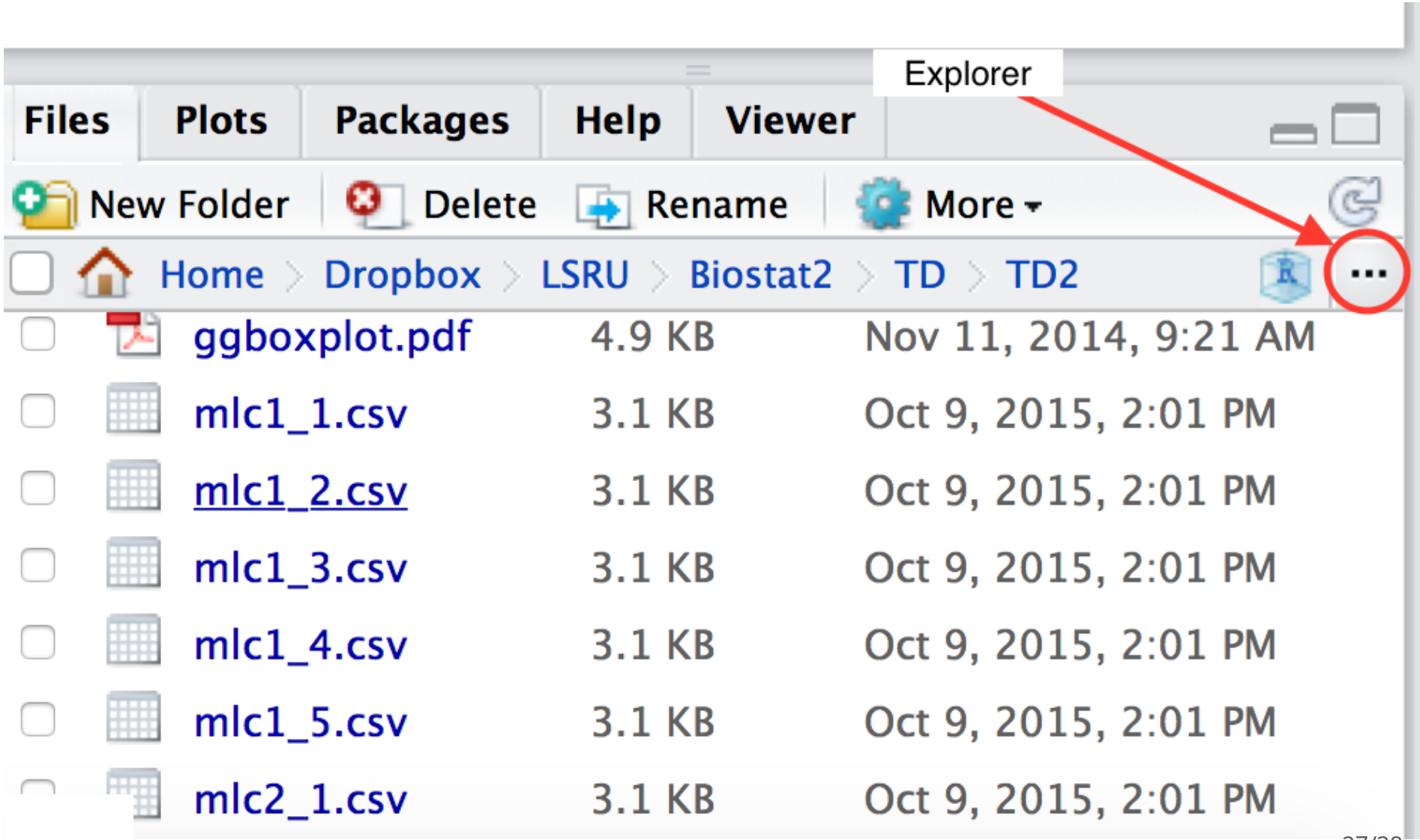
```
## Source: local data frame [3 x 4]
##
##    year month   day value
##   <chr> <chr> <chr> <chr>
## 1  2015    11    23  high
## 2  2014     2     1   low
## 3  2014     4    30   low
```

Reading data

readr

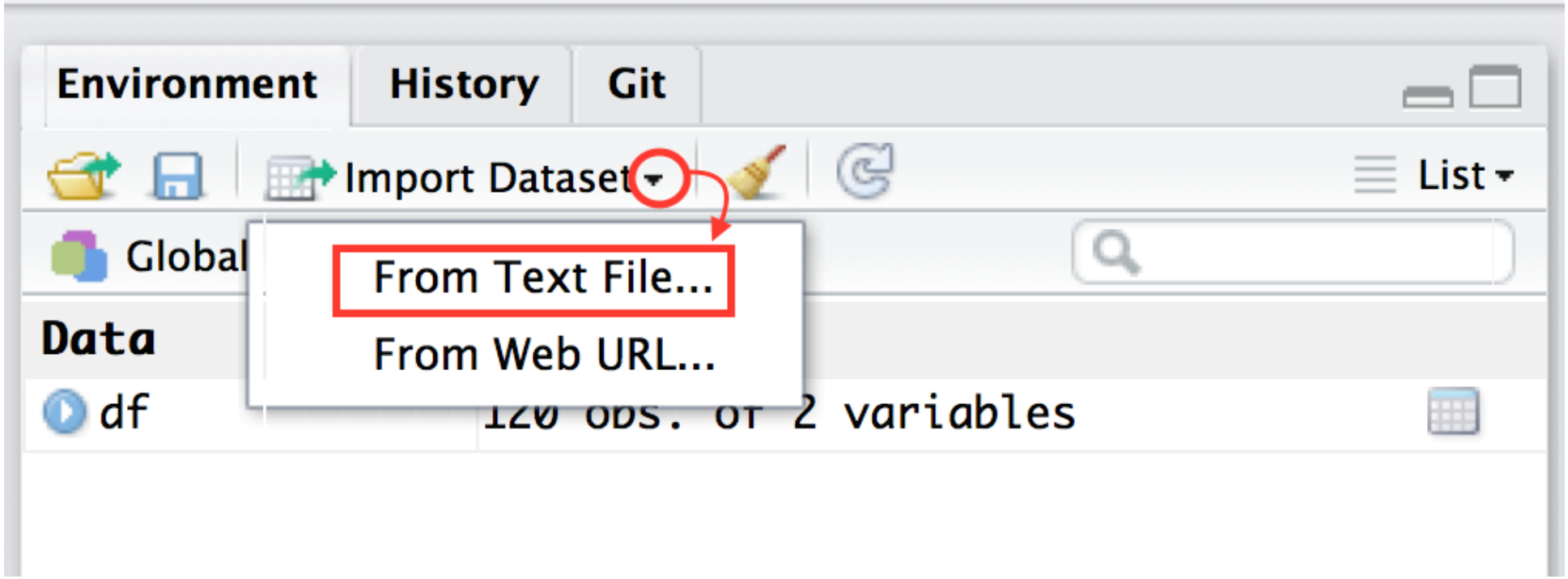
Use rstudio project

Using `Rstudio`, right bottom panel. Select the **folder**



import file

Using `Rstudio`, right top panel. Select directly your **file**



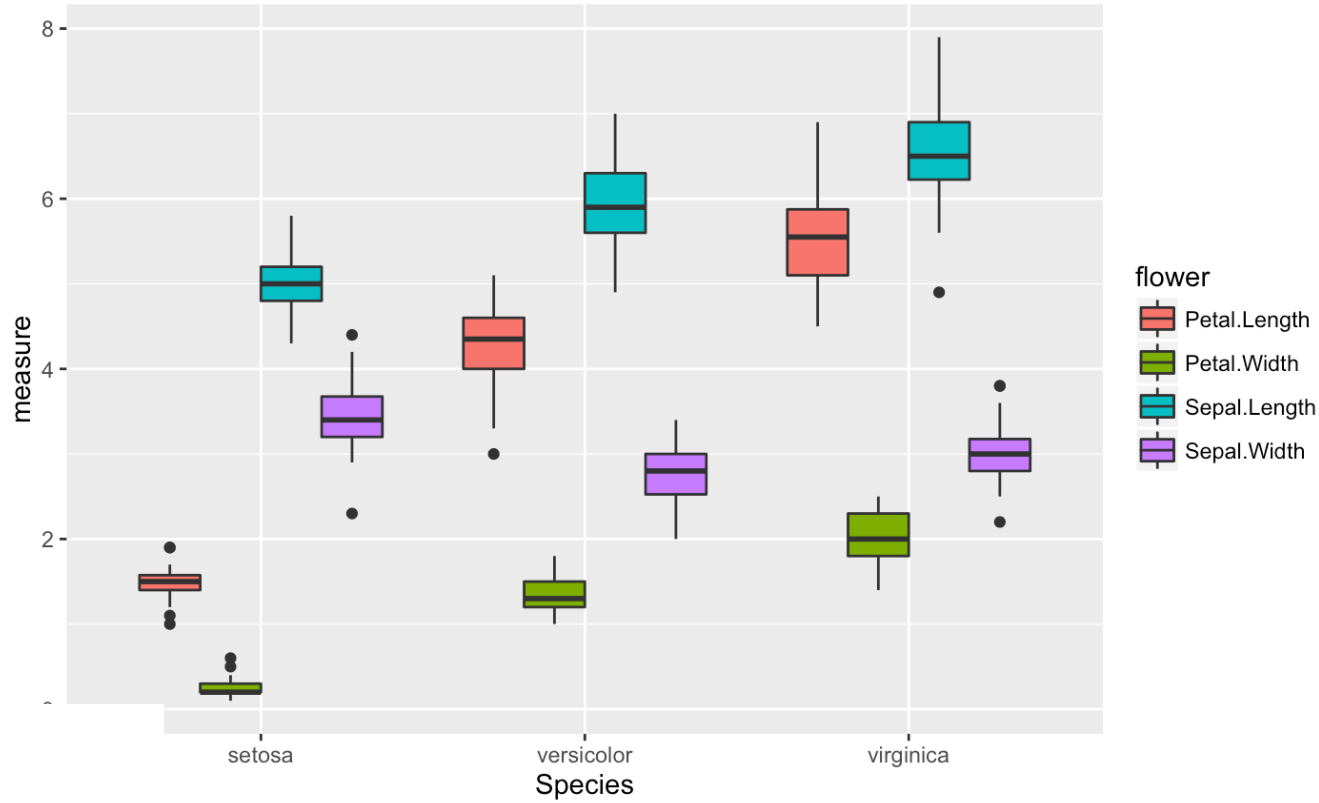
overview

Plotting

ggplot2

Why tidy is useful?

```
library("tidyr")
library("ggplot2")
iris %>%
  gather(flower, measure, 1:4) %>%
  ggplot()+
  geom_boxplot(aes(x = Species, y = measure, fill = flower))
```



Scatterplots

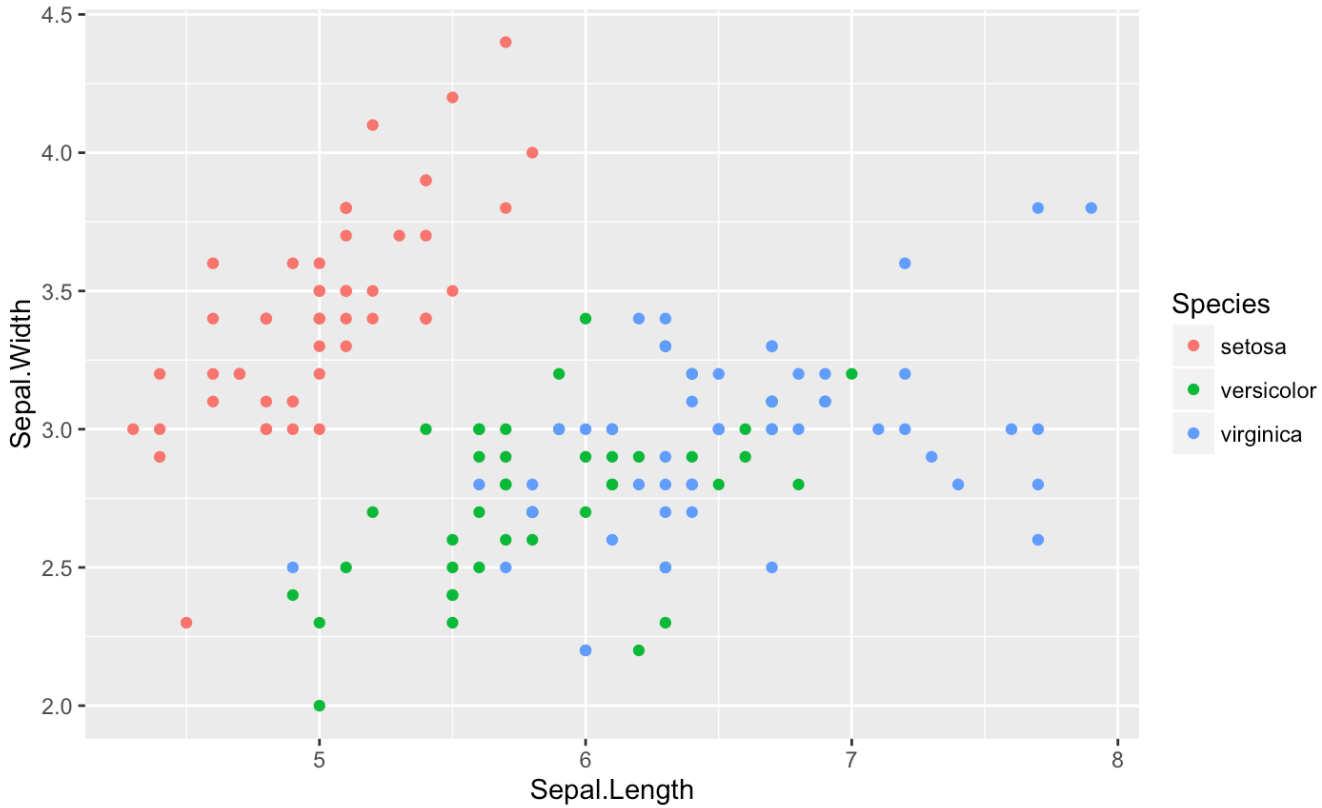
```
iris %>%
  ggplot(aes(x = Sepal.Length, y = Sepal.Width, colour = Species))+
  geom_point()+
  geom_smooth(method = "lm", se = FALSE)+
  xlab("Length")+
  ylab("Width")+
  ggtitle("Sepal")
```

More aesthetics

```
iris %>%
  ggplot(aes(x = Sepal.Length, y = Sepal.Width,
             size = Petal.Length / Petal.Width,
             colour = Species))+
  geom_point()+
  scale_size_area("Petal ratio Length / Width")+
  #scale_colour_brewer(palette = 1, type = "qual")+
  scale_colour_manual(values = c("blue", "red", "orange"))+
  xlab("Sepal.Length")+
  ylab("Sepal.Width")
```

in / out aesthetics

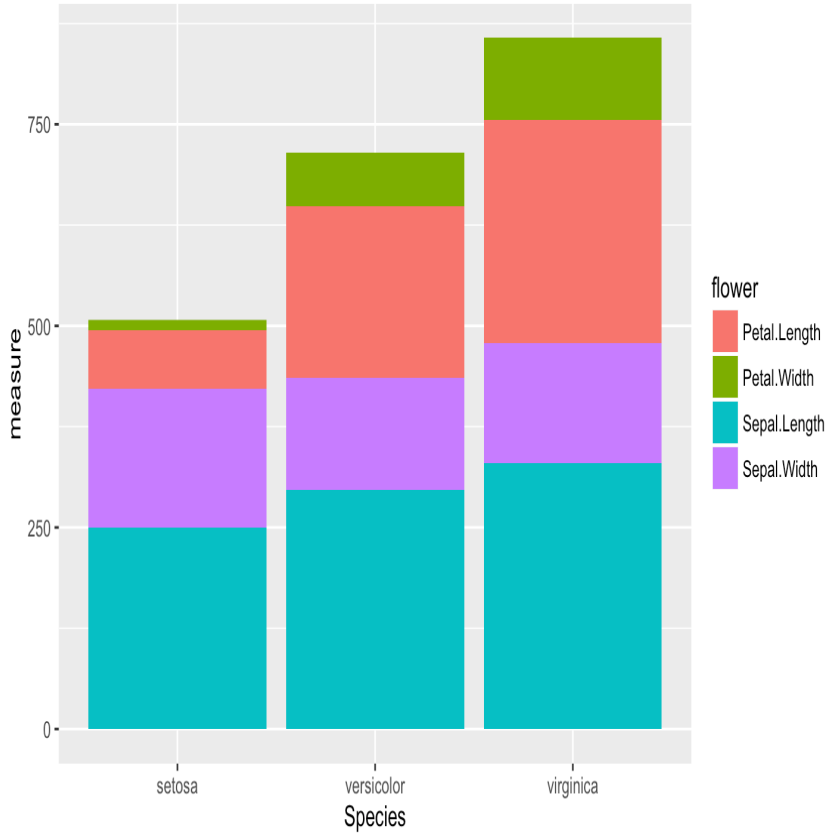
```
iris %>%
  ggplot(aes(x = Sepal.Length, y = Sepal.Width))+
  geom_point(aes(colour = Species))
```




```
iris %>%
```

Barplots

```
iris_melt %>%
  ggplot()+
  geom_bar(aes(x = Species, y = measure, fill = flower), stat = "identity")
```



Density and faceting

transparency using the `alpha` parameter

```
iris_melt %>%
  ggplot()+
  geom_density(aes(x = measure, fill = Species, colour = Species), alpha = 0.6)+
  facet_wrap(~ flower, scale = "free")+
  theme_bw()
```

facetting

transparency using the `alpha` parameter

```
iris_melt %>%
  ggplot()+
  geom_density(aes(x = measure, fill = Species, colour = Species), alpha = 0.6)+
  facet_grid(Species ~ flower, scale = "free")+
  theme_bw()
```

Recommended reading

- R for data science by Hadley & Garrett
- reading data
- tidy data
- plotting
- ggplot2 layer by layer by Hadley
- Excellent ressource about R (in French) Ewen Gallic by Ewen Gallic

Acknowledgments

Hadley Wickham Garrett Grolemund Jenny Bryan Ewen Gallic Simon David Robinson