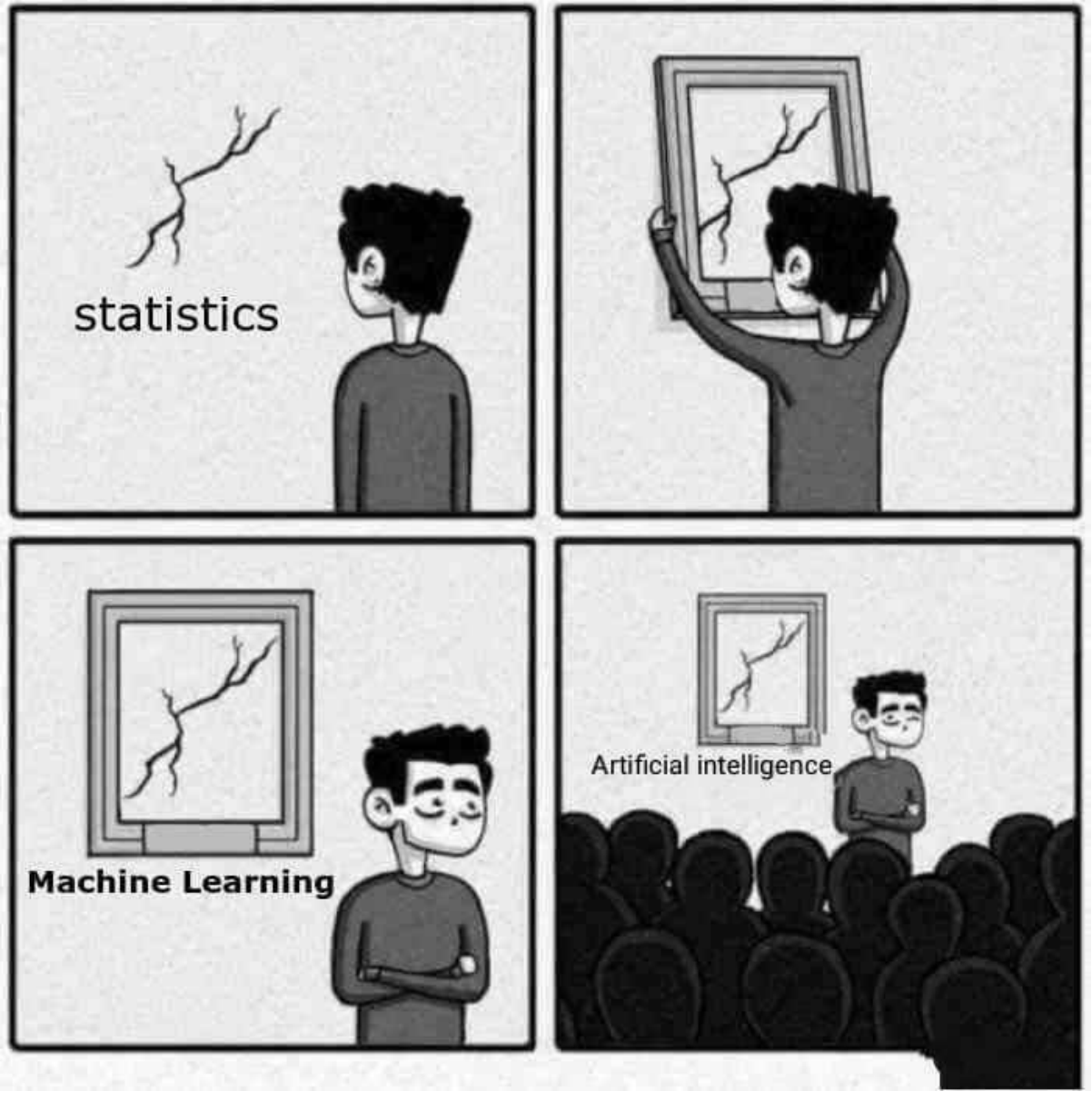
Introduction à la data science avec R

et au concept de “tidy data”

Ahmadou Dicko, PhD

SatRDay Abidjan 2020


statistics
Machine Learning
Artificial intelligence

Ce que nous allons voir ensemble

- C'est quoi R ?
- Bien démarrer avec R
- Notion de tidy data et mise en pratique

C'est quoi R ?

C'est quoi R ?

- R est un langage de programmation interpreté
- Quelques dates importantes :
  - 1990 : Ross Ihaka et Robert Gentleman développent R
  - 1996 : Le projet devient open source
  - 2000 : La version 1.0 de R voit le jour
  - 2020 : R 4.0 sortira normalement vers Avril et il y a environs 15000 packages (add-ons)

C'est quoi R ?

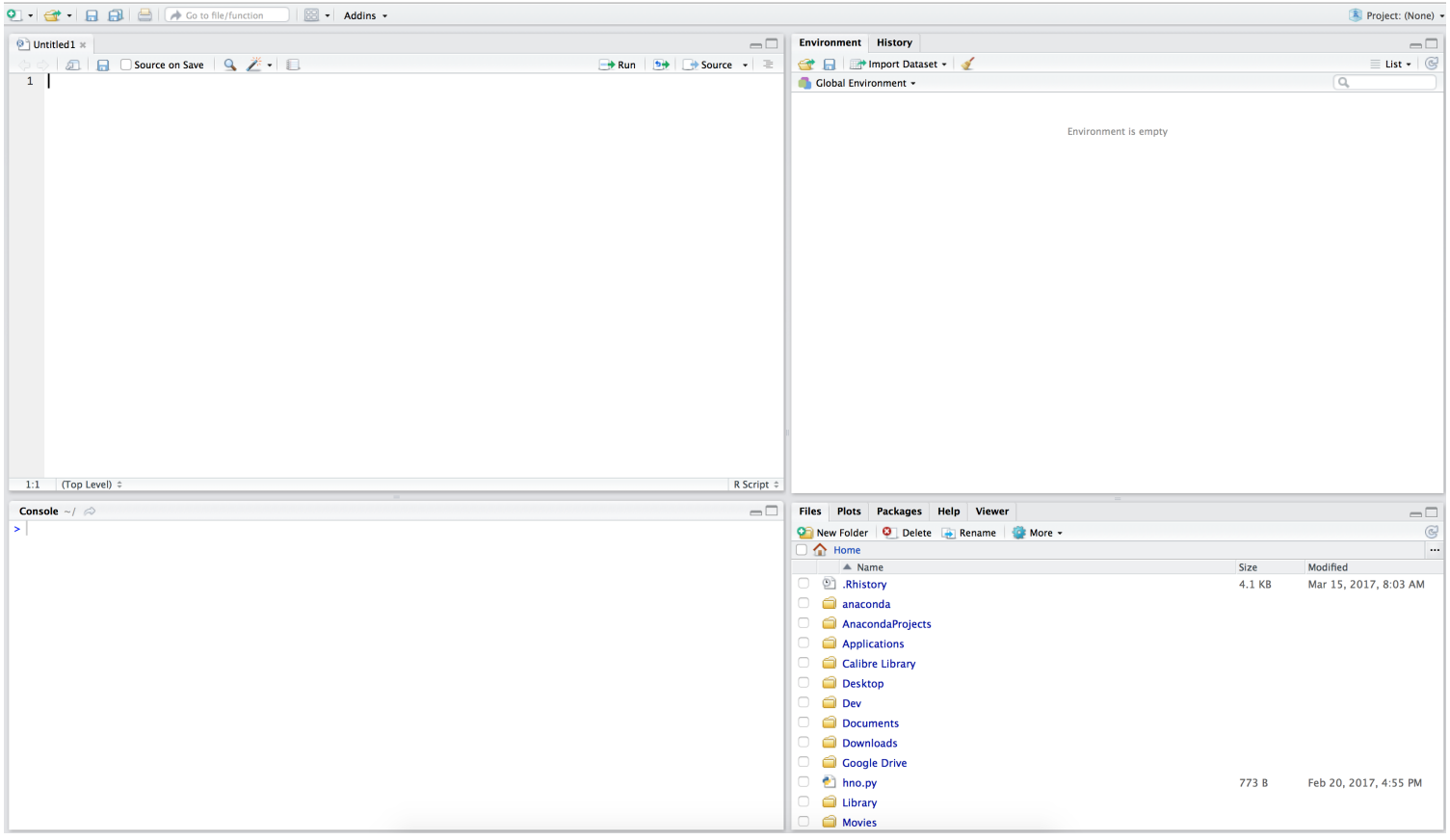
- R est gratuit et open source
- R il existe une grande communauté autour
- R a de bon outils pour la manipulation des données
- Les graphiques R sont de très bonnes qualités

Une petite intro à R

Concepts importants

- Un package `R` est en ensemble de fonctionnalité généralement organisé autour d'un thème (ou pas)
- Travailler sur `R` nécessite d'écrire des lignes de commandes (pas toujours)
- `Rstudio` est un environnement de travail pour `R` (mais y en d'autres comme `ESS`)

Environnement de travail: Rstudio



R et Rstudio

R: Engine	RStudio: Dashboard
	

Base du langage : R calculatrice ?

```
log(10)
```

```
## [1] 2.302585
```

```
cos(pi/3)
```

```
## [1] 0.5
```

```
 a <- 25
 a + 1
```

```
## [1] 26
```

```
 a * 3 + 5
```

```
## [1] 80
```

Base du langage : vecteur

```
c(1, 2, 10)
```

```
## [1]  1  2 10
```

```
c("Ali", "Modou", "Marie")
```

```
## [1] "Ali"   "Modou" "Marie"
```

Base du langage : matrice

```
(m <- matrix(c(1, 2, 10, 9), nrow = 2))
```

```
##      [,1] [,2]
## [1,]    1   10
## [2,]    2    9
```

```
m[1, 2]
```

```
## [1] 10
```

```
m[, 2]
```

```
## [1] 10  9
```

Base du langage : Les types

```
typeof("Open Data")
```

```
## [1] "character"
```

```
typeof(TRUE)
```

```
## [1] "logical"
```

```
typeof(25)
```

```
## [1] "double"
```

```
x <- c("Ali", 20)
typeof(x)
```

```
## [1] "character"
```

Base du langage : list

```
c("Ali", 10, "Marie")
```

```
## [1] "Ali"   "10"    "Marie"
```

```
list("Ali", 10, "Marie")
```

```
## [[1]]
## [1] "Ali"
##
## [[2]]
## [1] 10
##
## [[3]]
## [1] "Marie"
```

Base du langage: data.frame

```
(df <- data.frame(
  nom = c("Ali", "Modou", "Marie"),
  taille = c(170, 185, 165))
)
```

```
##     nom taille
## 1   Ali    170
## 2 Modou    185
## 3 Marie    165
```

Base du langage: data.frame

```
library(dplyr)
select(df, nom)
```

```
##     nom
## 1   Ali
## 2 Modou
## 3 Marie
```

```
filter(df, taille > 165)
```

```
##     nom taille
## 1   Ali    170
## 2 Modou    185
```

Base du langage: fonction

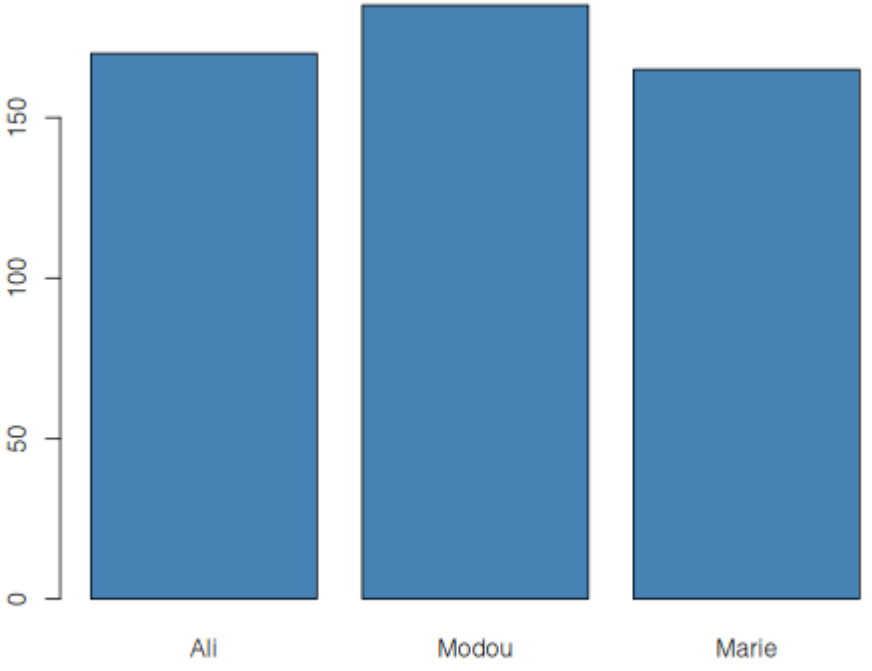
```
carré <- function(x) x^2
```

```
carré(c(1, 10, -5))
```

```
## [1]   1 100  25
```

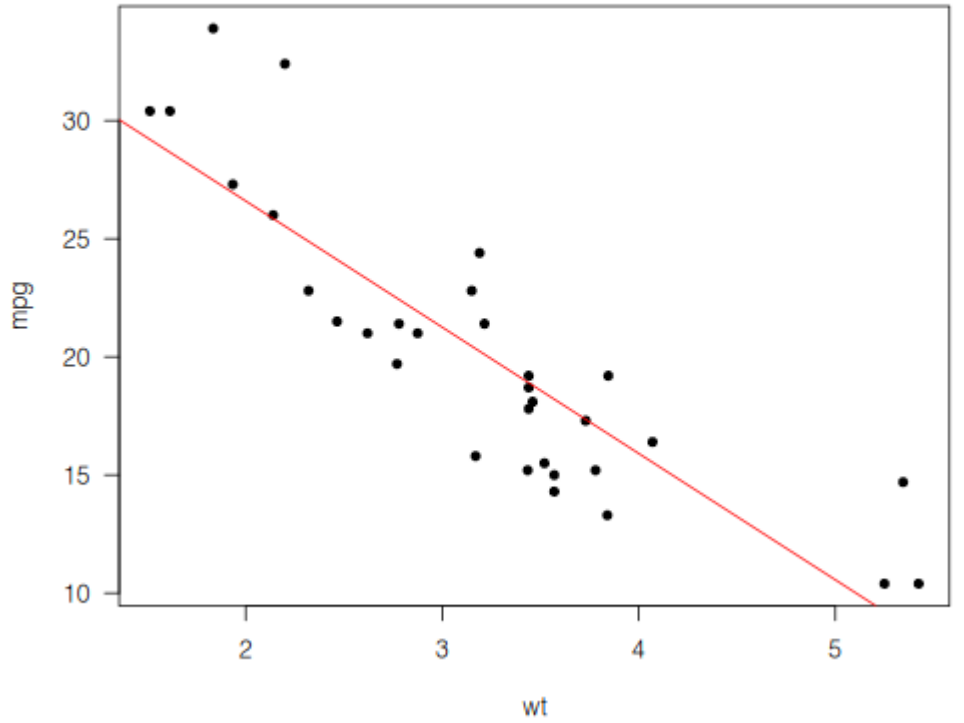
Base du langage: graphique

```
barplot(df$taille, names.arg = df$nom, col = "steelblue")
```



Base du langage: graphique

```
plot(mpg ~ wt, data = mtcars, pch = 19, las = 1, cex = 0.8)
abline(lm(mpg ~ wt, data = mtcars), col = "red")
```



Base du langage: graphique

```
library(ggplot2)
ggplot(df, aes(nom, taille)) +
  geom_col(fill = "steelblue")
```

Base du langage: graphique

```
ggplot(mtcars, aes(wt, mpg)) +
  geom_point(shape = 19) +
  geom_smooth(method = "lm", se = FALSE, colour = "red")
```

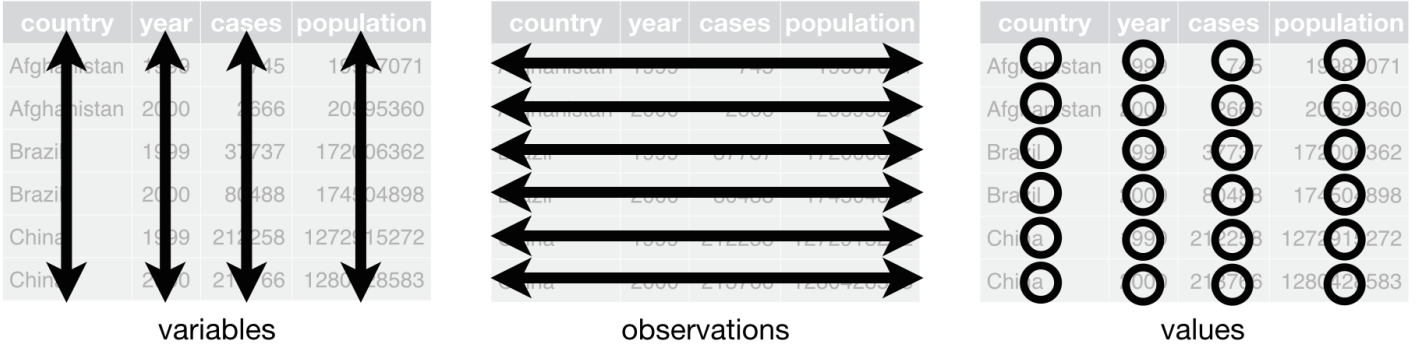
Tidy data

La notion de tidy data

> “Happy families are all alike; every unhappy family is unhappy in its own way.” – Leo Tolstoy
>
> “Tidy datasets are all alike, but every messy dataset is messy in its own way.” – Hadley Wickham

La notion de tidy data

1. À chaque **variable** sa propre **colonne**
2. À chaque **observation** sa propre **ligne**
3. À chaque **valeur** sa propre **cellule**



Tidy ou pas ?

	country	year	cases	population
1	Afghanistan	1999	745	19987071
2	Afghanistan	2000	2666	20595360
3	Brazil	1999	37737	172006362
4	Brazil	2000	80488	174504898
5	China	1999	212258	1272915272
6	China	2000	213766	1280428583

Tidy ou pas ?


country
year
cases
population
variables


country
year
cases
population
observations


country
year
cases
population
values

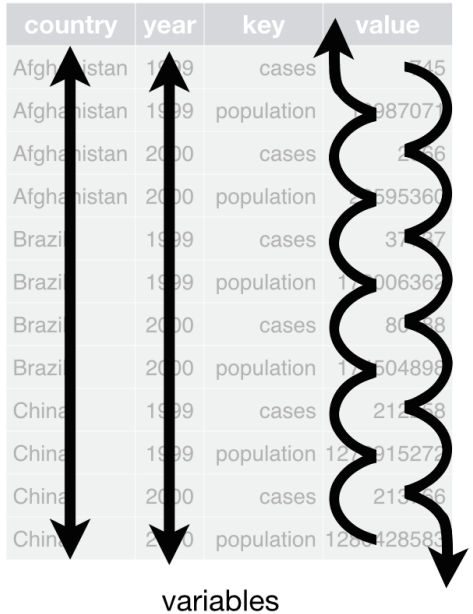
Tidy ou pas ?

	country	year	type	count
1	Afghanistan	1999	cases	745
2	Afghanistan	1999	population	19987071
3	Afghanistan	2000	cases	2666
4	Afghanistan	2000	population	20595360
5	Brazil	1999	cases	37737
6	Brazil	1999	population	172006362
7	Brazil	2000	cases	80488
8	Brazil	2000	population	174504898
9	China	1999	cases	212258
10	China	1999	population	1272915272

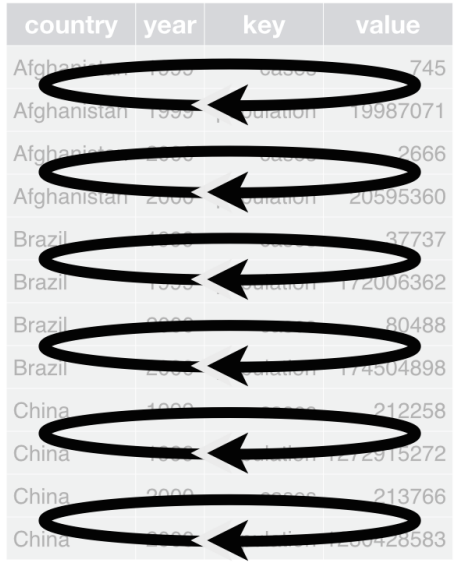
Tidy ou pas ?

country	year	key	value
Afghanistan	1999	cases	745
Afghanistan	1999	population	19987071
Afghanistan	2000	cases	2666
Afghanistan	2000	population	20595360
Brazil	1999	cases	37737
Brazil	1999	population	172006362
Brazil	2000	cases	80488
Brazil	2000	population	174504898
China	1999	cases	212258
China	1999	population	1272915272
China	2000	cases	213766
China	2000	population	1280428583

table2



variables



observations

Tidy ou pas ?

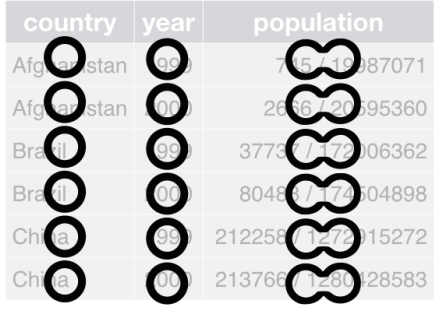
	country	year	rate
1	Afghanistan	1999	745/19987071
2	Afghanistan	2000	2666/20595360
3	Brazil	1999	37737/172006362
4	Brazil	2000	80488/174504898
5	China	1999	212258/1272915272
6	China	2000	213766/1280428583

Tidy ou pas ?

country	year	population
Afghanistan	1999	745 / 19987071
Afghanistan	2000	2666 / 20595360
Brazil	1999	37737 / 172006362
Brazil	2000	80488 / 174504898
China	1999	212258 / 1272915272
China	2000	213766 / 1280428583

table3

variables



values

Notion de tidy data

Les tidy data permettent de:

- Manipuler les données (pivot, etc)
- Combiner à d'autres données (jointure)
- Visualiser les données
- Exporter les données (i.e vers une BDD Postgres)

Le tidyverse

Le tidyverse

Les packages

- `tidyr` (rendre les données tidy)
- `readr` (lecture de données tabulaires)
- `tibble` (structure de données)
- `purrr` (programmation fonctionnelle)
- `dplyr` (manipulation des données)
- `ggplot2` (graphique)

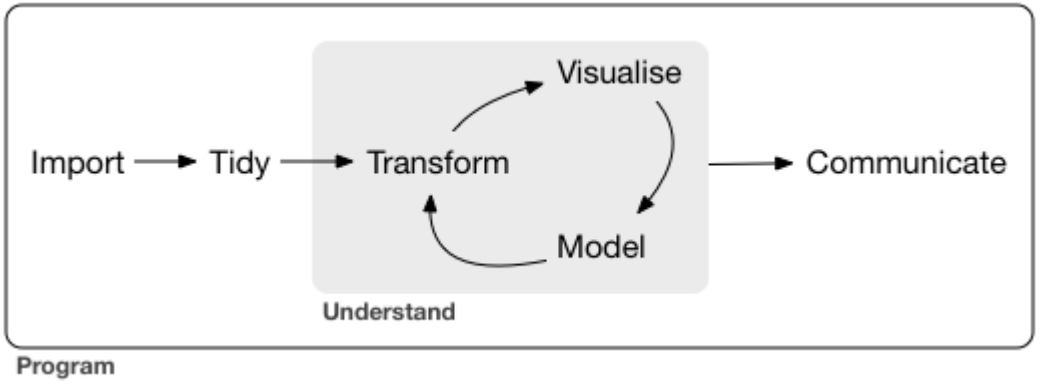
Comment les installer tous

Vous pouvez avoir tout les package de la tidyverse en executant cette commande:

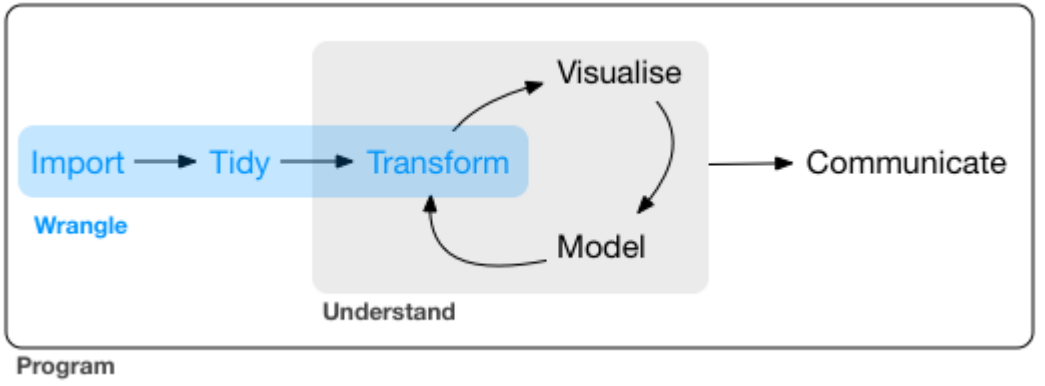
```
install.packages("tidyverse") ## télécharger le package
```

```
library(tidyverse) # Importer tidyverse
```

Le workflow d'un analyste



Lire les données



Ouvrir un fichier Excel

```
library(readxl)
```

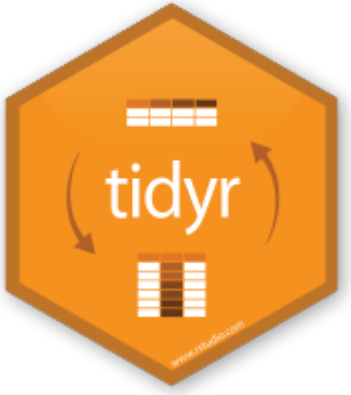
Ouvrir un fichier Excel

```
raw_tb <- read_excel(path = "data/tabular/tidy_data.xlsx", sheet = 2)
```

```
## # A tibble: 12 x 4
##    country      year key              value
##    <chr>       <dbl> <chr>            <dbl>
##  1 Afghanistan  1999 cases              745
##  2 Afghanistan  1999 population   19987071
##  3 Afghanistan  2000 cases             2666
##  4 Afghanistan  2000 population   20595360
##  5 Brazil       1999 cases            37737
##  6 Brazil       1999 population  172006362
##  7 Brazil       2000 cases            80488
##  8 Brazil       2000 population  174504898
##  9 China        1999 cases           212258
## 10 China        1999 population 1272915272
## 11 China        2000 cases           213766
## 12 China        2000 population 1280428583
```

Rendre les données tidy

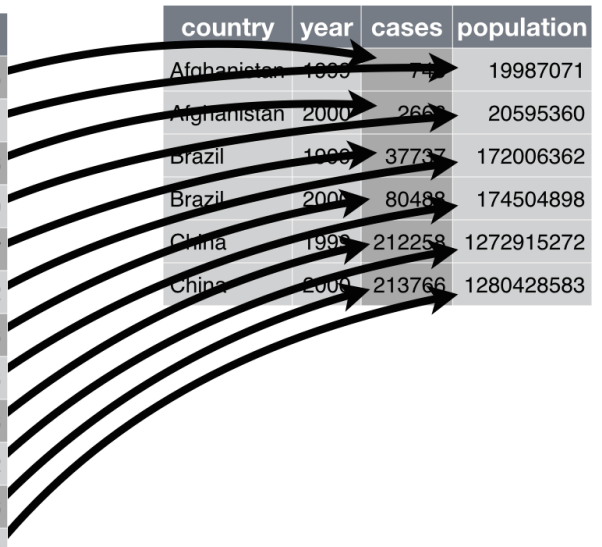
```
library(tidyr)
```



tidyr::pivot_wider

country	year	key	value
Afghanistan	1999	cases	745
Afghanistan	1999	population	19987071
Afghanistan	2000	cases	2666
Afghanistan	2000	population	20595360
Brazil	1999	cases	37737
Brazil	1999	population	172006362
Brazil	2000	cases	80488
Brazil	2000	population	174504898
China	1999	cases	212258
China	1999	population	1272915272
China	2000	cases	213766
China	2000	population	1280428583

table2



tidyr::pivot_wider

```
pivot_wider(raw_tb, names_from = "key", values_from = "value")
```

```
## # A tibble: 6 x 4
##   country      year  cases population
##   <chr>       <dbl>  <dbl>      <dbl>
## 1 Afghanistan  1999    745   19987071
## 2 Afghanistan  2000   2666   20595360
## 3 Brazil       1999  37737  172006362
## 4 Brazil       2000  80488  174504898
## 5 China        1999 212258 1272915272
## 6 China        2000 213766 1280428583
```

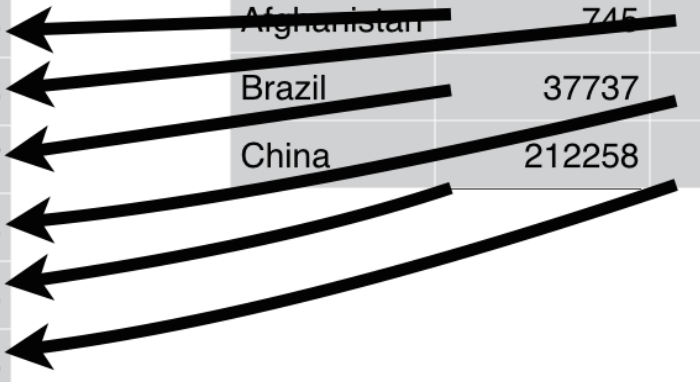
Ouvrir un fichier Excel

```
raw_tb <- read_excel(path = "data/tabular/tidy_data.xlsx", sheet = 4)
```

```
## # A tibble: 3 x 3
##   country     `1999` `2000`
##   <chr>        <dbl>  <dbl>
## 1 Afghanistan    745   2666
## 2 Brazil       37737  80488
## 3 China       212258 213766
```

tidyr::pivot_longer

country	year	cases
Afghanistan	1999	745
Afghanistan	2000	2666
Brazil	1999	37737
Brazil	2000	80488
China	1999	212258
China	2000	213766



country	1999	2000
Afghanistan	745	2666
Brazil	37737	80488
China	212258	213766

tidyr::pivot_longer

```
pivot_longer(raw_tb, cols = 2:3, names_to = "year", values_to = "case
```

```
## # A tibble: 6 x 3
##   country     year   cases
##   <chr>       <chr>  <dbl>
## 1 Afghanistan 1999     745
## 2 Afghanistan 2000    2666
## 3 Brazil      1999   37737
## 4 Brazil      2000   80488
## 5 China       1999  212258
## 6 China       2000  213766
```

Ouvrir un fichier CSV

```
library(readr)
```

Ouvrir un fichier CSV

```
raw_weather <- read_csv(file = "data/tabular/weather_tmax.csv", na =
```

```
## # A tibble: 11 x 10
##    id          year month    d1    d2    d3    d4    d5    d6    d7
##    <chr>      <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1 MX00001…  2010     1    NA    NA    NA    NA    NA    NA    NA
##  2 MX00001…  2010     2    NA   273   241    NA    NA    NA    NA
##  3 MX00001…  2010     3    NA    NA    NA    NA   321    NA    NA
##  4 MX00001…  2010     4    NA    NA    NA    NA    NA    NA    NA
##  5 MX00001…  2010     5    NA    NA    NA    NA    NA    NA    NA
##  6 MX00001…  2010     6    NA    NA    NA    NA    NA    NA    NA
##  7 MX00001…  2010     7    NA    NA   286    NA    NA    NA    NA
##  8 MX00001…  2010     8    NA    NA    NA    NA   296    NA    NA
##  9 MX00001…  2010    10    NA    NA    NA    NA   270    NA   281
## 10 MX00001…  2010    11    NA   313    NA   272   263    NA    NA
## 11 MX00001…  2010    12   299    NA    NA    NA    NA   278    NA
```

Rendre les données tidy

```
pivot_longer(raw_weather,cols = d1:d31, names_to = "day", values_to =
```

```
## # A tibble: 341 x 5
##    id           year month day    tmax
##    <chr>       <dbl> <dbl> <chr> <dbl>
##  1 MX000017004  2010     1 d1       NA
##  2 MX000017004  2010     1 d2       NA
##  3 MX000017004  2010     1 d3       NA
##  4 MX000017004  2010     1 d4       NA
##  5 MX000017004  2010     1 d5       NA
##  6 MX000017004  2010     1 d6       NA
##  7 MX000017004  2010     1 d7       NA
##  8 MX000017004  2010     1 d8       NA
##  9 MX000017004  2010     1 d9       NA
## 10 MX000017004  2010     1 d10      NA
## # … with 331 more rows
```

Autre chose que la tidyverse propose

Manipulation de données avec dplyr

Manipulation de données avec dplyr

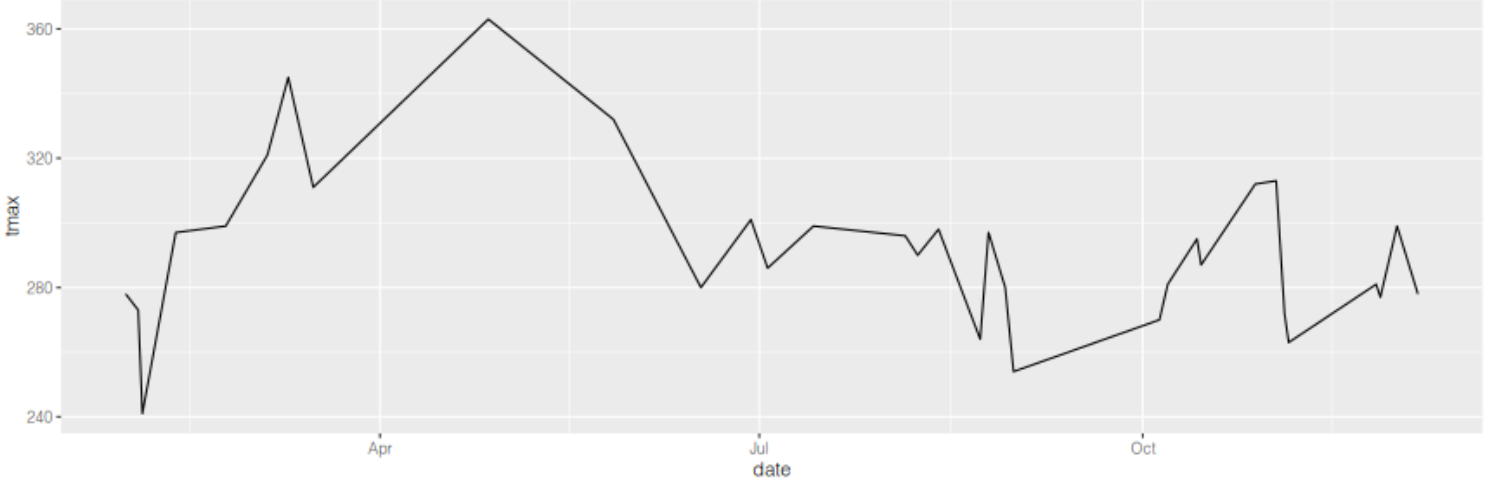
```
clean_weather %>%
  group_by(month) %>%
  summarise(tmax_moy = mean(tmax), tmax = max(tmax))
```

```
## # A tibble: 11 x 3
##    month tmax_moy  tmax
##    <dbl>    <dbl> <dbl>
##  1     1      278    278
##  2     2      278.   299
##  3     3      326.   345
##  4     4      363    363
##  5     5      332    332
##  6     6      290.   301
##  7     7      292.   299
##  8     8      283.   298
##  9    10      289    312
## 10    11      281.   313
## 11    12      288.   299
```

Graphique avec ggplot2

Graphique avec ggplot2

```
clean_weather %>%
  unite(date, year, month, day, sep = "-") %>%
  mutate(date = as.Date(date)) %>%
  ggplot(aes(date, tmax)) +
    geom_line()
```



Interaction avec SGBD

Interaction avec SGBD

```
library(DBI)
library(dbplyr)
conn <- dbConnect(RSQLite::SQLite(),  "data/db/taille.db")
```

```
dbListTables(conn)
```

```
## [1] "taille"
```

```
dbListFields(conn, "taille")
```

```
## [1] "nom"    "taille"
```

Interaction avec SGBD

```
dbGetQuery(conn, "SELECT * FROM taille")
```

```
##     nom taille
## 1   Ali    170
## 2 Modou    185
## 3 Marie    165
```

```
dbGetQuery(conn, "SELECT nom from taille WHERE taille > 165")
```

```
##     nom
## 1   Ali
## 2 Modou
```

Interaction avec SGBD

```
df <- tbl(conn, "taille")
df
```

```
## # Source:   table<taille> [?? x 2]
## # Database: sqlite 3.30.1
## #   [/home/ahmadou/Talks/ModelerMeetup/data/db/taille.db]
##   nom   taille
##   <chr>  <int>
## 1 Ali      170
## 2 Modou    185
## 3 Marie    165
```

Interaction avec SGBD

```
df %>%
  filter(taille > 165) %>%
  select(nom)
```

```
## # Source:   lazy query [?? x 1]
## # Database: sqlite 3.30.1
## #   [/home/ahmadou/Talks/ModelerMeetup/data/db/taille.db]
##   nom
##   <chr>
## 1 Ali
## 2 Modou
```

Interaction avec SGBD

```
df %>%
  filter(taille > 165) %>%
  select(nom) %>%
  collect()
```

```
## # A tibble: 2 x 1
##   nom
##   <chr>
## 1 Ali
## 2 Modou
```

Données spatiales avec R

Données spatiales avec R

```
library(sf)
village <- read_sf("data/vector/loc/loc.shp")
```

```
village
```

```
## Simple feature collection with 10537 features and 25 fields
## geometry type:  POINT
## dimension:      XY
## bbox:           xmin: 228460 ymin: 1366066 xmax: 894318 ymax: 184
## epsg (SRID):    32628
## proj4string:    +proj=utm +zone=28 +datum=WGS84 +units=m +no_defs
## # A tibble: 10,537 x 26
##    IDENTIFI TYPE  SOUSTYP      X      Y     X1     Y1 LONGUEUR
##       <dbl> <chr> <chr>    <dbl>  <dbl>  <dbl>  <dbl>    <dbl>
##  1   232434 TOPO… Village 516370 1.69e6 516370 1.69e6        0
##  2   167966 TOPO… Village 640269 1.51e6 640269 1.51e6        0
##  3   228092 TOPO… Village 553770 1.56e6 553770 1.56e6        0
##  4   232415 TOPO… Village 513432 1.74e6 513432 1.74e6        0
##  5   300844 TOPO… Village 437922 1.55e6 437922 1.55e6        0
##  6   228039 TOPO… Village 536320 1.57e6 536320 1.57e6        0
##  7   168250 TOPO… Village 635370 1.51e6 635370 1.51e6        0
```

Données spatiales avec R

```
routes <- read_sf("data/vector/routes/roads.shp")
```

```
glimpse(routes)
```

```
## Observations: 1,526
## Variables: 16
## $ IDENTIFI <dbl> 143372, 156041, 143509, 143811, 120425, 15587…
## $ TYPE     <chr> "RESEAU ROUTIER", "RESEAU ROUTIER", "RESEAU R…
## $ SOUSTYP  <chr> "Piste repertoriee non bitumee", "Piste reper…
## $ X        <dbl> 715246, 634109, 723301, 718196, 806334, 64532…
## $ Y        <dbl> 1641758, 1437479, 1659364, 1642148, 1425839, …
## $ X1       <dbl> 718196, 635779, 722516, 722516, 825661, 65668…
## $ Y1       <dbl> 1642148, 1432302, 1646055, 1646055, 1421319, …
## $ LONGUEUR <dbl> 2984.897, 5490.288, 13854.212, 6038.137, 2045…
## $ SURFACE  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, …
## $ ANGLE    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, …
## $ DISTANCE <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, …
## $ NOM      <chr> "P409", "P224", "P410", "P410", "59", "P221",…
## $ LONGUEU1 <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, …
## $ VITESSE  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, …
## $ BITUME   <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, …
```

Données spatiales avec R

```
region <- read_sf("data/vector/reg/reg.shp")
```

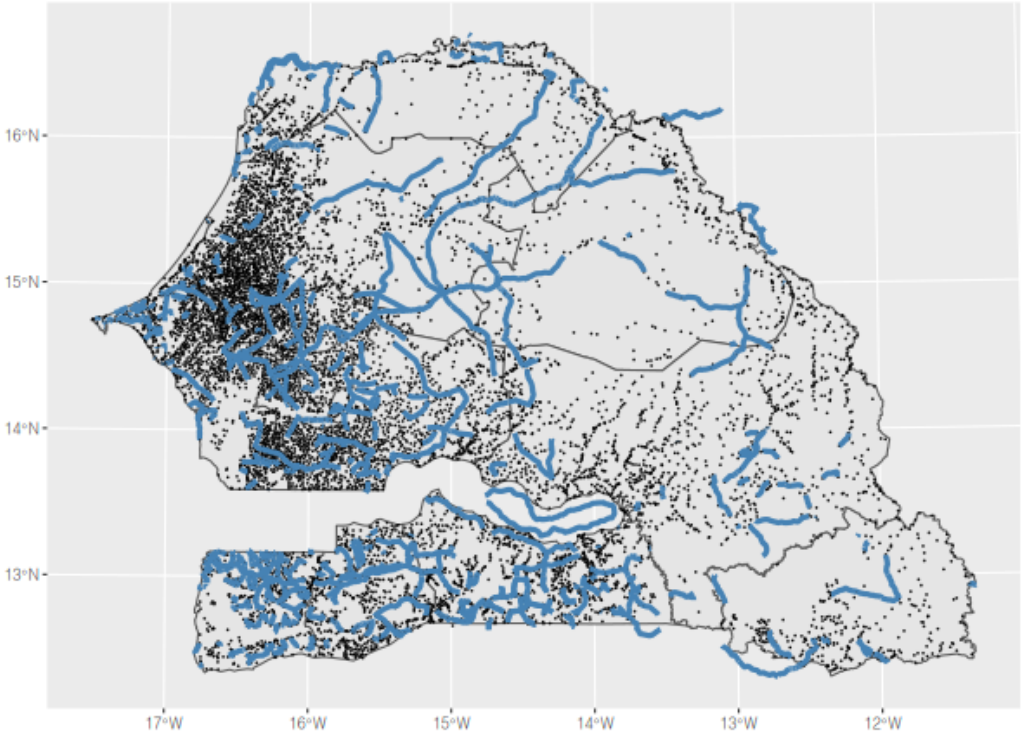
```
glimpse(region)
```

```
## Observations: 14
## Variables: 5
## $ SUPERFICE_ <dbl> 547, 4824, 6849, 5357, 16911, 13771, 24889,…
## $ NOMREG     <chr> "DAKAR", "DIOURBEL", "FATICK", "KOALACK", "…
## $ Code       <int> 1, 2, 3, 6, 5, 7, 8, 9, 10, 11, 12, 13, 14,…
## $ Code1      <int> 1, 2, 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14,…
## $ geometry   <POLYGON [m]> POLYGON ((271870 1633548, 2..., POL…
```

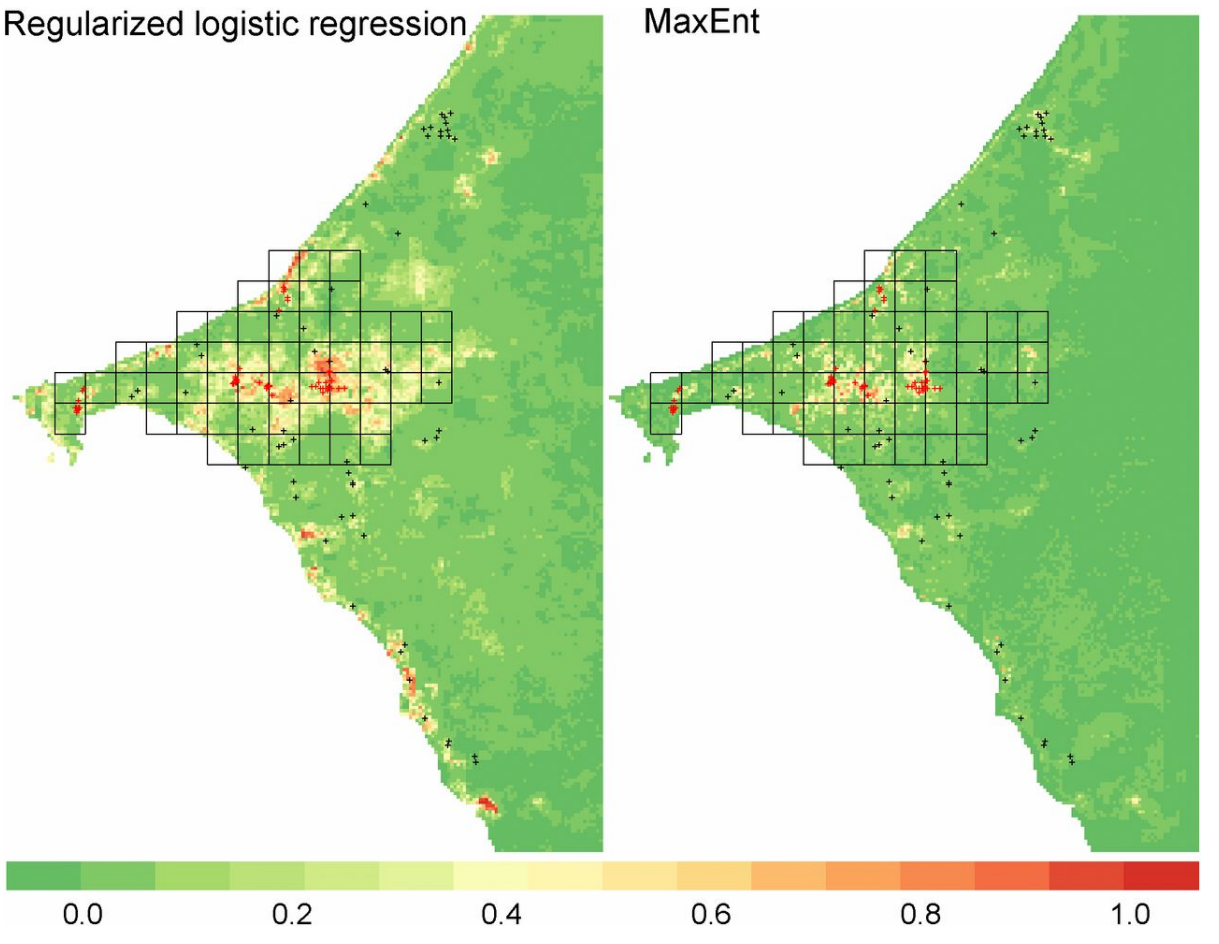
Données spatiales avec R

```
ggplot() +
  geom_sf(data = region) +
  geom_sf(data = village, size = 0.1, shape = 21) +
  geom_sf(data = routes, colour = "steelblue", size = 1.5)
```

Données spatiales avec R



Apprentissage statistique spatiale



Merci!

: mail@ahmadoudicko.com

: ahmadoudicko.com

: @dickoah