Tidyverse

Hugo Harari-Kermadec

$\text{EPOG}^+_2$ - Econometrics

2021
Well-ordered data

Tidyverse is a set of packages to plot and deal with data. Install it once and call it each time you restart R.

```r
install.packages("tidyverse")
library(tidyverse)
```

The 3 tidyverse’s commandments:

- a column for each variable,
- a row for each observation (individual),
- a table for each use.
Well-ordered data

A tidyverse table is called a tibble. You can directly import data as a tibble

```r
pop <- read_delim("population.csv", delim=";")
```

Or convert a dataframe to a tibble

```r
pop2 <- as_tibble(pop1)
```

Most of the time, tidyverse functions are able to deal with dataframes.
Verbs

arrange  sorts lines
slice  chooses lines.
filter  chooses lines according to a test.
select  chooses columns.
rename  renames columns.
mutate  creates new columns and calculates values for each line.

arrange(reduced, age)
slice(reduced, 1:5); filter(reduced, age>220)
select(reduced, age); select(reduced, -age);
rename(reduced, Company=Company_Name)
mutate(reduced, foundation=year-age)
Functions

starts_with selects every variable whose name starts with the argument.
ends_with the same but at the end.
case_when to use with mutate to set a new variable according to cases.

```r
select(reduced, starts_with("w_ln"))
mutate(reduced, Generation = case_when(
age < 10 ~ "young",
age >= 10 & age < 50 ~ "middle",
age >= 50 ~ "old"))
```
The pipe

The pipe allows to sequence many operations on the same data, as on an production line. No need to specify the database more than once. To move to another operation, use `%>%`

db_toy <- reduced %>% na.omit %>%
    select(Company = Company_Name, Year = year,
           Industry = ind_num, Assets = total_assets,
           Investment = w_ln_capex0, Profits_l = w_ln_profits_l) %>%
    arrange(Company, Year)
group_by

group_by allows to split the tibble in different groups of individuals, and to send them on parallel pipes. Operations are then applied independently on each group.

db_toy %>% group_by(Industry) %>%
  mutate(mean_assets=mean(Assets)) %>%
  slice(1) %>%
  select(Industry,mean_assets)
ggplot

ggplot produces great plots. Call ggplot with your data and the main *aesthetic* options you want `aes(x=.., y=.., color=" " , size=" ")` and add elements with +

`geom_point` for points

`geom_line` for lines.

`geom_smooth` to draw a curve according to the points.

ggplot(data=db_toy,
    aes(y=Investment,x=Profits_l))+
geom_point()+
theme( legend.position="none")