

A photograph of a modern building with a glass facade, featuring a prominent blue diagonal structural element. The building is set against a clear blue sky with a single white cloud. In the foreground, there are green plants and a white wall with a logo. A semi-transparent white box is overlaid on the center of the image, containing the title and author's name.

Manipulation de données avec dplyr

Vincent Guillemot

Enchaîner les commandes avec magrittr

- On utilise un opérateur

$\%>\%$

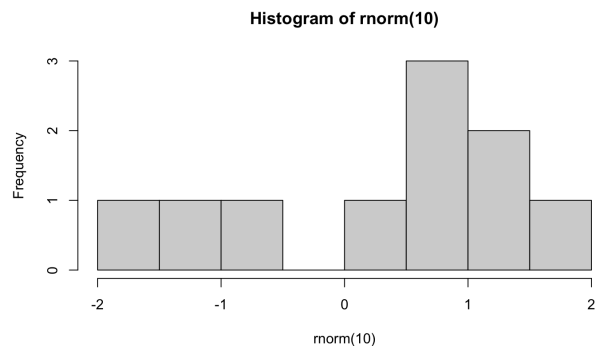


- Avant : $f(g(x))$
- Après : $g(x) \%>\% f()$

Exemple

```
set.seed(7895)
```

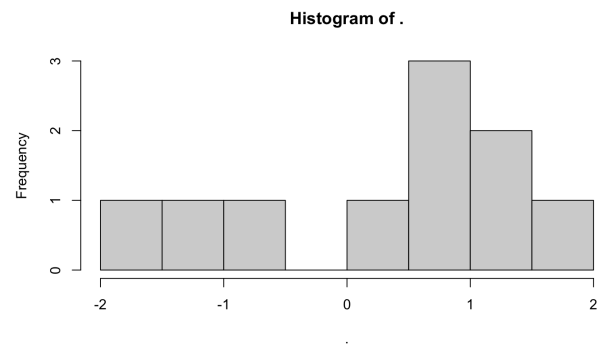
```
hist(rnorm(10))
```



```
library(magrittr)
```

```
set.seed(7895)
```

```
rnorm(10) %>% hist()
```



Le tidyverse



Avant toute chose

Charger le package dplyr...

```
library(dplyr) # ou require(dplyr)
```

Ou bien charger tidyverse...

```
library(tidyverse)
```

... mais cela chargera d'autres packages en plus

Et charger les données.

```
fruits <- readxl::read_excel("fruits.xlsx")
```

Le format “tibble”

Les données sont au format “tibble” : c’est comme des “data-frames” mais en mieux !

```
fruits
#> # A tibble: 51 × 18
#>   nom      groupe Energie   Eau Proteines Glucides Lipides Sucres Fructose Fibres
#>   <chr>    <chr>    <dbl> <dbl>    <dbl>    <dbl>    <dbl> <dbl>    <dbl> <dbl>
#> 1 Abricot  crus       194  87.1     0.81     9.01     0.25   6.7     1.3     1.7
#> 2 Abricot  secs      1010  24.7     2.88    59.1     0.5    34.3    10.6    8.3
#> 3 Ananas   exoti...   304  81.3     0.94    15.1     0.25   14.9     2.8     2.4
#> 4 Banane   exoti...   383  75.8     1.06    19.7     0.25   15.6     3.8     2.7
#> 5 Canneberge secs      1410  14.6     0.25    76.4     1      72.8    28.4    5.7
#> 6 Cerise   crus       235  85.7     0.81    13      0.25   10      4.6     1.6
#> 7 Citron   crus       118  91.3     0.25    1.56    0.25   0.8     0.4     0.25
#> 8 Clementine crus       200  87      0.81    9.17    0.25   8.6     1.5     1.7
#> 9 CompoteMult... compo...  279  82.9     0.25    15.3     0.08   14.6     6.92    1.6
#> 10 CompotePomme compo...  432  72.9     0.23    24.4     0.21   20.7     7.9     1.53
#> # i 41 more rows
#> # i 8 more variables: Calcium <dbl>, Magnesium <dbl>, Phosphore <dbl>,
#> #   Potassium <dbl>, Zinc <dbl>, BetaCarotene <dbl>, VitamineE <dbl>, VitamineC <dbl>
```

Les fonctions de dplyr

Nous allons voir ensemble quelques fonctions très pratiques de la librairie `dplyr`.

#	Fonction (US)	Fonction (UK)	Description
1	<code>mutate</code>	<code>mutate</code>	Créer ou modifier des colonnes
2	<code>select</code>	<code>select</code>	Sélectionner des colonnes
3	<code>arrange</code>	<code>arrange</code>	Trier les lignes
4	<code>filter</code>	<code>filter</code>	Sélectionner des lignes
5	<code>group_by</code>	<code>group_by</code>	Grouper des lignes
6	<code>summarize</code>	<code>summarise</code>	Résumer des groupes

Créer ou modifier des colonnes

Avec la fonction `mutate`.

```
fruits2 <- fruits %>%  
  mutate(Sucres_ratio = Sucres / 100)  
  
head(fruits2[, "Sucres_ratio"])  
#> # A tibble: 6 × 1  
#>   Sucres_ratio  
#>       <dbl>  
#> 1      0.067  
#> 2      0.343  
#> 3      0.149  
#> 4      0.156  
#> 5      0.728  
#> 6      0.1
```

Avec les fonctions classiques.

```
fruits2 <- fruits  
fruits2$Sucres_ratio <-  
  fruits2$Sucres / 100  
  
head(fruits2[, "Sucres_ratio"])  
#> # A tibble: 6 × 1  
#>   Sucres_ratio  
#>       <dbl>  
#> 1      0.067  
#> 2      0.343  
#> 3      0.149  
#> 4      0.156  
#> 5      0.728  
#> 6      0.1
```


Sélectionner des colonnes

Avec la fonction `select`.

```
fruits %>%  
  select(  
    Energie,  
    Sucres,  
    Lipides,  
    Proteines)
```

Avec les fonctions classiques.

```
fruits[,  
  c(  
    "Energie",  
    "Sucres",  
    "Lipides",  
    "Proteines") ]
```

Trier des lignes

Avec les fonctions `arrange` et `desc`.

```
fruits %>%
  select(Energie, Sucres, Fibres) %>%
  arrange(desc(Fibres))
#> # A tibble: 51 × 3
#>   Energie Sucres Fibres
#>   <dbl>  <dbl>  <dbl>
#> 1    1010   34.3    8.3
#> 2     425    8.5    6.8
#> 3    1410   72.8    5.7
#> 4     198    6.1    5.2
#> 5     969   38.1    5.1
#> 6     289    6.63   4.6
#> 7     206    5.4    4.3
#> 8     170    2.1    4.3
#> 9    1360   70.3    4.2
#> 10     293   12.2    4.1
#> # i 41 more rows
```

Avec les fonctions classiques

```
fruits[
  order(fruits$Fibres, decreasing = TRUE),
  c("Energie", "Sucres", "Fibres")]
#> # A tibble: 51 × 3
#>   Energie Sucres Fibres
#>   <dbl>  <dbl>  <dbl>
#> 1    1010   34.3    8.3
#> 2     425    8.5    6.8
#> 3    1410   72.8    5.7
#> 4     198    6.1    5.2
#> 5     969   38.1    5.1
#> 6     289    6.63   4.6
#> 7     206    5.4    4.3
#> 8     170    2.1    4.3
#> 9    1360   70.3    4.2
#> 10     293   12.2    4.1
#> # i 41 more rows
```

Sélectionner des lignes

Avec la fonction `filter`.

```
fruits %>%
  filter(Sucres > 60)
#> # A tibble: 2 × 18
#>   nom      groupe Energie   Eau Proteines Gluci
#>   <chr> <chr>      <dbl> <dbl>      <dbl>  <c
#> 1 Cann... secs      1410  14.6      0.25    7
#> 2 Rais... secs      1360  16        3      7
#> # i 7 more variables: Magnesium <dbl>, Phosphore <dbl>,
#> #   BetaCarotene <dbl>, VitamineE <dbl>, VitamineC <dbl>
```

Avec les fonctions classiques.

```
fruits[fruits$Sucres > 60, ]
#> # A tibble: 2 × 18
#>   nom      groupe Energie   Eau Proteines Glucides Lipide
#>   <chr> <chr>      <dbl> <dbl>      <dbl>  <dbl>  <dbl>
#> 1 Cann... secs      1410  14.6      0.25    76.4    1
#> 2 Rais... secs      1360  16        3    73.2    0.
#> # i 7 more variables: Magnesium <dbl>, Phosphore <dbl>,
#> #   BetaCarotene <dbl>, VitamineE <dbl>, VitamineC <dbl>
```

Agréger des colonnes

Avec la fonction `group_by` :

```
fruits %>% group_by(groupe)
#> # A tibble: 51 × 18
#> # Groups:   groupe [4]
#>   nom      groupe Energie   Eau Proteines Glucides Lipides Sucres Fructose Fibres
#>   <chr>    <chr>   <dbl> <dbl>    <dbl>   <dbl>   <dbl> <dbl>   <dbl> <dbl>
#> 1 Abricot   crus      194  87.1     0.81    9.01    0.25   6.7    1.3    1.7
#> 2 Abricot   secs     1010  24.7     2.88   59.1    0.5    34.3   10.6   8.3
#> 3 Ananas    exoti...   304  81.3     0.94   15.1    0.25   14.9    2.8    2.4
#> 4 Banane    exoti...   383  75.8     1.06   19.7    0.25   15.6    3.8    2.7
#> 5 Canneberge secs     1410  14.6     0.25   76.4    1      72.8   28.4   5.7
#> 6 Cerise    crus      235  85.7     0.81   13      0.25   10      4.6    1.6
#> 7 Citron    crus      118  91.3     0.25   1.56    0.25   0.8     0.4    0.25
#> 8 Clementine crus      200  87      0.81   9.17    0.25   8.6     1.5    1.7
#> 9 CompoteMult... compo...   279  82.9     0.25   15.3    0.08   14.6    6.92   1.6
#> 10 CompotePomme compo...   432  72.9     0.23   24.4    0.21   20.7    7.9    1.53
#> # i 41 more rows
#> # i 8 more variables: Calcium <dbl>, Magnesium <dbl>, Phosphore <dbl>,
#> #   Potassium <dbl>, Zinc <dbl>, BetaCarotene <dbl>, VitamineE <dbl>, VitamineC <dbl>
```

Les données sont prêtes à être “traitées” groupe par groupe. PS : L’opération `ungroup()` permet d’enlever les groupes.

Calculer une moyenne

Avec la fonction `summarize`.

```
fruits %>%
  group_by(groupe) %>%
  summarize(SucreMoyen = mean(Sucres))
#> # A tibble: 4 × 2
#>   groupe   SucreMoyen
#>   <chr>     <dbl>
#> 1 compote    15.5
#> 2 crus       9.68
#> 3 exotique   11.4
#> 4 secs      55.0
```

Avec les fonctions classiques.

```
aggregate(fruits$Sucres,
          by = list(fruits$groupe),
          FUN = mean)
#>   Group.1      x
#> 1  compote 15.533333
#> 2   crus   9.684242
#> 3 exotique 11.380000
#> 4   secs 54.980000
```

Exercice(s)

Calculer l'énergie moyenne, la teneur en sucres médiane et le maximum de la teneur en Fibres par groupe de fruits et trier le tout par ordre décroissant du maximum de la teneur en Fibres !

Deux autres fonctions

	Sélectionne	Ne sélectionne pas
Ne transforme pas	<code>select</code>	<code>rename</code>
Peut transformer	<code>transmute</code>	<code>mutate</code>



David Robinson

Principal Data Scientist at Heap



Mais il y en a tellement d'autres !
