Lecture 6 – simple stats within the tidyverse

AUTHOR
Paolo Crosetto

PUBLISHED
September 1, 2024

Intro

Jusque là on a utilisé R pour *manipuler* et representer *graphiquement* des *jeux de données* (en langage R, des `data.frame`). Mais R est né et a été développé pincipalement comme un langage de *statistique*. Aujourd'hui on va donc s'occuper de cela: comment faire des statistiques simples *(correlation, model linéaire, test statistique)* avec R.

Pour chaque commande / outil statistique qu'on couvrira aujourd'hui, l'exposition va se diviser en deux parties:

1. **statistiques avec base R**. Pour faire des stats on va abandonner de façon temporaire le *tidyverse* et utiliser le système R de base. En soi cela ne vous changera pas grand chose, mais vous verrez que les résultats des différentes fonctions (tests, régressions, etc…) ne sont pas dans un format *tidy* – n'ont pas une variable par colonne et une observation par ligne – et ne sont donc pas prêts tout de suite à être utilisés avec ce qu'on a appris jusque là (`filter`, `select`, `ggplot` …).
2. **lien stats de base -> tidyverse**. Heureusement il y a des solutions. On va donc explorer le package `broom` qui permet de 'faire le ménage' (broom signifie *balai* en anglais) et de transofrmer les résultats des régressions en `data.frame`s bien ordonnés, qu'on pourra utiliser pour, par exemple, visualiser les résultats de façon graphique. On pourra aussi utiliser la puissance du `%>%` et de `group_by()` pour mettre en place de façon rapide des analyses par groupe et les visualiser.

Régression linéaire.

Base R

> utilisez les données `airquality` – il s'agit de la qualité de l'air à NYC sur un certain interval de temps en 1973. Estimez une **régression linéaire** qui explique le niveau **d'Ozone** par la *température*, le *vent* et la *radiation solaire*. Sauvegardez le résultat de la régression dans un objet, `firstreg`.

```
## utilisons summary() d'abord
airquality %>% summary()
```

```
     Ozone           Solar.R          Wind             Temp      
 Min.   :  1.00   Min.   :  7.0   Min.   : 1.700   Min.   :56.00  
 1st Qu.: 18.00   1st Qu.:115.8   1st Qu.: 7.400   1st Qu.:72.00  
 Median : 31.50   Median :205.0   Median : 9.700   Median :79.00  
 Mean   : 42.13   Mean   :185.9   Mean   : 9.958   Mean   :77.88  
 3rd Qu.: 63.25   3rd Qu.:258.8   3rd Qu.:11.500   3rd Qu.:85.00  
 Max.   :168.00   Max.   :334.0   Max.   :20.700   Max.   :97.00  
 NA's   :37       NA's   :7                                       
     Month            Day      
 Min.   :5.000   Min.   : 1.0  
 1st Qu.:6.000   1st Qu.: 8.0  
 Median :7.000   Median :16.0  
 Mean   :6.993   Mean   :15.8  
 3rd Qu.:8.000   3rd Qu.:23.0  
 Max.   :9.000   Max.   :31.0  
```

```
## une régression s'écrit Y = K + b1*VAR + b2*VAR ....
## en R c'est Y ~ VAR1 + VAR2
firstreg <- lm(Ozone ~ Temp + Wind + Solar.R, data = airquality)
firstreg
```

```

Call:
lm(formula = Ozone ~ Temp + Wind + Solar.R, data = airquality)

Coefficients:
(Intercept)         Temp         Wind      Solar.R  
  -64.34208      1.65209     -3.33359      0.05982  
```

> explorez `firstreg` dans votre environnement. Il y a beaucoup de sous objets et sous parties. Essayez de visualiser le tableau récapitulatif. Utilisez `summary`

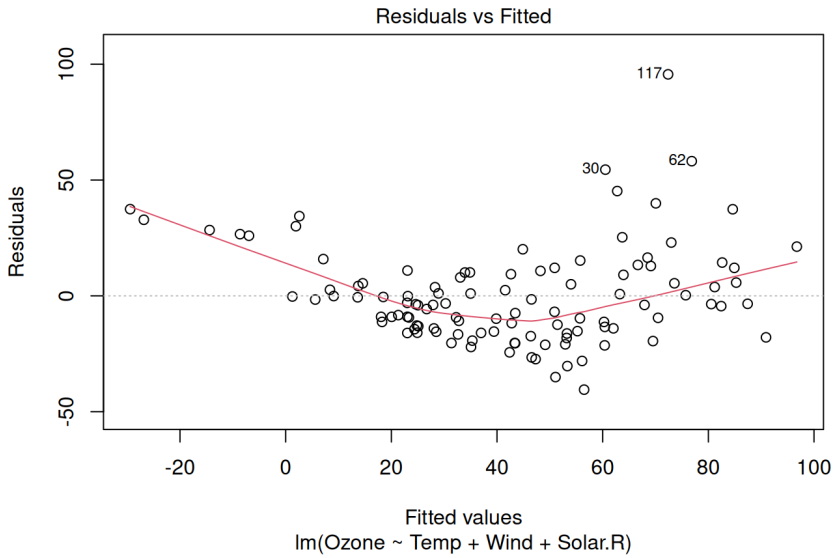
```
firstreg %>% summary()
```

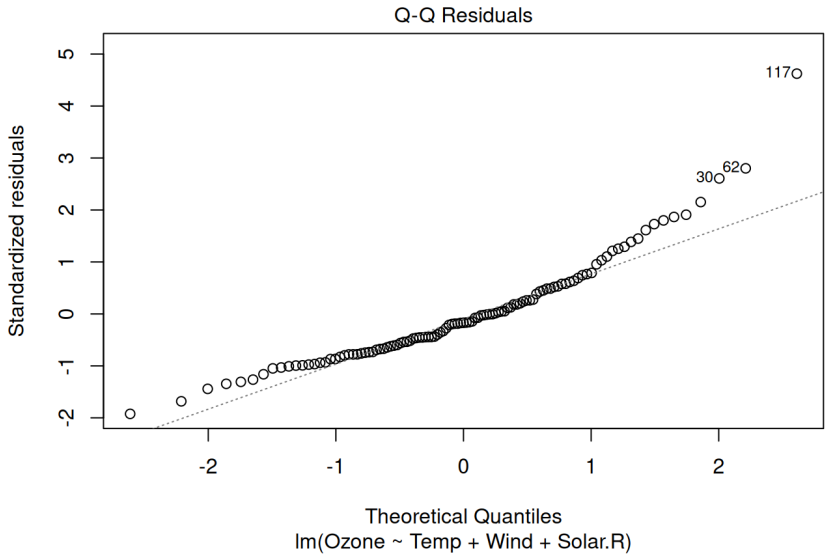
```

Call:
lm(formula = Ozone ~ Temp + Wind + Solar.R, data = airquality)

Residuals:
    Min      1Q  Median      3Q     Max 
-40.485 -14.219  -3.551  10.097  95.619 

Coefficients:
```

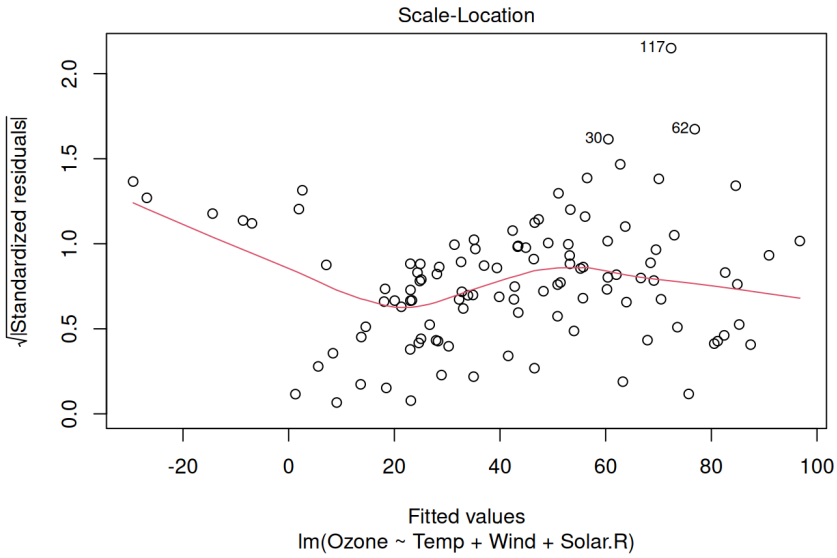
```
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -64.34208   23.05472  -2.791  0.00623 **
Temp          1.65209    0.25353   6.516 2.42e-09 ***
Wind         -3.33359    0.65441  -5.094 1.52e-06 ***
Solar.R       0.05982    0.02319   2.580  0.01124 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 21.18 on 107 degrees of freedom
  (42 observations deleted due to missingness)
Multiple R-squared:  0.6059,    Adjusted R-squared:  0.5948
F-statistic: 54.83 on 3 and 107 DF,  p-value: < 2.2e-16
```

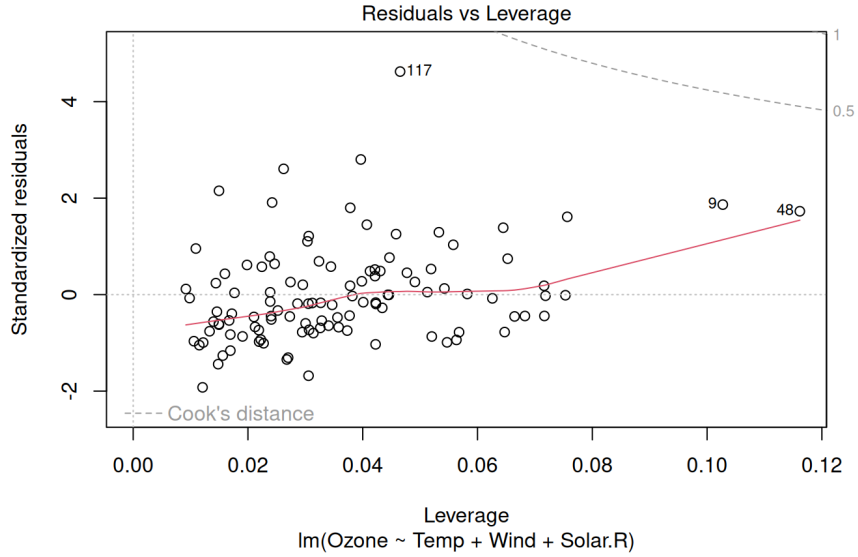
> il y d'autres sous objets que vous pouvez explorer. Si vous plottez l'objet, cela donne tous les diagnostics de régression (on ne rentre pas dans les détails ici). Notez que les plots ne sont pas des ggplots, mais ont un air très différents; il s'agit des plot produits par Base R.

```
plot(firstreg)
```









> vous pouvez accéder au coéfficients avec la fonction `coef()`

```
coef(firstreg)
```

```
 (Intercept)         Temp         Wind      Solar.R 
-64.34207893   1.65209291  -3.33359131   0.05982059 
```

exercice

> faites une régression de Ozone sur Wind en utilisant Base R et affichez son `summary()`

```
lm(Ozone ~ Wind, data = airquality) %>%
  summary()
```

```
Call:
lm(formula = Ozone ~ Wind, data = airquality)

Residuals:
    Min      1Q  Median      3Q     Max 
-51.572 -18.854  -4.868  15.234  90.000 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  96.8729     7.2387   13.38  < 2e-16 ***
Wind         -5.5509     0.6904   -8.04 9.27e-13 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 26.47 on 114 degrees of freedom
  (37 observations deleted due to missingness)
Multiple R-squared:  0.3619,    Adjusted R-squared:  0.3563 
F-statistic: 64.64 on 1 and 114 DF,  p-value: 9.272e-13
```

Broom

> tout cela n'est pas très pratique parce qu'on ne peut pas accéder aux données de la régression de façon *tidy*. On va donc utiliser `broom` pour le faire.

`broom` dispose de trois fonctions.

1. `tidy` retourne les coefficients, valeurs p et intervaux de confiance de la régression en format data.frame.
2. `glance` retourne les indicateurs de diagnostic de la régression en format data.frame (sur une ligne)
3. `augment` retourne les données initiales 'augmentées' avec les valuers estimées par la régression.

tidy

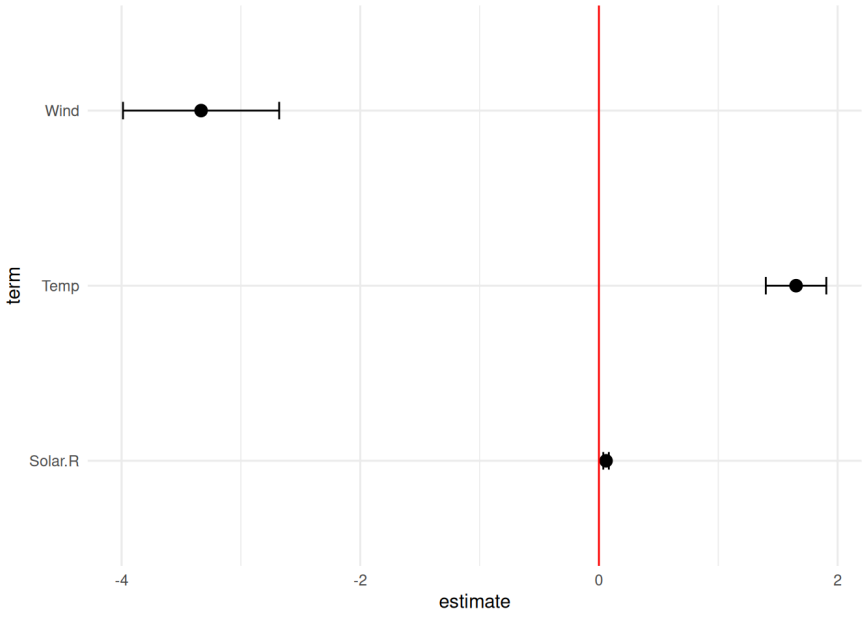
voilà le output de `tidy` pour notre objet `firstreg`:

```
library(broom)

firstreg %>% tidy()
```

```
# A tibble: 4 × 5
  term        estimate std.error statistic       p.value
  <chr>          <dbl>     <dbl>     <dbl>         <dbl>
1 (Intercept) -64.3      23.1        -2.79 0.00623
2 Temp          1.65      0.254       6.52 0.00000000242
3 Wind         -3.33      0.654      -5.09 0.00000152
4 Solar.R       0.0598    0.0232      2.58 0.0112
```

> et voilà, nos résultats sont maintenant en forme de data.frame et peuvent donc être utilisés pour nos analyses, plots... notamment: faites un **plot des coefficients** de la régression **avec des barres d'erreur**.

```
firstreg %>%
  tidy() %>%
  filter(term != "(Intercept)") %>%
  ggplot()+
  aes(y = term, x = estimate)+
  geom_point(size = 3)+
  geom_errorbarh(aes(xmin = estimate - std.error,
                     xmax = estimate + std.error), height = .1)+
  geom_vline(xintercept = 0, color = "red")+
  #coord_cartesian(xlim = c(-0.1, 0.2))+
  theme_minimal()
```

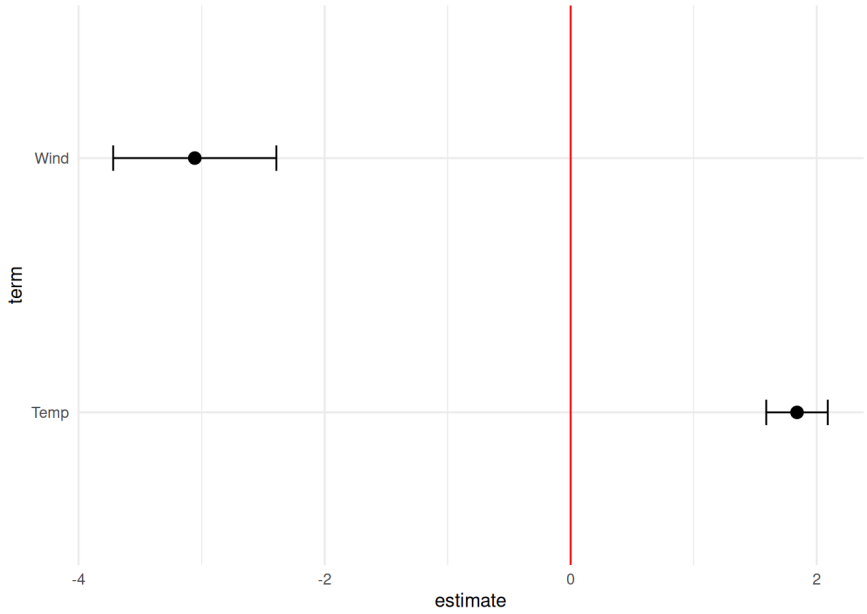


exercice

Regréssion de Ozone sur Wind et Temp (airquality) + plot des coefficients

```
# stratégie

# 1. faire le lm()
# 2. le passer par tidy()
# 3. faire le plot

# équivalent à lm(Ozone ~ Temp + Wind, data = airquality)

lm(Ozone ~ Temp + Wind, data = airquality) %>%
  tidy() %>%
  filter(term != "(Intercept)") %>%
```

```
ggplot()+
aes(y = term, x = estimate)+
geom_point(size = 3)+
geom_errorbarh(aes(xmin = estimate - std.error,
                   xmax = estimate + std.error), height = .1)+
geom_vline(xintercept = 0, color = "red")+
theme_minimal()
```



2. augment

augment ajoute les valuers estimées à notre data.frame. Cela nous permet, par exemple, de voir la régression de façon visuelle (sur deux variables uniquement) en plottant les points intiaux et les points estimés. On va faire cela par étapes, et pour la rélation Ozone ~ Temp

0. regardons ce que `augment` fait

```
airquality
```

```
    Ozone Solar.R Wind Temp Month Day
1      41     190  7.4   67     5   1
2      36     118  8.0   72     5   2
3      12     149 12.6   74     5   3
4      18     313 11.5   62     5   4
5      NA      NA 14.3   56     5   5
6      28      NA 14.9   66     5   6
7      23     299  8.6   65     5   7
8      19      99 13.8   59     5   8
9       8      19 20.1   61     5   9
10     NA     194  8.6   69     5  10
11      7      NA  6.9   74     5  11
12     16     256  9.7   69     5  12
13     11     290  9.2   66     5  13
14     14     274 10.9   68     5  14
15     18      65 13.2   58     5  15
16     14     334 11.5   64     5  16
17     34     307 12.0   66     5  17
18      6      78 18.4   57     5  18
19     30     322 11.5   68     5  19
20     11      44  9.7   62     5  20
21      1       8  9.7   59     5  21
22     11     320 16.6   73     5  22
23      4      25  9.7   61     5  23
24     32      92 12.0   61     5  24
25     NA      66 16.6   57     5  25
26     NA     266 14.9   58     5  26
27     NA      NA  8.0   57     5  27
28     23      13 12.0   67     5  28
29     45     252 14.9   81     5  29
30    115     223  5.7   79     5  30
31     37     279  7.4   76     5  31
32     NA     286  8.6   78     6   1
33     NA     287  9.7   74     6   2
34     NA     242 16.1   67     6   3
35     NA     186  9.2   84     6   4
36     NA     220  8.6   85     6   5
37     NA     264 14.3   79     6   6
38     29     127  9.7   82     6   7
39     NA     273  6.9   87     6   8
40     71     291 13.8   90     6   9
41     39     323 11.5   87     6  10
42     NA     259 10.9   93     6  11
```

43	NA	250	9.2	92	6	12
44	23	148	8.0	82	6	13
45	NA	332	13.8	80	6	14
46	NA	322	11.5	79	6	15
47	21	191	14.9	77	6	16
48	37	284	20.7	72	6	17
49	20	37	9.2	65	6	18
50	12	120	11.5	73	6	19
51	13	137	10.3	76	6	20
52	NA	150	6.3	77	6	21
53	NA	59	1.7	76	6	22
54	NA	91	4.6	76	6	23
55	NA	250	6.3	76	6	24
56	NA	135	8.0	75	6	25
57	NA	127	8.0	78	6	26
58	NA	47	10.3	73	6	27
59	NA	98	11.5	80	6	28
60	NA	31	14.9	77	6	29
61	NA	138	8.0	83	6	30
62	135	269	4.1	84	7	1
63	49	248	9.2	85	7	2
64	32	236	9.2	81	7	3
65	NA	101	10.9	84	7	4
66	64	175	4.6	83	7	5
67	40	314	10.9	83	7	6
68	77	276	5.1	88	7	7
69	97	267	6.3	92	7	8
70	97	272	5.7	92	7	9
71	85	175	7.4	89	7	10
72	NA	139	8.6	82	7	11
73	10	264	14.3	73	7	12
74	27	175	14.9	81	7	13
75	NA	291	14.9	91	7	14
76	7	48	14.3	80	7	15
77	48	260	6.9	81	7	16
78	35	274	10.3	82	7	17
79	61	285	6.3	84	7	18
80	79	187	5.1	87	7	19
81	63	220	11.5	85	7	20
82	16	7	6.9	74	7	21
83	NA	258	9.7	81	7	22
84	NA	295	11.5	82	7	23
85	80	294	8.6	86	7	24
86	108	223	8.0	85	7	25
87	20	81	8.6	82	7	26
88	52	82	12.0	86	7	27
89	82	213	7.4	88	7	28
90	50	275	7.4	86	7	29
91	64	253	7.4	83	7	30
92	59	254	9.2	81	7	31
93	39	83	6.9	81	8	1
94	9	24	13.8	81	8	2
95	16	77	7.4	82	8	3
96	78	NA	6.9	86	8	4
97	35	NA	7.4	85	8	5
98	66	NA	4.6	87	8	6
99	122	255	4.0	89	8	7
100	89	229	10.3	90	8	8
101	110	207	8.0	90	8	9
102	NA	222	8.6	92	8	10
103	NA	137	11.5	86	8	11
104	44	192	11.5	86	8	12
105	28	273	11.5	82	8	13
106	65	157	9.7	80	8	14
107	NA	64	11.5	79	8	15
108	22	71	10.3	77	8	16
109	59	51	6.3	79	8	17
110	23	115	7.4	76	8	18
111	31	244	10.9	78	8	19
112	44	190	10.3	78	8	20
113	21	259	15.5	77	8	21
114	9	36	14.3	72	8	22
115	NA	255	12.6	75	8	23
116	45	212	9.7	79	8	24
117	168	238	3.4	81	8	25
118	73	215	8.0	86	8	26
119	NA	153	5.7	88	8	27
120	76	203	9.7	97	8	28
121	118	225	2.3	94	8	29
122	84	237	6.3	96	8	30
123	85	188	6.3	94	8	31
124	96	167	6.9	91	9	1
125	78	197	5.1	92	9	2
126	73	183	2.8	93	9	3
127	91	189	4.6	93	9	4
128	47	95	7.4	87	9	5
129	32	92	15.5	84	9	6
130	20	252	10.9	80	9	7
131	23	220	10.3	78	9	8

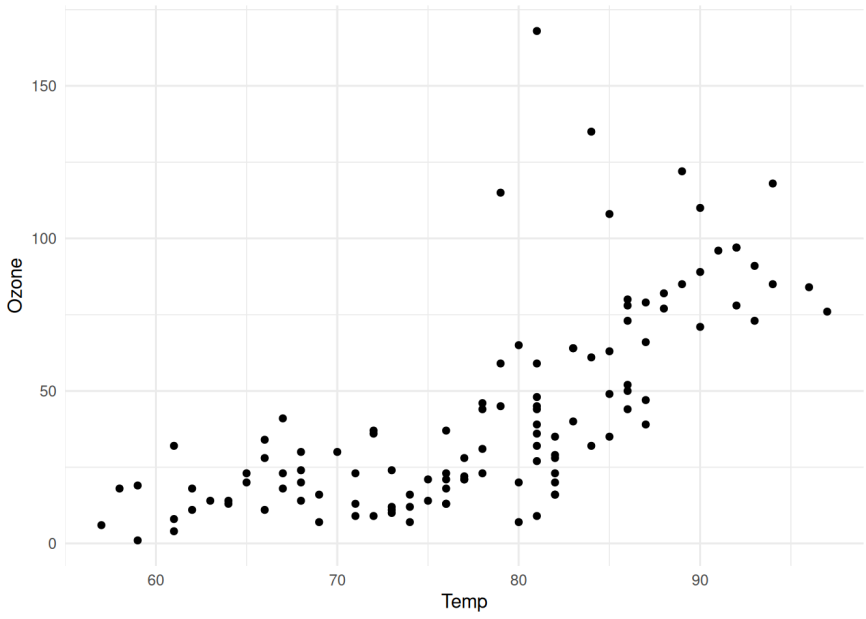
```
132    21     230 10.9   75     9   9
133    24     259  9.7   73     9  10
134    44     236 14.9   81     9  11
135    21     259 15.5   76     9  12
136    28     238  6.3   77     9  13
137     9      24 10.9   71     9  14
138    13     112 11.5   71     9  15
139    46     237  6.9   78     9  16
140    18     224 13.8   67     9  17
141    13      27 10.3   76     9  18
142    24     238 10.3   68     9  19
143    16     201  8.0   82     9  20
144    13     238 12.6   64     9  21
145    23      14  9.2   71     9  22
146    36     139 10.3   81     9  23
147     7      49 10.3   69     9  24
148    14      20 16.6   63     9  25
149    30     193  6.9   70     9  26
150    NA     145 13.2   77     9  27
151    14     191 14.3   75     9  28
152    18     131  8.0   76     9  29
153    20     223 11.5   68     9  30
```

```r
lm(Ozone ~ Wind, data = airquality) %>% augment()
```

```
# A tibble: 116 × 9
   .rownames Ozone  Wind .fitted .resid    .hat .sigma   .cooksd .std.resid
   <chr>     <int> <dbl>   <dbl>  <dbl>   <dbl>  <dbl>     <dbl>      <dbl>
 1 1            41   7.4    55.8 -14.8  0.0127    26.5 0.00204      -0.563
 2 2            36   8      52.5 -16.5  0.0110    26.5 0.00217      -0.626
 3 3            12  12.6    26.9 -14.9  0.0137    26.5 0.00224      -0.568
 4 4            18  11.5    33.0 -15.0  0.0104    26.5 0.00172      -0.571
 5 6            28  14.9    14.2  13.8  0.0259    26.6 0.00373       0.530
 6 7            23   8.6    49.1 -26.1  0.00970   26.5 0.00482      -0.992
 7 8            19  13.8    20.3  -1.27 0.0192    26.6 0.0000229    -0.0485
 8 9             8  20.1   -14.7  22.7  0.0799    26.5 0.0347        0.894
 9 11            7   6.9    58.6 -51.6  0.0146    26.1 0.0285       -1.96
10 12           16   9.7    43.0 -27.0  0.00864   26.5 0.00458      -1.03
# i 106 more rows
```

1. plot des points originaux de la rélation Ozone ~ Temp

```r
secondreg <- lm(Ozone ~ Temp, data = airquality)

plot1 <- secondreg %>%
  ggplot()+
  aes(y = Ozone, x = Temp)+
  geom_point()+
  theme_minimal()

plot1
```

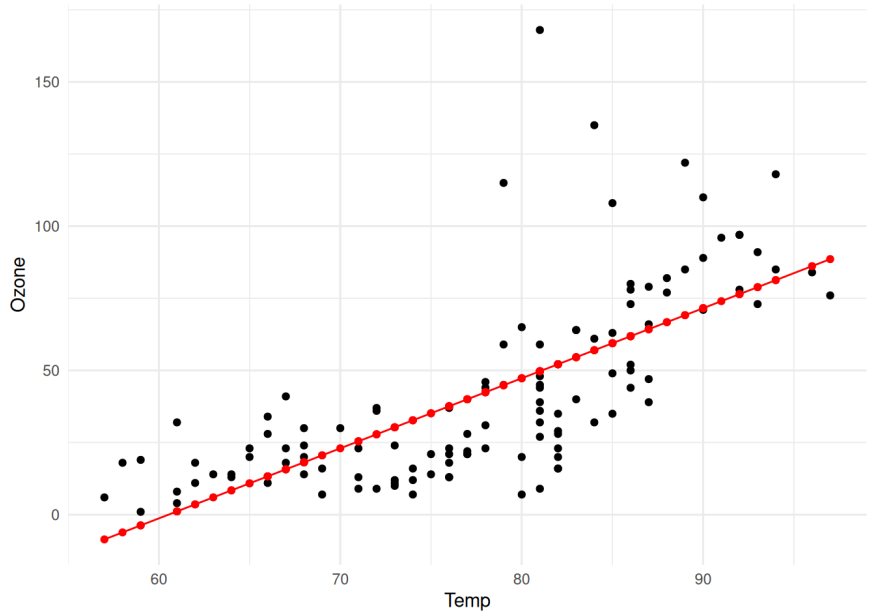


2. on ajoute les points estimés

(attention: ils seront sur une droite. Surpris?)

```r
plot2 <- secondreg %>%
  augment() %>%
  ggplot() +
  aes(y = Ozone, x = Temp)+
  geom_point()+
```

```
  geom_point(aes(y = .fitted), color = "red")+
  geom_line(aes(y = .fitted), color = "red")+
  theme_minimal()

plot2
```

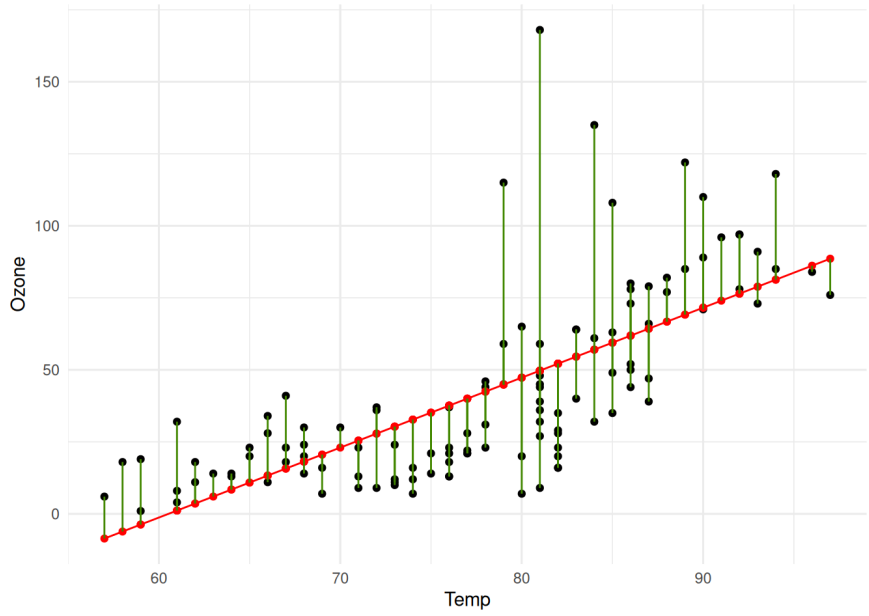


Vu qu'ils suivent une ligne droite, on peut aisément les plotter comme une `geom_line()`

```
## regarder le plot ci-dessus
```

Pour mieux visualiser les résidus on lie chaque point réel à sa prédiction par un `geom_segment()`

```
plot2 +
    geom_segment(aes(xend = Temp, yend = .fitted), color = "chartreuse4")
```



exercice

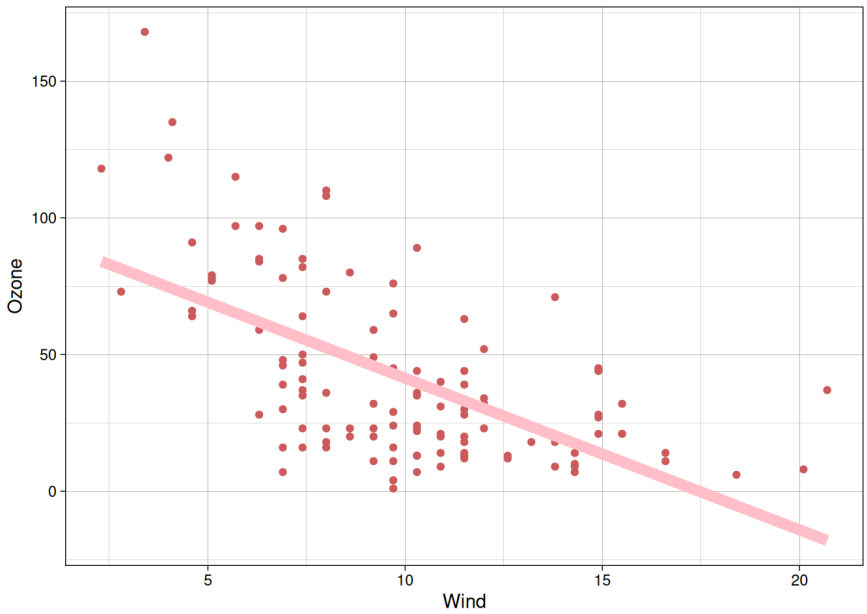
> faites une régression de Ozone sur Wind en utilisant le tidyverse et `tidy`

```
thirdreg <- lm(Ozone ~ Wind, data = airquality)

thirdreg %>% tidy()
```

```
# A tibble: 2 × 5
  term        estimate std.error statistic  p.value
  <chr>          <dbl>     <dbl>     <dbl>    <dbl>
1 (Intercept)    96.9       7.24      13.4  3.99e-25
2 Wind           -5.55      0.690     -8.04 9.27e-13
```

> montrez les points originaux et la droite de régression sur un plot en utilisant `augment`

```
thirdreg %>%
  augment() %>%
  ggplot()+
  aes(x = Wind, y = Ozone)+
  geom_point(color = "indianred")+
  geom_line(aes(y = .fitted), color = "pink", linewidth = 3)+
  theme_linedraw()
```



glance

glance est moins immédiatement utile mais le deviendra quand on pourra comparer différents modèles statistiques les uns à côté des autres. voilà ce que `glance()` donne.

```
firstreg %>% glance()
```

```
# A tibble: 1 × 12
  r.squared adj.r.squared sigma statistic  p.value    df logLik   AIC   BIC
      <dbl>         <dbl> <dbl>     <dbl>    <dbl> <dbl>  <dbl> <dbl> <dbl>
1     0.606         0.595  21.2      54.8 1.51e-21     3  -494.  999. 1012.
# ℹ 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

La puissance du tidyverse: plusieurs régressions à la fois, par groupe

Avec le tidyverse on peut lancer plusieurs régressions à la fois, et en visualiser les résultats avec un seul `ggplot`. On va travailler avec la base de données `gapminder`.

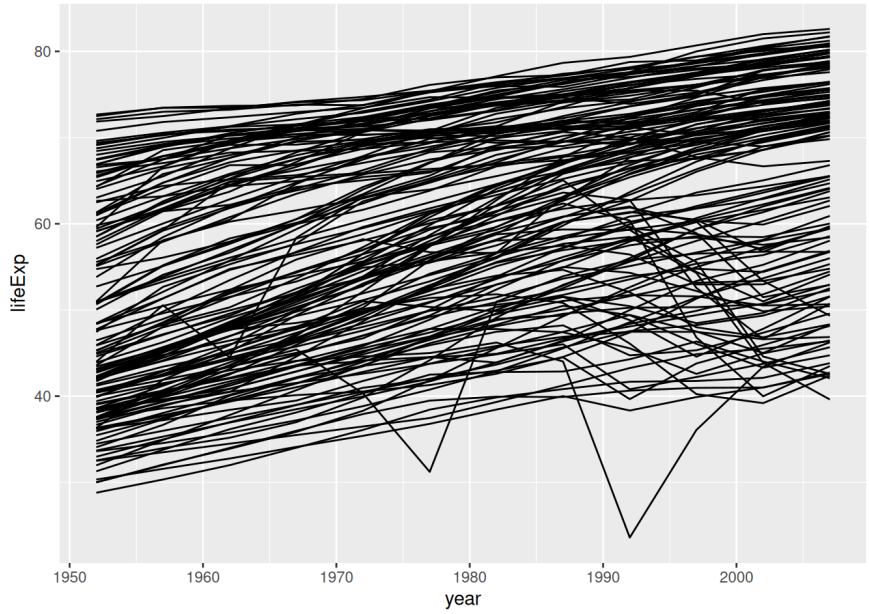
gapminder

`gapminder` est une base de données qui contient l'espérence de vie par pays sur plusieurs années. Il faut installer le package gapminder

```
#install.packages("gapminder")
library(gapminder)
```

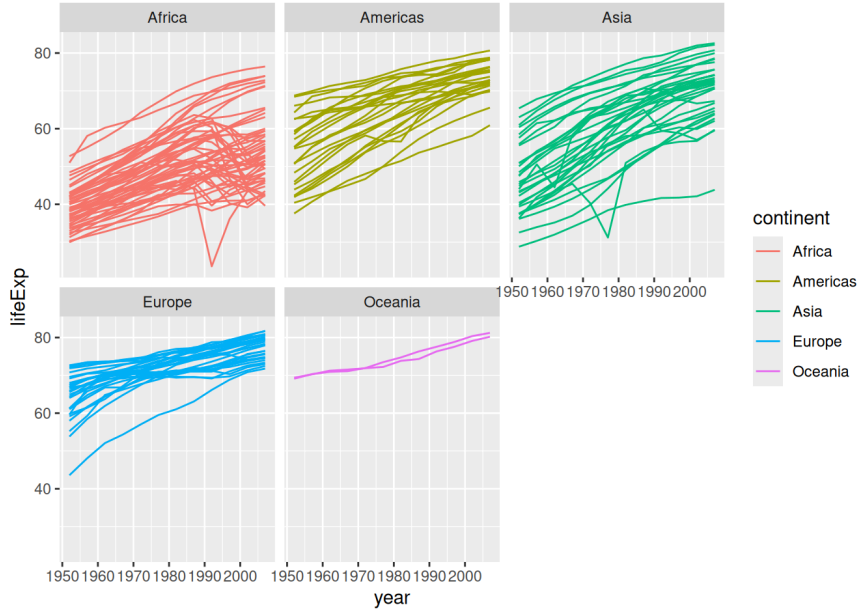
On va commencer par explorer les données. Comment l'espérance de vie a-t-elle évolué dans le temps pour tous les pays? un ggplot

```
gp <- gapminder

spaghetti <- gp %>%
  ggplot()+
  aes(x = year, y = lifeExp, group = country)+
  geom_line()
spaghetti
```



ça, c'est ce qu'on appelle un '**spaghetti plot**' – on n'y comprend rien. On va ajouter des facets et colorier par continent.

```
spaghetti +
  aes(color = continent)+
  facet_wrap(~continent)
```



L'espérance de vie à l'air d'avoir augmenté un peu partout. **Mais pourquoi?** s'agit-il d'un effet de richesse – plus on est riches, plus long on vit?

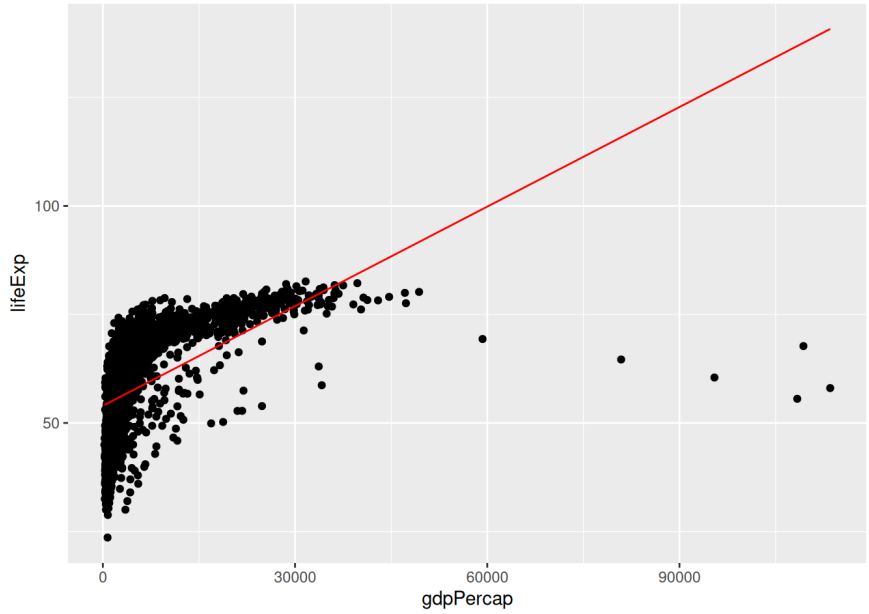
On va faire une régression pour cela. On sait comment faire:

```
reggp <- lm(lifeExp ~ gdpPercap, data = gp)


reggp %>% tidy()
```

```
# A tibble: 2 × 5
  term         estimate std.error statistic   p.value
  <chr>           <dbl>     <dbl>     <dbl>     <dbl>
1 (Intercept) 54.0      0.315         171.  0
2 gdpPercap    0.000765 0.0000258      29.7 3.57e-156
```

cela à l'air très significatif. On va faire un plot en utilisant `augment` pour vérifier de façon visuelle

```
reggp %>%
  augment() %>%
  ggplot()+
  aes(y = lifeExp, x = gdpPercap)+
  geom_point()+
  geom_line(aes(y = .fitted), color = "red")
```



pourquoi le fit est si mauvais? parce que les données contiennent une observation par pays **par an** et dans ce plot on ne tient pas en compte cela.

régressions par groupe

On peut bien se demander: est-ce que le coefficient de la régression a varié au fil du temps? autrement dit: peut-être dans les années 50 il y avait une forte corrélation entre le PIB et l'augmentation de l'espérance de vie, mais cette corrélation n'est plus là dans les années récentes.

Pour répondre à cela il faut faire une régression par an, et après regarder (plotter) les coefficients et leurs intervaux de confiance pour voir si l'effet est toujours bien vivant ou il s'affaiblit. De plus, le faire par continent aiderait. Peut-on faire cela?

Malheureusement, une approche simple et naive (groupez puis faites le lm) ne marche pas.

```
## on peut utiliser le pipe avec lm() si on utilise le point comme placeholder (marque-place)
gp %>%
  lm(lifeExp ~ gdpPercap, data = .)
```

```
Call:
lm(formula = lifeExp ~ gdpPercap, data = .)

Coefficients:
(Intercept)    gdpPercap
  5.396e+01    7.649e-04
```

```
## approche naïve: MARCHE PAS!
## parce que lm() ne connaît le concept de "groupe"
gp %>%
  group_by(year) %>%
  lm(lifeExp ~ gdpPercap, data = .)
```

```
Call:
lm(formula = lifeExp ~ gdpPercap, data = .)

Coefficients:
(Intercept)    gdpPercap
  5.396e+01    7.649e-04
```

```
## 2e approche naïve: avec summarise MARCHE PAS
## parce que summarise s'attend à trouver une VALEUR, pas un objet LM
#gp %>%
#  group_by(year) %>%
#  summarise(mean = lm(lifeExp ~ gdpPercap, data = .))
```

pourquoi? parce que `summarize` ne marche que si le output est une seule valeur; alors que `lm` retourne un objet complexe.

comment faire?

il faut passer par une fonction specialisé par groupe, `group_modify()`. Il s'agit d'une espèce de `summarise()`, parce que `group_modify()` applique une fonction à chaque groupe. Mais alors que `summarise()` prend en argument une variable et ne peut retourner qu'une valuer unique par groupe (par exemple, la moyenne, le max, le min, la numerosité), `group_modify()` prend en argument une base de données et retourne une base de données de sortie par groupe.

On a donc `variable -> summarise() -> valeur unique` et `grouped_df -> group_modify() -> df`

```
## un dernier truc qui marche pas
## parce que lm() ne retourne pas un data frame
#gp %>%
#  group_by(year) %>%
```

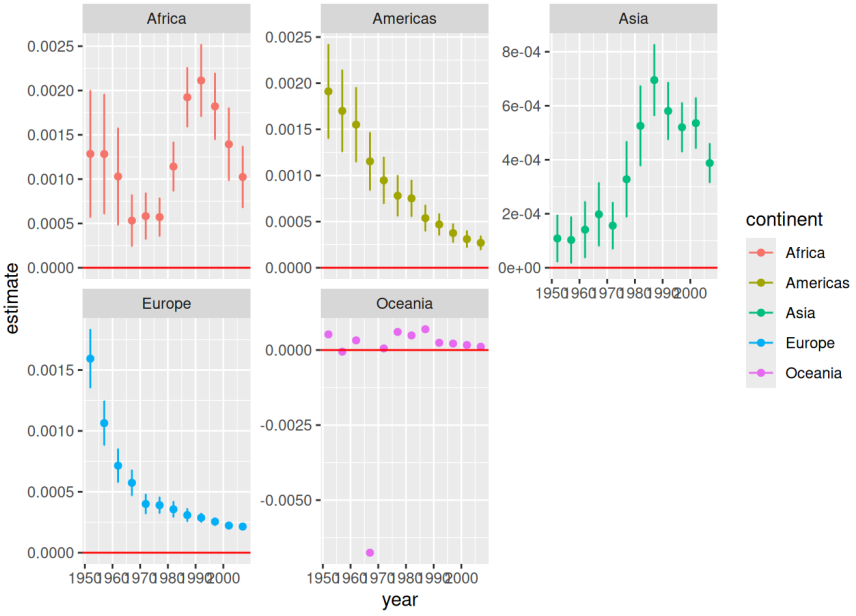
```
#  group_modify(~ lm(lifeExp ~ gdpPercap, data = .) )


## un truc qui marche ENFIN!!!
manyregs <- gp %>%
  group_by(year, continent) %>%
  group_modify(~ lm(lifeExp ~ gdpPercap, data = .) %>% tidy )


manyregs ## contient les résultats de 60 REGRESSIONS
```

```
# A tibble: 120 × 7
# Groups:   year, continent [60]
    year continent term         estimate   std.error statistic    p.value
   <int> <fct>     <chr>           <dbl>       <dbl>     <dbl>      <dbl>
 1  1952 Africa    (Intercept) 37.5        1.14          32.9    1.48e-35
 2  1952 Africa    gdpPercap    0.00128    0.000719       1.79   7.99e- 2
 3  1952 Americas  (Intercept) 45.5        2.57          17.7    6.76e-15
 4  1952 Americas  gdpPercap    0.00191    0.000511       3.74   1.07e- 3
 5  1952 Asia      (Intercept) 45.8        1.67          27.4    2.59e-23
 6  1952 Asia      gdpPercap    0.000108   0.0000874      1.24   2.24e- 1
 7  1952 Europe    (Intercept) 55.4        1.55          35.6    7.19e-25
 8  1952 Europe    gdpPercap    0.00159    0.000242       6.60   3.74e- 7
 9  1952 Oceania   (Intercept) 63.9      NaN           NaN    NaN
10  1952 Oceania   gdpPercap    0.000522 NaN           NaN    NaN
# i 110 more rows
```

Si on regarde l'objet obtenu on peut voir qu'on obtient un objet qui contient sur chaque ligne un coefficient de l'estimation par groupe – ici année/continent.

On peut maintenant avec aise plotter les résultats et voir si notre idée que la rélation entre PIB et espérance de vie s'estompe avec le temps est soutenue par les données.

```
manyregs %>%
  filter(term != "(Intercept)") %>%
  ggplot()+
  aes(x = year, y = estimate, color = continent)+
  geom_point()+
  geom_hline(yintercept = 0, color = "red")+
  geom_errorbar(aes(ymin = estimate - std.error,
                    ymax = estimate + std.error), width = .1)+
  facet_wrap(~continent, scales = "free_y")
```



rappel: un point dans ce plot **ne répresente pas les données** mais **une estimation de l'effet du PIB sur l'espérance de vie**. L'intuition que la relation soit moins forte au fil du temps tient bien la route pour l'Europe et les deux amériques; pas pour le reste du monde.

sommaire de la méthode

1. groupez le data frame
2. `group_modify()` : on applique une fonction complexe (comme `lm`) à chaque groupe.
3. utiliser `tidy()` pour que les résultats du modèle soient en format data.frame; si ce n'est pas le cas, cela ne marchera pas.

exercice

> faites une régression de `pop` sur `year` pour chaque continent – cela estime le taux de croissance moyen linéaire sur la période, par continent. Utilisez la méthode `group_modify()` décrite ci-dessus.

Corrélation

On peut suivre la même démarche pour des corrélations. Une corrélation indique la présence d'une rélation linéaire entre les données. On va à nouveau utiliser le jeu de données `gapminder` et tester la corrélation entre `gdpPercap` et `lifeExp`.

Base R

une corrélation se fait simplement avec `cor()`.

```
## correlation e base r
## cor()
cor(gp$lifeExp, gp$gdpPercap)
```

```
[1] 0.5837062
```

```
## même cor mais par année et par continent ## mais ça marche pas parce que cor() ne comprend pas les gr
gp %>%
  group_by(year, continent) %>%
  summarise(correlation = cor(.$lifeExp, .$gdpPercap))
```

```
`summarise()` has grouped output by 'year'. You can override using the
`.groups` argument.

# A tibble: 60 × 3
# Groups:   year [12]
    year continent correlation
   <int> <fct>           <dbl>
 1  1952 Africa          0.584
 2  1952 Americas        0.584
 3  1952 Asia            0.584
 4  1952 Europe          0.584
 5  1952 Oceania         0.584
 6  1957 Africa          0.584
 7  1957 Americas        0.584
 8  1957 Asia            0.584
 9  1957 Europe          0.584
10  1957 Oceania         0.584
# ℹ 50 more rows
```

par contre si on veut tester la significativité statistique de cette corrélation il faut utiliser `cor.test()`:

```
cor.test(gp$lifeExp, gp$gdpPercap) %>% tidy()
```

```
# A tibble: 1 × 8
  estimate statistic   p.value parameter conf.low conf.high method   alternative
     <dbl>     <dbl>     <dbl>     <int>    <dbl>     <dbl> <chr>    <chr>
1    0.584      29.7 3.57e-156      1702    0.552     0.614 Pearson… two.sided
```

notez que `cor()` retourne tout simplement une valeur; `cor_test()` retourne en revanche une liste, un objet complexe et avec plusieurs attributs, exactement comme `lm`.

Tidyverse, juste corrélation

La syntaxe du tidyverse est assez simple parce qu'on peut utiliser directement `summarise()`. Summarise marche avec toute fonction qui prend un vecteur et retourne un nombre. Mais on peut tout de même utiliser `group_modify` aussi.

par exemple, corrélation entre `lifeExp` et `gdpPercap` par continent

La corrélation est bcp plus forte pour Europe et Océanie.

tidyverse, corrélation et test

Il faut à nouveau passer par `group_modify()`. Il suffit de changer le `lm` par un `cor.test`:

```
gp %>%
  group_by(continent) %>%
  group_modify(~ cor.test(.$lifeExp, .$gdpPercap) %>% tidy)
```

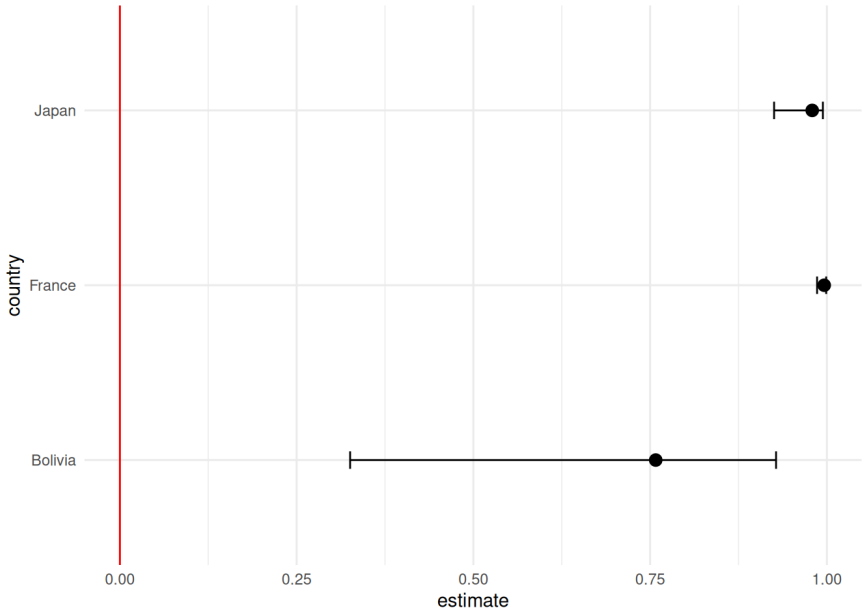
```
# A tibble: 5 × 9
# Groups:   continent [5]
  continent estimate statistic  p.value parameter conf.low conf.high method
  <fct>        <dbl>     <dbl>    <dbl>     <int>    <dbl>     <dbl> <chr>
1 Africa       0.426     11.7  7.60e-29       622    0.359     0.488 Pearson's …
2 Americas     0.558     11.6  5.45e-26       298    0.475     0.632 Pearson's …
3 Asia         0.382      8.21 3.29e-15       394    0.295     0.463 Pearson's …
4 Europe       0.781     23.6  4.05e-75       358    0.737     0.818 Pearson's …
5 Oceania      0.956     15.4  2.99e-13        22    0.901     0.981 Pearson's …
# ℹ 1 more variable: alternative <chr>
```

et voilà, on a les estimations (elles sont les mêmes que toute à l'heure) mais aussi les valuers p et les intervals de confiance.

Exercice

> calculez l'interval de confiance de la corrélation entre `lifeExp` et `gdpPercap` pour chaque pays. Plottez les résultats pour la **France**, le **Japon** et la **Bolivie**.

```
# stratégie de solution
# 1. grouper par country
# 2. appliquer group_modify et cor.test + tidy
# 3. filtrer les pays qu'il nous faut
# 4. fair e le plot des intervaux de confiance

gp %>%
  group_by(country) %>%
  filter(country %in% c("France", "Japan", "Bolivia")) %>%
  group_modify(~ cor.test(.$lifeExp, .$gdpPercap) %>% tidy) %>%
  ggplot()+
  aes(y = country, x = estimate)+
  geom_point(size = 3)+
  geom_errorbarh(aes(xmin = conf.low,
                     xmax = conf.high), height = .1)+
  geom_vline(xintercept = 0, color = "red")+
  #coord_cartesian(xlim = c(-0.1, 0.2))+
  theme_minimal()
```



t-test (et n'importe quel autre test)

Le test T à une variable sert à tester si la moyenne d'une variable est significativement différent d'une valeur de référence (par défaut: 0); à deux variables à tester si la moyenne d'une variable est statistiquement différente de l'autre.

On va tester une quesiton:

1. peut on dire que l'espérance de vie à la naissance est > 60 dans le monde?

Base R

```
t.test(gp$lifeExp, mu = 60)
```

```
    One Sample t-test

data:  gp$lifeExp
t = -1.6795, df = 1703, p-value = 0.09323
alternative hypothesis: true mean is not equal to 60
95 percent confidence interval:
 58.86070 60.08818
sample estimates:
mean of x
 59.47444
```

on ne peut pas rejeter l'hypothèse que l'espérance de vie soit inférieure à 60 – donc non, pris dans son ensemble l'espérance de vie dans le monde n'est pas supérieure à 60 ans (sur la période 1952-2012); la moyenne est juste inférieure à 60 et la différence avec 60 est significative.

Mais peut-être que cet âge de 60 a été dépassé à des temps différents dans des pays différents.

Si on ne voulait qu'afficher l'âge moyen par continent et année, il suffit un appel à `summarise`:

```r
## espérance de vie moyenne par continent / année
## il suffit un sumamrise()
gp %>%
  group_by(continent, year) %>%
  summarise(esperance = mean(lifeExp))
```

```
`summarise()` has grouped output by 'continent'. You can override using the
`.groups` argument.

# A tibble: 60 × 3
# Groups:   continent [5]
   continent  year esperance
   <fct>     <int>     <dbl>
 1 Africa     1952      39.1
 2 Africa     1957      41.3
 3 Africa     1962      43.3
 4 Africa     1967      45.3
 5 Africa     1972      47.5
 6 Africa     1977      49.6
 7 Africa     1982      51.6
 8 Africa     1987      53.3
 9 Africa     1992      53.6
10 Africa     1997      53.6
# ℹ 50 more rows
```

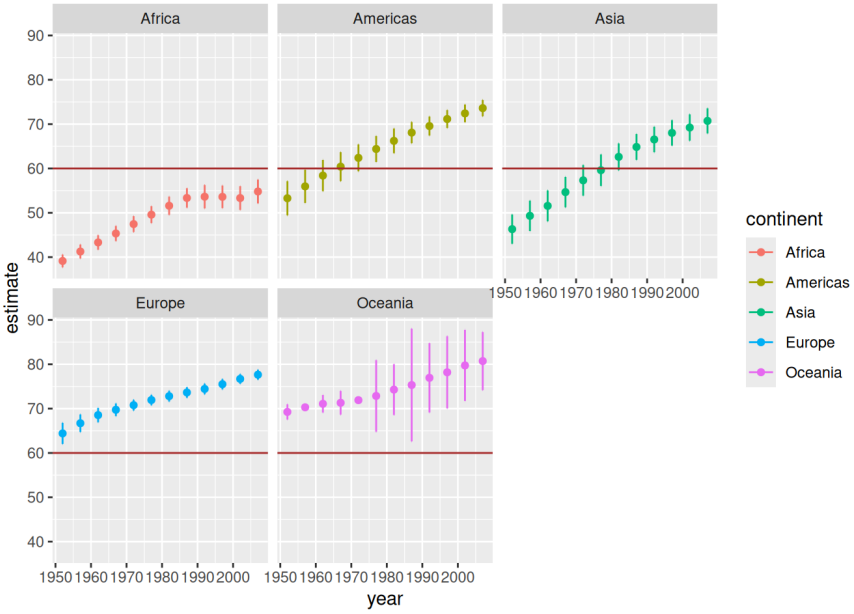
mais si on veut **tester** la différence par rapport à 60 avec un test T il faut passer par un `group_modify()`, comme on a vu pour le `lm` et la `cor` relation:

```r
## code dans une seule pipe

manytests <- gp %>%
  group_by(continent, year) %>%
  group_modify(~ t.test(.$lifeExp, mu = 60) %>%  tidy)

manytests
```

```
# A tibble: 60 × 10
# Groups:   continent, year [60]
   continent  year estimate statistic  p.value parameter conf.low conf.high
   <fct>     <int>    <dbl>     <dbl>    <dbl>     <dbl>    <dbl>     <dbl>
 1 Africa     1952     39.1    -29.2  1.66e-33        51     37.7      40.6
 2 Africa     1957     41.3    -24.0  1.76e-29        51     39.7      42.8
 3 Africa     1962     43.3    -20.5  2.96e-26        51     41.7      45.0
 4 Africa     1967     45.3    -17.4  4.41e-23        51     43.6      47.0
 5 Africa     1972     47.5    -14.1  3.11e-19        51     45.7      49.2
 6 Africa     1977     49.6    -11.0  4.02e-15        51     47.7      51.5
 7 Africa     1982     51.6     -8.22 6.61e-11        51     49.5      53.6
 8 Africa     1987     53.3     -6.10 1.42e- 7        51     51.2      55.5
 9 Africa     1992     53.6     -4.86 1.18e- 5        51     51.0      56.3
10 Africa     1997     53.6     -5.07 5.58e- 6        51     51.1      56.1
# ℹ 50 more rows
# ℹ 2 more variables: method <chr>, alternative <chr>
```

et avec un plot on obtient la réponse:

```r
  manytests %>%
    ggplot()+
    aes(x = year, y = estimate, color = continent)+
    geom_point()+
    geom_errorbar(aes(ymin = conf.low, ymax = conf.high), width = .1)+
    geom_hline(yintercept = 60, color = "brown")+
    facet_wrap(~continent)
```



réponse: pour l'Afrique , l'espérance de vie à la naissance n'a jamais été supérieure à 60 (les valeurs p sont toujours 1, ce qui indique qu'on ne peut pas rejeter l'hypothèse que la moyenne est inférieure à 60); pour Europe et Océanie, la moyenne a toujours été supérieure; pour Amérique et Asie, la transtition par 60 s'est effectuée pendant la periode d'observation.

exercice sur le t.test

> est-ce que le gdp percapita est inférieur à 2000€/an par continent par année?

```
# stratégie

# 1. group by
# 2. t.test (group_modify, tidy)
## t.test(var, mu = K, alternative = "greater"/"smaller")
# 3. plot

exo_test <- gp %>%
  group_by(year, continent) %>%
  group_modify(~ t.test(.$gdpPercap, mu = 2000) %>% tidy)

## plot
exo_test %>%
  ggplot()+
    aes(x = year, y = estimate, color = continent)+
    geom_point()+
    geom_errorbar(aes(ymin = conf.low, ymax = conf.high), width = .1)+
    geom_hline(yintercept = 2000, color = "brown")+
    facet_wrap(~continent, scales = "free_y")
```