Le modèle linéaire

Eric Marcon

31 January 2024

Régression linéaire simple

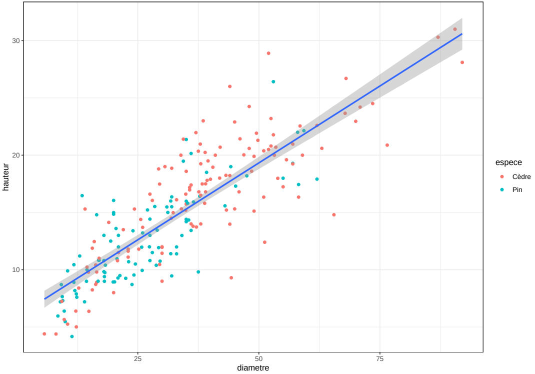
Ventoux

Données du projet de dendrométrie 2020, Mont Ventoux.

```
read_csv2("data/Inv_GEEFT_Ventoux_09-2020.csv") |>
  rename(
    espece = Espèce,
    diametre = `Diamètre (cm)`,
    hauteur = `Hauteur réelle (m)`
  ) |>
  mutate(
    espece = case_match(
      espece,
      "P" ~ "Pin",
      "C" ~ "Cèdre"
    )
  ) -> ventoux
```

Graphique hauteur ~ diamètre


```
ventoux |>
  ggplot(aes(x = diametre, y = hauteur)) +
  geom_point(aes(col = espece)) +
  geom_smooth(method = "lm")
```



Théorie

Modèle linéaire simple :

$$Y = \beta_0 + \beta_1 X + \mathrm{E}$$

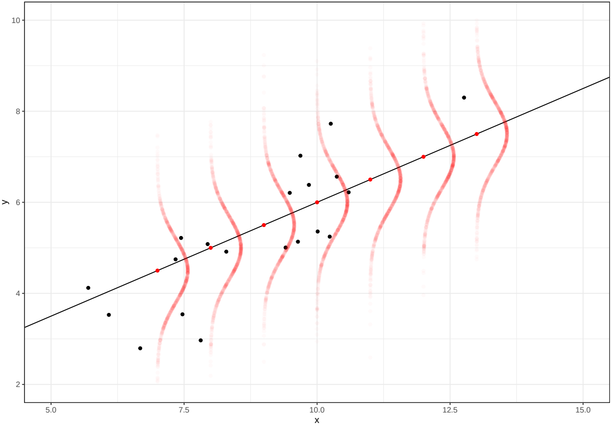
Y et X sont des vecteurs : $Y = \{y_i\}$ est l'ensemble des observations. Par abus d'écriture, Y est aussi la variable aléatoire dont les y_i sont des réalisations.

Vocabulaire : variable expliquée, exogène, coefficients, constante (intercept)...

$\mathrm{E} = \{\epsilon_i\}$ est l'erreur du modèle. $\mathrm{E} \sim \mathcal{N}(0, \sigma^2)$

Représentation

Le modèle prédit une densité de probabilité des valeurs de Y pour toute valeur de X distribuée normalement autour de la droite de régression.



Hypothèses

- Indépendance des erreurs : $\mathrm{Cov}(\epsilon_i, \epsilon_j) = 0$. Assurée par le design expérimental.
- Exogénéité : X n'est pas corrélé à E.
- Homoscédasticité : la variance de l'erreur est constante sur l'étendue de X.
- Normalité des termes d'erreur : $\mathrm{E} \sim \mathcal{N}(0, \sigma^2)$.

Exemple

Générer les données du modèle.

Coefficients :

```
beta_0 <- 1
beta_1 <- 0.5
sigma <- 1
```

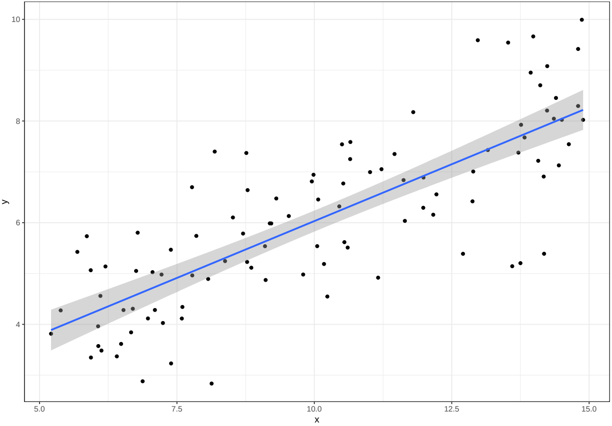
Tirage :

```
n <- 100
x <- runif(n, min = 5, max = 15)
# Jeu de points
mod_l1 <- tibble(x, y = rnorm(n, mean = beta_0 + beta_1*x, sd = sigma))
```

Estimation

Commencer par une figure.

```
mod_l1 |>
  ggplot(aes(x = x, y = y)) + geom_point() + geom_smooth(method = lm)
```



Estimation

La fonction `lm()` du package *stats* estime le modèle et permet de tester les hypothèses.

```
mod_l1_lm <- lm(y ~ x, data = mod_l1)
```

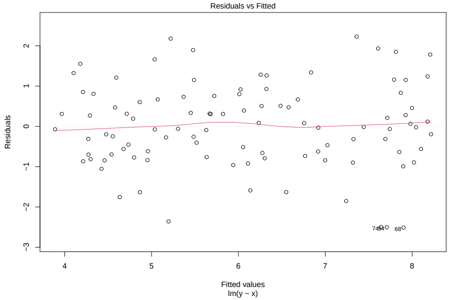
Syntaxe de la formule :

- variable expliquée à gauche, covariables à droite de `~`
- constante implicite `y ~ x` est identique à `y ~ 1 + x` alors que `y ~ 0 + x` force la constante à 0.
- possibilité de transformer les variables : `log(y) ~ I(x^2)` (Attention : `log(y) ~ x^2` est interprété comme l'interaction de `x` avec lui-même, c'est-à-dire `x`)

Homoscédasticité et indépendance des erreurs

Graphique $\mathrm{E} \sim Y*$

```
plot(mod_l1_lm, which = 1)
```

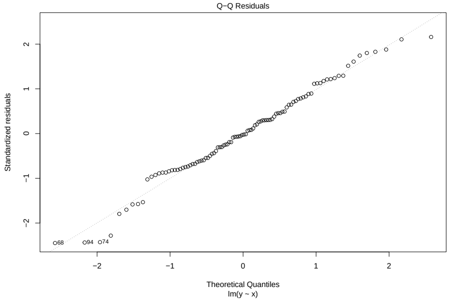


Les erreurs doivent être centrée sur 0 et uniformément réparties.

Normalité des erreurs

Graphique quantile - quantile (?qqplot)

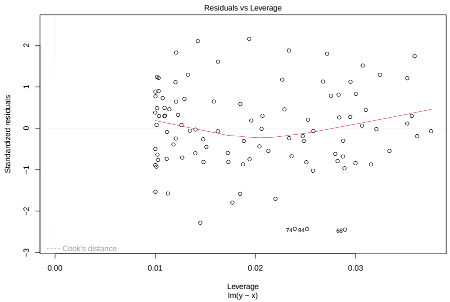
```
plot(mod_l1_lm, which = 2)
```



La non-normalité des résidus implique la non-normalité des estimateurs des coefficients.

Effet de levier


```
plot(mod_l1_lm, which = 5)
```



Les points avec fort effet de levier forte erreur ($\rightarrow$ grande distance de Cook) posent problème.

Rectification des données

Affaire d'expérience.

- Éliminer les points (réellement) aberrants ;
- Transformer Y si :
  - la relation n'est pas linéaire (ex.: quadratique) ;
  - l'erreur augmente avec $Y*$ ($\rightarrow$ racine carrée ou logarithme).
- Revoir les hypothèses à l'origine du modèle, le design expérimental...

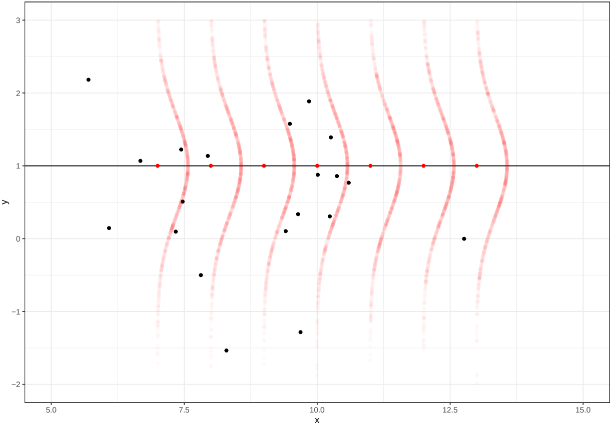
Interprétation des résultats : summary

```
##
## Call:
## lm(formula = y ~ x, data = mod_l1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.51229 -0.69782 -0.02673  0.68502  2.22763
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.55838    0.37138   4.196 5.97e-05
## x            0.44729    0.03525  12.689  < 2e-16
##
## (Intercept) ***
## x           ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.042 on 98 degrees of freedom
## Multiple R-squared:  0.6216, Adjusted R-squared:  0.6178
## F-statistic:   161 on 1 and 98 DF,  p-value: < 2.2e-16
```

Statistique F

La statistique F décrit la probabilité que le modèle n'explique rien.

Modèle nul: $Y = \bar{Y} = \beta_0$



R^2

R^2 mesure la proportion de la variance de Y expliquée par le modèle :

$$R^2 = \frac{\mathrm{Var}(Y^\star)}{\mathrm{Var}(Y)} = 1 - \frac{\sigma}{\mathrm{Var}(Y)}$$

→ Que devient R^2 en doublant σ ? Estimer rapidement puis re-simuler le modèle pour vérifier.

R^2 ajusté pénalise le R^2 par le nombre de paramètres du modèle.

Les degrés de liberté sont le nombre d'observations moins le nombre de paramètres moins 1.

Estimation des coefficients

Les coefficients sont estimés par la méthode des moindres carrés : minimisation des écarts

$$\sum (y_i - y_i^\star)^2$$

.

Résultat identique à la maximisation de la vraisemblance

$$\prod f(\epsilon_i)$$

où $f(\dot{)}$ est la densité de $\mathcal{N}(0, \sigma^2)$.

Estimation des coefficients

L'estimateur de chaque coefficient est sa valeur la plus probable.

L'estimateur est distribué normalement (quand E est normal) :

$$\hat{\beta}_1 \sim \mathcal{N}(0.447, 0.0352^2)$$

Un test de Student donne la probabilité de se tromper en affirmant que l'estimateur n'est pas nul.

Synthèse 1/2

Un bon modèle a un grand R^2 et des petites p-values.

- R^2 diminue avec la variance de l'erreur ;
- L'écart-type des estimateur diminue comme $\sqrt{n}$.

Mais les deux dépendent du design expérimental.

Design expérimental

Quadrupler l'effort d'échantillonnage divise par deux l'intervalle de confiance

```
mod_l1x4 <- tibble(
  x = rnorm(n * 4, mean = 10, sd = 2), # x est calculé avant y
  y = rnorm(n * 4, mean = beta_0 + beta_1 * x, sd =sigma) # y utilise x
)
mod_l1x4_lm <- lm(y ~ x, data = mod_l1x4)
summary(mod_l1x4_lm)$coefficients
```

```
##              Estimate Std. Error   t value
## (Intercept) 1.8435413 0.25344512  7.273927
## x           0.4226806 0.02443802 17.296027
##                 Pr(>|t|)
## (Intercept) 1.870813e-12
## x           2.180247e-50
```

Choix économique.

Design expérimental

Retirer les valeurs intermédiaires de X augmente le R^2 (*design factoriel*) alors que σ ne change pas.

```
mod_l1x4 |>
  filter(x < 6 | x >14) %>% # pas |> pour "data = ."
  lm(y ~ x, data = .) |>
  summary() |>
  pluck("r.squared")
```

```
## [1] 0.8153628
```

contre 0.4291063 avec toutes les données.

Design expérimental

Le R^2 d'un modèle avec des données individuelles est plus faible qu'avec des données agrégées.

→ Estimer le modèle hauteur ~ diamètre des données Ventoux.

→ Regrouper les données par espèce.

→ Estimer le modèle à nouveau.

Synthèse 2/2

Considérer R^2 et p-values en fonction du modèle :

- beaucoup de données individuelles $\rightarrow$ faible R^2 mais petites p-values pour montrer l'influence d'un facteur ;
- possibilité d'un très grand R^2 sans aucun coefficient significatif si peu de points ;
- un grand R^2 et des petites p-values permettent de faire des prédictions.

Prédictions

`predict()` permet d'extrapoler le modèle.

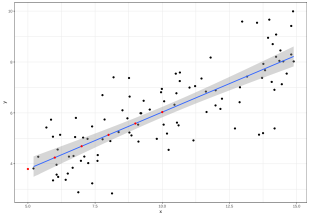
```
mod_l1_lm |> predict(newdata = data.frame(x = 5:10))
```

```
##        1        2        3        4        5
## 3.794805 4.242090 4.689375 5.136661 5.583946
##        6
## 6.031231
```

Prédictions

Ajout des points sur la figure :

```
# Estimation du modèle
mod_l1 |>
  ggplot(aes(x, y)) + geom_point() + geom_smooth(method = lm) ->
  mod_l1_ggplot
# Choix des x pour lesquels y est à prédire
mod_l1_predict <- data.frame(x = 5:10)
# Ajout des prédictions
mod_l1_predict$y <- predict(mod_l1_lm, newdata = mod_l1_predict)
# Ajout des points à la figure précédente
mod_l1_ggplot +
  geom_point(data = mod_l1_predict, aes(x = x, y = y), col = "red")
```

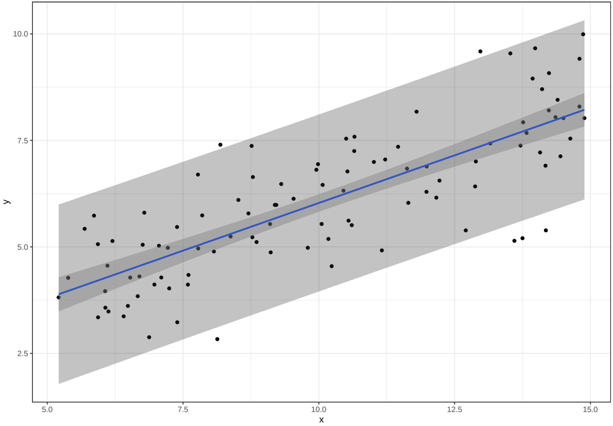


Intervalles de confiance et de prédiction

La zone grisée de `geom_smooth` est l'intervalle de confiance de l'espérance de $Y|X$, c'est-à-dire de la moyenne des prédictions. Il est bien plus étroit que l'intervalle de prédiction, qui correspond à 95% des prédictions :

```
mod_l1_predict <- data.frame(
  x = seq(from = min(mod_l1$x), to = max(mod_l1$x), length.out = 50)
)
mod_l1_predict <- cbind(
  mod_l1_predict,
  predict(
    mod_l1_lm,
    newdata = mod_l1_predict,
    interval = "prediction"
  )
)
mod_l1_ggplot +
  geom_ribbon(
    data = mod_l1_predict,
    aes(y = fit, ymin = lwr, ymax = upr),
    alpha = 0.3
  ) -> mod_l1_ggplot_predict
```

Intervalles de confiance et de prédiction



Régression linéaire multiple

Théorie

Modèle linéaire multiple :

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \mathrm{E}$$

Y et X_j sont des vecteurs : $X_1 = \{x_{i,1}\}$ est l'ensemble des valeurs du premier prédicteur (= variable explicative, variable exogène ou covariable).

Représentation

Multidimensionnelle donc plus difficile.

Le dimension de Y est égale au nombre de covariables moins 1: le modèle linéaire réduit la dimension des données.

Ajout d'un coefficient à l'exemple précédent :

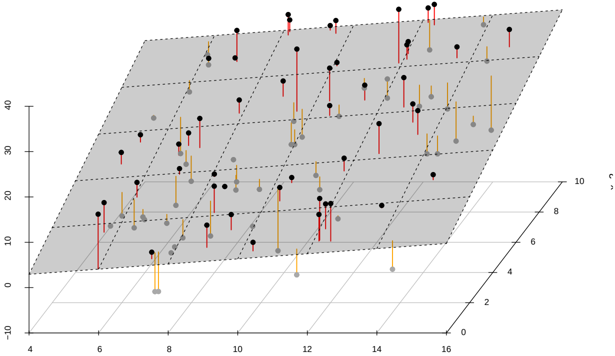
```
beta_0 <- 1
beta_1 <- 0.5
beta_2 <- 2
sigma <- 5
```

Tirage :

```
n <- 100
x_1 <- runif(n, min = 5, max = 15)
x_2 <- runif(n, min = 0, max = 10)
# Jeu de points
mod_l2 <- tibble(
  x_1, x_2,
  y = rnorm(n, mean = beta_0 + beta_1*x_1 + beta_2*x_2, sd = sigma)
)
```

Représentation

```
mod_l2_lm  <- lm(y ~ x_1 + x_2, data = mod_l2)
```



Hypothèses

En plus des précédentes :

- Non colinéarité des covariables.

Si une des covariables est une combinaison linéaire des autres, le modèle ne peut pas être estimé.

En pratique, les covariables doivent être aussi peu corrélées que possible.

Interactions

On peut tester l'effet de l'interaction de deux variables :

```
lm(y ~ x_1 + x_2 + x_1*x_2, data = mod_l2) |> summary()
```

```
##
## Call:
## lm(formula = y ~ x_1 + x_2 + x_1 * x_2, data = mod_l2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13.6196  -3.4068   0.0407   3.2955  13.6722
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.96537    3.76069  -0.257  0.79796
## x_1          0.73680    0.38248   1.926  0.05701
## x_2          2.14510    0.65736   3.263  0.00153
## x_1:x_2     -0.03291    0.06440  -0.511  0.61049
##
## (Intercept)
## x_1         .
## x_2         **
## x_1:x_2
## ---
```

Ancova

Théorie

Modèle de régression multiple avec des covariables catégorielles, codées sous forme d'indicatrices (autant d'indicatrices que de modalités - 1).

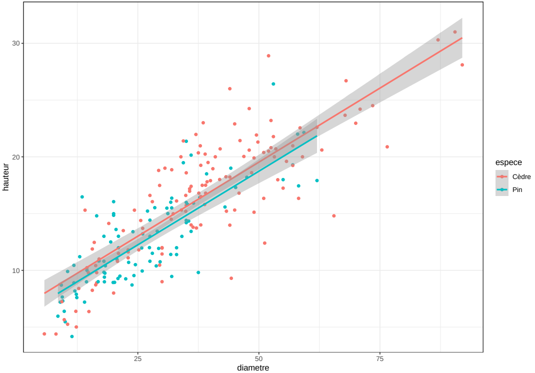
$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \mathrm{E}$$

Exemple du Ventoux :

- Y est la hauteur des arbres ;
- X_1 est leur diamètre ;
- L'espèce est codée par une variable indicatrice, par exemple $X_2 = \mathbb{1}("Cedre")$.

Exemple

```r
ventoux |>
  ggplot(aes(x = diametre, y = hauteur, color = espece)) +
  geom_point() +
  geom_smooth(method = "lm")
```



Estimation

La figure représente *deux régressions séparées* : les pentes pourraient être différentes. Une Ancova est donc appropriée.

`lm` crée automatiquement des indicatrices pour les variables catégorielles.

```
(ventoux_lm <- lm(hauteur ~ diametre + espece, data = ventoux))
```

```
##
## Call:
## lm(formula = hauteur ~ diametre + espece, data = ventoux)
##
## Coefficients:
## (Intercept)     diametre    especePin
##      6.5018       0.2605      -0.7696
```

Ici, l'indicatrice vaut 1 pour les pins, 0 pour les cèdres.

Représentation

La figure doit être construite manuellement

```
ventoux |>
  bind_cols(predict(ventoux_lm, interval = "confidence")) |>
  ggplot(aes(x = diametre, color = espece)) +
    geom_point(aes(y = hauteur)) +
    geom_line(aes(y = fit)) +
    geom_ribbon(aes(y = fit, ymin = lwr, ymax = upr),  alpha = 0.3)
```