Régression linéaire simple

Timothée Flutre

11/02/2015

Contents

1 Préambule

Ce document a été généré à partir d'un fichier texte au format Rmd utilisé avec le logiciel libre R. Pour exporter un tel fichier vers les formats HTML et PDF, installez le paquet rmarkdown disponible sur CRAN (il va vraisemblablement vous être demandé d'installer d'autres paquets), puis ouvrez R et entrez:

```
library(rmarkdown)
render("myanalysis.Rmd", "all")
```

Il est généralement plus simple d'utiliser le logiciel libre RStudio, mais ce n'est pas obligatoire. Pour plus de détails, lisez cette page. Pour écrire des équations avec LaTeX, reportez-vous au livre en ligne.

2 Contexte

Ce document fait partie de l'atelier "Prédiction Génomique" organisé et animé par Jacques David et Timothée Flutre en 2015 à Montpellier SupAgro dans le cadre de l'option APIMET (Amélioration des Plantes et Ingénierie végétale Méditerranéennes et Tropicales) couplée à la spécialité SEPMET (Semences Et Plants Méditerranéens Et Tropicaux) du Master 3A (Agronomie et Agroalimentaire).

Ce document a pour but d'introduire concrètement les étudiants à l'un des aspects de la modélisation statistique, la simulation. Il prend comme exemple la régression linéaire simple, historiquement développée par Galton (1886) pour étudier l'hérédité de la taille dans l'espèce humaine.

3 Introduction

3.1 A propos de la modélisation

"Essentially, all models are wrong, but some are useful." (Box, 1987). Cette célèbre citation illustre parfaitement le fait qu'un modèle est une simplification du phénomène étudié, mais qu'après tout, si cette simplification nous apporte des enseignements et nous permet de prendre de bonnes décisions, cela importe tout autant.

Il est donc important de rappeler que la première question à se poser, en tant que modélisateur, concerne la validité du modèle. Bien que cela paraisse évident, ceci consiste avant tout à vérifier que les données à analyser correspondent bien à la question à laquelle on veut répondre (Gelman & Hill, 2006).

Il est également utile, pour mieux comprendre le processus de modélisation statistique, de distinguer le "monde réel", dans lequel vivent les données, du "monde théorique", dans lequel vivent les modèles: "When we use a statistical model to make a statistical inference, we implicitly assert that the variation exhibited by data is captured reasonably well by the statistical model, so that the theoretical world corresponds reasonably well to the real world." (Kass, 2011).

En particulier, il ne faut pas confondre les données avec des variables aléatoires, même si on fait souvent le raccourci: "In both approaches [frequentist and Bayesian], a statistical model is introduced and we may say that the inference is based on what *would* happen if the data *were* to be random variables distributed according to the statistical model. This modeling assumption would be reasonable if the model *were* to describe accurately the variation in the data." (Kass, 2011).

3.2 Notations et vocabulaire

L'inférence avec un modèle statistique consiste généralement à *estimer* les paramètres, puis à s'en servir pour *prédire* de nouvelles données.

Lorsque l'on propose un modèle pour répondre à une question donnée, on commence donc par expliquer les notations. Suivant les conventions, nous utilisons des lettres grecques pour dénoter les paramètres (non-observés), par exemple θ, des lettres romaines pour dénoter les données observées, y, et un tilde pour les données prédites, $\tilde{y}$. L'ensemble des paramètres est généralement noté en majuscule, Θ. De plus, s'il y a plusieurs paramètres ou données, on utilise des vecteurs notés en gras, ce qui donne $\boldsymbol{\theta}$ et $\boldsymbol{y}$.

Une fois les notations établies, on écrit la vraisemblance (*likelihood*), souvent présentée comme étant la "probabilité des données sachant les paramètres". En fait, si les données sont des variables continues, c'est la densité de probabilité des données sachant les paramètres, notée $p(y|\theta)$, et si les données sont des variables discrètes, c'est la fonction de masse, notée $P(y|\theta)$. Mais le plus important est de réaliser que, dans la vraisemblance, ce ne sont pas les données qui varient, mais les paramètres: la vraisemblance est une fonction des paramètres, d'où le fait qu'on la note $\mathcal{L}(\theta)$.

Tout naturellement, la méthode du maximum de vraisemblance cherche donc à identifier la valeur du paramètre, notée $\hat{\theta}$, qui maximise la vraisemblance.

$$\hat{\theta} = \text{argmax}_\theta \, \mathcal{L} \quad \Leftrightarrow \quad \frac{\partial \mathcal{L}}{\partial \theta}(\hat{\theta}) = 0$$

3.3 Comprendre par l'exemple

Le vocabulaire esquissé au paragraphe précédent n'est pas forcément très intuitif... Par exemple, supposons que l'on étudie une quantité physique dont la valeur résulte de la somme d'une très grande quantité de facteurs indépendants, chacun ayant un faible impact sur la valeur finale. Prenons trois mesures de cette quantité d'intérêt. Comme il y a de la variation, on choisit d'introduire une *variable aléatoire* Y correspondant à la quantité d'intérêt, et dénotons par y_1, y_2 et y_3 les trois observations, vues comme des réalisations de la variable aléatoire Y.

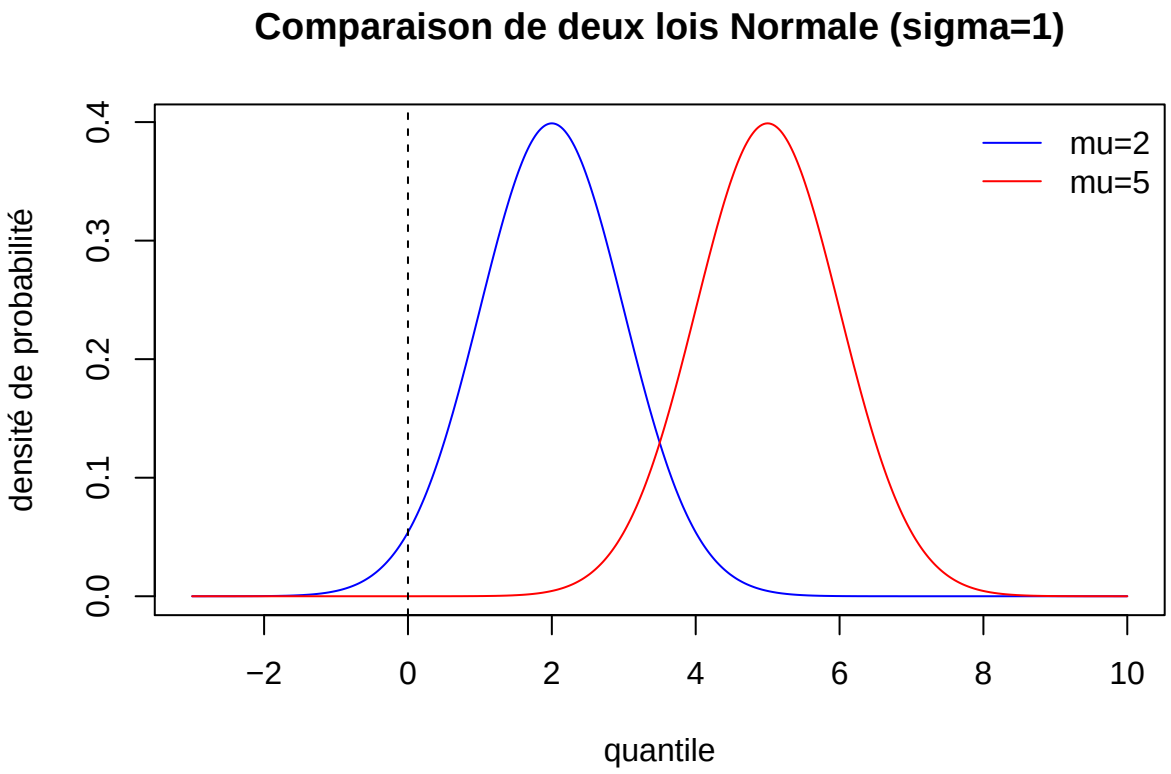
Par exemple, supposons les valeurs suivantes:

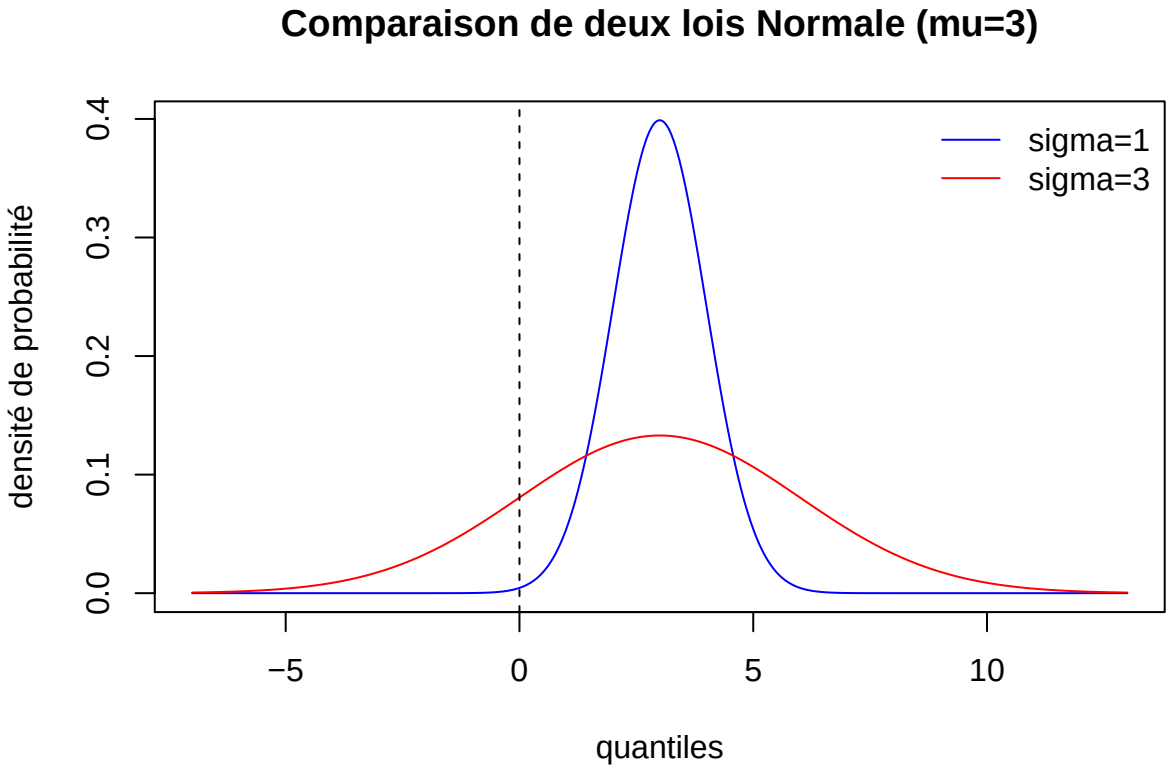
- y_1=4.374
- y_2=5.184
- y_3=4.164

Etant donné les caractéristiques du phénomène ("somme d'une très grande quantité de facteurs indépendants, chacun ayant un faible impact"), il est raisonnable de supposer que la variable Y suit une *loi Normale* (*c.f.* théorème central limite). Cette distribution de probabilité est caractérisée par deux paramètres, sa *moyenne* que l'on note généralement μ, et sa *variance* que l'on note généralement σ^2 (σ étant l'écart-type). En terme de notation, on écrit $Y \sim \mathcal{N}(\mu, \sigma^2)$, et la densité de probabilité de la réalisation y de Y s'écrit:

$$p(y|\mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(y-\mu)^2}{2\sigma^2}\right)$$

L'intérêt de ce modèle paramétrique est de pouvoir *résumer* les données, par exemple un million de mesures, par seulement deux valeurs, les paramètres. Mais bien entendu, nous ne connaissons pas les valeurs de paramètres! La moyenne μ peut prendre toutes les valeurs entre $-\infty$ et $+\infty$, et la variance σ^2 n'a pour seul restriction que d'être positive. Or comme le montrent les graphiques ci-dessous, la loi Normale peut être assez différente selon les valeurs de ces paramètres:

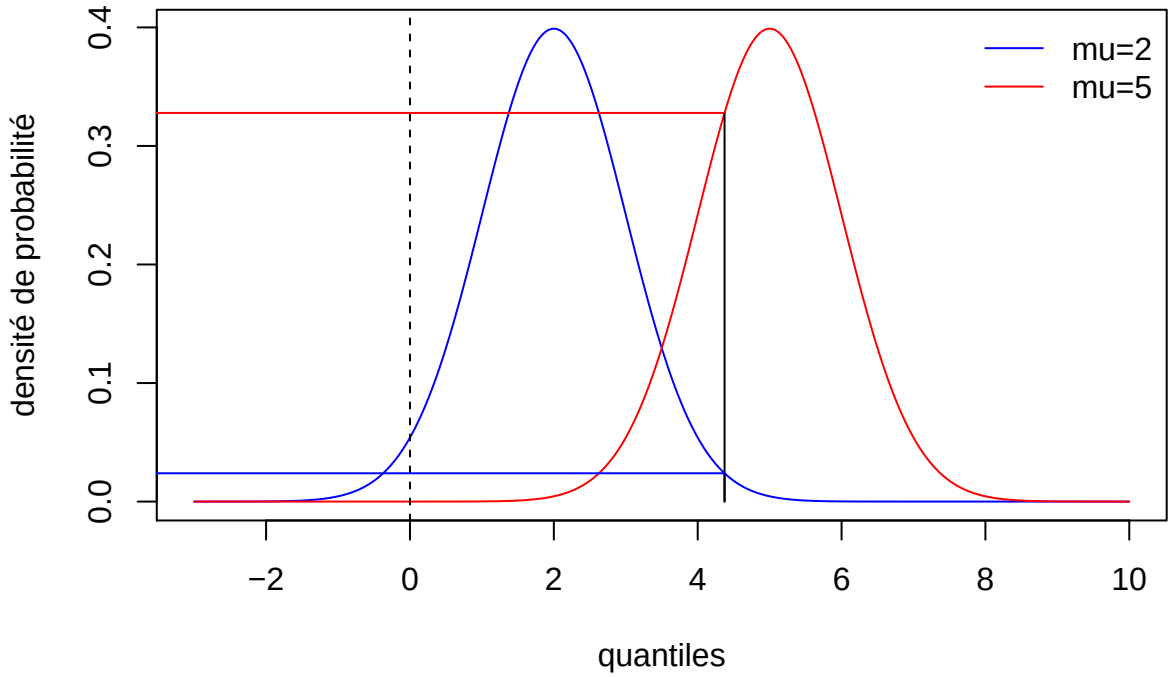




Revenons à nos trois mesures: 4.374, 5.184, 4.164. Parmi toutes les valeurs possibles des paramètres, le défi consiste donc à trouver celles pour lesquelles la loi Normale est une bonne description du mécanisme qui a généré nos données. On parle donc de trouver les valeurs des paramètres de telle sorte que la *vraisemblance* d'obtenir ces données soit la plus élevée possible pour ces valeurs des paramètres.

Pour rendre les choses plus facile, supposons que l'on sache déjà que la variance vaut 1. Il ne nous reste plus

qu'à trouver la moyenne. Regardons cela de plus près pour la première observation, y_1=4.374:



D'après le graphique, la densité de probabilité $p(y_1|\mu = 5, \sigma = 1)$ est strictement supérieur à $p(y_1|\mu = 2, \sigma = 1)$. Cela se vérifie si l'on fait le calcul avec la formule ci-dessus: $p(y_1|\mu = 5, \sigma = 1)$=0.328 et $p(y_1|\mu = 2, \sigma = 1)$=0.024. Comme les deux densité ont la même valeur pour σ, la différence vient bien du terme $(y - \mu)^2$ dans l'exponentielle, qui représente l'écart à la moyenne. Au final, nous pouvons donc dire qu'en ce qui concerne la première observation, la vraisemblance $\mathcal{L}(\mu = 5, \sigma = 1)$ est plus grande que $\mathcal{L}(\mu = 2, \sigma = 1)$.

Comme l'on dispose de plusieurs observations et que l'on suppose qu'elles sont toutes des réalisations de la même variable aléatoire, il est pertinent de calculer la vraisemblance de toutes ces observations conjointement plutôt que séparément:

$$\mathcal{L}(\mu, \sigma) = p(y_1, y_2, y_3|\mu, \sigma)$$

Si l'on fait aussi l'hypothèse que les observations sont indépendantes, cela se simplifie en:

$$\begin{aligned}\mathcal{L}(\mu, \sigma) &= p(y_1|\mu, \sigma) \times p(y_2|\mu, \sigma) \times p(y_3|\mu, \sigma) \\ &= \prod_{i=1}^{3} p(y_i|\mu, \sigma)\end{aligned}$$

Il n'est pas très pratique de maximiser la vraisemblance directement, on préfère donc passer au log (qui est monotone, donc le maximum de l'un est le maximum de l'autre):

$$
\begin{aligned}
l(\mu, \sigma) &= \log \mathcal{L}(\mu, \sigma) \\
&= \sum_{i=1}^{3} \log p(y_i|\mu, \sigma) \\
&= \sum_{i=1}^{3} \log \left[\frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(y_i - \mu)^2}{2\sigma^2} \right) \right] \\
&= -3 \log \sigma \; - \frac{3}{2} \log(2\pi) \; - \sum_{i=1}^{3} \frac{(y_i - \mu)^2}{2\sigma^2}
\end{aligned}
$$

Lorsque le nombre d'observations augmente, un simple examen graphique n'est pas très pratique ni suffisant. D'où le fait, qu'en pratique, (i) l'on écrive une fonction qui calcule la log-vraisemblance, et (ii) l'on cherche le maximum de cette fonction:

```
compute.log.likelihood <- function(parameters, data){
  mu <- parameters[1]
  sigma <- parameters[2]
  y <- data
  n <- length(y)
  log.lik <- - n * log(sigma) - (n/2) * log(2 * pi) - sum(((y - mu)^2) / (2 * sigma^2))
  return(log.lik)
}

compute.log.likelihood(c(5,1), y)
```

```
## [1] -3.32
```

```
compute.log.likelihood(c(2,1), y)
```

```
## [1] -13
```

Dans le cas de la loi Normale, il existe déjà dans R des fonctions implémentant la densité de probablité, ce qui nous permet de vérifier que nous n'avons pas fait d'erreur:

```
sum(dnorm(x=y, mean=5, sd=1, log=TRUE))
```

```
## [1] -3.32
```

```
sum(dnorm(x=y, mean=2, sd=1, log=TRUE))
```

```
## [1] -13
```

Cette section devrait vous avoir donné les bases du raisonnement ainsi que des techniques que nous allons utiliser dans la suite de l'atelier, commençant dès maintenant avec la régression linéaire simple.

4 Ecrire le modèle

4.1 Notations

- n: nombre d'individus (diploïdes, supposés non-apparentés)
- i: indice indiquant le i-ème individu, donc $i \in \{1, \ldots, n\}$
- y_i: phénotype de l'individu i pour la caractère d'intérêt
- μ: moyenne globale du phénotype des n individus
- f: fréquence de l'allèle minoritaire au marqueur SNP d'intérêt (situé sur un autosome)
- x_i: génotype de l'individu i à ce SNP, codé comme le nombre de copie(s) de l'allèle minoritaire à ce SNP chez cet individu ($\forall i\ x_i \in \{0, 1, 2\}$)
- β: effet additif de chaque copie de l'allèle minoritaire en unité du phénotype
- ϵ_i: erreur pour l'individu i
- σ^2: variance des erreurs
- $\mathcal{N}(\mu, \sigma^2)$: distribution Normale univariée de moyenne μ et variance σ^2

4.2 Vraisemblance

Dans notre cas, nous supposons que le génotype au SNP d'intérêt a un effet additif sur la moyenne du phénotype, ce qui s'écrit généralement:

$$\forall i \quad y_i = \mu + \beta x_i + \epsilon_i \text{ avec } \epsilon_i \overset{\text{i.i.d}}{\sim} \mathcal{N}(0, \sigma^2)$$

Une autre façon, mais équivalente, de l'écrire est:

$$\forall i \quad y_i | x_i, \mu, \beta, \sigma \overset{\text{i.i.d}}{\sim} \mathcal{N}(\mu + \beta x_i, \sigma^2)$$

A partir de cela, il devient maintenant facile de *simuler des données*: si vous choisissez des valeurs pour les x_i, ainsi que pour les paramètres, vous pouvez facilement générer des valeurs pour les y_i.

De plus, on peut écrire la vraisemblance sous forme plus explicite de fonction des paramètres $\Theta = \{\mu, \beta, \sigma\}$:

$$\begin{aligned}
\mathcal{L}(\Theta) &= p(\boldsymbol{y}|\boldsymbol{x}, \Theta) = p(y_1, \ldots, y_n | x_1, \ldots, x_n, \mu, \beta, \sigma) \\
&= \prod_{i=1}^{n} p(y_i | x_i, \mu, \beta, \sigma) \\
&= \prod_{i=1}^{n} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(y_i - \mu - \beta x_i)^2}{2\sigma^2}\right)
\end{aligned}$$

Pour trouver les valeurs des paramètres qui maximisent la vraisemblance, on travaille généralement avec la log-vraisemblance, $l(\Theta) = \log \mathcal{L}(\Theta)$, puis on utilise les règles d'analyse pour calculer $\frac{\partial l}{\partial \beta}$, etc.

5 Simuler des données

Afin de simuler des données, nous allons utiliser un générateur de nombres pseudo-aléatoires. Le mot "pseudo" est là pour rappeler que les générateurs informatiques sont déterministes et peuvent donc être initialisés avec une graine (*seed*), très utile pour la reproducibilité des analyses:

```
seed <- 1859
set.seed(seed)
```

Commençons par fixer le nombre d'individus:

```
n <- 500
```

Puis la moyenne globale (de manière arbitraire, ce n'est pas très important car on peut toujours centrer les phénotypes en début d'analyse):

```
mu <- 50
```

Pour simuler les génotypes, nous allons supposer que la population est à l'équilibre d'Hardy-Weinberg:

```
##' Calculate the genotype frequencies (AA, Aa, aa) at Hardy-Weinberg equilibrium.
##'
##' https://en.wikipedia.org/wiki/Hardy%E2%80%93Weinberg_principle
##' @param maf frequency of the minor allele, a
##' @return vector of genotype frequencies
##' @author Timothée Flutre
calc.geno.freq <- function(maf){
  geno.freq <- c((1 - maf)^2,
                 2 * (1 - maf) * maf,
                 maf^2)
  names(geno.freq) <- c("AA", "Aa", "aa")
  return(geno.freq)
}

f <- 0.3
genotypes <- sample(x=c(0,1,2), size=n, replace=TRUE, prob=calc.geno.freq(f))
```

Le morceau de code ci-dessus vous montre aussi les bonnes pratiques de programmation:

- choisir des noms de fonctions et variables clairs et explicites;
- documenter le code;
- citer des référence si nécessaire.

Regardons à quoi ressemblent les génotypes que nous venons de simuler:

```
table(genotypes)
```

```
## genotypes
##   0   1   2
## 267 192  41
```

```r
sum(genotypes) / (2 * n)
```

```
## [1] 0.274
```

```r
var(genotypes)
```

```
## [1] 0.413
```

Tirons une valeur pour l'effet du génotype sur le phénotype:

```r
(beta <- rnorm(n=1, mean=1, sd=1))
```

```
## [1] 1.51
```

Simulons des erreurs (par simplicité, fixons σ à 1):

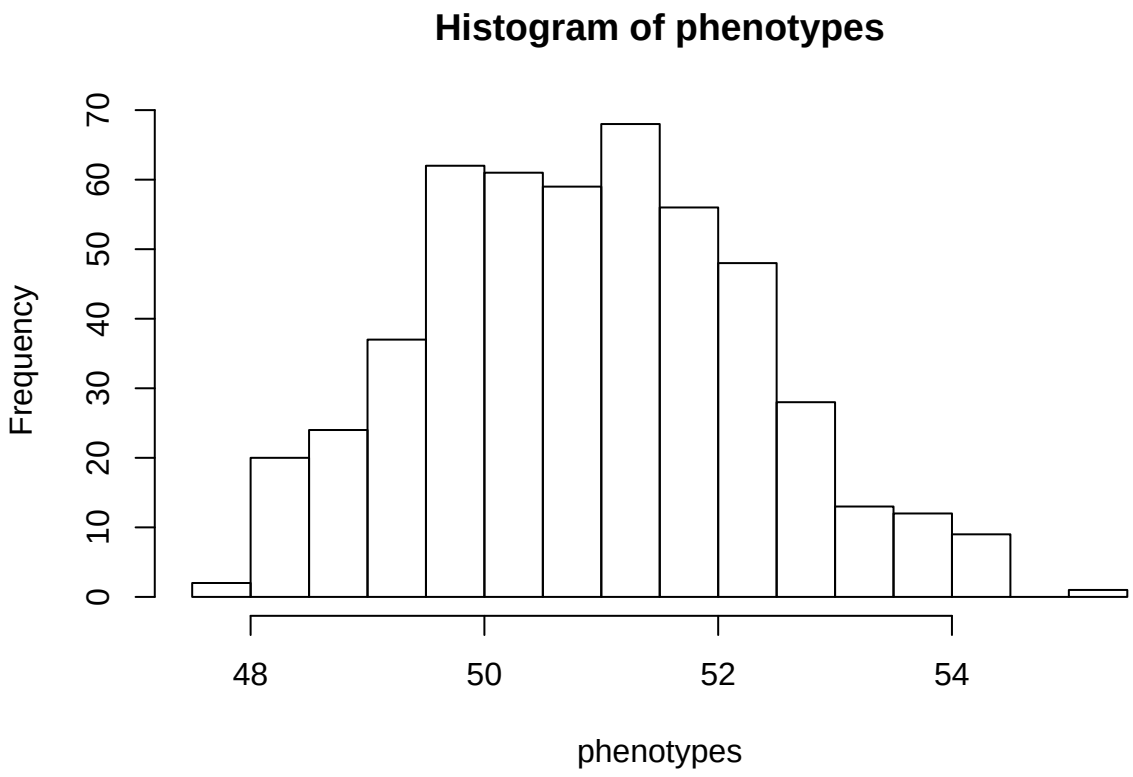
```r
sigma <- 1
errors <- rnorm(n=n, mean=0, sd=sigma)
```

Nous avons maintenant tout ce qu'il faut pour simuler les phénotypes:

```r
phenotypes <- mu + beta * genotypes + errors
```

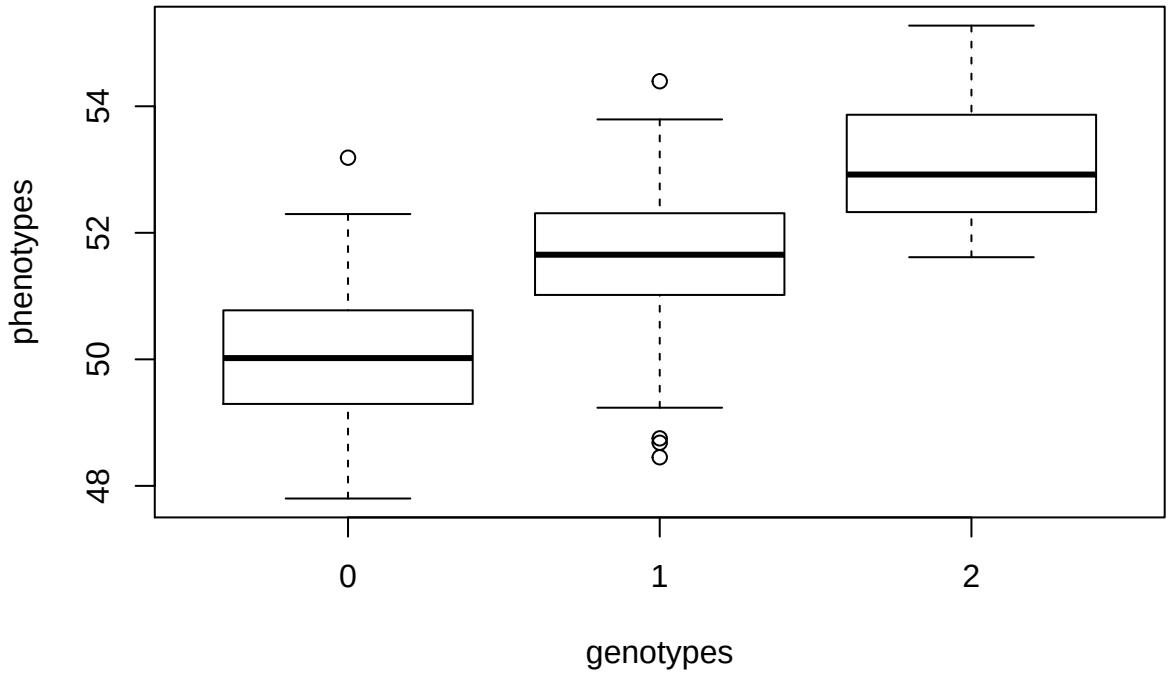
Regardons à quoi ils ressemblent:

```r
hist(phenotypes, breaks="FD")
```



Comme ce qui nous intéresse ici, ce ne sont pas uniquement les phénotypes, mais bien la relation entre les génotypes et les phénotypes, un autre type de graphique semble plus approprié:

```
boxplot(phenotypes ~ genotypes, xlab="genotypes", ylab="phenotypes")
```



Pour la suite, il est habituel dans R d'organiser les données dans un tableau:

```
dat <- data.frame(x=genotypes, y=phenotypes)
summary(dat)
```

```
##        x               y        
##  Min.   :0.000   Min.   :47.8  
##  1st Qu.:0.000   1st Qu.:49.8  
##  Median :0.000   Median :50.8  
##  Mean   :0.548   Mean   :50.9  
##  3rd Qu.:1.000   3rd Qu.:51.8  
##  Max.   :2.000   Max.   :55.3
```

6 Réaliser l'inférence

6.1 Dérivation mathématique

TODO

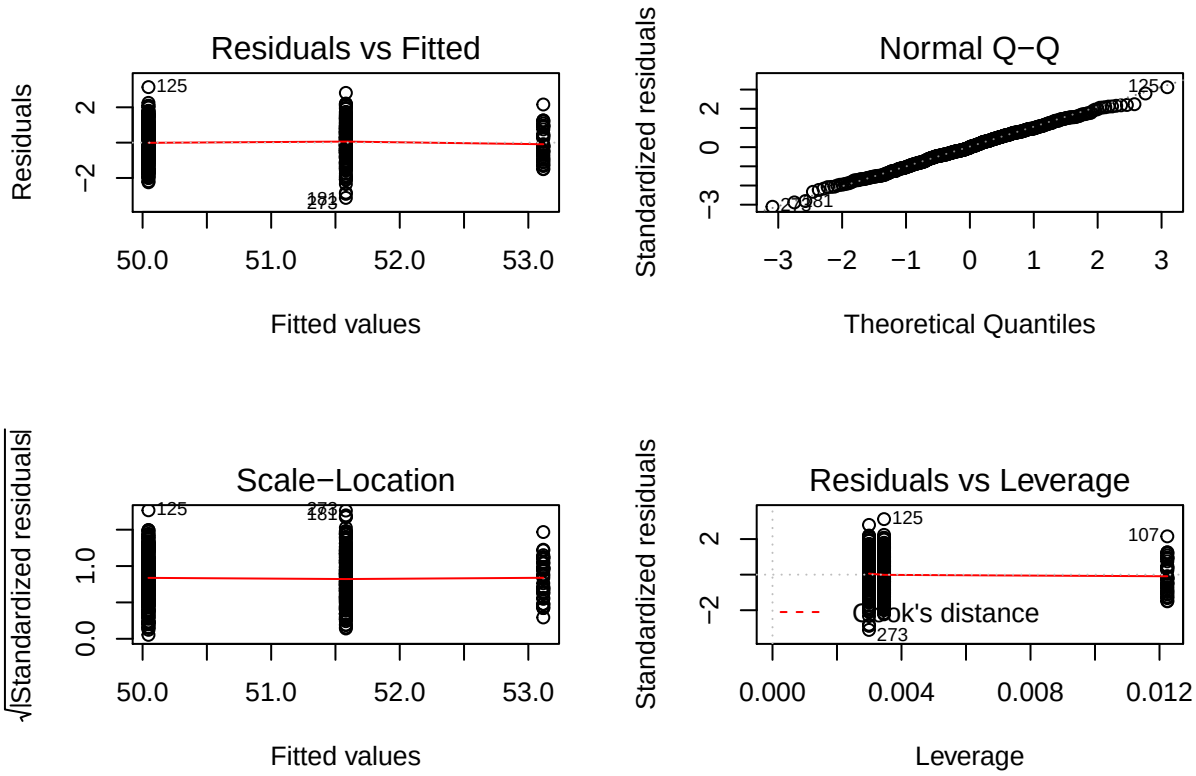
6.2 Implémentation (R stats)

Bien entendu, R a nativement une fonction implémentant l'estimation par maximum de vraisemblance d'un modèle linéaire:

```
fit <- lm(y ~ x, data=dat)
```

Avant toute autre chose, il nous faut vérifier que les hypothèses du modèles sont vérifiées (homoscédasticité, Normalité, indépendence):

```
par(mfrow=c(2,2))
plot(fit)
```



Ceci semble être bien le cas (évidemment puisque nous avons simulé nous-même les données...).

Nous pouvons donc extraire de l'objet les quantités qui nous intéresse et les comparer aux vraies valeurs utilisées pour simuler les données:

```
summary(fit)
```

```
##
## Call:
## lm(formula = y ~ x, data = dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.1284 -0.6718 -0.0062  0.7347  3.1419
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  50.0458     0.0593   844.1   <2e-16 ***
## x             1.5352     0.0703    21.9   <2e-16 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.01 on 498 degrees of freedom
## Multiple R-squared:  0.489,  Adjusted R-squared:  0.488
## F-statistic:  477 on 1 and 498 DF,  p-value: <2e-16
```

```
c(mu, beta, sigma)
```

```
## [1] 50.00  1.51  1.00
```

```
(mu.hat <- coefficients(fit)[1])
```

```
## (Intercept)
##          50
```

```
(beta.hat <- coefficients(fit)[2])
```

```
##    x
## 1.54
```

```
(sigma.hat <- sqrt((1/(n-2) * sum(fit$residuals^2))))
```

```
## [1] 1.01
```

parler du R2

6.3 Implémentation (R base)

Il peut être intéressant, surtout dans ce cas simple, d'implémenter cette méthode par soi-même, histoire de mieux la comprendre. Pour cela, il faut d'abord écrire une fonction R à qui on donne les données et les valeurs de paramètres et qui renvoie la valeur de la log-vraisemblance. Puis on peut utiliser la fonction mle du paquet stats4.

TODO

7 Explorer les simulations possibles

La simulation est un outil particulièrement utile pour explorer comment un modèle répond à des changements dans les données.

Maintenant, c'est à vous: que voudriez-vous explorer en premier?

8 Perspectives

Naturellement, l'activité de modélisation statistique ne se limite pas à simuler des données sur ordinateur. Bien au contraire, elle est au coeur de l'activité de recherche en ce qu'elle vise à identifier les caractéristiques saillantes d'un phénomène naturel afin d'en réaliser l'inférence.

Concernant le thème de l'atelier, la prédiction génomique, quelles sont les limites du modèle exploré ci-dessus? Que proposez-vous pour y remédier?

9 Annexe

```
print(sessionInfo(), locale=FALSE)
```

```
## R version 3.1.2 (2014-10-31)
## Platform: x86_64-pc-linux-gnu (64-bit)
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] MASS_7.3-37     knitr_1.9       rmarkdown_0.4.2
##
## loaded via a namespace (and not attached):
## [1] digest_0.6.8    evaluate_0.5.5  formatR_1.0     htmltools_0.2.6
## [5] stringr_0.6.2   tools_3.1.2     yaml_2.1.13
```