

Prédiction génomique

Timothée Flutre

26/04/2015

Contents

1	Contexte	1
2	Introduction	2
3	Simuler des génotypes et des phénotypes	3
3.1	Modèle	3
3.2	Simulation	4
3.3	Inférence	7
3.3.1	En fréquentiste	7
3.3.2	En bayésien	8
3.3.3	Comparaison	13
3.4	Critiques	13
4	Simuler des phénotypes à partir de vrais génotypes	13
5	Prédire grâce au génome	15
6	Explorer des jeux de données réels librement disponibles	15
7	Perspectives	15
8	Références	16
9	Annexe	16

1 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'explorer par simulation l'intérêt de la prédiction génomique en sélection artificielle pour la création variétale. Il est recommandé d'avoir déjà lu et suivi le document “Premiers pas” de l'atelier.

2 Introduction

Modèle standard de la génétique quantitative (voir les références en fin de document):

$$y_i = g_i + \epsilon_i \quad (1)$$

- i : indice du i ème individu parmi les N qui composent l'échantillon ($i \in \{1, \dots, N\}$)
- y_i : valeur phénotypique de l'individu i , considérée comme continue
- g_i : valeur génotypique de l'individu i (peut être interprétée comme la valeur phénotypique moyenne de l'individu s'il est cloné dans tous les environnements possibles)
- ϵ_i : composante non-génétique pour l'individu i ("déviation environnementale")

Quel que soit l'individu, si l'on suppose que ses valeur génotypique et composante non-génétique ne sont pas corrélées, alors sa variance phénotypique est égale à $\sigma_p^2 = \sigma_g^2 + \sigma_\epsilon^2$. A ce stade, l'héritabilité au sens large est définie comme étant $H^2 = \frac{\sigma_g^2}{\sigma_g^2 + \sigma_\epsilon^2}$.

La valeur génotypique peut également se décomposer en composantes additive, de dominance et d'épistasie: $g_i = a_i + d_i + \zeta_i$. On suppose généralement aussi que ces composantes ne sont pas corrélées, et donc $\sigma_g^2 = \sigma_a^2 + \sigma_d^2 + \sigma_\zeta^2$. Ceci amène à définir l'héritabilité au sens strict par $h^2 = \frac{\sigma_a^2}{\sigma_g^2 + \sigma_\epsilon^2}$.

Le même modèle, mais en notation matricielle:

$$\mathbf{y} = \mathbf{g} + \boldsymbol{\epsilon} \quad (2)$$

- G : matrice de variance-covariance $N \times N$ des valeurs génotypiques;
- R : matrice de variance-covariance $N \times N$ des composantes non-génétiques.

La matrice G , dite "d'apparentement", se décompose aussi en relations additives, de dominance et d'épistasie, même si les premières sont généralement les seules utilisées en pratique. Dans ce cas, $G = \sigma_a^2 A$ où σ_a^2 est estimé et A , la matrice des relations génétiques additives, est calculée à partir de l'arbre généalogique (pédigrée) des individus. De plus, la matrice R est généralement diagonale, telle que $R = \sigma_\epsilon^2 I$ où σ_ϵ^2 est estimé simultanément à σ_a^2 , et I est la matrice identité.

Si l'on suppose que $\mathbf{g} \sim \mathcal{N}_N(\mathbf{0}, G)$ et $\boldsymbol{\epsilon} \sim \mathcal{N}_N(\mathbf{0}, R)$, alors $\hat{\mathbf{g}} = E[\mathbf{g}|\mathbf{y}] = G(G + R)^{-1}\mathbf{y}$ où $\hat{\mathbf{g}}$ est le meilleur prédicteur linéaire sans biais de \mathbf{g} (Best Linear Unbiased Predictor, BLUP) et $H = G(G + R)^{-1}$ est une généralisation matricielle de l'héritabilité.

Or il faut bien remarquer que la généalogie ne permet que de calculer la matrice d'apparentement *attendue*, celle-ci pouvant donc différer de la matrice d'apparentement *réalisée*. En effet, bien qu'en moyenne le coefficient d'apparentement (identité par descendance) entre un allèle d'un parent et un allèle de son enfant soit de 1/4, cette proportion varie le long du génome, à cause, entre autres, de l'échantillonage mendélien des chromosomes et de la variation du taux de recombinaison le long des chromosomes. De plus, la généalogie seule ne permet pas d'identifier quelles régions du génomes ont une variation génotypique plus ou moins associée à la variation phénotypique, les fameux locus influençant les traits quantitatifs (Quantitative Trait Locus, QTL).

Si maintenant l'on dispose des génotypes $\{\mathbf{x}_i\}$ à un ensemble de P marqueurs génétiques, le modèle devient:

$$y_i = g(\mathbf{x}_i, \boldsymbol{\beta}) + \epsilon_i \quad (3)$$

où l'erreur est différente du modèle précédent, mais gardons la même notation par simplicité.

On peut n'utiliser les marqueurs que pour estimer G (généralement uniquement A en pratique) plus précisément, mais on peut aussi inclure directement les marqueurs comme variables explicatives dans le modèle. Les marqueurs sont souvent des SNP dont le génotype est codé en terme de dosage allélique ($x_{ij} \in \{0, 1, 2\}$). Comme précédemment, seuls les effets additifs sont généralement pris en compte et la valeur génotypique s'écrit alors $g_i \approx g(\mathbf{x}_i, \boldsymbol{\beta}) = \sum_{j=1}^P x_{ij}\beta_j$.

Avec le toujours plus grand débit des technologies des séquençage, il est très fréquent qu'il y ait beaucoup plus de marqueurs que d'individus: $P >> N$. Dans ce tels cas, la méthode traditionnelle du maximum de vraisemblance présentée dans le document “Premiers pas” ne donne plus de bonnes estimations des effets des marqueurs, les $\{\beta_j\}$, et doit être “pénalisée” (on dit aussi “régularisée”). On parle encore de “rétrécir” les estimations des effets (shrinkage en anglais). Dans l'approche bayésienne, que $P >> N$ ou non, c'est simplement le prior sur les $\{\beta_j\}$ qui change.

Explicitement incorporer les effets de dominance, et surtout d'épistasie, semble infaisable étant donné l'explosion combinatoire qui en résulte. Certains auteurs ont donc proposé des modèles semi-paramétriques (espace de Hilbert à noyau reproduisant, RKHS en anglais; réseaux neuronaux).

Quoi qu'il en soit, une abondance d'articles existe sur ces sujets (voir les revues listées en fin de document) et, pour se familiariser avec ces questions à moindre coût, rien de mieux que de faire des simulations!

3 Simuler des génotypes et des phénotypes

3.1 Modèle

L'un des modèles le plus utilisé est la régression d'arête (ridge regression en anglais):

$$\mathbf{y} = \mathbf{1}\mu + \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon} \quad \text{avec } \boldsymbol{\beta}, \boldsymbol{\epsilon} | \sigma_\beta^2, \sigma_\epsilon^2 \sim \mathcal{N}_N(\mathbf{0}, \sigma_\beta^2 I) \mathcal{N}_N(\mathbf{0}, \sigma_\epsilon^2 I)$$

où $Var(\mathbf{y}) = \sigma_\beta^2 \mathbf{X} \mathbf{X}' + \sigma_\epsilon^2 I$.

Comme vous pouvez le voir, ce modèle presuppose que le caractère d'intérêt a une architecture génétique où tous les marqueurs ont un effet, que donc, vraisemblablement, ces effets sont faibles, et qu'ils s'additionnent les uns aux autres. On parle de “modèle additif infinitésimal”.

Il est d'ailleurs possible de montrer qu'il est équivalent au modèle dit “animal” de l'ère pré-marqueurs:

$$\mathbf{y} = \mathbf{1}\mu' + \mathbf{Z}\mathbf{u} + \boldsymbol{\epsilon}' \quad \text{avec } \mathbf{u}, \boldsymbol{\epsilon}' | \sigma_a^2, \sigma_{\epsilon'}^2 \sim \mathcal{N}_N(\mathbf{0}, \sigma_a^2 A) \mathcal{N}_N(\mathbf{0}, \sigma_{\epsilon'}^2 I)$$

où Z n'est qu'une matrice d'incidence (ici l'identité), le vecteur \mathbf{u} contient les valeurs génotypiques additives, aussi appelées valeurs en croisement (breeding values en anglais), et $Var(\mathbf{y}) = \sigma_a^2 A + \sigma_{\epsilon'}^2 I$. Avec quelques hypothèses et en supposant un nombre infini de marqueurs, la matrice A est égale à $\mathbf{X} \mathbf{X}' / [2 \sum_{j=1}^P p_j(1 - p_j)]$ où p_j est la fréquence de l'allèle minoritaire au SNP j .

Dans cette section, choisissons donc le modèle de régression d'arête avec les notations suivantes:

- N : nombre d'individus ($i \in \{1, \dots, N\}$);
- \mathbf{y} : vecteur de longueur N des valeurs phénotypiques;
- μ : valeur phénotypique moyenne;
- W : matrice de dimension $N \times Q$ d'incidence de covariables;
- $\boldsymbol{\alpha}$: vecteur de longueur Q des effets de ces covariables (modélisés comme “fixes” dans la terminologie traditionnelle);

- X : matrice de dimension $N \times P$ de codage des génotypes aux marqueurs;
- β : vecteur de longueur P des effets de ces génotypes (modélisés comme “aléatoires” dans la terminologie traditionnelle);
- ϵ : vecteur de longueur N des erreurs;
- σ_β^2 : composant de la variance contribué par β ;
- σ_ϵ^2 : composant de la variance contribué par ϵ .

La vraisemblance s'écrit généralement comme suit:

$$\mathbf{y} = \mathbf{1}\mu + W\boldsymbol{\alpha} + X\beta + \epsilon$$

avec, dans l'approche fréquentiste, les distributions suivantes pour les effets aléatoires:

- $\beta \sim \mathcal{N}_N(\mathbf{0}, \sigma_\beta^2 I)$;
- $\epsilon \sim \mathcal{N}_N(\mathbf{0}, \sigma_\epsilon^2 I)$;
- $Cov(\beta, \epsilon) = 0$.

Dans l'approche bayésienne, on l'écrira généralement de la façon suivante:

$$\mathbf{y} | \mu, \boldsymbol{\alpha}, \beta, \sigma_\beta^2, \sigma_\epsilon^2 \sim \mathcal{N}_N(\mathbf{1}\mu + W\boldsymbol{\alpha} + X\beta, \sigma_\epsilon^2 I)$$

avec comme paramètres $\Theta = \{\mu, \boldsymbol{\alpha}, \beta, \sigma_\beta^2, \sigma_\epsilon^2\}$. Plusieurs priors sont possibles, mais choisissons la simplicité avec $p(\Theta) = p(\mu)p(\boldsymbol{\alpha})p(\beta|\sigma_\beta^2)p(\epsilon|\sigma_\epsilon^2)p(\sigma_\beta^2)p(\sigma_\epsilon^2)$ où:

- $\mu \sim \mathcal{N}_N(\mathbf{0}, 10^{10}I)$, un prior “plat”(flat en anglais);
- $\boldsymbol{\alpha} \sim \mathcal{N}_N(\mathbf{0}, 10^{10}I)$, idem;
- $\beta | \sigma_\beta^2 \sim \mathcal{N}_N(\mathbf{0}, \sigma_\beta^2 I)$;
- $\epsilon | \sigma_\epsilon^2 \sim \mathcal{N}_N(\mathbf{0}, \sigma_\epsilon^2 I)$;
- $\sigma_\beta^2 \sim \chi^{-2}(\nu = 5, \tau^2 = 1)$;
- $\sigma_\epsilon^2 \sim \chi^{-2}(\nu = 5, \tau^2 = 1)$.

TODO: faire des graphiques explorant les distributions *a priori*

3.2 Simulation

```
seed <- 1859; set.seed(seed)
N <- 1000
inds.id <- sprintf(fmt=paste0("ind%0", floor(log10(N))+1, "i"), 1:N)
mu <- 54
Q <- 1
W <- matrix(data=rnorm(N*Q, mean=12, sd=2), nrow=N, ncol=Q,
             dimnames=list(inds.id, paste0("fix", 1:Q)))
(alpha <- rnorm(n=Q, mean=5, sd=1))

## [1] 5.92
```

```

P <- 5000
snps.id <- sprintf(fmt=paste0("snp%0", floor(log10(P))+1, "i"), 1:P)

##' Calculate the genotype frequencies (AA, Aa, aa) at Hardy-Weinberg equilibrium.
##'
##' https://en.wikipedia.org/wiki/Hardy%2880%93Weinberg_principle
##' @param maf frequency of the minor allele, a
##' @return vector of genotype frequencies
calc.geno.freq <- function(maf=0.3){
  geno.freq <- c((1 - maf)^2,
                  2 * (1 - maf) * maf,
                  maf^2)
  names(geno.freq) <- c("AA", "Aa", "aa")
  return(geno.freq)
}

X <- matrix(sample(x=c(0,1,2), size=N*P, replace=TRUE, prob=calc.geno.freq()),
             nrow=N, ncol=P,
             dimnames=list(ind.s.id, snps.id))
table(X)

```

```

## X
##      0      1      2
## 2449348 2100291 450361

```

```

A <- A.mat(X-1)
A[1:3,1:3]

```

```

##           ind0001 ind0002 ind0003
## ind0001  0.9808 -0.0406 -0.0131
## ind0002 -0.0406  0.9935 -0.0255
## ind0003 -0.0131 -0.0255  1.0204

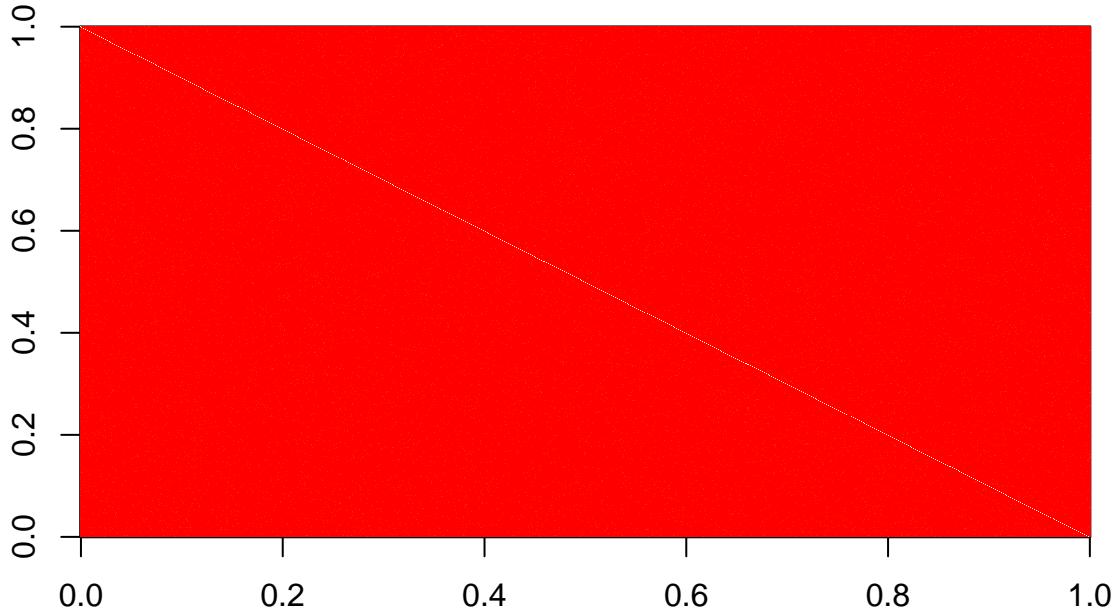
```

```

##' Plot a matrix as an image in the right orientation.
##'
##' @param M matrix to plot
##' @return none
image.mat <- function(M, ...){
  image(t(M)[,nrow(M):1], ...)
}

image.mat(A)

```



```

beta <- rep(0, P)
nb.qtls <- P # all markers have an effect
sigma.beta2 <- 1
beta[sample.int(P, nb.qtls)] <- rnorm(n=nb.qtls, mean=0, sd=sigma.beta2)

##' Estimate the minor allele frequency of a SNP whose genotypes are coded in allele dose.
##'
##' @param x vector of {0,1,2}
##' @return numeric
estim.maf.1SNP <- function(x){
  stopifnot(is.vector(x))
  N <- length(x)
  f.hat <- sum(x) / (2 * N)
  if(f.hat > 0.5)
    f.hat <- 1 - f.hat
  return(f.hat)
}

##' Estimate the minor allele frequencies of many SNPs whose genotypes are coded in allele dose.
##'
##' @param X matrix with individuals in rows and SNPs in columns
##' @return vector of numeric
estim.maf.allSNPs <- function(X){
  stopifnot(is.matrix(X))
  P <- ncol(X)
  out <- rep(NA, P)
  names(out) <- colnames(X)
  for(j in 1:P)
    out[j] <- estim.maf.1SNP(X[,j])
  return(out)
}

mafs <- estim.maf.allSNPs(X)
(sigma.a2 <- var(X %*% beta)[1,1]) # var of additive relationships

```

```

## [1] 2014

(sigma.beta2 * 2 * sum(mafs * (1 - mafs)))

## [1] 2099

h2 <- 0.60
(sigma.epsilon2 <- ((1 - h2) / h2) * sigma.a2)

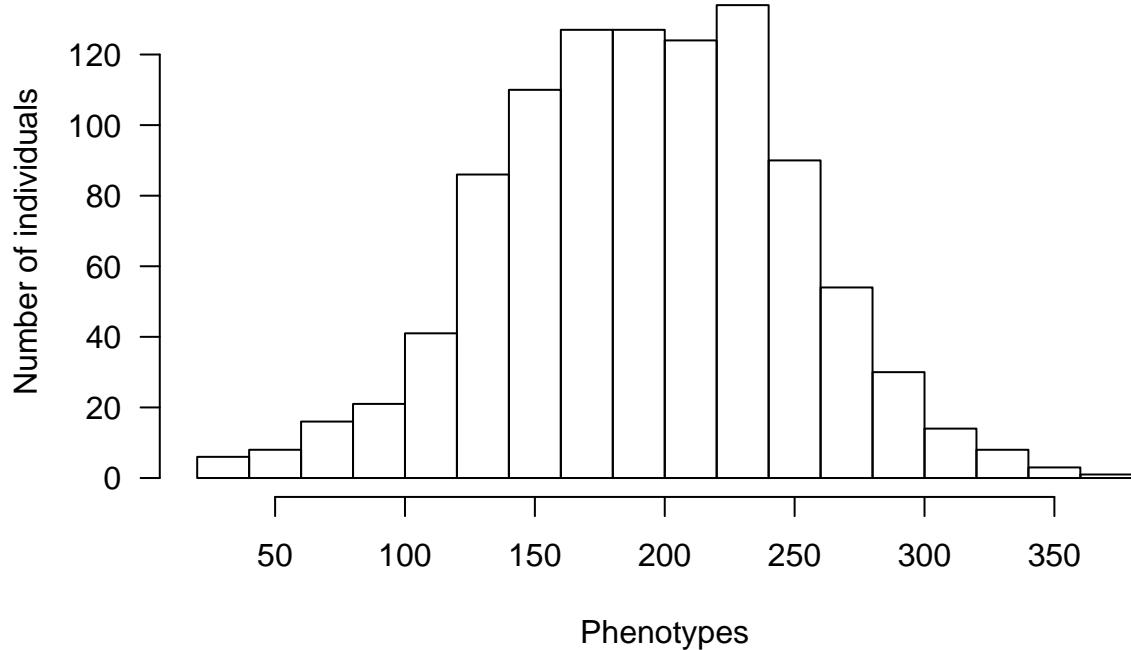
## [1] 1343

epsilon <- matrix(mvrnorm(n=1, mu=rep(0, N), Sigma=sigma.epsilon2 * diag(N)))
y <- matrix(1, nrow=N) * mu + W %*% alpha + X %*% beta + epsilon
summary(y)

##      V1
##  Min.   : 28
##  1st Qu.:154
##  Median :194
##  Mean   :192
##  3rd Qu.:232
##  Max.   :366

hist(y, breaks="FD", main="", las=1,
      xlab="Phenotypes", ylab="Number of individuals")

```



3.3 Inférence

3.3.1 En fréquentiste

Paquet rrBLUP:

```

system.time(
  fit.rrBLUP <- mixed.solve(y=y, Z=X, X=cbind(rep(1,N), W),
                            method="REML", SE=TRUE, return.Hinv=TRUE)
)

##      user  system elapsed
##  8.517   0.254   8.889

cbind(c(mu, alpha), fit.rrBLUP$beta)

##      [,1]    [,2]
## 54.00 129.98
## fix1  5.92   5.02

fit.rrBLUP$beta.SE

##          fix1
## 38.331  0.832

c(sigma.epsilon2, fit.rrBLUP$Ve)

## [1] 1343 1287

c(sigma.beta2, fit.rrBLUP$Vu)

## [1] 1.000 0.846

cor(beta, fit.rrBLUP$u)

## [1] 0.324

```

3.3.2 En bayésien

En bref, il suffit d'appliquer la formule de Bayes:

$$p(\Theta|\mathbf{y}, W, X) = \frac{p(\Theta, \mathbf{y}|W, X)}{p(\mathbf{y}|W, X)} \quad (4)$$

$$= \frac{p(\Theta)p(\mathbf{y}|W, X, \Theta)}{\int_{\Theta} p(\Theta)p(\mathbf{y}|W, X, \Theta)d\Theta} \quad (5)$$

$$\propto p(\Theta)p(\mathbf{y}|W, X, \Theta) \quad (6)$$

Cette formule est la simple application de la définition de la [probabilité conditionnelle](#), ainsi que de la formule des [probabilités totales](#). Pour mieux retenir la formule de Bayes, remarquez que le posterior est proportionnel au prior multiplié par la vraisemblance.

Paquet [BGLR](#):

```

system.time(
  fit.BGLR <- BGLR(y=y, ETA=list(list(X=W, model="FIXED"),
                                list(X=X, model="BRR")),
                        verbose=FALSE, nIter=7000, burnIn=1000, thin=5)
)

##      user    system elapsed
##  69.773   0.073  69.861

c(mu, fit.BGLR$mu)

## [1] 54 111

fit.BGLR$SD.mu

## [1] 36.5

c(alpha, fit.BGLR$ETA[[1]]$b)

##      fix1
## 5.92 5.16

fit.BGLR$ETA[[1]]$SD.b

##      fix1
## 0.828

c(sigma.epsilon2, fit.BGLR$varE)

## [1] 1343 1421

fit.BGLR$SD.varE

## [1] 260

c(sigma.beta2, fit.BGLR$ETA[[2]]$varB)

## [1] 1.000 0.779

cor(beta, fit.BGLR$ETA[[2]]$b)

## [1] 0.323

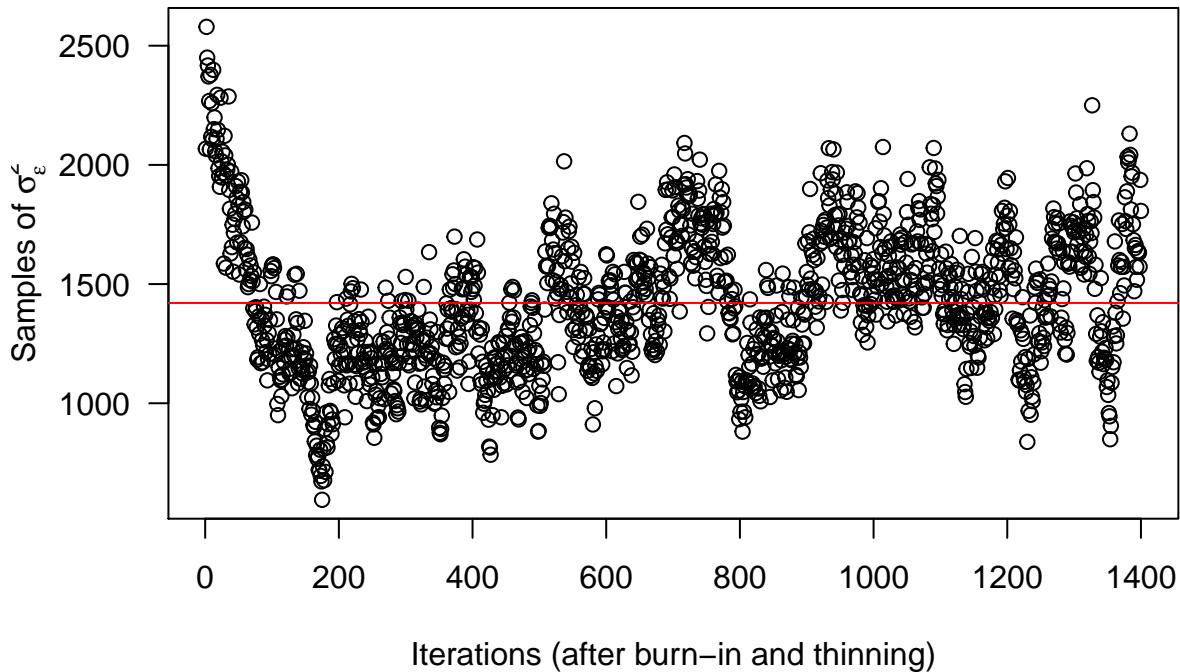
```

Lorsque l'on utilise un algorithme MCMC, il est toujours important de vérifier qu'il a convergé, au moins visuellement:

```

plot(scan("varE.dat"), las=1,
      xlab="Iterations (after burn-in and thinning)",
      ylab=expression(paste("Samples of ", sigma[epsilon]^2)))
abline(h=fit.BGLR$varE, col="red")

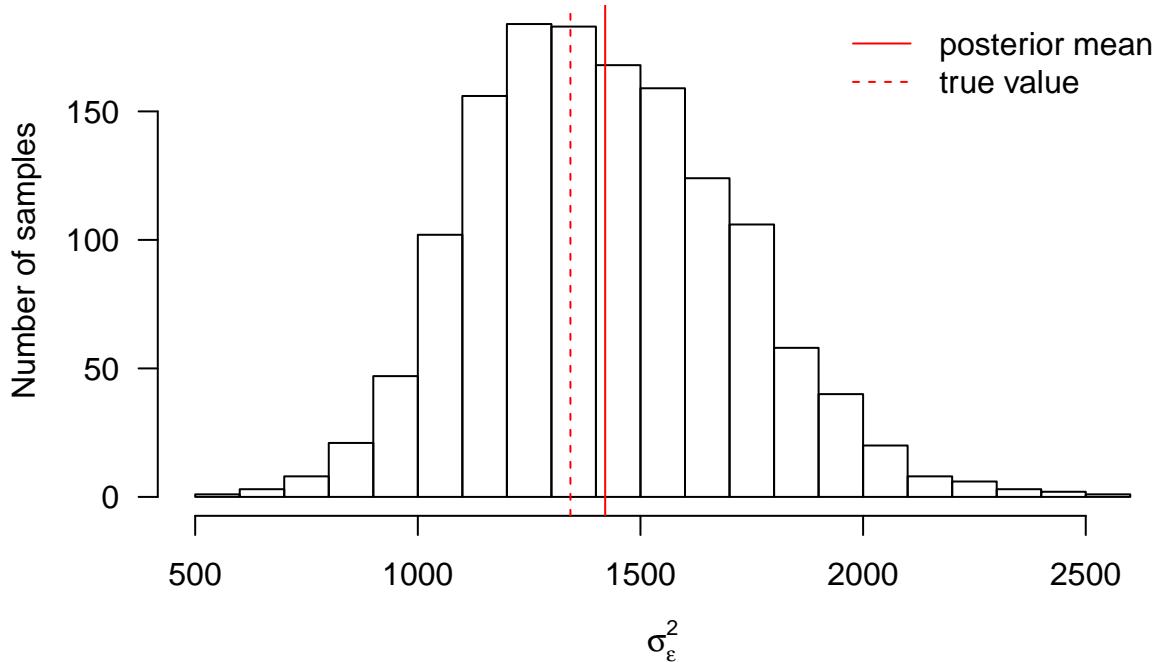
```



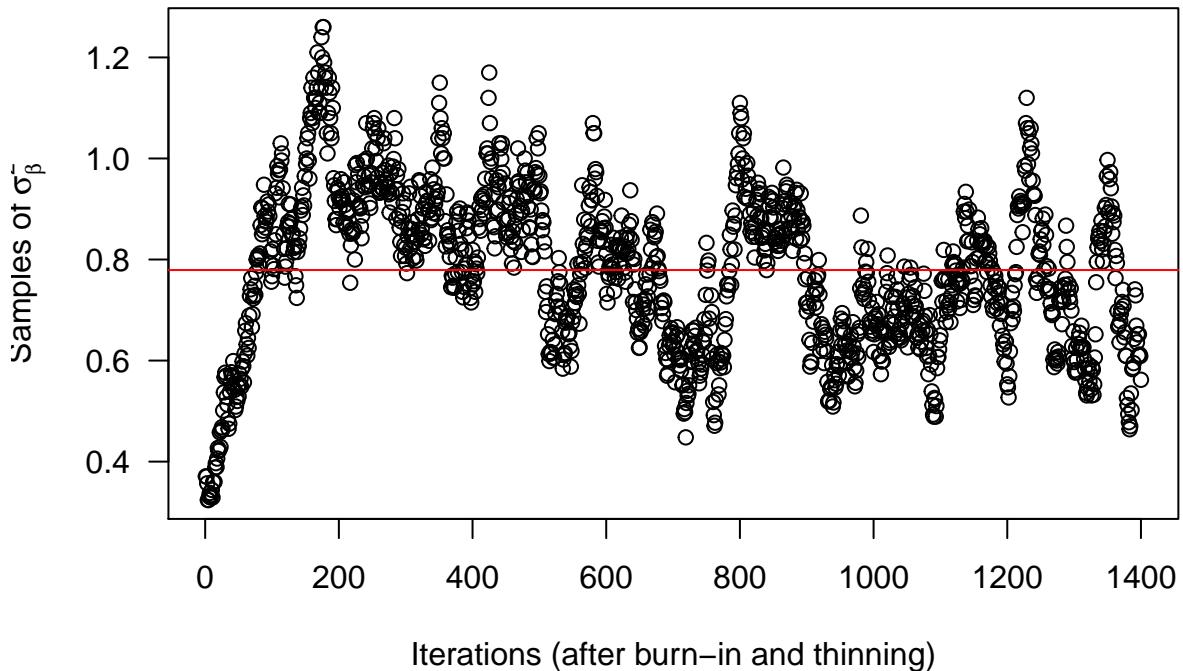
```

hist(scan("varE.dat"), breaks="FD", main="", las=1,
      xlab=expression(sigma[epsilon]^2), ylab="Number of samples")
abline(v=fit.BGLR$varE, col="red")
abline(v=sigma.epsilon2, col="red", lty=2)
legend("topright", legend=c("posterior mean","true value"),
      col="red", lty=c(1,2), bty="n")

```

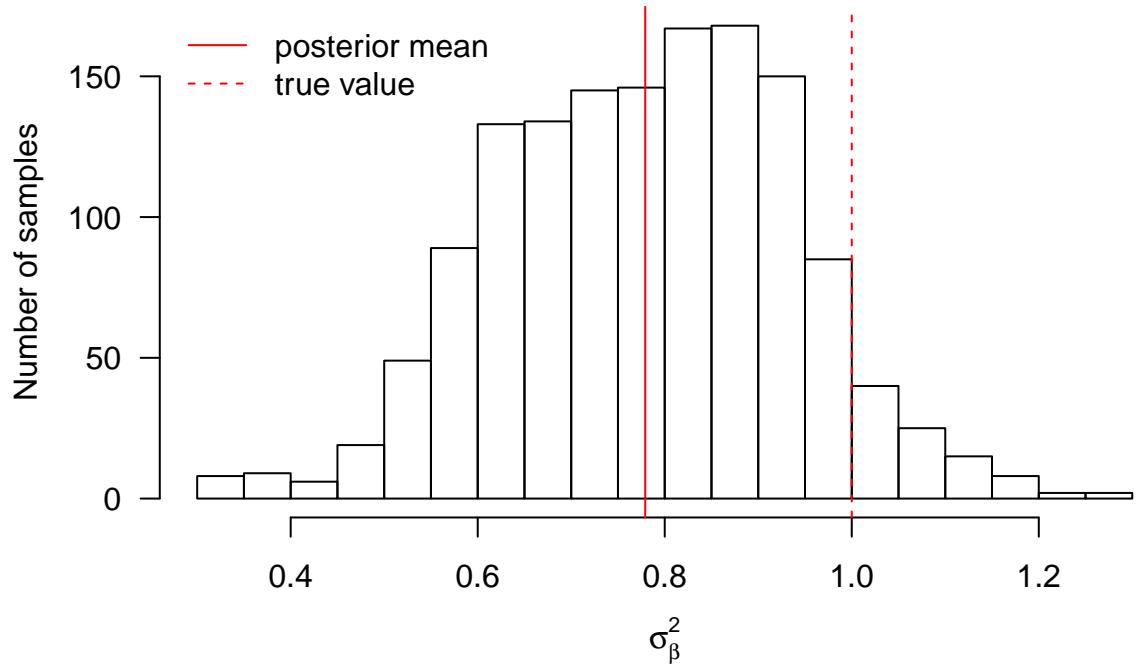


```
plot(scan("ETA_2_varB.dat"), las=1,
     xlab="Iterations (after burn-in and thinning)",
     ylab=expression(paste("Samples of ", sigma[beta]^2)))
abline(h=fit.BGLR$ETA[[2]]$varB, col="red")
```

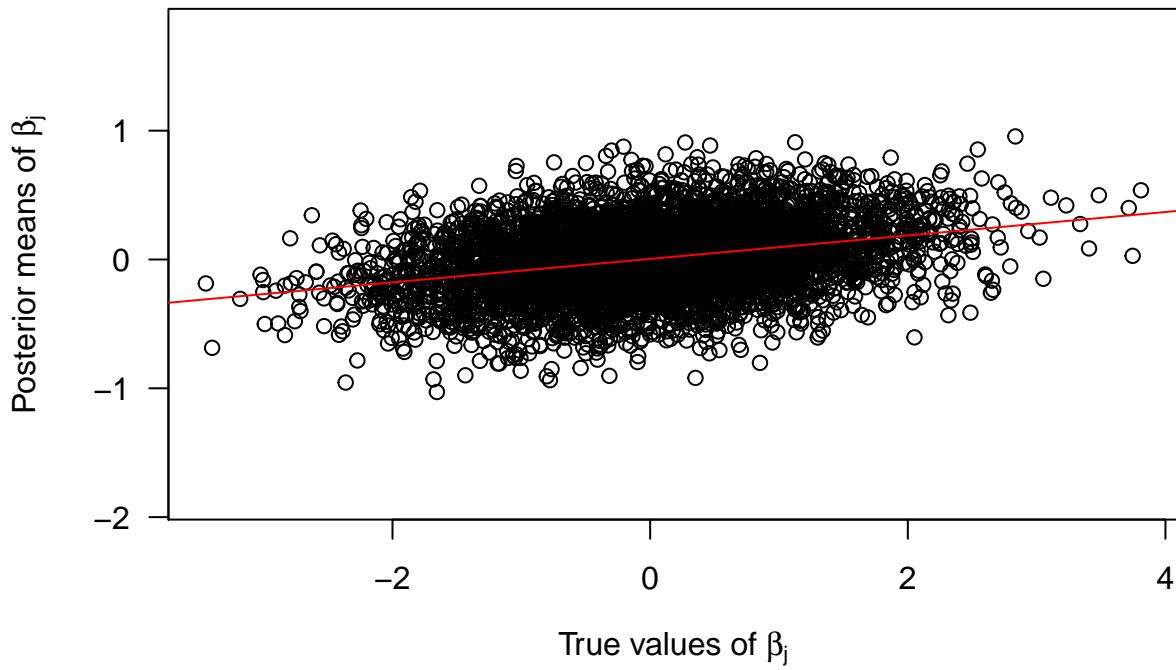


```
hist(scan("ETA_2_varB.dat"), breaks="FD", main="", las=1,
      xlab=expression(sigma[beta]^2), ylab="Number of samples")
abline(v=fit.BGLR$ETA[[2]]$varB, col="red")
abline(v=sigma.beta2, col="red", lty=2)
```

```
legend("topleft", legend=c("posterior mean", "true value"),
      col="red", lty=c(1,2), bty="n")
```



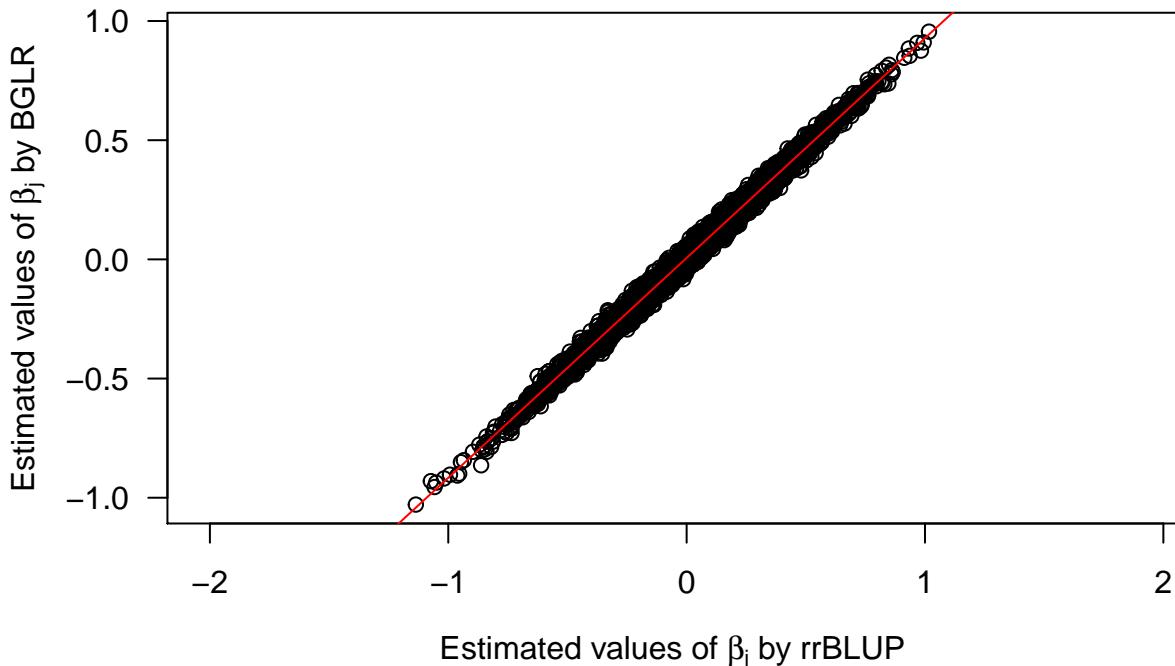
```
plot(beta, fit.BGLR$ETA[[2]]$b, asp=1, las=1,
      xlab=expression(paste("True values of ", beta[j])),
      ylab=expression(paste("Posterior means of ", beta[j])))
abline(lm(fit.BGLR$ETA[[2]]$b ~ beta), col="red")
```



3.3.3 Comparaison

Les deux approches donnent des résultats très similaires concernant les effets des marqueurs:

```
plot(fit.rrBLUP$u, fit.BGLR$ETA[[2]]$b, asp=1, las=1,
      xlab=expression(paste("Estimated values of ", beta[j], " by rrBLUP")),
      ylab=expression(paste("Estimated values of ", beta[j], " by BGLR")))
abline(lm(fit.BGLR$ETA[[2]]$b ~ fit.rrBLUP$u), col="red")
```



3.4 Critiques

Il est important de réaliser à quel point la simulation ci-dessus est simpliste. Tout d'abord, les génotypes ont été simulés indépendamment les uns des autres, sans tenir compte de l'apparentement entre individus ni du déséquilibre de liaison entre marqueurs adjacents le long du génome. Pour remédier à cela, le plus simple serait de simuler des génotypes via le “coalescent avec recombinaison” (voir paquet [scrm](#)).

De plus, tous les caractères n'ont pas une architecture génétique additive et infinitésimale. Dans certains cas (la majorité?), seul un sous-ensemble des marqueurs a un effet. D'autres priors doivent alors être utilisés, qui visent à sélectionner les marqueurs à effet non-nul, par exemple “BayesB” dans la fonction BGLR().

Par ailleurs, certains caractères ne sont pas toujours bien capturés par la loi Normale, comme les caractères discrets, qu'ils soient binaires (résistant versus sensible) ou non (nombre de grains par épis).

Quoi qu'il en soit, afin de se rapprocher de cas réels, commençons par utiliser de vrais génotypes.

4 Simuler des phénotypes à partir de vrais génotypes

Commençons par charger le fichier au format CSV avec les noms des SNP en première ligne, les noms des individus en première colonne, et le dosage alléliques dans le reste:

```

## system.time(X <- read.csv("true_genotypes.csv", header=TRUE, nrows=381))
## system.time(
##   X <- read.csv(unz("true_genotypes.csv.zip", "true_genotypes.csv"),
##                 header=TRUE, row.names=1, nrows=381,
##                 colClasses=c("character", rep("integer", 62220)))
## )
system.time(tmp <- as.data.frame(fread("true_genotypes.csv")))

```

```

##    user  system elapsed
##  4.240   0.036   4.672

```

```

X <- as.matrix(tmp[,-1])
rownames(X) <- tmp[,1]
dim(X)

```

```

## [1] 380 62220

```

```

X[1:3,1:5]

```

```

##      snp00001 snp00002 snp00003 snp00004 snp00005
## ind001      2      2      0      0      2
## ind002      2      2      2      0      2
## ind003      2      2      0      0      2

```

```

str(X[1:3,1:5])

```

```

##  int [1:3, 1:5] 2 2 2 2 2 2 0 2 0 0 ...
##  - attr(*, "dimnames")=List of 2
##    ..$ : chr [1:3] "ind001" "ind002" "ind003"
##    ..$ : chr [1:5] "snp00001" "snp00002" "snp00003" "snp00004" ...

```

TODO: tracer la distribution des fréquences alléliques

Calculer les relations génétiques additives et les visualiser:

```

A <- A.mat(X - 1)
A[1:3,1:3]

```

```

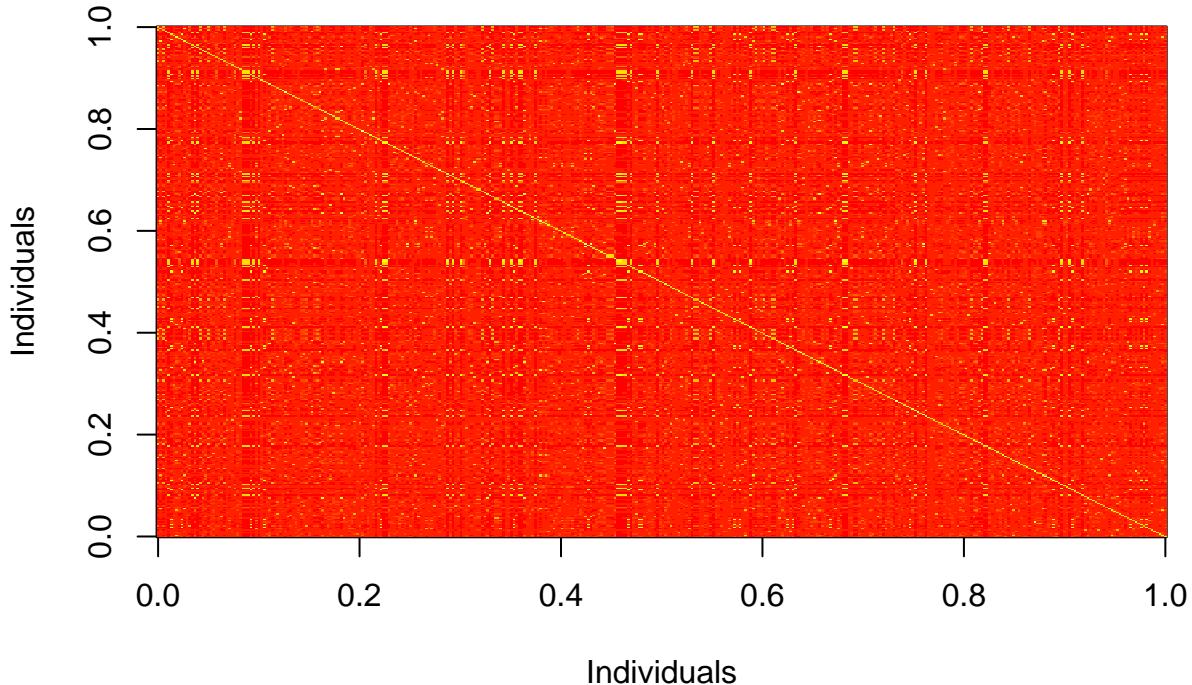
##      ind001 ind002 ind003
## ind001  1.4254 -0.0215  1.4210
## ind002 -0.0215  2.0584 -0.0221
## ind003  1.4210 -0.0221  1.4270

```

```

image.mat(A, xlab="Individuals", ylab="Individuals")

```



TODO: définir différents sous-ensembles de marqueurs

5 Prédire grâce au génome

Inspirez-vous du [tutoriel](#) d'Amy Jacobson (Univ. Minnesota); le pdf est [ici](#).

6 Explorer des jeux de données réels librement disponibles

Comme l'a fait justement remarquer Zamir (PLoS Biology 2013, Science 2014), il est très difficile de trouver des jeux de données avec phénotypes en libre accès. Cependant, en voici quelques uns:

- [Crossa et al \(Genetics, 2010\)](#): blé (599 lignées, 4 conditions, rendement en grains, pédigrée, 1279 marqueurs DArT) et maïs (300 lignées, 1148 marqueurs SNP, 3 caractères, deux conditions)
- [Resende et al \(Genetics, 2012\)](#): pin (951 individus de 61 familles, pédigrée, 4853 marqueurs SNP, phénotypes dérégréssés)
- [Cleveland et al \(G3, 2012\)](#): porc (3534 animaux, pédigrée, 5 caractères, 53000 marqueurs SNP)

7 Perspectives

Au-delà des critiques faites dans la section 3, les deux plus grandes simplifications de ce travail ont été de ne se concentrer que sur un seul caractère, et de complètement ignorer l'environnement.

Or c'est bien là le défi des sélectionneurs, qu'ils soient dans des entreprises semencières ou dans des collectifs de paysans: créer de nouvelles variétés combinant plusieurs caractères d'intérêt et adaptées à l'itinéraire technique, à la filière économique, à l'agriculteur et au consommateur, etc.

Mais ce sera pour un autre cours!

8 Références

- Barton, N. H. and P. D. Keightley (2002, January). Understanding quantitative genetic variation. *Nature Reviews Genetics* 3 (1), 11-21. [DOI](#)
- Weir, B. S., A. D. Anderson, and A. B. Hepler (2006, October). Genetic relatedness analysis: modern data and new challenges. *Nature Reviews Genetics* 7 (10), 771-780. [DOI](#)
- Visscher, P. M., W. G. Hill, and N. R. Wray (2008, March). Heritability in the genomics era — concepts and misconceptions. *Nature Reviews Genetics* 9 (4), 255-266. [DOI](#)
- Slatkin, M. (2008, June). Linkage disequilibrium — understanding the evolutionary past and mapping the medical future. *Nature Reviews Genetics* 9 (6), 477-485. [DOI](#)
- Stephens, M. and D. J. Balding (2009, October). Bayesian statistical methods for genetic association studies. *Nature Reviews Genetics* 10 (10), 681-690. [DOI](#)
- de los Campos, G., D. Gianola, and D. B. Allison (2010, December). Predicting genetic predisposition in humans: the promise of whole-genome markers. *Nature Reviews Genetics* 11 (12), 880-886. [DOI](#)
- Morrell, P. L., E. S. Buckler, and J. Ross-Ibarra (2012, February). Crop genomics: advances and applications. *Nature Reviews Genetics* 13 (2), 85-96. [DOI](#)

9 Annexe

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

## R version 3.2.0 (2015-04-16)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 14.04.2 LTS
##
## attached base packages:
## [1] stats      graphics   grDevices utils      datasets   methods    base
##
## other attached packages:
## [1] data.table_1.9.4 BGLR_1.0.4       rrBLUP_4.3        MASS_7.3-40
## [5] knitr_1.9        rmarkdown_0.5.1
##
## loaded via a namespace (and not attached):
## [1] plyr_1.8.2       formatR_1.2       tools_3.2.0       htmltools_0.2.6
## [5] reshape2_1.4.1    yaml_2.1.13      Rcpp_0.11.5       stringr_0.6.2
## [9] digest_0.6.8     chron_2.3-45     evaluate_0.7
```