Joint estimaton of all marker effects (= Models for Genomic selection)

Andrés Legarra - INRA

Joint estimation

- We are we interested in estimating all markers simultaneously
  - (for genomic evaluation)
  - And for QTL business:
    - Less biased estimates of effects
    - More correct location, free from contamination
    - Less problems from multiple testing

- But there is a price to pay
  - More complex models & programming
  - Less clear results & interpretation
  - (Often) long computing times – but this is because we do MCMC

The problem of "related" populations

- Animal populations are very related
- And typical traits have many QTLs
- Thus, "false" signals appear:
  - either in markers indicating "overall relationships" (e.g. two bull families)
  - or in two markers linked to the same QTL
- Fitting all markers at once we solved both problems

Models for Genomic selection

- Single marker
- Whole-genome (multiple marker) genomic selection
  - Different priors

Single marker

- Assume there is a marker in complete LD with a QTL
- For example, the polymorphism in the halothane gene (HAL) which is a predictor of bad meat quality in swine

Single marker

- Estimate breeding values including the marker is a piece of cake
- $\mathbf{y}_i$= marker effect in animal *i* + *e*
  - We substitute the true, possibly unknown gene by a proxy observed marker and estimate effects of the latter using a linear model
  - We can include an additional polygenic genetic value of animal *i*

Base model

- **y= …+ Za + e**
  - **Z**= incidence matrix of marker effects
  - **a**= marker effect
  - **e**=residuals

3 individuals, 1 marker with 4 alleles

$$\mathbf{Za} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 2 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix}$$

- This can be solved, for example, by least squares

Base model

- **y= …+ Za + u + e**
  - **Z**= incidence matrix of marker effects
  - **a**= marker effect
  - **u**= polygenic breeding value (Var(u)= $\mathbf{A}\sigma_u^2$)
  - **e**=residuals

3 individuals, 1 marker with 4 alleles

$$\mathbf{Za} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 2 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix}$$

- This can be solved, for example, by BLUP

Single marker

- Fine, if there are no more QTLs around
- Can try an iterative procedure (*forward selection*)
  - Find QTL1
  - Fit QTL1 as fixed effect, then find QTL2
  - Fit QTL1, QTL2 as fixed effect, then find QTL3
  - …
- This will not work because we'll find only large QTL but not the small ones
  - And those that you find are certainly exaggerated

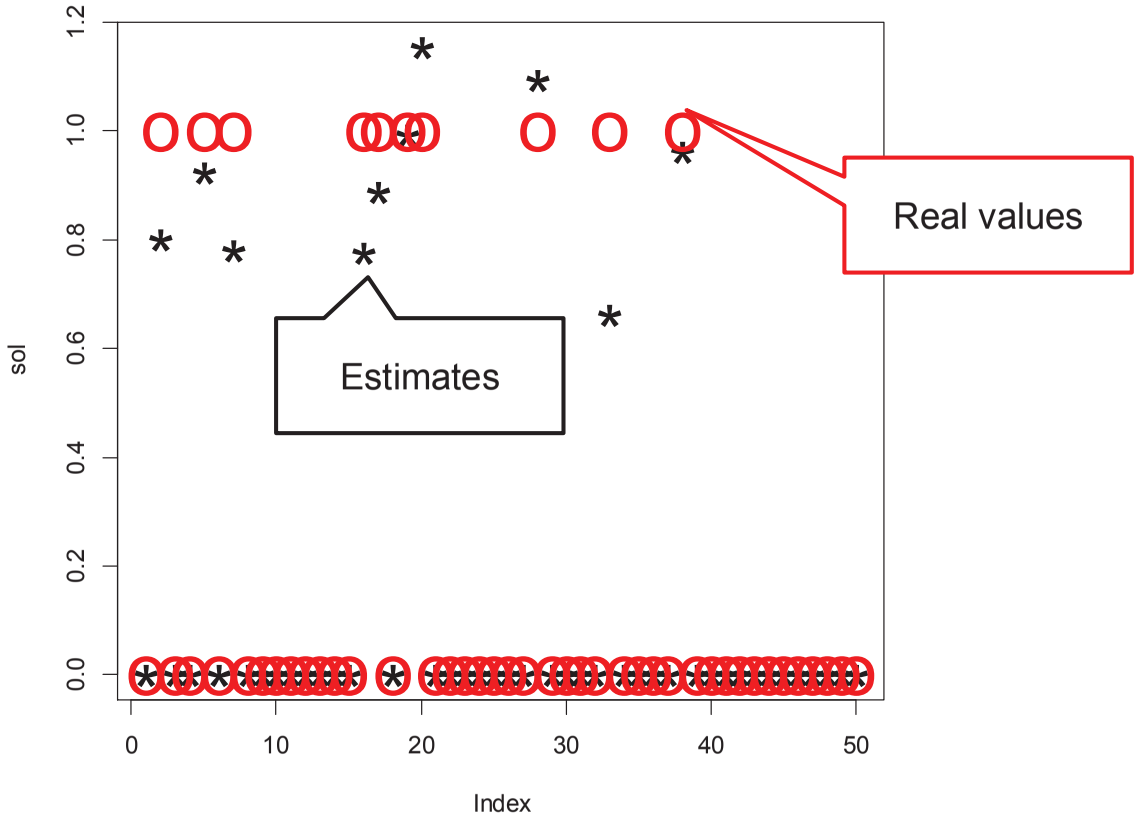
Beavis effect

- We are mapping QTLs
- To declare a QTL in a position, we perform a test (for example a t-test)
- This test depends on the estimated effect of the QTL
  - estimated effect = real effect + « estimation noise »
  - by keeping selected QTLs, we often keep large and positive noises
    - this is negligible if there were few QTLs with large effects but this is not the case
    - large noises will occur in analysis with *many* markers
  - this biases the estimated QTL effect

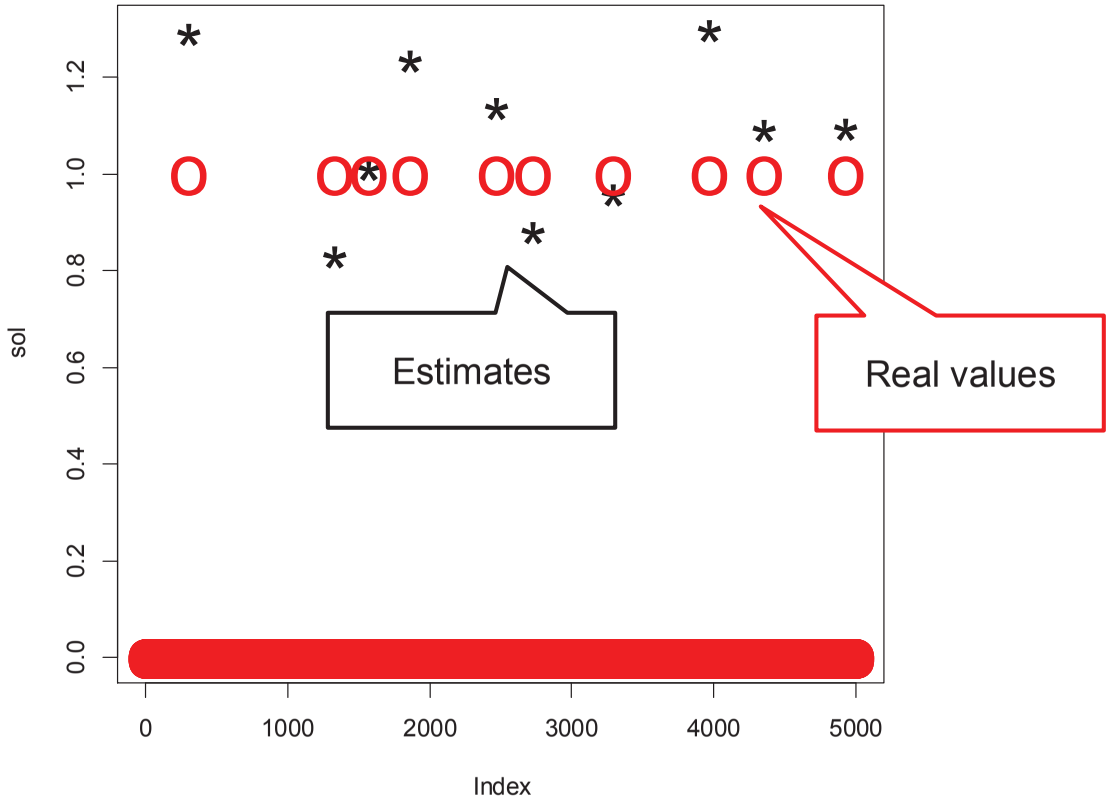

```
# this is a simulation concerning the beavis effect
# we simulate a system with 10 SNPs among 5000, set a p-value, and try to
estimate the effect
# of the selected SNPs

set.seed(1234)
n=1000 # individuals
p=5000 # markers
tr=10  # true QTLs
Z=matrix( sample(c(-1,0,1),n*p,replace=TRUE,prob=c(.25,.5,.25)), nrow=n )
pos=sample(1:p,tr)
a=rep(0,p)
a[pos]=1 # effect of 1
y=100+Z%*%a+rnorm(n)*sqrt(var(Z%*%a)) # h2=0.5
# GWAS
bonf=0.05/p # bonferroni level
sol=rep(NA,p)
for (i in 1:p){
    pp=lm(formula = y ~ Z[, i])
    sol[i]=coefficients(pp)[2]
    # get pvalue
    pval=anova(pp)[1,5]
    if(pval>bonf) sol[i]=0
}
plot(sol,pch="*",cex=3)
points(a,col="red",pch="o",cex=3)
```

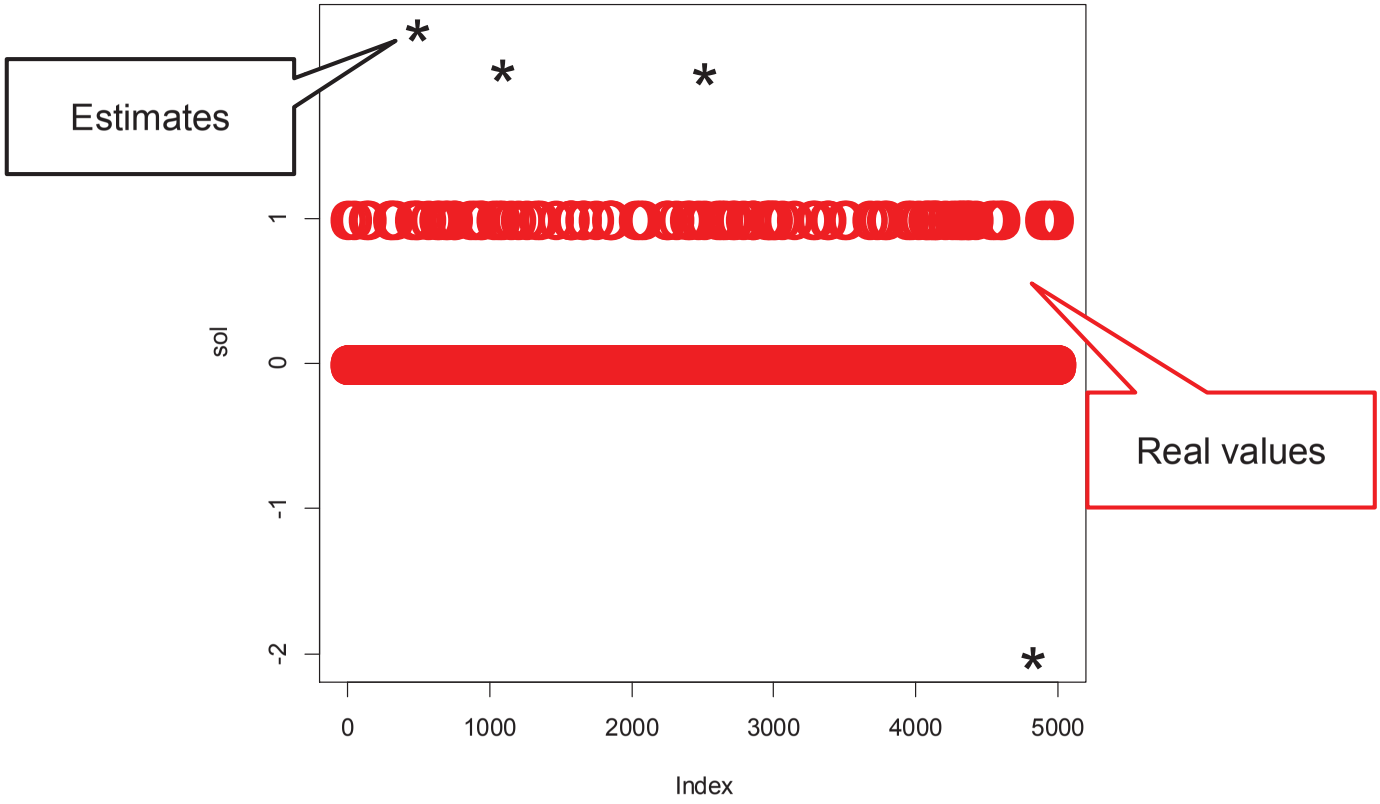
10 true QTLs in 50 markers, 1000 individuals



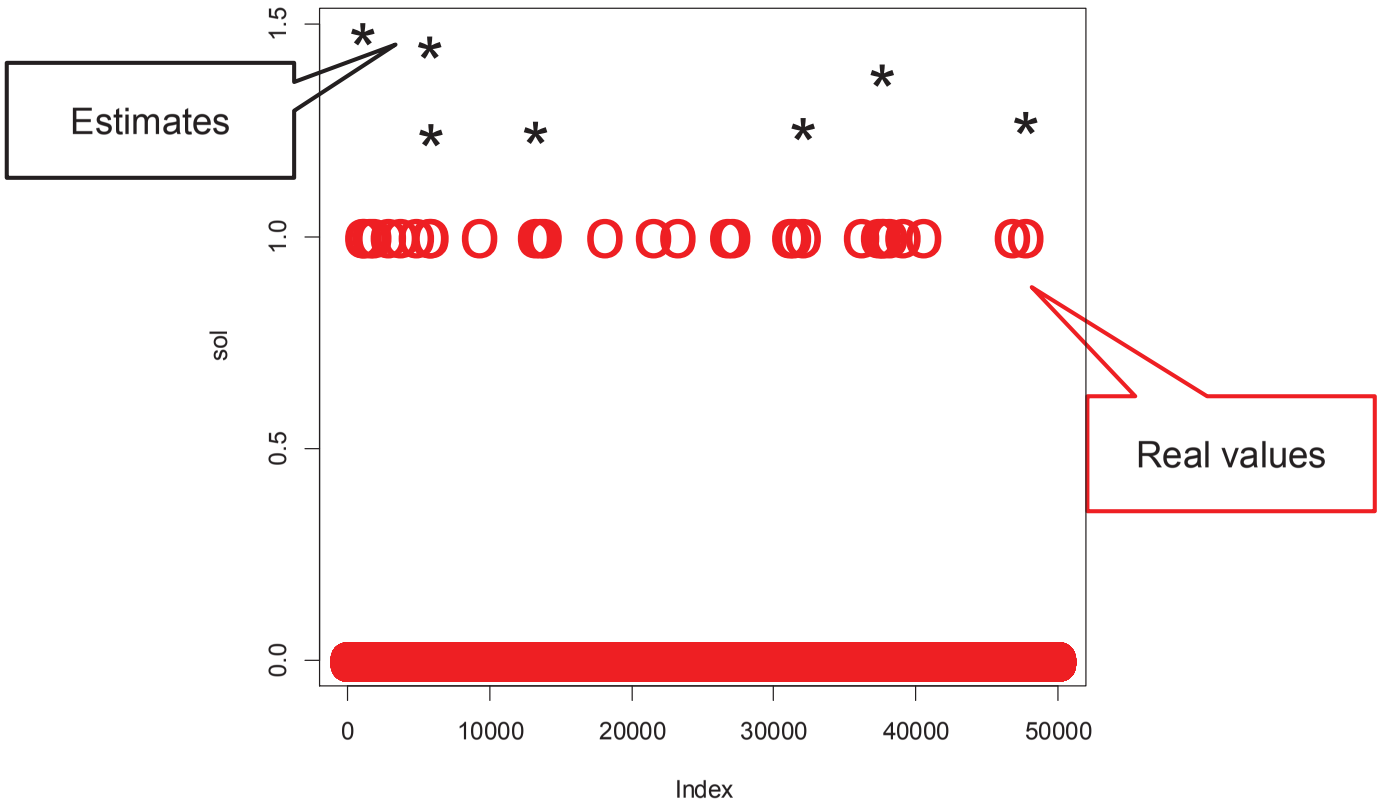
10 true QTLs in 5000 markers, 1000 individuals



100 true QTLs in 5000 markers, 1000 individuals



30 true QTLs in 50000 markers, 1000 individuals



Whole genome

- If we don't select QTL regions we skip the problem of bias and of lack of power
- Therefore :
  - Genetic value = sum of effects of all regions
    - We effectively treat all regions as being carriers of a QTL
  - How do we estimate the effects of all regions?

Whole genome

- The simpler is to do an extension of single marker analysis
- Do multiple marker regression
- You want to cover all the genome => many markers

Multiple marker additive model

- **y= Za + e**
  - **Z**= incidence matrix of marker effects
  - **a**= marker effect
  - **e**=residuals

4 individuals, 2 markers each

2 alleles in 1st marker

$$\mathbf{Za} = \left[\begin{array}{cc|cccc} 1 & 1 & 0 & 1 & 1 & 0 \\ 2 & 0 & 2 & 0 & 0 & 0 \\ 0 & 2 & 1 & 0 & 0 & 1 \\ 0 & 2 & 1 & 1 & 0 & 0 \end{array}\right] \left[\begin{array}{c} a_{1,1} \\ a_{1,2} \\ \hline a_{2,1} \\ a_{2,2} \\ a_{2,3} \\ a_{2,4} \end{array}\right]$$

4 alleles in 2nd marker

Estimating SNP effects

- The simultaneous estimates of many markers by least squares are very poor,
  - if we have much more SNPs than individuals
  - They are thus terribly bad for genomic predictions as well
- Even if we had many individuals, there is a missing piece of information:
  - we think that most SNPs do *not* have an effect or at least a big one
  - this is a « prior » information
- Can we do something?

Integration

- Need to postulate p(**a**) : this means SNP are random effects
- Best Predictor is optimal for selection (Cochran, 1951; Goffinet and Elsen 1984 GSE; Gianola and Fernando 1986 JAS; etc etc)
- The Best Predictor comes from the conditional mean of SNP effects after observing the data

$$\hat{\mathbf{a}} = E(\mathbf{a}\,|\,\mathbf{y}) = \frac{\int \mathbf{a}\, p(\mathbf{y}\,|\,\mathbf{a})\, p(\mathbf{a})\, d\mathbf{a}}{\int p(\mathbf{y}\,|\,\mathbf{a})\, p(\mathbf{a})\, d\mathbf{a}}$$

- Remember: classical BLUP is an example of BP if
  - $p(\mathbf{g}) \sim N(0, \mathbf{A}\sigma^2_u)$

Best Predictor as a Bayesian estimator

« Likelihood » (how SNP effects affect the phenotype)

« Prior » (how we think SNP effects are)

$$\hat{\mathbf{a}} = E(\mathbf{a}\,|\,\mathbf{y}) = \frac{\int \mathbf{a}\, p(\mathbf{y}\,|\,\mathbf{a})\, p(\mathbf{a})\, d\mathbf{a}}{\int p(\mathbf{y}\,|\,\mathbf{a})\, p(\mathbf{a})\, d\mathbf{a}}$$

Estimate of SNP effects

Best Predictor as a « penalized » estimator

- Statisticians & « machine learners » aim using « penalized » estimators (Ridge regression, Elastic net…)

- A penalized estimator is the same as a « best predictor » (or as a Bayesian estimator) before with prior now called « penalization »

A priori Distributions for marker effects

- *Unknown*
- *Still we don't have a good theory*
- Agreement on
  - Small effects more frequent than big ones
  - marker effects are assumed *a priori* independent

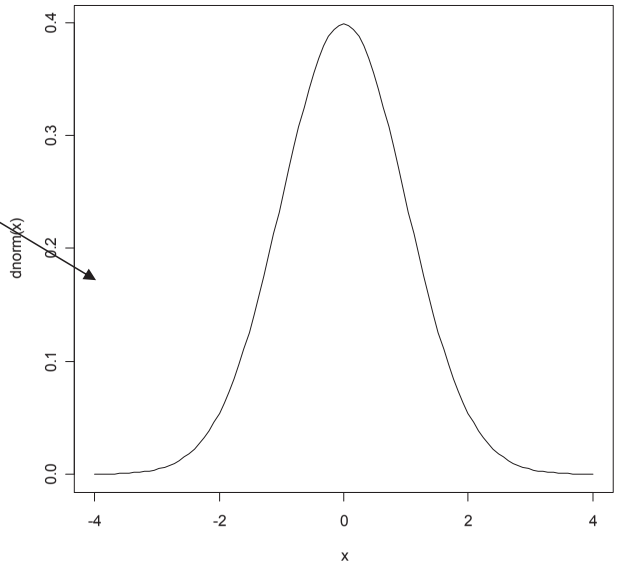
A priori Distributions for marker effects

- Several distributions for SNP effects have been proposed
  - Normal (Meuwissen et al., Genetics 2001; Van Raden JDS 2008) -> BLUP_SNP or GBLUP or RR-BLUP
  - BayesA, BayesB, (Meuwissen et al. 2001; Habier et al., 2011)
  - Mixture of normal , BayesC(Pi) (Van Raden JDS 2008, Habier et al., 2011)
  - (Bayesian) Lasso (Usai et al., 2009; De los Campos, et al., 2009)

Normal distribution

$$a_i \sim N\left(0, \sigma_a^2\right)$$

Few « big » effects



SNP-based Normal (BLUP_SNP) models for genomic evaluation.

Andrés Legarra - INRA

Recall

- We want to estimate SNP effects **a** by a model
  - **y = Xb + Za + e**
- **Z** contains (somehow) genotypes

Useful parameterizations

- We have SNPs, thus biallelic
- We could say that each allele has an effect
- This results in *2n* effects
- it's possible to fit ONE effect by locus

Allele coding

- we fit a regression of genetic value on gene content (as in Falconer)
- a_i is the effect of the SNP
- We code explanatory variables in **Z** (**y=Xb+Za+e**) as:

– « CC » =	0	$-a_i$	$(-2p)a_i$
– « CG » =	a_i	0	(1-2p)ai
– « GG » =	$2a_i$	a_i	(2-2p)ai

Strandén & Christensen (http://www.gsejournal.org/content/43/1/25) refer to these as « 012 », « 101 » and « centered »

So we have different **Z**'s depending on how we code…

Example

- 2 SNP, 4 individuals

CC CG
AA GG
AC AA
GG GG

$$\mathbf{Z} = \begin{bmatrix} 0 & 1 \\ 2 & 0 \\ 1 & 0 \\ 0 & 0 \end{bmatrix}$$

« 012 »

$$\mathbf{Z} = \begin{bmatrix} -1 & 0 \\ 1 & -1 \\ 0 & -1 \\ -1 & -1 \end{bmatrix}$$

« 101 »

$$\mathbf{Z} = \begin{bmatrix} -0.75 & 0.75 \\ 1.25 & -0.25 \\ 0.25 & -0.25 \\ -0.75 & -0.25 \end{bmatrix}$$

« centered » (the sum of each column of Z is 0)

All give the same estimates of SNP effects

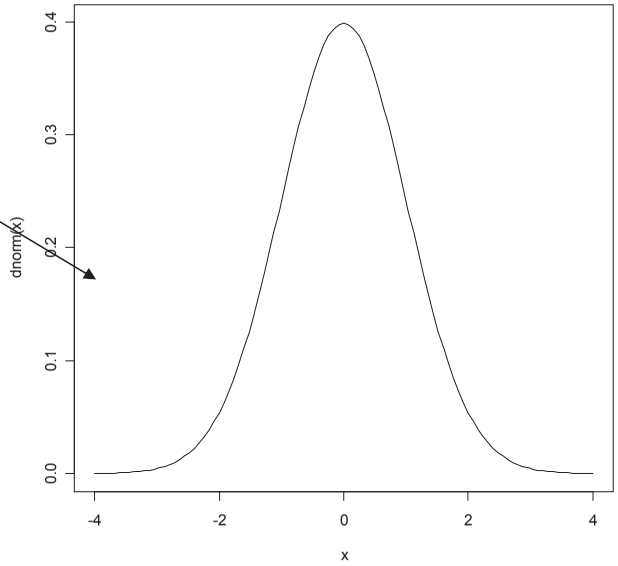
A priori Distributions for marker effects

- Several distributions for SNP effects have been proposed
  - Normal (Meuwissen et al., Genetics 2001; Van Raden JDS 2008)
  - BayesA, BayesB, (Meuwissen et al. 2001; Habier et al., 2011)
  - Mixture of normal , BayesC(Pi) (Van Raden JDS 2008, Habier et al., 2011)
  - (Bayesian) Lasso (Usai et al., 2009; De los Campos, et al., 2009)

Normal distribution

$$a_i \sim N\left(0, \sigma_a^2\right)$$

Few « big » effects



Normal equations for genomic selection (BLUP_SNP)

- If we assume normality there are closed expressions for **â**
- This is called « BLUP », and also « genomic BLUP » , BLUP_SNP, or GBLUP, but also « ridge regression » or Random Regression-BLUP
  - I will keep GBLUP for the use of the genomic relationship matrix
  - and BLUP_SNP for the direct estimation of SNP effects

Mixed model equations for BLUP_SNP

- Henderson's MME
- **Z'Z** is *not* diagonal
- Var(**a**)=**D** is diagonal if we assume uncorrelated SNP effects

$$\begin{bmatrix} \mathbf{X}'\mathbf{R}^{-1}\mathbf{X} & \mathbf{X}'\mathbf{R}^{-1}\mathbf{Z} \\ \mathbf{Z}'\mathbf{R}^{-1}\mathbf{X} & \mathbf{Z}'\mathbf{R}^{-1}\mathbf{Z}+\mathbf{D}^{-1} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{b}} \\ \hat{\mathbf{a}} \end{bmatrix} = \begin{bmatrix} \mathbf{X}'\mathbf{R}^{-1}\mathbf{y} \\ \mathbf{Z}'\mathbf{R}^{-1}\mathbf{y} \end{bmatrix}$$

Could (will) be something different !!

$$\mathbf{D} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \sigma_a^2 = \mathbf{I}\sigma_a^2$$

Efficient solvers for BLUP_SNP:

- Gauss-Seidel with Residual Update: (Legarra and Misztal J. Dairy Sci. 2008) (reinvented many times) implemented in GS3
  - Form the basis of the Gibbs Sampling Algorithms in BayesC, etc.
  - Iterate on:
    1. Estimate SNP effect
    2. Correct data for this SNP effect
- Preconditioned Conjugate Gradients (not in GS3)
  - Estimate all SNP simultaneously using "search" based on residuals at each iteration

Fortran pseudocode

```
Double precision:: xpx(neq),y(ndata),e(ndata),X(ndata,neq), &
sol(neq),lambda,lhs,rhs,val
do i=1,neq
   xpx(i)=dot_product(X(:,i),X(:,i)) !form diagonal of X'X
enddo
e=y
do until convergence
   do i=1,neq
     !form lhs X'R-1X + G-1
     lhs=xpx(i)/vare+1/vara
     ! form rhs with y corrected by other effects (formula 1) !X'R-1y
     rhs=dot_product(X(:,i),e)/vare +xpx(i) *sol(i)/vare
     ! do Gauss Seidel
     val=rhs/lhs
     ! MCMC sample solution from its conditional (commented out here)
     ! val=normal(rhs/lhs,1d0/lhs)
     ! update e with current estimate (formula 2)
     e=e - X(:,i)*(val-sol(i))
     !update sol
     sol(i)=val
   enddo
enddo
```

Iterate

Solve for this SNP

Correct data for this SNP

Estimate of this SNP

BLUP_SNP solution

$$\hat{a}_{BLUP} = \frac{\frac{\boldsymbol{x}'\boldsymbol{y}}{\sigma_e^2}}{\frac{\boldsymbol{x}'\boldsymbol{x}}{\sigma_e^2} + \frac{1}{\sigma_a^2}}$$

Variance of the SNP

$$\hat{a}_{GWAS} = \frac{\frac{\boldsymbol{x}'\boldsymbol{y}}{\sigma_e^2}}{\frac{\boldsymbol{x}'\boldsymbol{x}}{\sigma_e^2}}$$

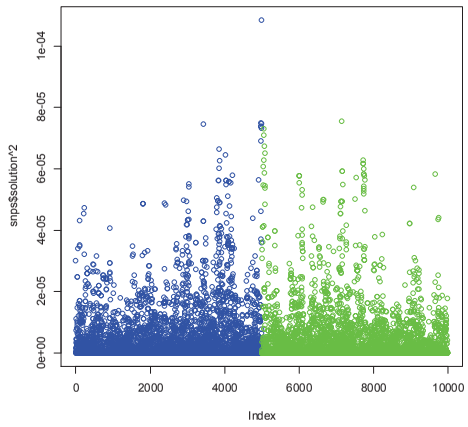
Least squares solution (e.g. in GWAS)

In BLUP_SNP, we shrink the least square estimate towards 0 because usually $\frac{1}{\sigma_a^2}$ is a large number

Estimate of this SNP

- So, the estimate is much smaller than the GWAS estimate
- But we can fit all SNPs simultaneously
- And this provides unbiased (in some sense) estimates

- However, the result is very confusing for QTL detection:



Estimate of this SNP

This suggests an iterative/adaptive strategy

$$\hat{a}_{i\,BLUP} = \frac{\frac{\boldsymbol{x}'\boldsymbol{y}}{\sigma_e^2}}{\frac{\boldsymbol{x}'\boldsymbol{x}}{\sigma_e^2} + \frac{1}{\sigma_{ai}^2}}$$

Variance of THIS SNP

If $\sigma_{ai}^2 \to 0$ we get the least square estimate

The more important the SNP, the larger σ_{ai}^2

-BayesA, etc etc (see later)

BLUP_SNP parameters

- For BLUP_SNP, we need σ^2_a and σ^2_e in

$$\begin{bmatrix} \mathbf{X}'\mathbf{R}^{-1}\mathbf{X} & \mathbf{X}'\mathbf{R}^{-1}\mathbf{Z} \\ \mathbf{Z}'\mathbf{R}^{-1}\mathbf{X} & \mathbf{Z}'\mathbf{R}^{-1}\mathbf{Z}+\mathbf{D}^{-1} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{b}} \\ \hat{\mathbf{a}} \end{bmatrix} = \begin{bmatrix} \mathbf{X}'\mathbf{R}^{-1}\mathbf{y} \\ \mathbf{Z}'\mathbf{R}^{-1}\mathbf{y} \end{bmatrix}; \mathbf{R} = \mathbf{I}\sigma_e^2; \mathbf{D} = \mathbf{I}\sigma_a^2$$

- Option 1: estimate them (e.g. Legarra et al., 2008)
  - Not always easy with small data sets or pre-corrected data
- Option 2: deduce from previous pedigree-based estimates, σ^2_g
  - Using a formula proposed by several authors (e.g. Gianola et al., 2009)

BLUP_SNP parameters

- How do we get the variance of SNP effects, σ^2_a, from a genetic variance σ^2_g ?
- The formula comes from the variance explained by each SNP, which is $2p_i q_i a_i^2$
  - We try to explain all genetic variance as if « caused » by SNP effects, and these SNP effects have a variance of σ^2_a
- Assumes Hardy-Weinberg and Linkage equilibrium

$$\sigma_a^2 \approx \frac{\sigma_g^2}{2\sum_{all\,SNPs} p_i(1-p_i)}$$

BLUP_SNP parameters

- Yet another faster way:
  - Construct a genomic relationship matrix **G**
  - Fit a REML model $y = Xb + Wg + \cdots$ with this matrix of relationship ($Var(g) = \boldsymbol{G}$)and estimate σ_g^2
  - Use then

$$\sigma_a^2 = \frac{\sigma_g^2}{2\sum_{all\ SNPs} p_i(1-p_i)}$$

  - This is the same as estimating σ_a^2 directly

BLUP_SNP parameters

- Both options (fix or estimate) usually agree but not always ☹

From pedigree

$$\frac{\sigma_g^2}{2\sum_{all\ SNPs} p_i(1-p_i)} \approx \sigma_a^2$$

Estimated from marker data

- Advice: use the formula and <u>also</u> estimate σ_a^2

BLUP_SNP parameters

- Still, we need σ^2_a and σ^2_e in

$$\begin{bmatrix} \mathbf{X}'\mathbf{R}^{-1}\mathbf{X} & \mathbf{X}'\mathbf{R}^{-1}\mathbf{Z} \\ \mathbf{Z}'\mathbf{R}^{-1}\mathbf{X} & \mathbf{Z}'\mathbf{R}^{-1}\mathbf{Z}+\mathbf{D}^{-1} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{b}} \\ \hat{\mathbf{a}} \end{bmatrix} = \begin{bmatrix} \mathbf{X}'\mathbf{R}^{-1}\mathbf{y} \\ \mathbf{Z}'\mathbf{R}^{-1}\mathbf{y} \end{bmatrix}; \mathbf{R} = \mathbf{I}\sigma_e^2; \mathbf{D} = \mathbf{I}\sigma_a^2$$

- For the residual variance σ^2_e the same two options exist (estimate or « fix »)

- There are technical, but important, details associated with use of precorrected data (DYD's)

Residual variance with pseudo-data

- What is the residual variance if we use DYD's?
- DYD=performance of the daughters, corrected by dams' BVs and other effects. Ideally:

$$2DYD_i = u_i + 2\frac{1}{n_i}\sum_j \phi_j + 2\frac{1}{n_i}\sum_j e_j = u_i + \varepsilon_i$$

Bull BV

Mendelian sampling of his daughters

« True » residuals

« Pseudo » residuals

But $Var(\phi) = \frac{1}{2}\sigma_u^2$

n_i=« edc », equivalent daughter contribution

And therefore $Var(\varepsilon_i) = \frac{1}{n_i}\left(2\sigma_u^2 + 4\sigma_e^2\right)$

This is an approximation because we don't know exactly the other effects or dam's genetic values

Heterogenous residual variances

- In dairy, it's typical to use DYD's or some kind of deregressed proofs with varying precision (edc's)
  - e.g.; **y= Xb** + **Za** +**ε** ; Var(ε) = **F**σ^2_ε

$$\sigma^2_\varepsilon = 2\sigma^2_u + 4\sigma^2_e$$

if using DYD's

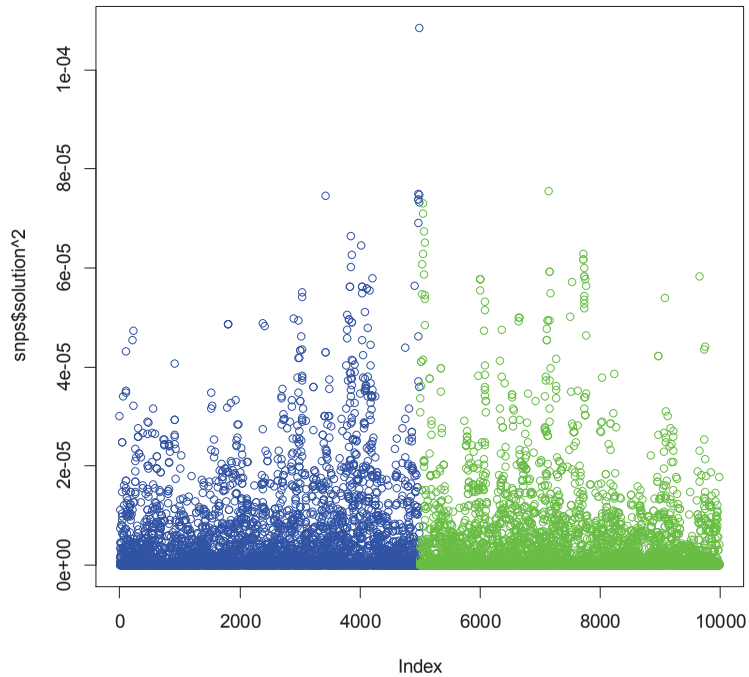
  - **F** is tipically assumed diagonal with 1/edc$_i$ for the i-th bull

- How do we fit this in a BLUP_SNP context ? (trick by G de los Campos, detailed in doi:10.1017/S0016672310000534)
  - the model above can be transformed in an equivalent model, yielding the same solutions

$$\mathbf{y}^* = \mathbf{F}^{-1/2}\mathbf{y};\ \mathbf{X}^* = \mathbf{F}^{-1/2}\mathbf{X};\ \mathbf{Z}^* = \mathbf{F}^{-1/2}\mathbf{Z};\ \mathbf{e}^* = \mathbf{F}^{-1/2}\mathbf{e}$$

    - multiply each element in **y** by sqrt(edc$_i$)
    - multiply each row of **X** and **Z** by sqrt(edc$_i$)
      - then, **y*****= X*****b** + **Z*****a** +ε* **;** Var(ε*) = **I**σ^2_ε
  - in blupf90 and GS3, this is done using edc's as WEIGHTS

What to do with estimates



BLUP_SNP output

- Only SNP effects
- We can in principle obtain their s.e. but they are never significant – estimates are too shrunken towards 0
  - And it doesn't make much sense
- So no clear way of declaring "significance"
- Perhaps some local-FDR ideas (Efron)?

3- SNP- based models for Genomic evaluation. Bayesian methods: BayesA, BayesB, BayesCPi, Lasso

Andrés Legarra - INRA

Estimating variances = BayesC (with π=0)

- It simply consists in a BLUP_SNP where we estimate (and simultaneously « integrate out ») σ^2_a and σ^2_e
  - i.e., a regular Gibbs sampler applied to SNPs instead of EBV's (Gibbs-SNP ??)
  - Legarra et al., 2008 (we didn't call it BayesC), Habier et al., 2011

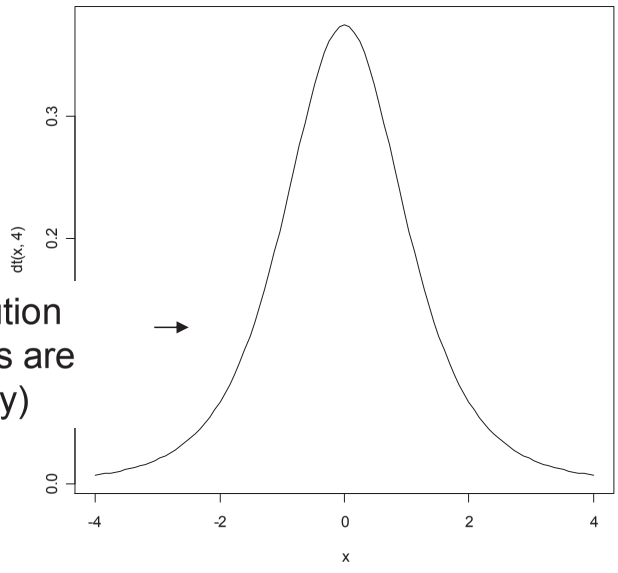
$$\mathbf{y} = \mathbf{Xb} + \mathbf{Za} + \mathbf{e},$$

$$\mathbf{a} \mid \sigma_a^2 \sim MVN\left(\mathbf{0}, \mathbf{I}\sigma_a^2\right); \sigma_a^2 \sim S_a \chi_\nu^{-2}$$

$$\mathbf{e} \mid \sigma^2 \sim MVN\left(\mathbf{0}, \mathbf{I}\sigma_e^2\right); \sigma_e^2 \sim S_e \chi_\nu^{-2}$$

BayesA (*t* distribution)

a *t* distribution (big effects are not unlikely)



representation as « t »

$$a_i \sim t\left(0, \nu, \sigma_a^2\right)$$

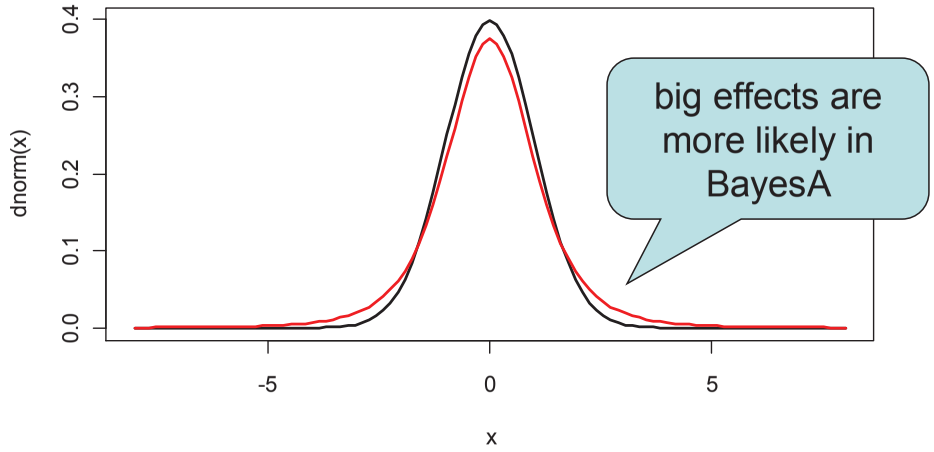
$$\equiv$$

$$a_i \sim N\left(0, \sigma_{a,i}^2\right) \chi_\nu^{-2} \sigma_a^{-2}$$

Meuwissen et al. representation

Gianola et al. (2009) proved that fitting a variance by locus is equivalent to postulating *t* distribution for all locus

Normal vs. BayesA



BayesA

- We « estimate » a different σ^2_a *for each SNP*
  - this estimate is horribly bad
  - but SNP solutions correspond to a model with « t » distributions

$$\mathbf{y} = \mathbf{Xb} + \mathbf{Za} + \mathbf{e},$$

$$\mathbf{e} \mid \sigma^2 \sim MVN\left(\mathbf{0}, \mathbf{I}\sigma_e^2\right); \sigma_e^2 \sim S_e \chi_{ve}^{-2}$$

$$\left\{ \begin{array}{c} a_i \sim t\left(0, \nu_a, \sigma_a^2\right) \\ \equiv \\ a_i \sim N\left(0, \sigma_{a,i}^2\right); \sigma_{a,i}^2 \sim S_a \chi_{va}^{-2} \end{array} \right\}$$

representation as « t »

$$a_i \sim t\left(0, \nu, \sigma_a^2\right)$$

$$\equiv$$

$$a_i \sim N\left(0, \sigma_{a,i}^2\right) \chi_\nu^{-2} \sigma_a^{-2}$$

Meuwissen et al. representation

- Pretty straightforward from GSRU

Why not BayesA

- In BayesA it is unclear what “is” a QTL and what “is not” a QTL
- And typically results from BayesA don’t “show” QTLs

BayesCPi

- e.g. Habier et al., 2011 (see also Rohan Fernando course notes)
- What if some SNP had no effect?
  - This is the original idea of BayesB
  - needs the probability that a given SNP is at the model or not
  - can be computed by MCMC

$$\mathbf{y} = \mathbf{Xb} + \mathbf{Za} + \mathbf{e},$$

$$\mathbf{a} \mid \boldsymbol{\delta}, \sigma_a^2 \sim \begin{cases} a_i \sim N\left(\mathbf{0}, \sigma_a^2\right) if\ \delta_i = 1 \\ a_i = 0\ if\ \delta_i = 0 \end{cases}$$

$$\sigma_a^2 \sim S_a \chi_v^{-2}$$

$$\delta_i \sim \begin{cases} 0\, with\ probability\ \pi \\ 1\ with\ probability\ 1-\pi \end{cases}$$

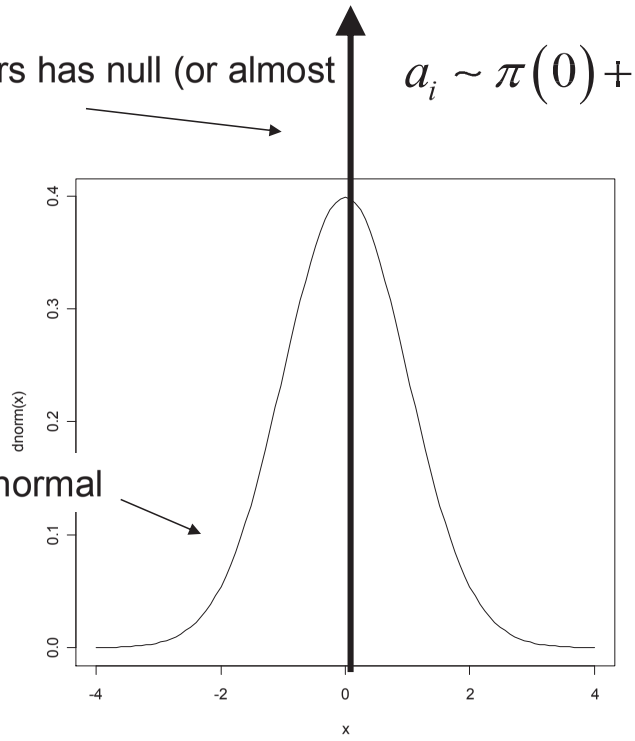
π can be fixed or estimated

$$\mathbf{e} \mid \sigma^2 \sim MVN\left(\mathbf{0}, \mathbf{I}\sigma_e^2\right); \sigma_e^2 \sim S_e \chi_v^{-2}$$

Mixture distribution or BayesC(Pi)

A fraction π of markers has null (or almost null) effects.

$$a_i \sim \pi(0) + (1-\pi)\, N\left(0, \sigma_a^2\right)$$



Otherwise they are normal

BayesCPi

- Back to the idea that each SNP has a variance
  - « In » the model: $\delta_i = 1 \rightarrow \sigma_{ai}^2 = \delta_i \sigma_a^2 = \sigma_a^2$
  - « Not in » the model: $\delta_i = 0 \rightarrow \sigma_{ai}^2 = \delta_i \sigma_a^2 = 0$
  - This is more explicit than SNP effects
  - In practice, when $\sigma_{ai}^2 \rightarrow \infty$ the estimate of the SNP is like a fixed effect

This is the same variance for all SNP in the model

- For a single SNP, the probability of including it in the model is a function of the apparent effect of the SNP and the « expected » fraction of SNPs in the model
- $p(\delta_i = 1) = p(\boldsymbol{y}|\delta_i = 1)\, p(\delta_i = 1)$

Prior information π (how many SNPs we enter in the model)

Likelihood of data if we fit this SNP

BayesCPi

- Algorithm consists in a BLUP_SNP by GSRU where we estimate (and simultaneously « integrate out ») σ^2_a and σ^2_e
  - for each SNP we compute the probability of it being « in » the model (indicator variable δ)
    - This was a nightmare in original BayesB
    - R Fernando found out a simple way of computing it (course notes: http://www.ans.iastate.edu/stud/courses/short/2010short.html) that is « like » GSRU
  - we can equally compute the proportions π or fix them previously

Fortran pseudocode for BayesCPi

```
...
do j=1,niter
   do i=1,neq
      ...
       ! compute loglikelihood for state 1 (i -> in model) and 0 (not in model)
       ! Notes by RLF (2010, Bayesian Methods in
       ! Genome Association Studies, p 47/67)
       v1=xpx(i)*vare+(xpx(i)**2)*vara
       v0=xpx(i)*vare
       rj=rhs*vare ! because rhs=X'R-1(y corrected)
       ! prob state delta=0
       like0=density_normal((/rj/),v0) !rj = N(0,v0)
       ! prob state delta=1
       like1=density_normal((/rj/),v1) !rj = N(0,v1)
       ! add prior for delta
       like1=like1*pi; like0=like0*(1-pi)
       !standardize
       like0=like0/(like0+like1); like1=like1/(like0+like1)
       delta(i)=sample(states=(/0,1/),prob=(/like0,like1/)
       if(delta(i)==1) then
         val=normal(rhs/lhs,1d0/lhs)
       else
         val=0
       endif
    ...
   enddo
   pi=1- beta(count(delta==1)+apriori_included,count(delta==0)+apriori_not_included)
   ss=sum(sol**2)+nua*Sa
   vara=ss/chi(nua+count(delta==1))
   …
enddo
```

Likelihood of data if we fit this SNP

Prior information π (how many SNPs we enter in the model)

BayesCPi

- So far this looks simple
- But BayesCPi has many details & caveats
  - How to run the Gibbs sampler?
    - Rule of thumb: iterate $100 > n > 5$ times the number of markers
      - (need to find the good combination of markers)
  - Do we estimate or fix π ? At which values?
  - What do we get as results?

BayesCPi

- Parameter π (or $(1 - \pi)$) is the number of SNPs in the model
- Do we estimate or fix it?
  - In theory we can estimate it
  - In practice it is very tricky
    - Colombani et al. could estimate it in Holstein but not in Montbéliarde (estimate too imprecise)
  - Usually we “fix” it to 1/1000 (50 SNP out of 50,000)

BayesCPi

- Parameter π and genetic variance
- In the case of BayesCPi, $\sigma_g^2 = 2\pi\sum p_i q_i \sigma_a^2$
- So, to recover all genetic variance from SNPs, we need to modify π and σ_a^2 at the same time
  - Then $\sigma_a^2 = \frac{\sigma_g^2}{2\pi\sum p_i q_i}$
- So, $\pi = 0.001$ implies that σ_a^2 is 1000 times larger than in BLUP_SNP and estimates are less "shrunken"

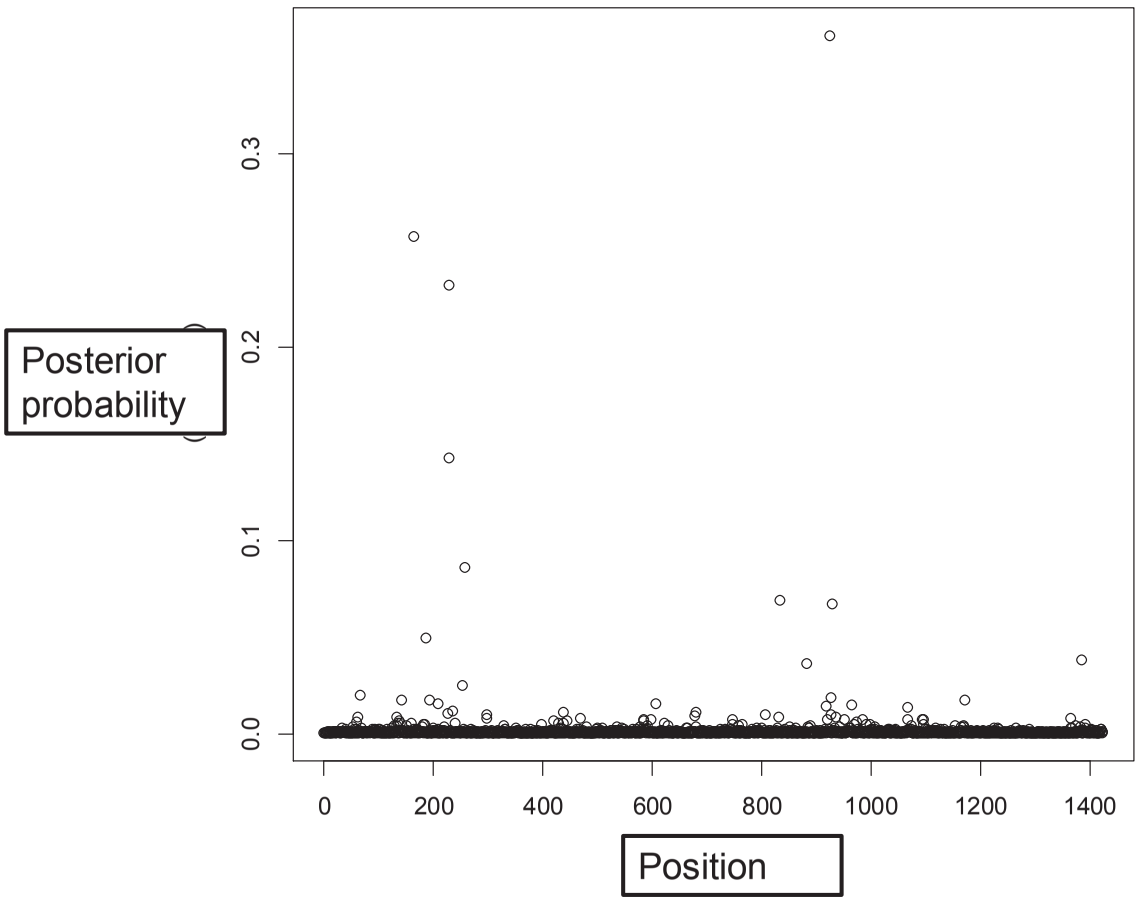
BayesCPi

- Output of BayesCPi
  - For each SNP, the marginal posterior probability of being "in the model"
  - Not a single subset of SNPs "in the model"

effect	level	solution	sderror	p
2	1	0.49637122E-02	0.63842196E-01	0.69375000E-02
2	2	0.49501460E-03	0.17864670E-01	0.10375000E-02
2	3	0.38664734E-04	0.79524430E-02	0.32500000E-03
2	4	0.18222423E-04	0.59148438E-02	0.25000000E-03
2	5	0.21643136E-03	0.11477947E-01	0.53750000E-03
2	6	-0.55016190E-03	0.28990326E-01	0.97500000E-03
2	7	0.94168849E-04	0.74293395E-02	0.28750000E-03

- This implies that most SNP are in LD with some QTL somewhere
- Sometimes, a single SNP stands out → large QTL

OAR12, Salle et al. (JAS)



BayesCPi

- How do we declare a “positive” QTL?
- Have no p-values in this analysis
  - Bayesians insist in using the Bayes Factor for that (Wakefield; Bertrand & Stephens, etc.) but no clear rules how
- Construct the Bayes Factor:

$$\frac{\frac{p(SNP\ in\ the\ model|data)}{p(SNP\ not\ in\ the\ model|data)}}{\frac{p(SNP\ in\ the\ model)}{p(SNP\ not\ in\ the\ model)}}$$

Posterior «odds »

Prior « odds »

In our case:

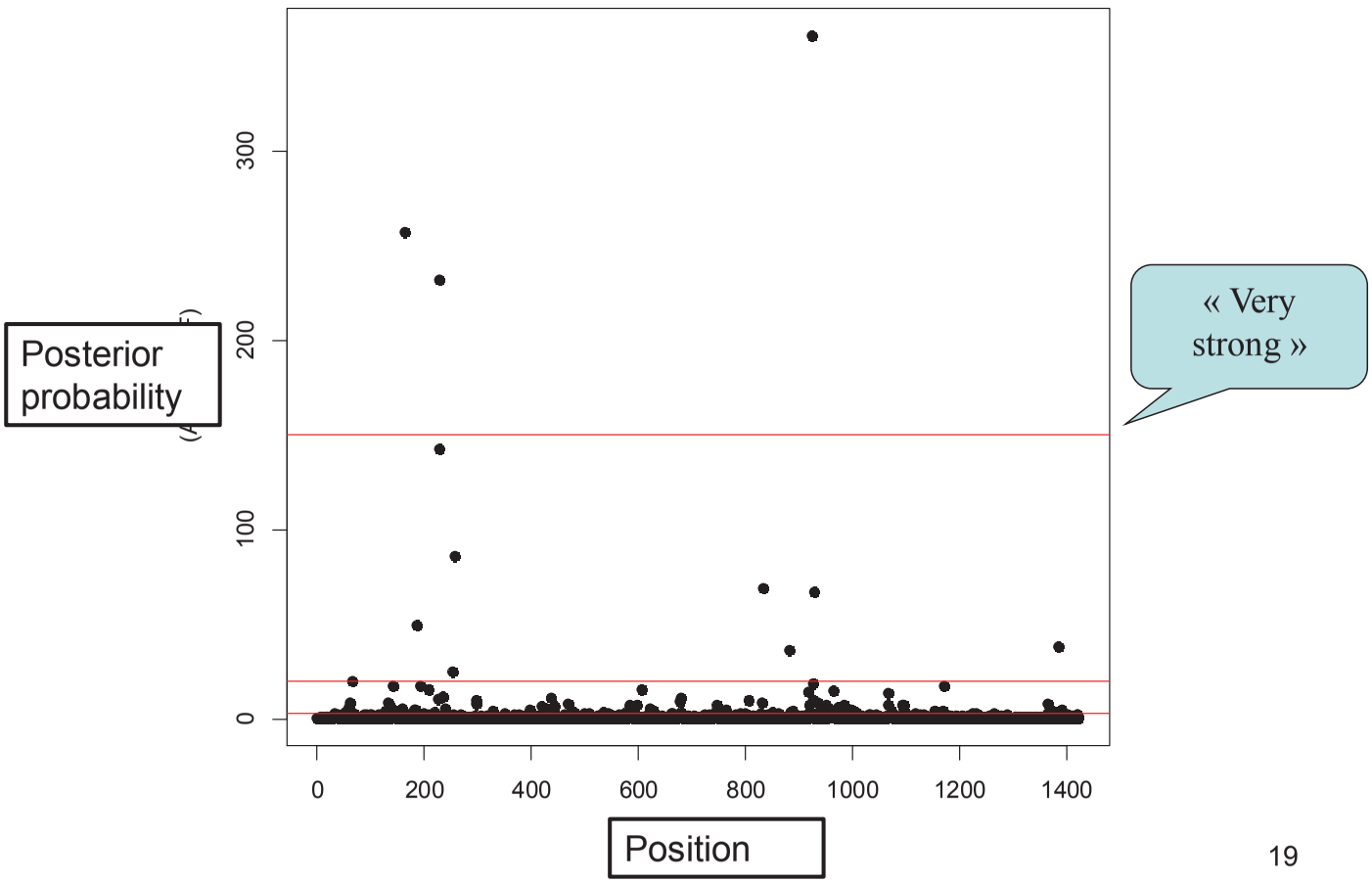
$$BF_i = \frac{(1-\pi)}{\pi}\frac{p(\delta_i=1|\boldsymbol{y})}{1-p(\delta_i=1|\boldsymbol{y})}$$

BayesCPi

- What thresholds for BF?
- Some people suggest using permutations → too long
- Use a scale adapted by Kaas & Raftery (1995) used in QTL detection by Varona et al. (2001, GSE) and Vidal et al. (2005, JAS)
  - BF= 3-20 "suggestive“
  - BF= 20-150 "strong “
  - BF>150 "very strong”

- We don’t need correction for multiple testing (Bonferroni):
  - all SNP were introduced at the same time
  - And the prior already « penalizes » their estimates

OAR12, Salle et al. (JAS)



BayesB

- e.g. Meuwissen et al., 2001
- What if some SNP had no effect in Bayes A?
  - This is the original idea of BayesB
  - needs the probability that a given SNP is at the model or not
  - can be computed by MCMC but is notoriously more difficult than BayesCPi (see for instance Villanueva et al., JAS)

$$\mathbf{y} = \mathbf{Xb} + \mathbf{Za} + \mathbf{e},$$

$$\mathbf{a} \mid \boldsymbol{\delta}, \sigma_a^2 \sim \begin{cases} a_i \sim t\left(\mathbf{0}, \sigma_a^2, \nu\right) if\ \delta_i = 1 \\ a_i = 0\ if\ \delta_i = 0 \end{cases}$$

$$\sigma_a^2 \sim S_a \chi_\nu^{-2}$$

$$\delta_i \sim \begin{cases} 0\ with\ probability\ \pi \\ 1\ with\ probability\ 1-\pi \end{cases}$$

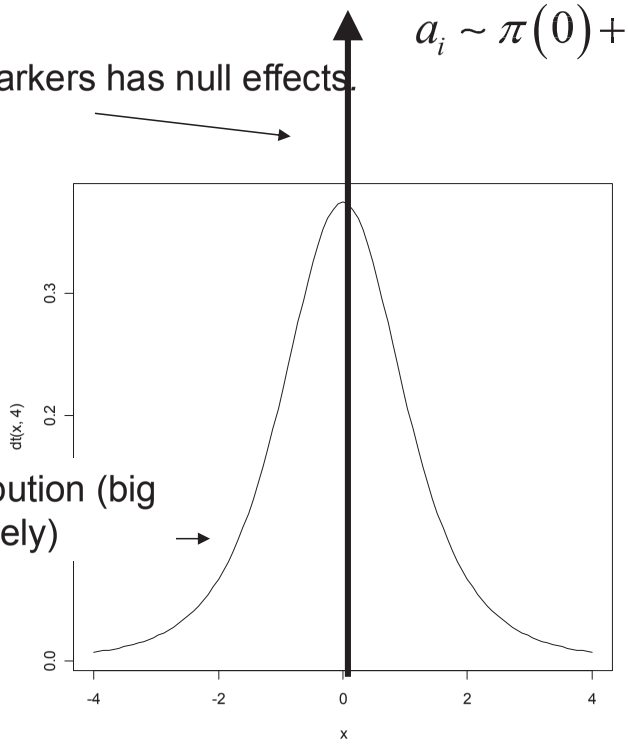
π is fixed

$$\mathbf{e} \mid \sigma^2 \sim MVN\left(\mathbf{0}, \mathbf{I}\sigma_e^2\right); \sigma_e^2 \sim S_e \chi_\nu^{-2}$$

BayesB (mixture with *t* distribution)

$$a_i \sim \pi(0) + (1-\pi)\, t\left(0, \nu, \sigma_a^2\right)$$

A fraction π of markers has null effects.



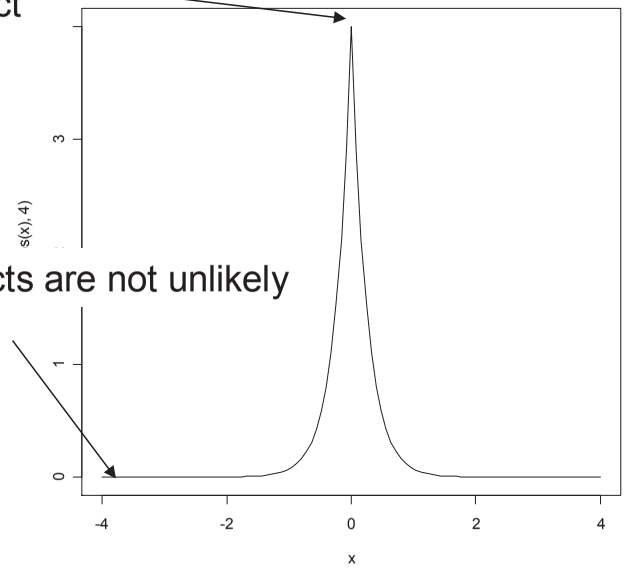
Otherwise a *t* distribution (big effects are not unlikely)

Lasso (double exponential)

$$a_i \sim \frac{\lambda}{2}\exp\left(-\lambda|a_i|\right)$$

Often marker has *almost* null effect

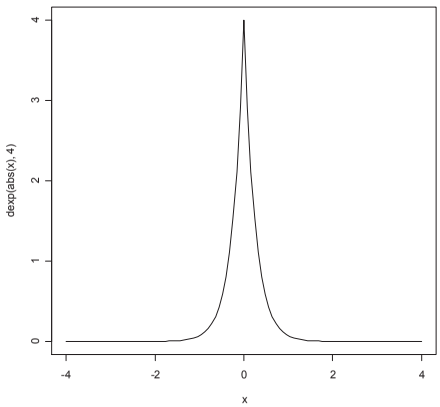
Otherwise big effects are not unlikely



Lasso

Hierarchical representation of Lasso

- **y** : data
- **a** : SNP effects



$$\mathbf{y} = \mathbf{Xb} + \mathbf{Za} + \mathbf{e},$$

$$\mathbf{a} \mid \lambda, \sigma^2 \sim \prod_i \frac{\lambda}{2}\exp\left(-\lambda|a_i|\right)$$

$$\mathbf{e} \mid \sigma^2 \sim MVN\left(\mathbf{0}, \mathbf{I}\sigma_e^2\right)$$

Distribution of SNP effects

Bayesian Lasso 1Var

- the Bayesian Lasso (Park & Casella 2008) uses an equivalent hierarchical model:
- Again, we give different variances to each SNP. Large SNP receive more variance
- But no SNP is fixed to 0

$$\boldsymbol{y} = \boldsymbol{Xb} + \boldsymbol{Za} + \boldsymbol{e}$$

$$p(\boldsymbol{a}|\boldsymbol{\tau}) \sim \boldsymbol{N}(\boldsymbol{0}, \boldsymbol{D}\sigma^2) \Rightarrow \boldsymbol{D} = \begin{bmatrix} \tau_1^2 & 0 & 0 & 0 \\ 0 & \tau_2^2 & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \tau_n^2 \end{bmatrix} \Rightarrow p(\boldsymbol{\tau}|\lambda) = \prod_i \frac{\lambda^2}{2} e^{-\lambda^2 \tau_i^2}$$

$$p(\boldsymbol{e}) \sim \boldsymbol{N}(\boldsymbol{0}, \boldsymbol{I}\sigma^2)$$

$\sigma_{ai}^2 = \tau^2 \, \sigma^2$

Prior for the variances

Bayesian Lasso 2 Var

- There is this another Bayesian Lasso (BL2Var) which I found to make more sense:

$$\boldsymbol{y} = \boldsymbol{Xb} + \boldsymbol{Za} + \boldsymbol{e}$$

$$p\left(\boldsymbol{a}\middle|\boldsymbol{\sigma_{ai}^2}\right) \sim \boldsymbol{N}(\boldsymbol{0}, \boldsymbol{D}) \Rightarrow \boldsymbol{D} = \begin{bmatrix} \sigma_{a1}^2 & 0 & 0 & 0 \\ 0 & \sigma_{a2}^2 & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \sigma_{an}^2 \end{bmatrix} \Rightarrow p(\sigma_{ai}^2|\lambda) = \prod_i \frac{\lambda^2}{2} e^{-\lambda^2 \sigma_{ai}^2}$$

$$p(\boldsymbol{e}) \sim \boldsymbol{N}(\boldsymbol{0}, \boldsymbol{I}\sigma_e^2)$$

σ_{ai}^2 are « variances » of SNP effects

Prior for the variances

Bayesian Lasso 2 Var vs. BayesA

$$\boldsymbol{y} = \boldsymbol{Xb} + \boldsymbol{Za} + \boldsymbol{e}$$

$$p(\boldsymbol{a}|\sigma_{ai}^2) \sim \boldsymbol{N}(\boldsymbol{0}, \boldsymbol{D}) \Rightarrow \boldsymbol{D} = \begin{bmatrix} \sigma_{a1}^2 & 0 & 0 & 0 \\ 0 & \sigma_{a2}^2 & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \sigma_{an}^2 \end{bmatrix}$$

σ_{ai}^2 are « variances » of SNP effects

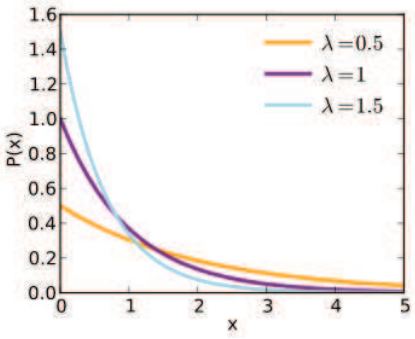
$$p(\boldsymbol{e}) \sim \boldsymbol{N}(\boldsymbol{0}, \boldsymbol{I}\sigma_e^2)$$

Distribution of the variances

BL

$$p(\boldsymbol{\tau}|\lambda) = \prod_i \frac{\lambda^2}{2} e^{-\lambda^2 \sigma_{ai}^2}$$

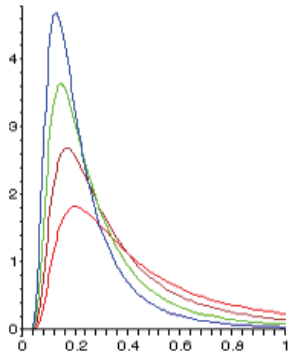
Exponential



BayesA

$$p(\sigma_{ai}^2|\lambda) \sim \prod_i \chi_\nu^{-2} S_a^{-2}$$

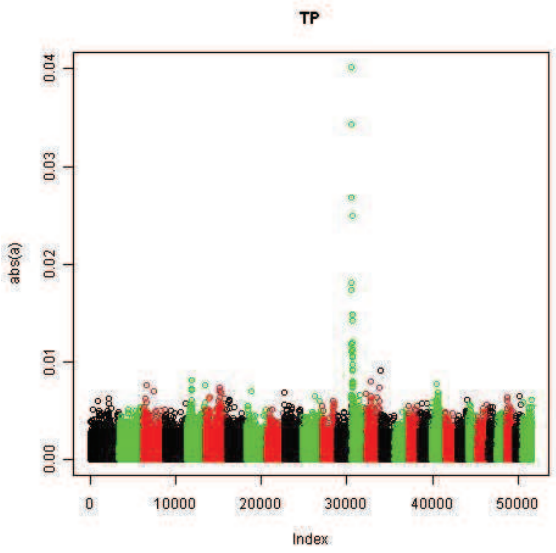
Inverted chi-squared



Bayesian Lasso

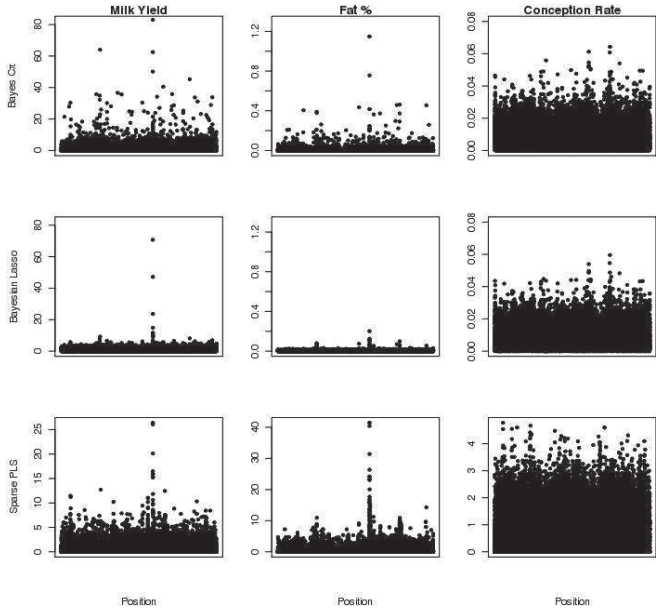
- It gives different weights to larger SNPs
- Mixing is better than BayesCPi
- Performance in Genomic Selection is as good (Colombani et al., 2013, JDS)
- But there is no clear notion of what SNP is a QTL and a few papers with “lasso for QTL” are disappointing.

- Holstein dairy cattle (Legarra et al., 2011)



C Colombani (JDS)

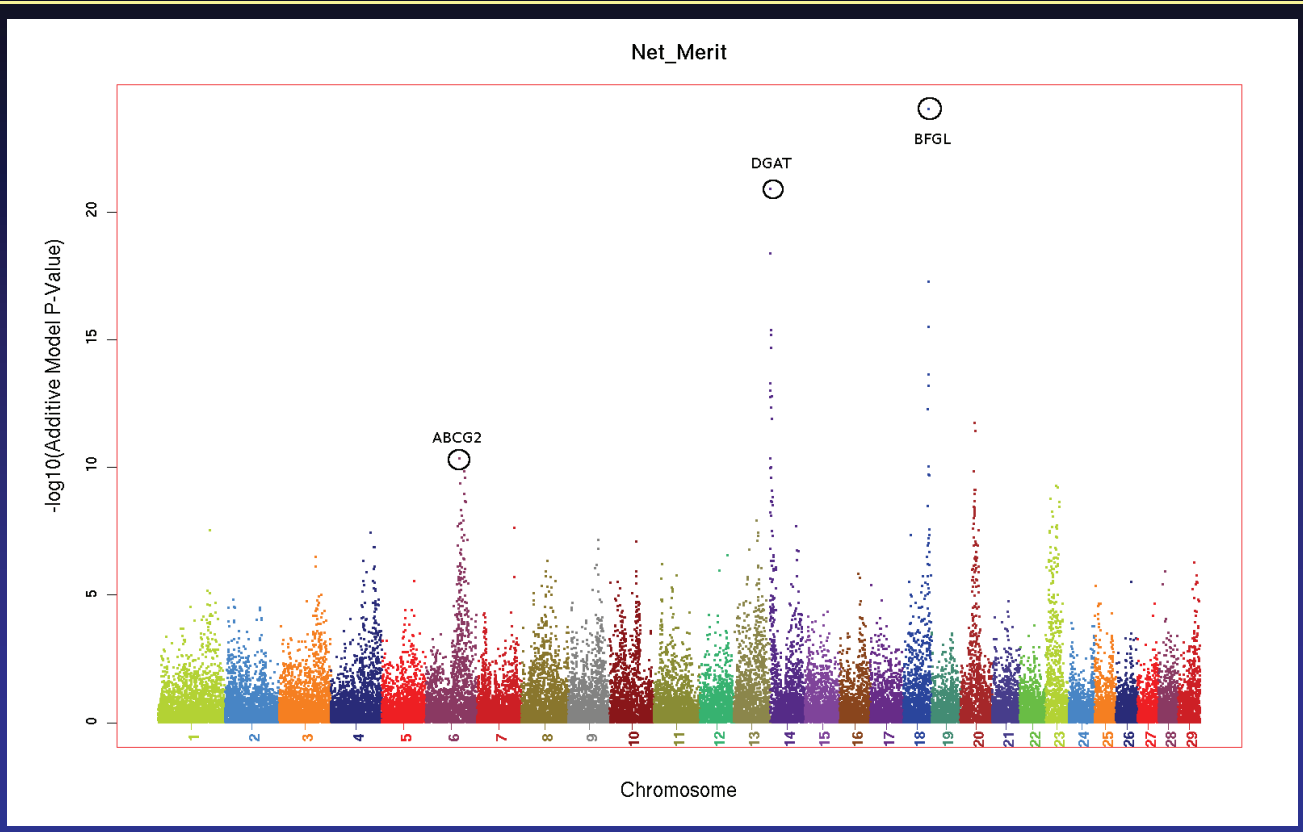
- Shape of effects



Non-MCMC methods

- So far we have seen mostly Gibbs
  - Lasso (and its cousin the Elastic Net) have very fast algorithms
    - The Lasso produces the optimal subset of SNP that explain $\boldsymbol{y}$ best
    - But has received very little attention
  - VanRaden's (2008) nonlinearA is a fast and efficient EM algorithm for BayesA,
  - There is also FastBayesB, fastBayesA
- No one uses these approaches at least in publications
  - Except at USDA : see next slide by John Cole (16th QTLMAS)

We got right to the point…



Cole, J.B. et al. 2009. Distribution and location of genetic effects for dairy traits. ICAR Tech Ser. 13:355–360.

Interest

- Multiple marker methods are nice because…
  - Handle very well (cryptic) relatedness
  - Show different things (potentially, QTLs hidden by large ones)
  - Do not overestimate enormously QTL effects (as typical GWAS or genome scan do)
  - Avoid multiple testing problem
- They are ugly because…
  - MCMC is looooong
  - No one knows how to declare significance

Advice

- Use everything: GWAS, LA, LDLA, BayesCPi, Bayesian Lasso
- Don't trust one method blindly because it seems formidable
- Multiple marker methods need attention to details: correct priors and initial values, computation time, verification of convergence
  - *GWAS is easier than BLUP_SNP that is easier than Bayesian Lasso that is easier than BayesCPi*
- Details are typically overlooked by most users !!!