# Alencar Xavier

- Quantitative Geneticist, Corteva Agrisciences
- Adjunct Professor at Purdue University
- https://alenxav.github.io/

## Outline

# Get some data

The code below fetches soybean data, a small subset from the SoyNAM project. We use two families (5,15) as training set, one (04) as prediction target. Phenotypes come from 10 incomplete blocks observed over 3 years.

```
tr = function(x) sum(diag(x)) # Trace function
data(met,package='NAM')
Obs$Sp = with(Obs,NAM::SPC(YLD,Block,Row,Col,4,4))
Obs$Block = factor(Obs$Block)
Gen2 = data.matrix(Gen[grep('-04',rownames(Gen)),])
Gen = data.matrix(Gen[grep('-05|-15',rownames(Gen)),])
Obs = droplevels(Obs[grep('-05|-15',Obs$ID),])
Obs = Obs[,c('Block','Sp','ID','YLD')]
Gen = Gen[,apply(Gen,2,var)>0.1]  # Observed individuals
Gen2 = Gen2[,colnames(Gen)] # Prediction target
```

## Get some data

```
dim(Obs)
## [1] 1117    4
dim(Gen)
## [1]  280 3974
head(Obs)
##     Block       Sp         ID   YLD
## 16      9 56.56612 DS11-15004 52.16
## 28      7 55.90857 DS11-05089 68.25
## 36      8 57.66000 DS11-05042 58.94
## 76      3 66.27364 DS11-05156 75.38
## 83      9 56.98647 DS11-15133 66.93
## 113     8 53.03104 DS11-15119 72.64
```

## Build GRM

From Habier et al. 2007 and VanRaden 2008, we have

$$A = \frac{(M-P)(M-P)'}{2\sum(1+f_j)p_j(1-p_j)}$$

Pre-center the markers ($\tilde{Z} = M - P$), then you get

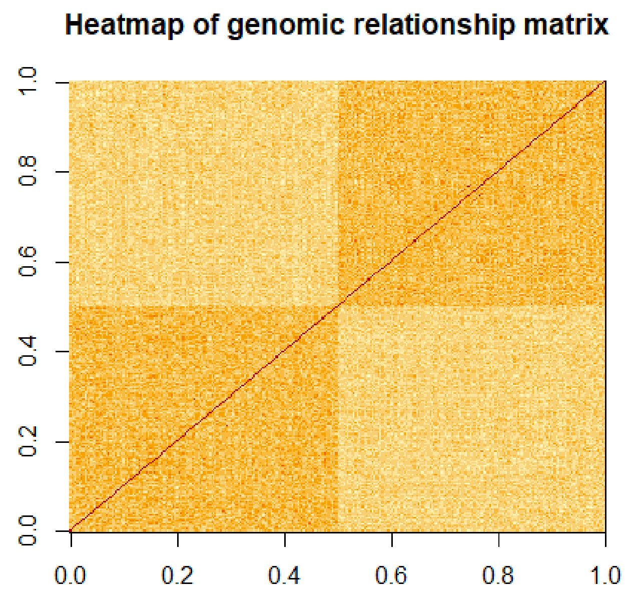
$$A = \alpha\tilde{Z}\tilde{Z}'$$

Where $\alpha^{-1} = n^{-1}\sum_{j=1}^{p}(\tilde{z}_j{}'\tilde{z}_j)$. Also, note that $A$ stands here for additive relationship, **not as a pedigree matrix**.

## Build GRM

```
Z1 = apply(Gen,2,function(x) x-mean(x)) # Observed
individuals
Z2 = apply(Gen2,2,function(x) x-mean(x)) # Prediction target
A = tcrossprod(Z1)
alpha = 1/mean(diag(A))
A = A*alpha
diag(A) = diag(A)+0.01 # Stabilize GRM
iA = solve(A) # Get inverse
```

# Build GRM



## Statistical model

- Linear model

$$y = Xb + Zu + e$$
$$y \sim N(Xb, V)$$

- Variances

$$Var(y) = V = ZGZ' + R$$
$$Var(u) = G = A\sigma_a^2$$
$$Var(e) = R = I\sigma_e^2$$

## Statistical model

The joint variance, including marker effects $\beta$, is defined as

$$Var\begin{bmatrix} y \\ u \\ \beta \\ e \end{bmatrix} = \begin{bmatrix} V & ZG & Z\tilde{Z}D & R \\ GZ' & G & \tilde{Z}D & 0 \\ D\tilde{Z}'Z' & D\tilde{Z}' & D & 0 \\ R & 0 & 0 & R \end{bmatrix}$$

where $u = \tilde{Z}\beta$, $\beta \sim N(0, D)$, $D = I_m\sigma_\beta^2$, and $G = A'\sigma_u^2 = \tilde{Z}D\tilde{Z}'$. The model could be described as:

$$y = Xb + Z\left(\tilde{Z}\beta\right) + e$$

## Matrices - Design matrices

```r
y = Obs$YLD
X = model.matrix(~Block+Block:Sp-1,Obs)
Z = model.matrix(~ID-1,Obs)

# Constants
n = length(y) # number of obs
q = ncol(Z) # levels of Z
rX = ncol(X) # rank of X

# If you are not sure X is full-rank, run
# rX = qr(X)$rank

# Check dimensions
print(c(n=n,q=q,rX=rX))
##     n    q   rX
## 1117  280   18
```

## Matrices - Variance starting values

Let’s assume a starting values of $h_0^2 \cong 0.25$. Now what?

If $\hat{\sigma}_{Y0}^2 = \frac{(y - Xb_{LS})\prime(y - Xb_{LS})}{n - r_X}$. Then,

$\hat{\sigma}_e^2 = (1 - h_0^2)\hat{\sigma}_{Y0}^2$ and $\hat{\sigma}_e^2 = h_0^2 \frac{\hat{\sigma}_{Y0}^2}{\sum_{j=1}^{q} \sigma_{Z_j}^2}$.

```
# Starting values for variance components
b = qr.solve(X,y)
vy0 = c(crossprod(y-X%*%b)/(n-rX))
vu = 0.25*vy0/sum(apply(Z,2,var))
ve = 0.75*vy0
vc = c(vu=vu,ve=ve)
print(vc)
```

```
##        vu        ve
## 15.42179 46.14134
```

# Matrices - Variance-covariances

```
I = diag(n)
R = I*ve
G = A*vu
ZAZ = Z %*% A %*% t(Z)
V = Z%*%G%*%t(Z) + R
iG = solve(G)
iR = solve(I*ve)
iV = solve(V)
```

## Henderson’s equation

- Henderson’s mixed model equation (HMME) for

$$y = Xb + Zu + e$$

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

$$\begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} \begin{bmatrix} \hat{g}_1 \\ \hat{g}_2 \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \end{bmatrix}$$

$$C\hat{g} = r$$

where $C$ is the *left-hand side*, $r$ is the *right-hand side*.

## Henderson’s equation

$$y = Xb + Zu + e$$

Alternative notation

$$y = Wg + e$$

where $W = \{X, Z\}$, and $g = \{b, u\}$. Thus, MME

$$[W'R^{-1}W + \Sigma]\hat{g} = W'R^{-1}y$$

where

$$\Sigma = \begin{bmatrix} 0 & 0 \\ 0 & G^{-1} \end{bmatrix}$$

## Henderson’s equation

System of equations

```
Sigma = Matrix::bdiag(diag(0,rX),iG)
W = cbind(X,Z)
C = t(W) %*% iR %*% W + Sigma
iC = solve(C)
r = t(W) %*% iR %*% y
```

Compute coefficients

```
g = iC %*% r
b = g[1:rX]
u = g[-c(1:rX)]
```

# Reduced animal model

The “reduced animal model” simplifies the mixed model equation

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

into

$$\begin{bmatrix} X'X & Z'X \\ X'Z & Z'Z + \lambda A^{-1} \end{bmatrix} \begin{bmatrix} \hat{b} \\ \hat{u} \end{bmatrix} = \begin{bmatrix} X'y \\ Z'y \end{bmatrix}$$

where $\lambda = \sigma_e^2 \sigma_u^{-2}$.

That is achieved by multiplying every thing by $\sigma_e^2$.

## Reduced animal model

From a data-augmentation standpoint, the reduced model is:

$$\begin{bmatrix} X'X & Z'X \\ X'Z & Z'Z \\ 0 & \lambda A^{-1} \end{bmatrix} \begin{bmatrix} \hat{b} \\ \hat{u} \end{bmatrix} = \begin{bmatrix} X'y \\ Z'y \\ 0 \end{bmatrix}$$

## Variance of BLUEs

Let $C^{-1}$ be described as

$$C^{-1} = \begin{bmatrix} C^{11} & C^{12} \\ C^{21} & C^{22} \end{bmatrix}$$

Useful because

$$Var(\hat{b}) = C^{11} = (X'V^{-1}X)^{-1}$$

Also, we'll need $C^{22}$ to estimate variances later

```
  C22 = as.matrix(iC[-c(1:rX),-c(1:rX)])
```

## Projection matrices

LS absorption

$$S = I - X(X'X)^{-1}X'$$

BLUE absorption

$$M = I - X(X'V^{-1}X)^{-1}X'V^{-1}$$

Random effect absorption

$$V^{-1} = R^{-1} - R^{-1}Z(Z'R^{-1}Z + G^{-1})^{-1}Z'R^{-1}$$

Mixed model absorption

$$P = V^{-1} - V^{-1}X(X'V^{-1}X)^{-1}X'V^{-1}$$

## Projection matrices

$$H = WC^{-1}W'R^{-1}$$
$$Hy = \hat{y}$$
$$P = R^{-1}(I - H)$$

```
S = I - X %*% solve( t(X)%*%X ) %*% t(X)
M = I - X %*% solve( t(X) %*% iV %*%X ) %*% t(X) %*% iV
P = iV %*% M
H = W %*% iC %*% t(W) %*% iR
yHat = H %*% y
e = y - yHat
```

# Projection matrices - Identities

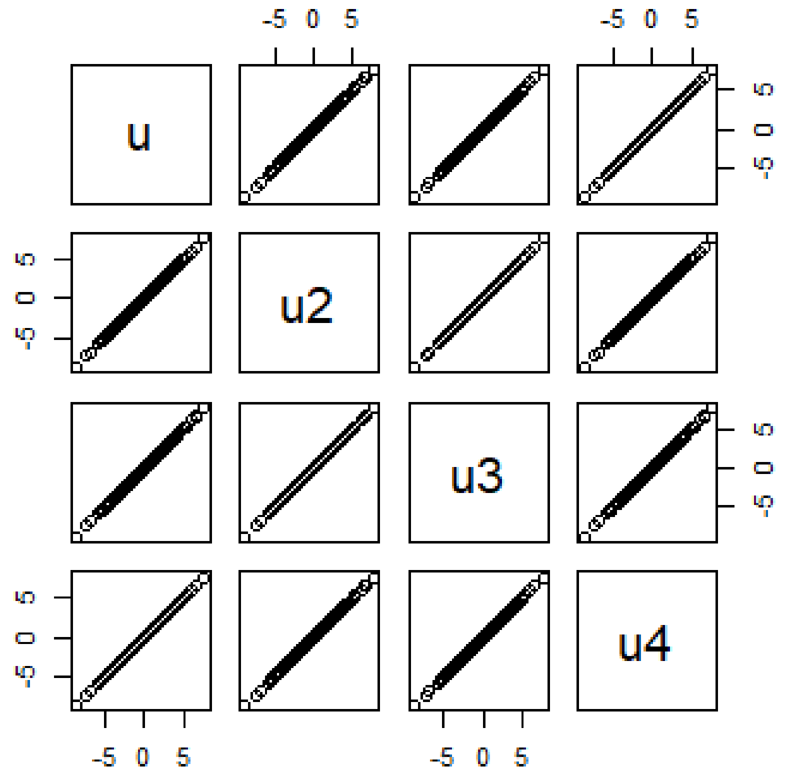
$$SX = 0$$
$$PX = 0$$
$$SS = S$$
$$MX = 0$$
$$My = y - X\hat{b}$$
$$MM = M$$
$$MS = M$$
$$SM = S$$
$$PVP = P$$
$$P = V^{-1}M$$
$$P = (XX'\sigma_{\infty}^{2} + V)^{-1}$$
$$ZAZ'Py = \frac{Z\hat{u}}{\hat{\sigma}_{u}^{2}}$$
$$Py = \frac{\hat{e}}{\hat{\sigma}_{e}^{2}}$$

## BLUP solutions

$$\hat{u} = \hat{g}_2$$
$$\hat{u} = GZ'Py$$
$$\hat{u} = (Z'R^{-1}Z + G^{-1})^{-1}Z'R^{-1}\left(y - X\hat{b}\right)$$
$$\hat{u} = (Z'Z + \sigma_e^2\sigma_u^{-2}A^{-1})^{-1}Z'\left(y - X\hat{b}\right)$$

## BLUP solutions

```
# Other ways to get BLUPs
u2 = G %*% t(Z) %*% P %*% y
u3 = solve( t(Z)%*%iR%*%Z + iG , t(Z)%*%iR%*%c(y-
X%*%b) )
u4 = solve( t(Z)%*%Z + iA*(ve/vu) , t(Z)%*% c(y-
X%*%b) )
plot(data.frame(u,u2,u3,u4))
```



## Variance of BLUPs

The variance of $u$ is

$$Var(u) = G$$

However, the variance of $\hat{u}$

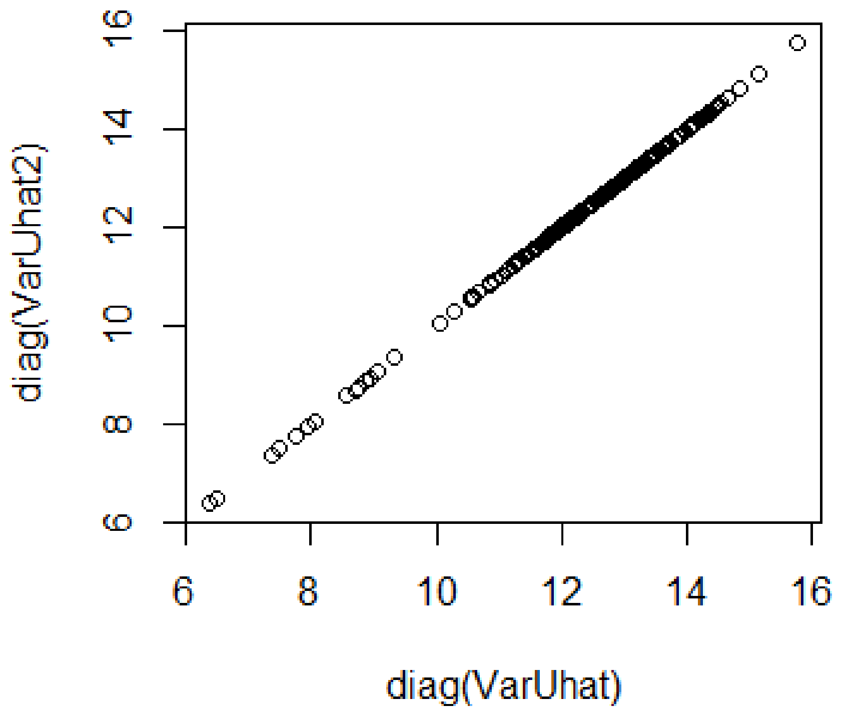
$$Var(\hat{u}) = GZ'PZG$$

Or, in terms of C,

$$Var(\hat{u}) = G - C^{22}$$

## Variance of BLUPs

```
# Variance of u hat
VarUhat = G %*% t(Z) %*% P %*% Z %*% G
VarUhat2 = G - C22
plot(diag(VarUhat),diag(VarUhat2))
```



## Reliability

Reliability is the observation-level heritability ($r^2$)

Under ML

$$r^2 = Z'V^{-1}ZG$$

Under REML

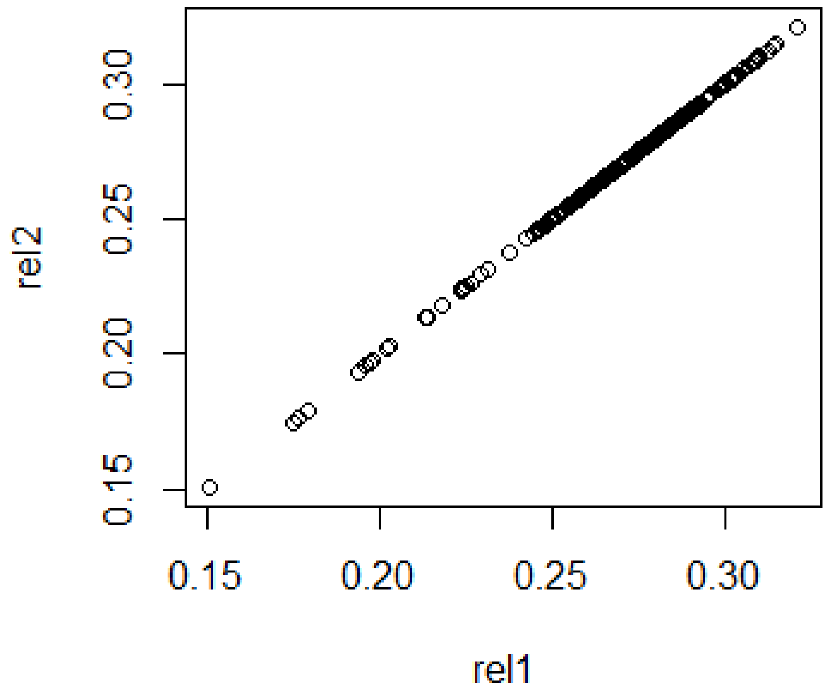
$$r^2 = Z'P^{-1}ZG$$
$$r^2 = I - C^{22}G^{-1}$$

Alternatively

$$r^2 = \frac{Var(u)}{Var(\hat{u})}$$

```
rel1 = diag(diag(q)-C22 %*% iG)
rel2 = diag( t(Z) %*% P %*% Z %*% G )
plot(rel1,rel2)
```

## Reliability



## Accuracy

Accuracy is the square root of reliability ($a = \sqrt{r^2}$). Accuracy is generally defined as the correlation between estimated and true breeding values.

$$a = cor(u, \hat{u}) = \frac{cov(u, \hat{u})}{sd(u) \times sd(\hat{u})}$$

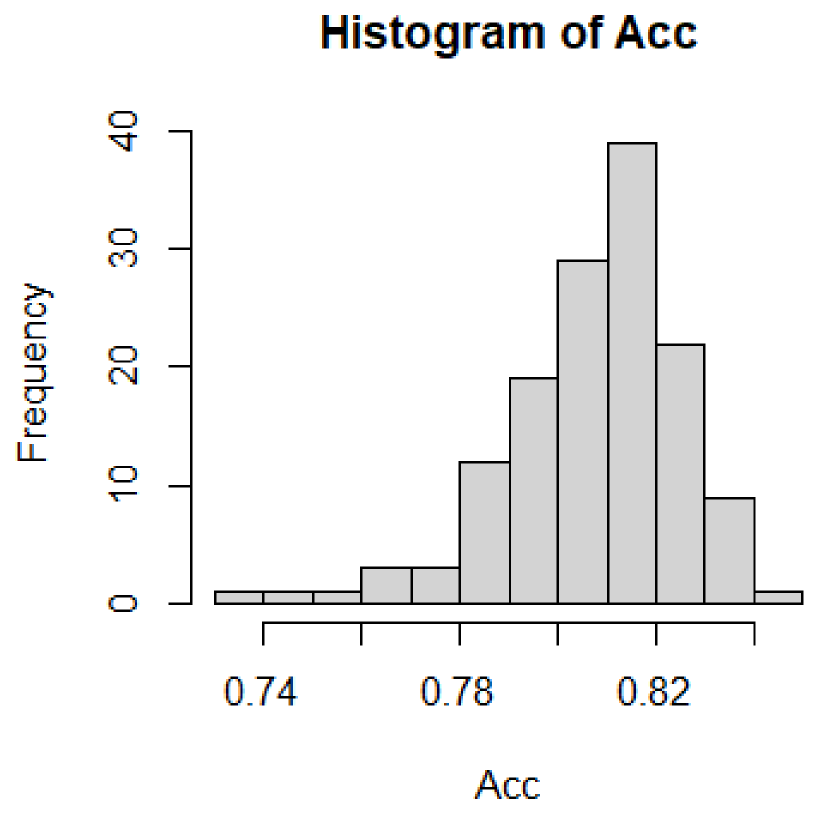
If the statistical model is the true model, $\sigma_i^2 = \hat{\sigma}_i^2$

$$a_i = \sqrt{\frac{cov(u, \hat{u})^2}{var(\hat{u})}} = \sqrt{\frac{G_{iy} Z' V^{-1} Z G_{yi}}{G_{ii}}}$$

When all individuals are observed, the denominator $G_{ii}$ cancels out with the nominator $G_{iy}$, yielding $a = \sqrt{Z' V^{-1} Z G}$.

## Accuracy

```
R2_by_ind = function(z){
Cov12=(z%*%t(Z1)*alpha)*vu
  Var22 = c(crossprod(z)*alpha)*vu
  R2 = Cov12%*%t(Z)%*%iV%*%Z%*%t(Cov12)/Var22}
Acc = sqrt(apply(Z2,1,R2_by_ind)); hist(Acc)
```



## REML

Gaussian likelihood

$$L(X|\theta) = \frac{exp(y - Xb)'V^{-1}(y - Xb)}{2\pi|V|}$$

Log-likelihood

$$-2LL(X|\theta) = ln|V| + (y - Xb)'V^{-1}(y - Xb)$$

Restricted Log-likelihood (REML)

$$-2LL(X|\theta) = ln|V| + ln|X'V^{-1}X| + (y - Xb)'V^{-1}(y - Xb)$$

REML derived as pseudo-random

$$-2LL(X|\theta) = ln|P| + y'Py$$

# Expectation-Maximization

We EM simply by rearranging $\partial L / \partial \sigma_i^2$.

Genetic variance

$$\partial L / \partial \sigma_u^2 = tr(PZAZ') - y'PZAZ'Py$$

$$\partial L / \partial \sigma_u^2 = \frac{q}{\sigma_u^2} - \frac{u'A^{-1}u}{\sigma_u^4} - \frac{tr(A^{-1}C^{22})}{\sigma_u^4}$$

Residual variance

$$\partial L / \partial \sigma_e^2 = tr(PI) - y'PIPy = tr(P) - y'P^2y$$

$$\partial L / \partial \sigma_e^2 = \frac{n - r_X}{\sigma_e^2} - \frac{q}{\sigma_e^2} - \frac{tr(A^{-1}C^{22})}{\sigma_e^2 \sigma_u^2} - \frac{e'e}{\sigma_e^4}$$

# Expectation-Maximization

The final estimators turn out:

$$\hat{\sigma}_u^2 = \frac{\hat{u}'A^{-1}\hat{u} + tr(A^{-1}C^{22})}{q} = \frac{\hat{u}'A^{-1}\hat{u}}{q - tr(G^{-1}C^{22})}$$

$$\hat{\sigma}_e^2 = \frac{\hat{e}'\hat{e} + tr(WC^{-1}W')\hat{\sigma}_e^2}{n} = \frac{y'e}{n - r_X}$$

## Expectation-Maximization

```
# Var e
Ve = c(crossprod(y,e)) / (n-rX)
# Var U
Vu = c(t(u) %*% iA %*% u + tr(iA%*%C22)) / q
# Var U (faster converging alternative)
Vu2 = c(t(u) %*% iA %*% u) / ( q - tr(iG%*%C22) )
# Check
print(c(Ve=Ve, Vu=Vu, Vu2=Vu2))
##        Ve       Vu      Vu2
## 52.18652 14.67906 12.66814
```

# Pseudo-Expectation

Simplification of the likelihood. From

$$\partial L / \partial \sigma_i^2 = tr(PV_i) - y'PV_i'Py$$

To

$$\partial PL / \partial \sigma_i^2 = tr(SV_i) - y'SV_i'Py$$

Yielding (Schaeffer 1986, VanRaden 1988):

$\hat{\sigma}_u^2 = \frac{\tilde{u}'\hat{u}}{tr(AZ'SZ)}$ and $\hat{\sigma}_e^2 = \frac{y'S'\hat{e}}{n - r_X}$

where $\tilde{u} = ySZ$.

## Pseudo-Expectation

```
Sy = S %*% y
ZSy = t(Z) %*% Sy
trAZSZ = tr( S %*% ZAZ ) # In practice, only diagonals are
computed
# Var U
c(u %*% ZSy) / trAZSZ
## [1] 9.506624
# Var E
c(t(e) %*% Sy) / (n-rX)
## [1] 52.18652
```

**P.S.**: Pseudo-Expectation works great for SNP-BLUP.

## MIVQUE

Minimum Variance Quadratic Unbiased Estimator (Rao 1971):

- Unbiased variances: $E[\hat{\sigma}] = \sigma$
- Quadratic unbiasedness: $E[y'Qy] = E[\epsilon_i{}'Q\epsilon_i] = 0$
- Invariant: $QX = 0$
- $y'Qy = y'Q\epsilon_0 + y'Q\epsilon_1$ where $\epsilon_0 = e$ and $\epsilon_1 = Zu$.
- Iterations of MIVQUE yields REML, we use $Q = P$.
- General formulation:

$$\hat{\sigma} = S^{-1}q$$

Where $S$ and $q$ are defined next.

## MIVQUE

$$\hat{\sigma} = S^{-1}q$$

Terms are defined as

$$S = \begin{bmatrix} P\frac{\partial V}{\partial \sigma_u^2}P\frac{\partial V}{\partial \sigma_u^2} & P\frac{\partial V}{\partial \sigma_u^2}P\frac{\partial V}{\partial \sigma_e^2} \\ P\frac{\partial V}{\partial \sigma_e^2}P\frac{\partial V}{\partial \sigma_u^2} & P\frac{\partial V}{\partial \sigma_e^2}P\frac{\partial V}{\partial \sigma_e^2} \end{bmatrix}$$

$$q = \begin{bmatrix} y'P\frac{\partial V}{\partial \sigma_u^2}Py \\ y'P\frac{\partial V}{\partial \sigma_e^2}Py \end{bmatrix}$$

where $\frac{\partial V}{\partial \sigma_u^2} = ZAZ'$ and $\frac{\partial V}{\partial \sigma_e^2} = I$

# MIVQUE

```
PZAZ = P %*% ZAZ
S = matrix(c(
  tr( PZAZ %*% PZAZ ),    tr( PZAZ %*% P ),
  tr( P %*% I %*% PZAZ ), tr( P %*% P     )),2,2)
qs = c(Vu = c(t(y) %*% PZAZ %*% P %*% y),
       Ve = c(t(y) %*% P %*% P %*% y))
solve(S,qs)
## [1]  8.495934 54.161688
```

# Average-Information

Second-derivative methods (NR, FS, AI) work with:

$$\theta^{t+1} = \theta^{t} - \frac{f'}{f''}$$

Where $f'$ and $f''$ are first and second derivatives, respectively. Where $f' = \Delta(\theta)$ is computed as

$$\Delta\left(\sigma_i^2\right) = \partial L / \partial\sigma_i^2 = tr\left(P\frac{\partial V}{\partial\sigma_i^2}\right) - y'P\frac{\partial V}{\partial\sigma_i^2}Py$$

And the (average-Information) $f'' = AI(\theta)$ is computed as:

$$AI\left(\sigma_i^2, \sigma_j^2\right) = y'P\frac{\partial V}{\partial\sigma_i^2}P\frac{\partial V}{\partial\sigma_j^2}Py$$

## Average-Information

```
SecDer1 = matrix(c(
  t(y)%*%PZAZ%*%PZAZ%*%P%*%y, t(y)%*%PZAZ%*%P%*%P%*% y,
  t(y)%*%P%*%PZAZ%*%P%*%y, t(y) %*%P%*%P%*%P%*%y ),2,2)
FirDer1 = c( vu = tr(PZAZ) - t(y) %*% PZAZ %*% P %*% y ,
             ve = tr(P %*% I) - t(y) %*% P %*% P %*% y )
vc - solve(SecDer1,FirDer1)
##         vu        ve
##   5.614032 53.308581
SecDer1 # AI matrix
##             [,1]       [,2]
## [1,] 0.12285971 0.04612028
## [2,] 0.04612028 0.53927449
```

## Average-Information

Building $V$ and $P$ is often **not feasible**. Let's check how to solve using the mixed model equations (Meyer 1997). We have seen solutions for first derivative, $\Delta(\theta)$, through $C$:

$$\partial L/\,\partial\sigma_u^2 = \frac{q}{\sigma_u^2} - \frac{u'A^{-1}u}{\sigma_u^4} - \frac{tr(A^{-1}C^{22})}{\sigma_u^4}$$

$$\partial L/\,\partial\sigma_e^2 = \frac{n - r_X}{\sigma_e^2} - \frac{q}{\sigma_e^2} - \frac{tr(A^{-1}C^{22})}{\sigma_e^2\sigma_u^2} - \frac{e'e}{\sigma_e^4}$$

## Average-Information

The $AI(\theta)$ is obtained as follows.

$$M_B = \begin{bmatrix} C & W'R^{-1}B \\ B'R^{-1}W & B'R^{-1}B \end{bmatrix}$$

$$B = [ZAZ'Py \quad Py] = [Zu\sigma_u^{-2} \quad e\sigma_e^{-2}]$$

Take the Cholesky

$$M_B = LL'$$

Then we reconstruct the sub-matrix corresponding to $B'R^{-1}B$.

$$AI(\theta) = L_{22}{}'L_{22}$$

## Average-Information

```
B = cbind( vu = Z %*% u / vu, ve = e / ve )
MB = chol(rbind(cbind(as.matrix(C),       t(W) %*% iR %*% B),
                cbind( t(B) %*% iR %*% W, t(B) %*% iR %*% B)))
LB = MB[(ncol(MB)-1):ncol(MB),(ncol(MB)-1):ncol(MB)]
SecDer2 =crossprod(LB)
FirDer2 =c(q/vu-(t(u)%*%iA%*%u)/(vu^2)-tr(iA%*%C22)/(vu^2),
    ((n-rX)/ve-(1/ve)*(q-tr(iA%*%C22)/(vu))-crossprod(e)/(ve^2))
vc - solve(SecDer2,FirDer2)
##        vu        ve
##  5.614032 53.308581
SecDer2 # AI matrix
##
##  0.12285971 0.04612028
##  0.04612028 0.53927449
```

## Gibbs sampling

Simple Hierarchical Bayesian approach to estimate variance components, where the final estimates are the **average** of the samples *a posteriori*, $\pi(\theta|x)$, the distribution of parameters ($\theta = \{\sigma_u^2, \sigma_u^2, b, u\}$) given data ($x = \{y, X, Z, A\}$). The posterior is a function of the probability of the data, $p(x|\theta)$, and probability of priors, $\pi(\theta)$.

$$\pi(\theta|x) \propto p(x|\theta)\pi(\theta)$$
$$p(x|\theta) = N(Xb + Zu, R)N(u, G|\sigma_u^2)N(e, R|\sigma_e^2)$$
$$\pi(\theta) = \chi^{-2}(\sigma_u^2|S_u, \nu_u)\chi^{-2}(\sigma_e^2|S_e, \nu_e)$$

## Gibbs sampling

We sample variance components from a scaled inverse-chi square distribution:

$$\sigma_u^2 | u, S_u, \nu_u \sim \frac{u' A^{-1} u + S_u \nu_u}{\chi^2(q + \nu_u)}$$

$$\sigma_e^2 | e, S_e, \nu_e \sim \frac{e' e + S_e \nu_e}{\chi^2(n + \nu_e)}$$

Using the mixed model equation ($Cg = r$) notation, we sample coefficients from:

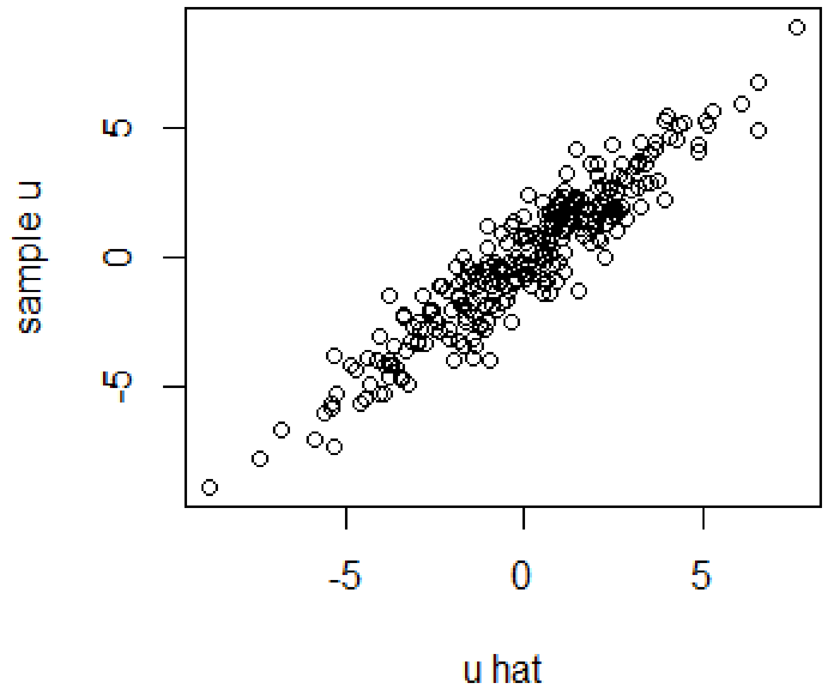
$$g_j | x, g_{-j} \sim N\left(\frac{r_j - C_{-j,j} g_{-j}}{C_{j,j}}, \frac{1}{C_{j,j}}\right)$$

## Gibbs sampling

```
# Priors
df0 = 5
Su = vu
Se = ve
# Sample genetic variance
c(t(u) %*% iA %*% u + Su*df0 ) / rchisq(1,q+df0)
## [1] 3.645489
# Sample residual variance
c(t(e) %*% e + Se*df0 ) / rchisq(1,n+df0)
## [1] 46.12222
```

```
g_samp = sapply(1:(rX+q), function(j) rnorm(n=1,
        mean=c(r[j]-C[j,-j]%*%g[-j])/C[j,j],
sd=sqrt(1/C[j,j])))
plot(g[-c(1:rX)],g_samp[-c(1:rX)],xlab='u
hat',ylab='sample u')
```

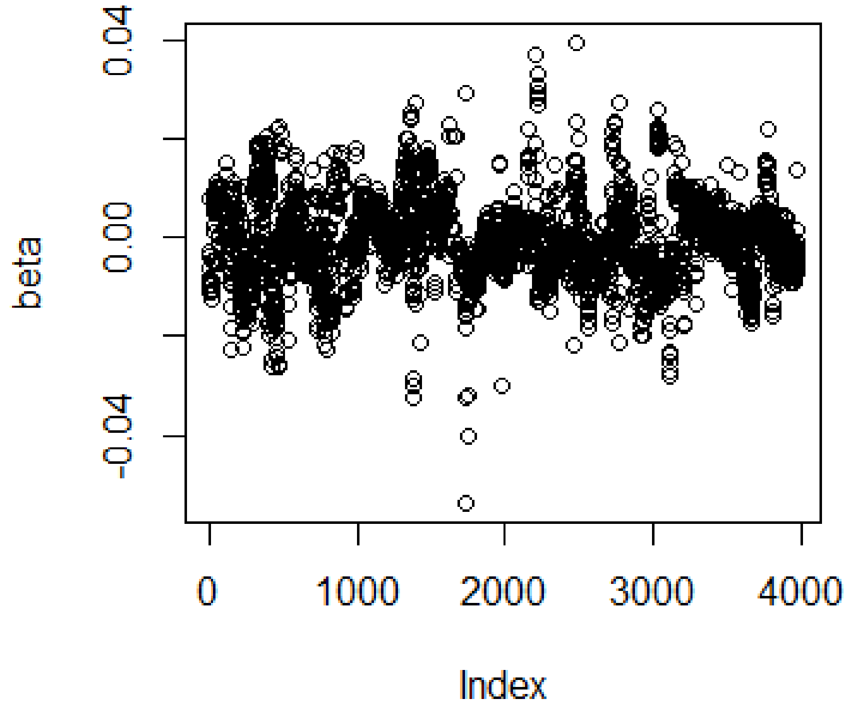
**Gibbs sampling**



## Marker effects

$$\hat{\beta}|\hat{u} = \hat{\sigma}_{\beta}^{2}\tilde{Z}'G^{-1}\hat{u}$$

```
vb = vu * alpha
beta = vb * c( t(Z1) %*% iG %*% u)
plot(beta)
```



# Final remarks

- Design matrices ($Z$,$X$) as sparse (*Matrix* pkg)
- Symmetric matrices ($C$,$C^{-1}$,$A^{-1}$) are store are upper diagonals
- Implement with good linear algebra libraries (*RcppEigen*, *RcppArmadillo*)
- Same matrices are never computed ($P$,$V$,$R$,$A$)
- When only traces are needed, only diagonals are computed (e.g., $tr(Z'SZ)$)
- Absorption can be efficiently done one vector at a time (e.g., $SZ$)
- Factorization can help in specific situations (SVD, EVD, Cholesky)

## THANK YOU