Lecture 4 - Advanced topics

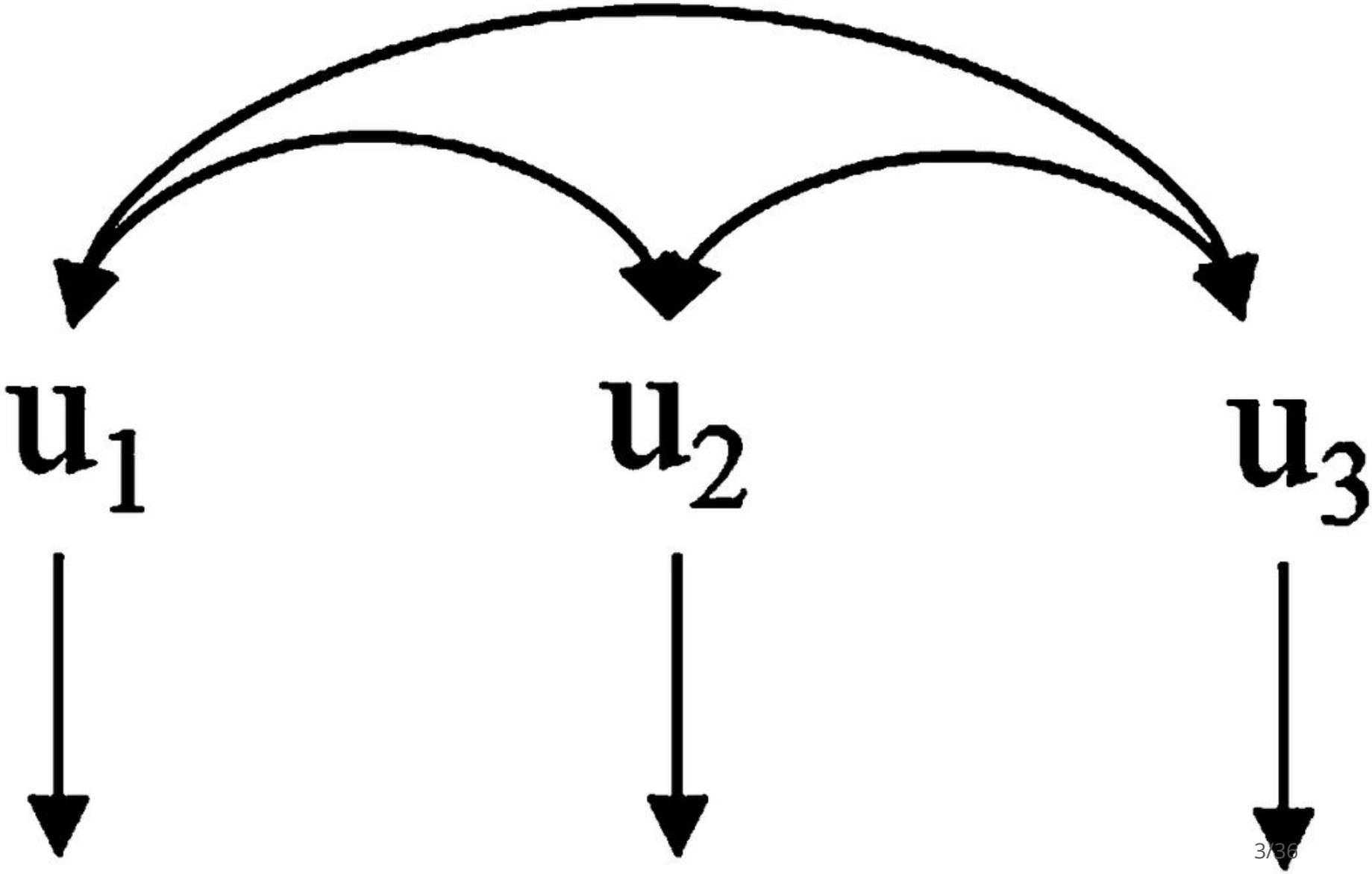
Alencar Xavier, Gota Morota

October 26, 2018

Outline

- Multivariate models
- Bayesian methods
- Machine learning
- G x E interactions

Multivariate models



Multivariate models

Mixed models also enable us to evaluate multiple traits:

- More accurate parameters: BV and variance components
- Information: Inform how traits relate to each other
- Constrains: May increase computation time considerably

It preserves the same formulation

$$y = Xb + Zu + e$$

However, we now stack the traits together:

$y = \{y_1, y_2, \ldots, y_k\}$, $X = \{X_1|X_2|\ldots|X_k\}'$, $b = \{b_1, b_2, \ldots, b_k\}$, $Z = \{Z_1|Z_2|\ldots|Z_k\}'$, $u = \{u_1, u_2, \ldots, u_k\}$, $e = \{e_1, e_2, \ldots, e_k\}$.

Multivariate models

The multivariate variance looks nice at first

$$Var(y) = Var(u) + Var(e)$$

But can get ugly with a closer look:

$$Var(u) = Z(G \otimes \Sigma_a)Z' = \begin{bmatrix} Z_1'GZ_1\sigma^2_{a_1} & Z_1'GZ_2\sigma_{a_1a_2} \\ Z_2'GZ_1\sigma_{a_2a_1} & Z_2'GZ_2\sigma^2_{a_2} \end{bmatrix}$$

and

$$Var(e) = R \otimes \Sigma_e = \begin{bmatrix} R\sigma^2_{e_1} & R\sigma_{e_1e_2} \\ R\sigma_{e_2e_1} & R\sigma^2_{e_2} \end{bmatrix}$$

Multivariate models

You can still think the multivariate mixed model as

$$y = Wg + e$$

Where

$$y = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, W = \begin{bmatrix} X_1 & 0 & Z_1 & 0 \\ 0 & X_2 & 0 & Z_2 \end{bmatrix}, g = \begin{bmatrix} b_1 \\ b_2 \\ u_1 \\ u_2 \end{bmatrix}, e = \begin{bmatrix} e_1 \\ e_2 \end{bmatrix}$$

Multivariate models

Left-hand side ($W'R^{-1}W + \Sigma$)

$$\begin{bmatrix} X_1'X_1\Sigma_{e_{11}}^{-1} & X_1'X_2\Sigma_{e_{12}}^{-1} & X_1'Z_1\Sigma_{e_{11}}^{-1} & X_1'Z_2\Sigma_{e_{12}}^{-1} \\ X_2'X_1\Sigma_{e_{12}}^{-1} & X_2'X_2\Sigma_{e_{22}}^{-1} & X_2'Z_1\Sigma_{e_{12}}^{-1} & X_2'Z_2\Sigma_{e_{22}}^{-1} \\ Z_1'X_1\Sigma_{e_{11}}^{-1} & Z_1'X_2\Sigma_{e_{12}}^{-1} & G^{-1}\Sigma_{a_{11}}^{-1} + Z_1'Z_1\Sigma_{e_{11}}^{-1} & G^{-1}\Sigma_{a_{12}}^{-1} + Z_1'Z_2\Sigma_{e_{12}}^{-1} \\ Z_2'X_1\Sigma_{e_{12}}^{-1} & Z_2'X_2\Sigma_{e_{22}}^{-1} & G^{-1}\Sigma_{a_{11}}^{-1} + Z_2'Z_1\Sigma_{e_{12}}^{-1} & G^{-1}\Sigma_{a_{22}}^{-1} + Z_2'Z_2\Sigma_{e_{22}}^{-1} \end{bmatrix}$$

Right-hand side ($W'R^{-1}y$)

$$\begin{bmatrix} X_1'y\Sigma_{e_1}^{-1} \\ X_2'y\Sigma_{e_1e_2}^{-1} \\ Z_1'y\Sigma_{e_1}^{-1} \\ Z_2'y\Sigma_{e_1e_2}^{-1} \end{bmatrix}$$

Multivariate models

```
data(wheat, package = 'BGLR')
G = NAM::GRM(wheat.X)
Y = wheat.Y; colnames(Y) = c('E1','E2','E3','E4')
mmm = NAM::reml( y = Y, K = G )
knitr::kable( round(mmm$VC$GenCor,2) )
```

	E1	E2	E3	E4
E1	1.00	-0.25	-0.22	-0.50
E2	-0.25	1.00	0.96	0.55
E3	-0.22	0.96	1.00	0.72
E4	-0.50	0.55	0.72	1.00

Multivariate models

```
mmm$VC$Vg
```

```
##            E1         E2         E3         E4
## E1  0.6277835 -0.1446924 -0.1102175 -0.2743640
## E2 -0.1446924  0.5440731  0.4419945  0.2822577
## E3 -0.1102175  0.4419945  0.3919626  0.3130735
## E4 -0.2743640  0.2822577  0.3130735  0.4828705
```

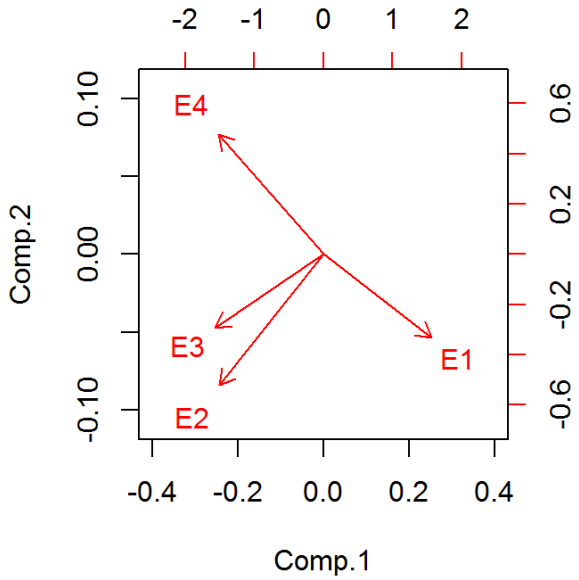
```
mmm$VC$Ve
```

```
##             E1         E2         E3         E4
## E1  0.53504246 0.08247812 -0.1159118 0.06882868
## E2  0.08247812 0.56214755  0.2973841 0.15795801
## E3 -0.11591175 0.29738408  0.6714234 0.11086214
## E4  0.06882868 0.15795801  0.1108621 0.59405228
```

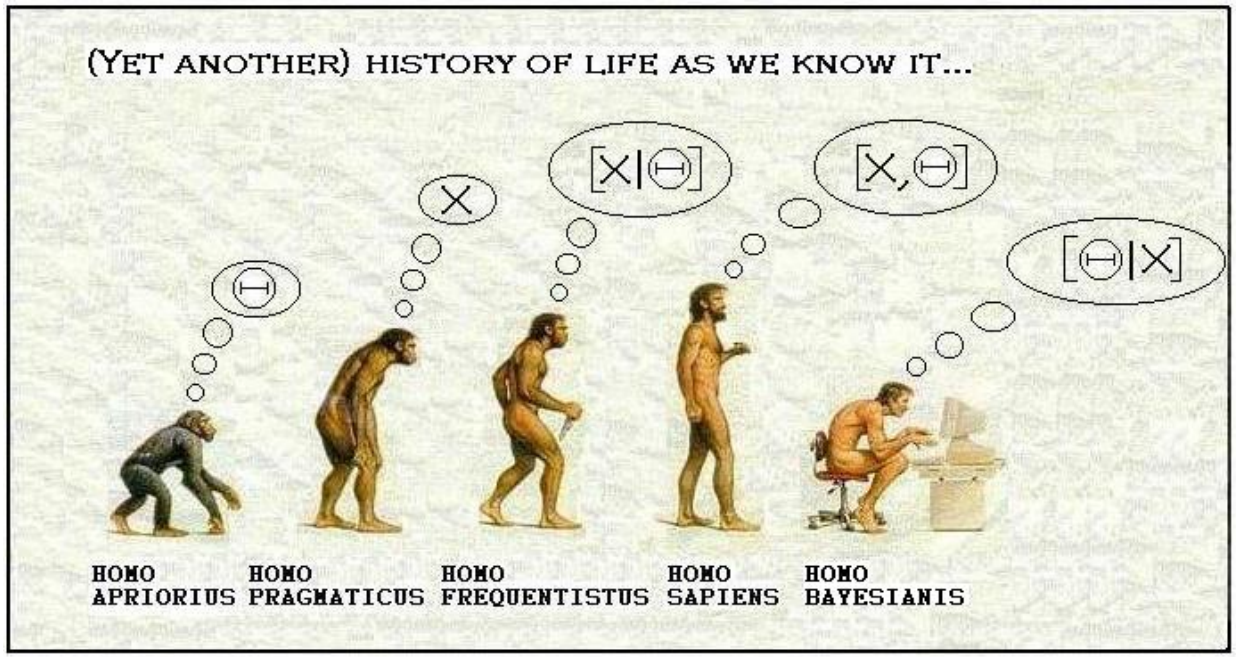
Multivariate models

- Selection indeces, co-heritability, indirect response to selection
- Study residual and additive genetic association among traits

```
biplot(princomp(mmm$VC$GenCor,cor=T),xlim=c(-.4,.4),ylim=c(-.11,.11))
```



Bayesian methods



Bayesian methods

The general framework on a hierarchical Bayesian model follows:

$$p(\theta|x) \propto p(x|\theta)p(\theta)$$

Where:

- Posterior probability: $p(\theta|x)$
- Likelihood: $p(x|\theta)$
- Prior probability: $p(\theta)$

Bayesian methods

For the model:

$$y = Xb + Zu + e, \quad u \sim N(0, K\sigma_a^2),\ e \sim N(0, I\sigma_e^2)$$

- Data ($x = \{y, X, Z, K\}$)
- Parameters ($\theta = \{b, u, \sigma_a^2, \sigma_e^2\}$)

Probabilistic model:

$$p(b, u, \sigma_a^2, \sigma_e^2|y, X, Z, K) \propto N(y, X, Z, K|b, u, \sigma_a^2, \sigma_e^2) \times N(b, u|\sigma_a^2, \sigma_e^2) \times \chi^{-2}(\sigma_a^2, \sigma_e^2|S_a, S_e, \nu_a, \nu_e)$$

Bayesian methods

REML: the priors (S_a, S_e, ν_a, ν_e) are estimated from data.

Hierarchical Bayes: You provide priors. Here is how:

$$\sigma_a^2 = \frac{u' K^{-1} u + S_a \nu_a}{\chi^2 (q + \nu_a)}$$

```
sigma2a=(t(u)%*%iK%*%u+Sa*dfa)/rchisq(df=ncol(Z)+dfa,n=1)
```

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

```
sigma2e=(t(e)%*%e+Se*dfe)/rchisq(df=length(y)+dfe,n=1)
```

Bayesian methods

What does it mean for **you**? If your "prior knowledge" tells you that a given trait has approximately $h^2 = 0.5$ (nothing unreasonable). In which case, half of the phenotypic variance is due to genetics, and the other half is due to error. Your prior shape is:

$$S_a = S_e = \sigma_y^2 \times 0.5$$

We usually assign small a prior degrees of freeds. Samething like four or five prior degrees of freedom. That means that assuming $\nu_0 = 5$, you are yielding to your model 5 data points that support heritability 0.5

$$\nu_a = \nu_e = 5$$

Example of prior influence: In a dataset with 300 data points, 1.6% of the variance components information comes from prior (5/305), and 98.4% comes from data (300/305).

Bayesian methods

For whole-genome regression models

$$y = \mu + Ma + e, \quad a \sim N(0, I\sigma_b^2),\ e \sim N(0, I\sigma_e^2)$$

We scale the prior genetic variance based on allele frequencies

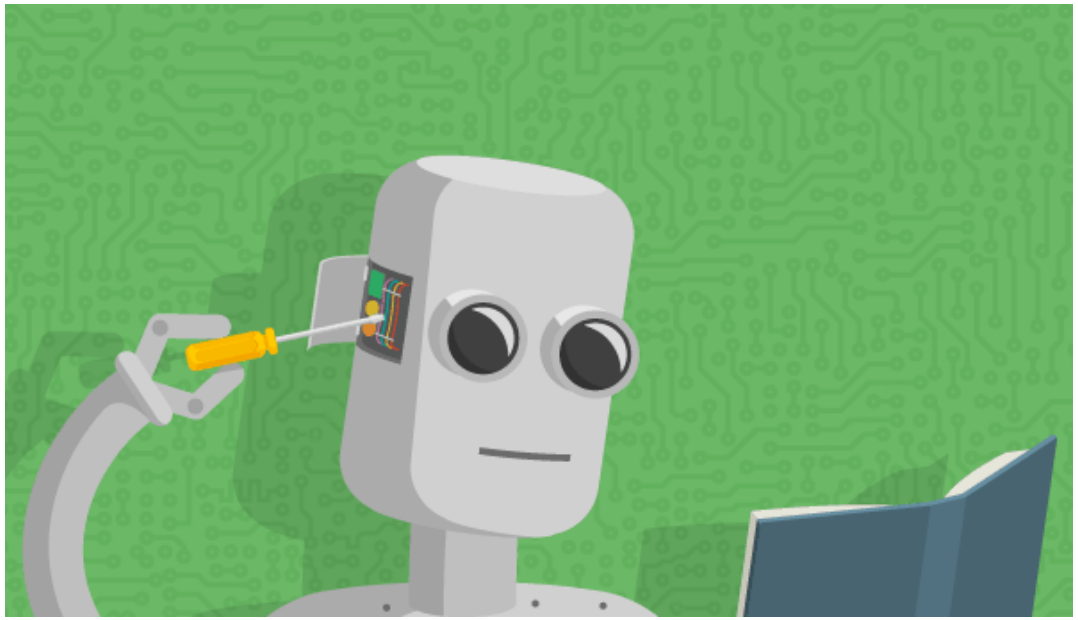
$$S_b = \frac{\sigma_y^2 \times 0.5}{2\sum p_j(1-p_j)}$$

Two common settings:

- All markers, one random effect:
- Each markers as a random effect:

Machine learning methods

- Parametric methods for prediction: L1-L2
- Semi-parametric methods for prediction: Kernels
- Non-parametric methods for prediction: Trees and nets



Machine learning methods

L1-L2 machines include all mixed and Bayesian models we have seen so far. The basic framework is driven by a single (random) term model:

$$y = Xb + e$$

The univariate soltion indicates how the model is solved. A model without regularization yields the least square (LS) solution. If we regularize by deflating the nominator, we get the L1 regularization (LASSO). If we regularize by inflating the denominator, we get the L2 regularization (Ridge). For any combination of both, we get a elastic-net (EN). Thus:

$$b_{LS} = \frac{x'y}{x'x}, \;\; b_{Lasso} = \frac{x'y - \lambda}{x'x}, \;\; b_{Ridge} = \frac{x'y}{x'x + \lambda}, \;\; b_{EN} = \frac{x'y - \lambda_1}{x'x + \lambda_2}$$

Whereas the Bayesian and mixed model framework resolves the regularization as $\lambda = \sigma_e^2/\sigma_b^2$, ML methods search for λ through (-fold) cross-validation.

Machine learning methods

Common loss functions in L1-L2 machines

- LS (no prior, no shrinkage): $argmin(\sum e_i^2)$
- L1 (Laplace prior with variable selection): $argmin(\sum e_i^2 + \lambda \sum |b_j|)$
- L2 (Gaussian prior, unique solution): $argmin(\sum e_i^2 + \lambda \sum b_j^2)$

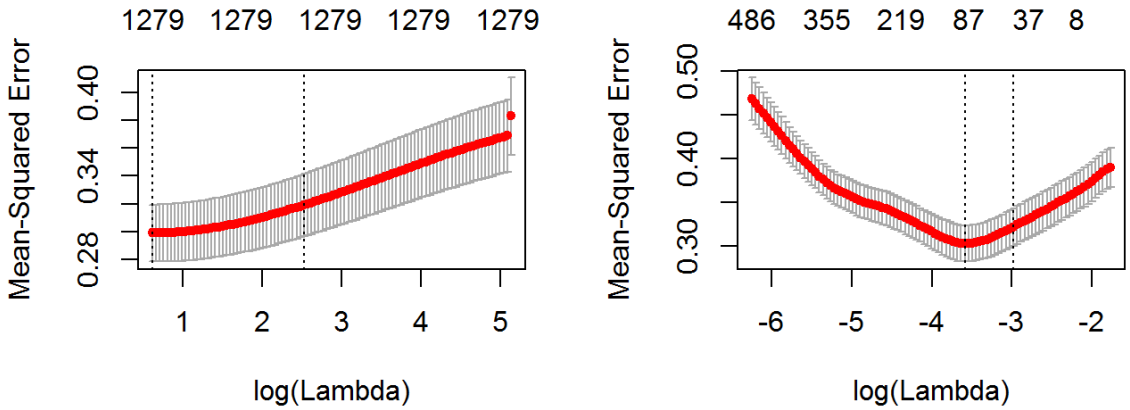
Other losses that are less popular

- Least absolute: $argmin(\sum |e_i|)$ based on $b_{LA} = \frac{MD(x \times y)}{x'x}$
- ϵ-loss: $argmin(\sum e_i^2, |e_i| > \epsilon)$ - used in support vector machines

Machine learning methods

Cross-validations to search for best value of lambda

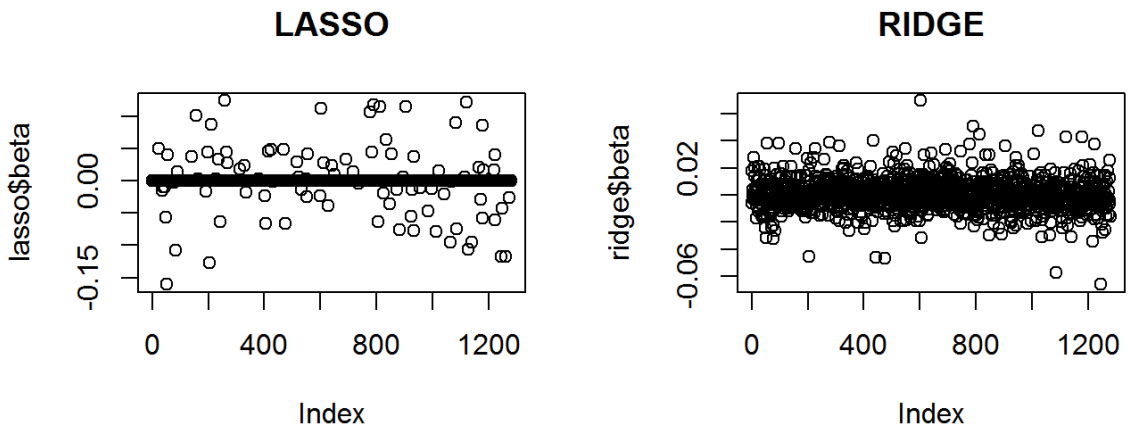
```
lasso = glmnet::cv.glmnet(x=wheat.X,y=rowMeans(Y),alpha=1);
ridge = glmnet::cv.glmnet(x=wheat.X,y=rowMeans(Y),alpha=0);
par(mfrow=c(1,2)); plot(ridge); plot(lasso)
```



Machine learning methods

Re-fit the model using this best value

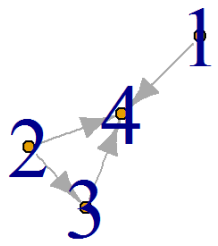
```
lasso = glmnet::glmnet(x=wheat.X,y=rowMeans(Y),lambda=lasso$lambda.min,alpha=1)
ridge = glmnet::glmnet(x=wheat.X,y=rowMeans(Y),lambda=ridge$lambda.min,alpha=0)
par(mfrow=c(1,2)); plot(lasso$beta,main='LASSO'); plot(ridge$beta,main='RIDGE');
```



Machine learning methods

Of course, the losses presented above are not limited to the application of prediction and classification. Below, we see an example of deploying LASSO for a graphical model (Markov Random Field): How the traits of the multivariate model relate in terms of additive genetics:

```
ADJ=huge::huge(mmm$VC$GenCor,.3,method='glasso',verbose=F)$path[[1]]
plot(igraph::graph.adjacency(adjmatrix=ADJ),vertex.label.cex=3)
```



Machine learning methods

Reproducing kernel Hilbert Spaces (RKHS), is a generalization of a GBLUP… Most commonly instead of using the linear kernel ($ZZ'\alpha$), RKHS commonly uses one or more Gaussian or exponential kernels:

$$K = \exp(-\theta D^2)$$

Where D^2 is the squared Euclidean distance, and θ is a bandwidth:

- Single kernel: $1/mean(D^2)$
- Three kernels: θ={5/q, 1/q, 0.2/q}, where `q=quantile(D2,0.05)`

Machine learning methods

We can use REML, PRESS (=cross-validation) or Bayesian approach to solve RKHS

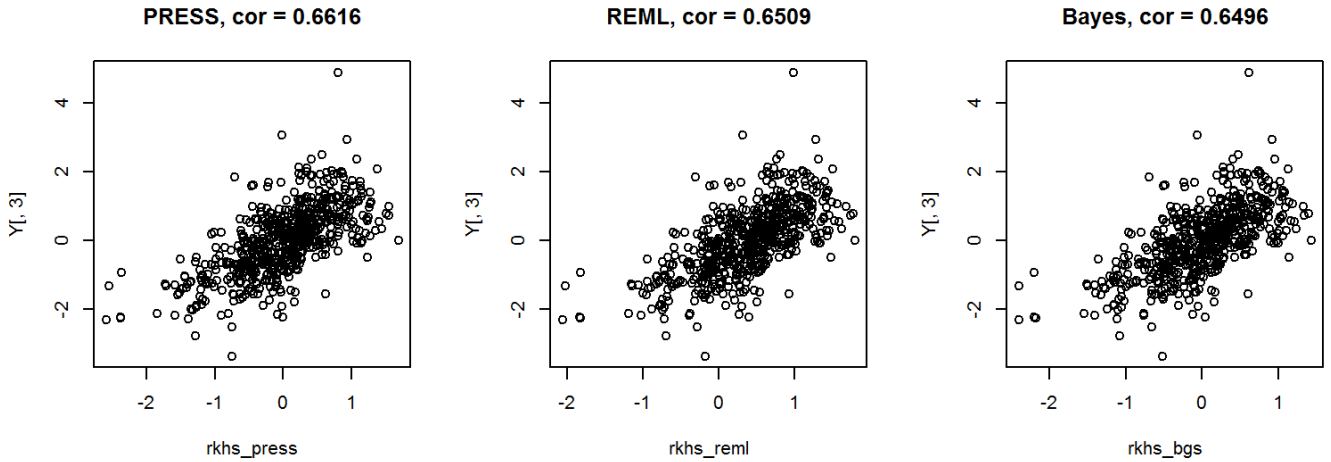
```
# Make the kernel
D2 = as.matrix(dist(wheat.X)^2)
K = exp(-D2/mean(D2))

# Below we are going to calibrate models on Env 2 and predict Env 3
rkhs_press = NAM::press(y=Y[,2],K=K)$hat
rkhs_reml = NAM::reml(y=Y[,2],K=K)$EBV
rkhs_bgs = NAM::gibbs(y=Y[,2],iK=solve(K))$Fit.mean
```

```
##
  |
  |                                                      |   0%
  |
  |                                                      |   1%
  |
  |=                                                     |   1%
  |
  |=                                                     |   2%
  |
  |==                                                    |   2%
  |
```

Machine learning methods

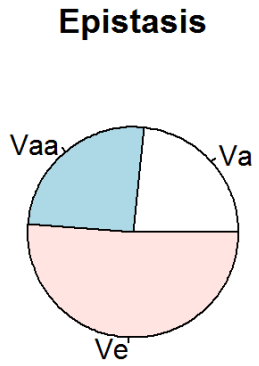
```
par(mfrow=c(1,3))
plot(rkhs_press,Y[,3],main=paste('PRESS, cor =',round(cor(rkhs_press,Y[,3]),4) ))
plot(rkhs_reml,Y[,3],main=paste('REML, cor =',round(cor(rkhs_reml,Y[,3]),4) ))
plot(rkhs_bgs,Y[,3],main=paste('Bayes, cor =',round(cor(rkhs_bgs,Y[,3]),4) ))
```



Machine learning methods

RKHS for epistasis and variance component analysis

```
Ks = NAM::G2A_Kernels(wheat.X) # Get all sorts of linear kernels
FIT = BGLR::BGLR(rowMeans(Y),verbose=FALSE,
      ETA=list(A=list(K=Ks$A,model='RKHS'),AA=list(K=Ks$A,model='RKHS')))
pie(c(Va=FIT$ETA$A$varU,Vaa=FIT$ETA$AA$varU,Ve=FIT$varE),main='Epistasis')
```



Machine learning methods

For the same task (E2 predict E3), let's check members of the Bayesian alphabet

```
fit_BRR = bWGR::wgr(Y[,2],wheat.X); cor(c(fit_BRR$hat),Y[,3])
```

```
## [1] 0.5768394
```

```
fit_BayesB = bWGR::BayesB(Y[,2],wheat.X); cor(fit_BayesB$hat,Y[,3])
```

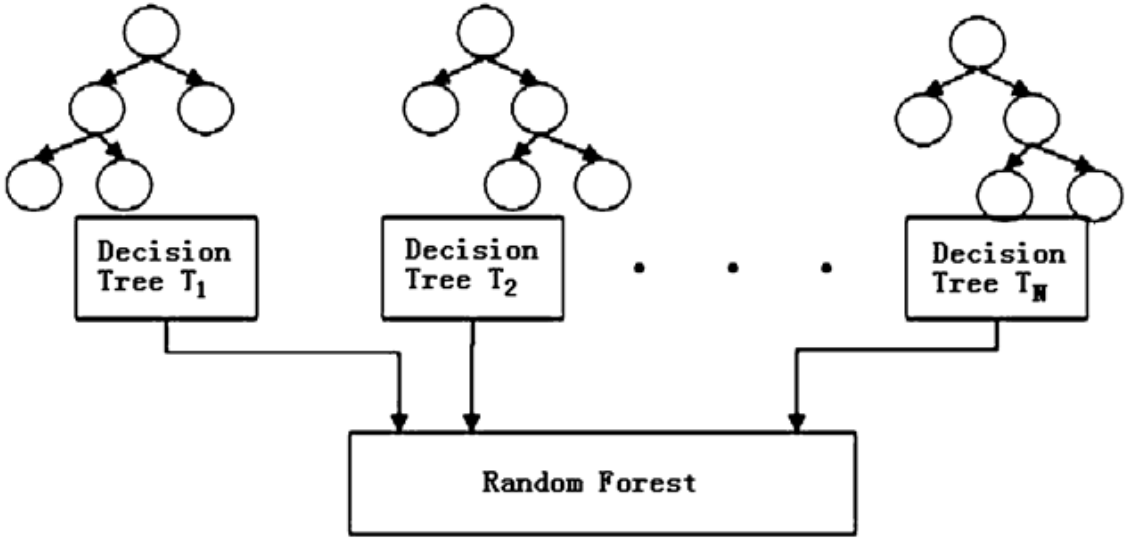
```
## [1] 0.5379385
```

```
fit_emBayesA = bWGR::emBA(Y[,2],wheat.X); cor(fit_emBayesA$hat,Y[,3])
```

```
## [1] 0.6388318
```

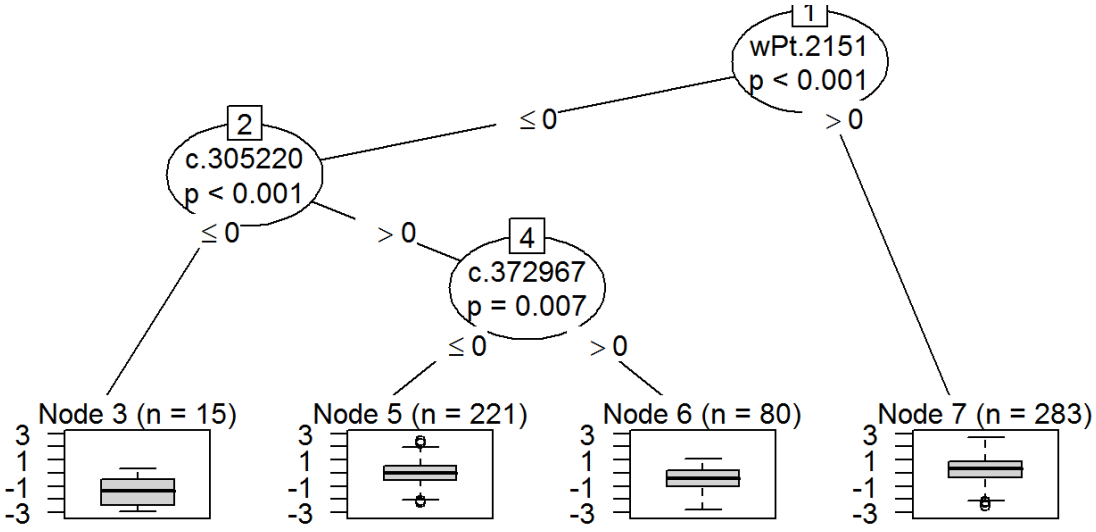
Machine learning methods

Tree regression and classifiers



Machine learning methods

```
fit_tree = party::ctree(y~.,data.frame(y=Y[,2],wheat.X)); plot(fit_tree)
```

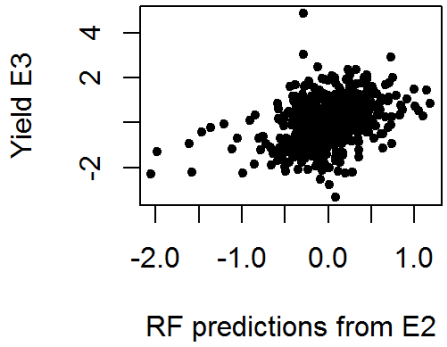



```
cor(c(fit_tree@predict_response()),Y[,3])
```

```
## [1] 0.265622
```

Machine learning methods

```
fit_rf = ranger::ranger(y~.,data.frame(y=Y[,2],wheat.X))
plot(fit_rf$predictions,Y[,3],xlab='RF predictions from E2',ylab='Yield E3',pch=20)
```




```
cor(fit_rf$predictions,Y[,3])
```

```
## [1] 0.4028364
```

Genotype-Environmnet interactions

Genotype-Environmnet interactions

```
y=as.vector(wheat.Y); Z=wheat.X; Zge=as.matrix(Matrix::bdiag(Z,Z,Z,Z))
#
fit_g = bWGR::BayesRR(rowMeans(wheat.Y),Z)
fit_ge = bWGR::BayesRR(y,Zge)
fit_gge = bWGR::BayesRR2(y,rbind(Z,Z,Z,Z),Zge)
#
fit_g$h2
```

```
## [1] 0.4563049
```

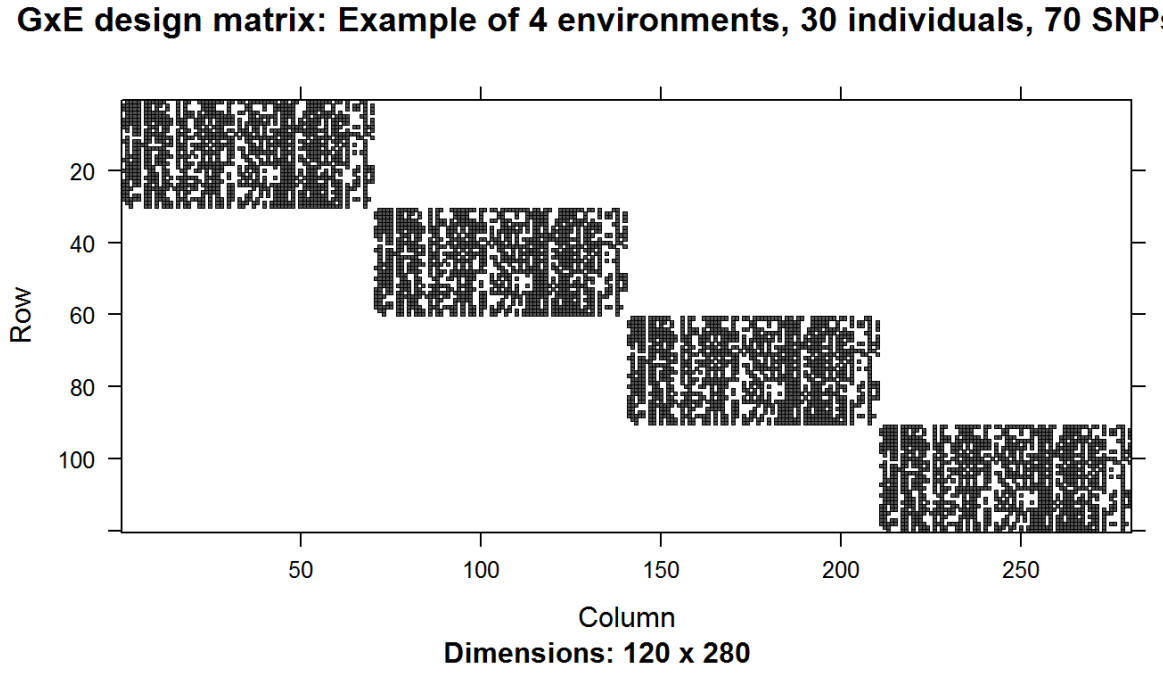
```
fit_ge$h2
```

```
## [1] 0.6831601
```

```
fit_gge$h2
```

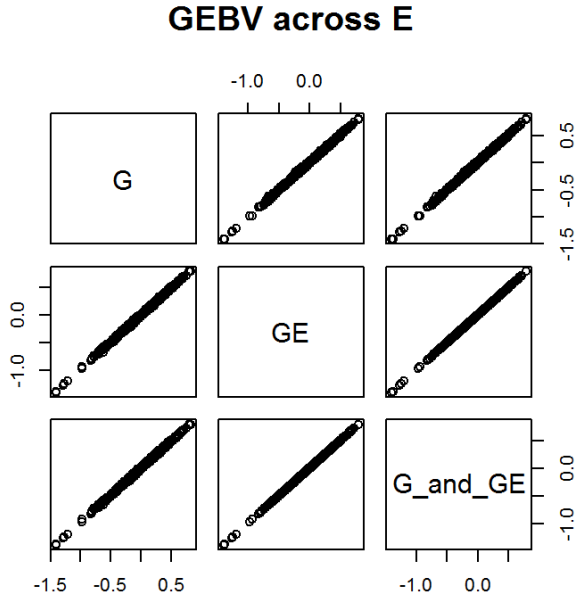
```
## [1] 0.6796175
```

Genotype-Environmnet interactions



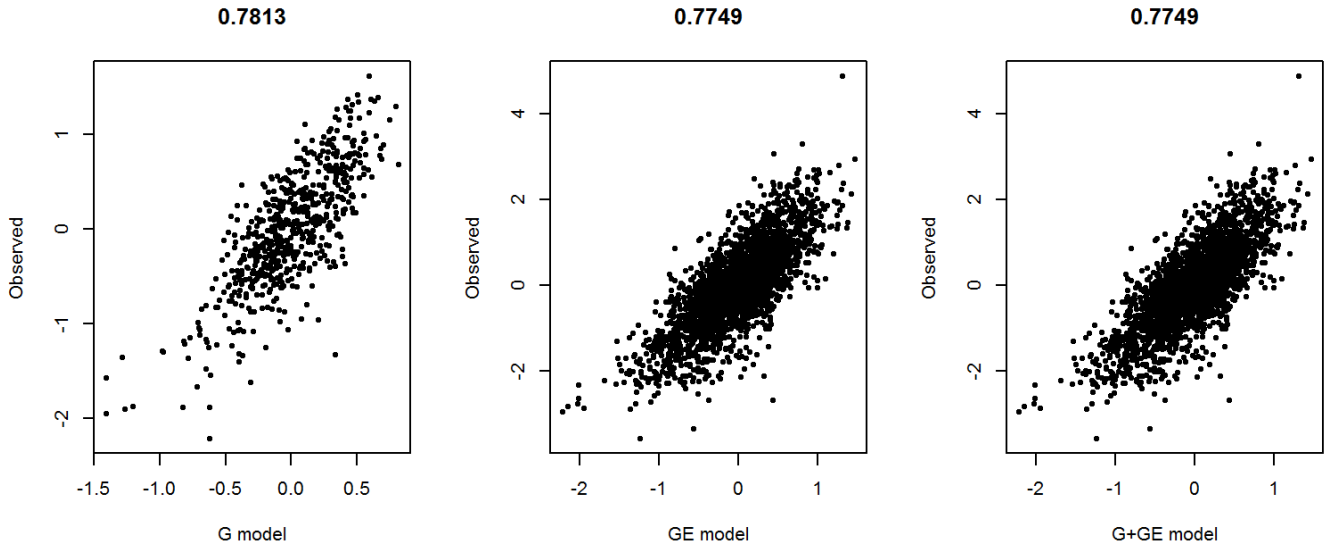
Genotype-Environmnet interactions

```
GE1=matrix(fit_ge$hat,ncol=4); GE2=matrix(fit_ge$hat,ncol=4)
plot(data.frame(G=fit_g$hat,GE=rowMeans(GE1),G_and_GE=rowMeans(GE2)),main='GEBV across E')
```



Genotype-Environmnet interactions

```
par(mfrow=c(1,3))
plot(fit_g$hat,rowMeans(Y),main=round(cor(fit_g$hat,rowMeans(Y)),4),xlab='G model',ylab='Observed',pch=2
plot(c(GE1),y,main=round(cor(c(GE1),y),4),xlab='GE model',ylab='Observed',pch=20)
plot(c(GE2),y,main=round(cor(c(GE2),y),4),xlab='G+GE model',ylab='Observed',pch=20)
```



Thanks!