Análise de Dados no Melhoramento Genético de Pinus

Table of Contents

Neste documento estão os scripts para as análises abordadas no capítulo “Análise de Dados no Melhoramento Genético de Pinus”.

# Pacotes necessários

require(tidyverse)  
require(asreml)  
require(gghighlight)  
require(ComplexHeatmap)  
require(RColorBrewer)  
require(ggrepel)

# 1. Análise individual (somente um ambiente)

## Carregando o conjunto de dados

Para realizar as análises individuais, utilizaremos somente o primeiro ambiente do conjunto de dados disponibilizados.

data = read.table("EstCov.txt", header = T)  
  
data\_e1 = data %>% filter(amb == "E1") %>% select(gen, rept,y)  
data\_e1$gen = as.factor(data\_e1$gen)  
data\_e1$rept = as.factor(data\_e1$rept)

O experimento consiste em 15 tratamentos delineados em 10 blocos completos casualizados. Perceba a presença de dados perdidos (NA) no conjunto de dados. Isso não será problema para a execução da análise.

## Modelo para análise dos dados

m = asreml(fixed = y ~ rept,  
 random = ~ gen,  
 data = data\_e1)  
summary(m)$varcomp

## Teste de significância: Likelihood ratio test (LRT)

Uma vez construído o modelo, é necessário determinar a significância do efeito de tratamentos. Isto é realizado utilizando o teste de deviance ou Likelihood Ratio Test (LRT). Para isto, devemos construir o modelo reduzido, isto é, sem efeito a ser testado. Com o modelo reduzido em mãos, o próprio ASReml encarrega-se de calcular o LRT utilizando a função “lrt”.

mr = asreml(fixed = y ~ rept,  
 data = data\_e1)  
  
lrt(m,mr)

## Likelihood ratio test(s) assuming nested random models.  
## (See Self & Liang, 1987)  
##   
## df LR-statistic Pr(Chisq)   
## m/mr 1 23.979 4.87e-07 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

Veja que o p-valor [Pr(Chisq)] é menor que 0.05, portanto o efeito em teste é significativo.

## Acurácia e herdabilidade

Uma vez aferida a significância, podemos calcular alguns parâmetros genéticos, como a acurácia e a herdabilidade. Para isto, utilizaremos as seguintes fórmulas:

* Acurácia [(Mrode, 2014)](https://cabidigitallibrary.org/doi/book/10.1079/9781780643915.0000)

em que PEV é a variância do erro de predição, obtida da diagonal da matriz de coeficientes da equação de modelos mistos

* Herdabilidade individual
* Herdabilidade média (fórmula padrão)

em que r é o número de repetições

* Herdabilidade média [(Cullis et al., 2006)](https://link.springer.com/article/10.1198/108571106X154443)

em que é a variância média da diferença entre dois BLUPs

Para estimar e , utilizaremos a função “predict” do ASReml. Esta função também servirá para obtenção dos BLUPs

predm\_sed = predict(m,"gen", sed = T)  
predm\_vcov = predict(m, "gen", vcov = T)  
  
PEV = mean(diag(predm\_vcov$vcov))  
Acurácia = sqrt(1-(PEV/summary(m)$varcomp[grep("gen",rownames(summary(m)$varcomp)),1]))  
  
Herdabilidade\_ind = summary(m)$varcomp[grep("gen",rownames(summary(m)$varcomp)),1] /   
 (sum(summary(m)$varcomp[,1]))  
  
Herdabilidade\_med\_pad = summary(m)$varcomp[grep("gen",rownames(summary(m)$varcomp)),1] /  
 (summary(m)$varcomp[grep("gen",rownames(summary(m)$varcomp)),1] +  
 (summary(m)$varcomp[grep("R",rownames(summary(m)$varcomp)),1]/10))  
  
MVdelta = mean((predm\_sed$sed^2)[upper.tri(predm\_sed$sed^2, diag = F)])  
Herdabilidade\_med\_cullis = 1-(MVdelta)/  
 (2\*summary(m)$varcomp[grep("gen",rownames(summary(m)$varcomp)),1])

Parâmetro

Valor

r

0.8937149

0.3340860

0.8338030

0.8020476

Veja que e são diferentes, ainda que ambos indiquem a herdabilidade média no banco de dados. Isto ocorre devido ao desbalanceamento, que, no exemplo acima, é devido à perda de dados. é uma fórmula proposta para quando os dados são balanceados e, por vezes, pode distorcer o valor real da herdabilidade média. Além de , existem outras alternativas que podem ser utilizadas quando os dados são desbalanceados. O leitor pode encontrá-las em [Schmidt et al. (2019a)](https://doi.org/10.2135/cropsci2018.06.0376) e [Schmidt et al. (2019b)](https://doi.org/10.1534/genetics.119.302134).

## BLUPs

Ainda utilizando a função “predict”, obteremos os BLUPs, ou valores genotípicos, dos tratamentos em análise. Com eles, realizaremos a seleção e calcularemos os ganhos genéticos estimados:

em que é a média dos BLUPs dos genótipos selecionados e é a média do experimento. No exemplo, selecionamos os cinco melhores tratamentos.

predm\_sed = predict(m,"gen", sed = T)  
predm\_vcov = predict(m, "gen", vcov = T)  
  
BLUP = predm\_sed$pvals  
  
BLUP = BLUP %>% arrange(desc(predicted.value))  
  
BLUPs = BLUP[1:5,]  
  
Ganho = (mean(BLUPs$predicted.value) - mean(BLUP$predicted.value)) /  
 mean(BLUP$predicted.value) \* 100

# 2. Análise conjunta (múltiplos ambientes)

## Carregando o conjunto de dados

Desta vez utilizaremos todo o conjunto de dados disponibilizados.

data$amb = as.factor(data$amb)  
data$gen = as.factor(data$gen)  
data$rept = as.factor(data$rept)

Agora temos 15 tratamentos delineados em 10 blocos completos casualizados em 4 ambientes.

## Modelagem das estruturas de covariância

### Modelo 1: Simetria composta para genótipo e identidade para resíduo

m1 = asreml(fixed = y ~ rept:amb,  
 random = ~gen + gen:amb,  
 data = data)  
summary(m1)$varcomp

### Modelo 2: Simetria composta para genótipo e bloco diagonal para resíduo

m2 = asreml(fixed = y ~ rept:amb,  
 random = ~gen + gen:amb,  
 residual = ~dsum(~id(units)|amb),  
 data = data)  
summary(m2)$varcomp

### Modelo 3: Simetria composta heterogênea para genótipo e bloco diagonal para resíduo

m3 = asreml(fixed = y ~ rept:amb,  
 random = ~corh(amb):gen,  
 residual = ~dsum(~id(units)|amb),  
 data = data)  
summary(m3)$varcomp

### Modelo 4: Estrutura multivariada para genótipo e bloco diagonal para resíduo

m4 = asreml(fixed = y ~ rept:amb,  
 random = ~us(amb):gen,  
 residual = ~dsum(~id(units)|amb),  
 data = data)  
summary(m4)$varcomp

### Seleção do modelo

data.frame("Modelo" = seq(1,4),  
 "AIC" = c(summary(m1)$aic,summary(m2)$aic,summary(m3)$aic,  
 summary(m4)$aic))

O modelo 4 possui o menor AIC e, portanto, o melhor ajuste.

## Modelo Fator Analítico Multiplicativo Misto

### Carregando o conjunto de dados

Neste exemplo, utilizamos o conjunto de dados presente neste [link](https://github.com/Kaio-Olimpio/Probability-for-GEI/blob/master/maize_dataset.csv)

dataset = read.csv("https://raw.githubusercontent.com/Kaio-Olimpio/Probability-for-GEI/master/maize\_dataset.csv",   
 sep = ",")  
  
dataset <- transform(dataset, env = factor(Location), gen = factor(Hybrid), rept = factor(Rep),   
 block = factor(Block))  
  
num.env = nlevels(factor(dataset$env))  
num.gen = nlevels(factor(dataset$gen))  
name.env = levels(factor(dataset$env))  
name.gen = levels(factor(dataset$gen))

### Seleção do modelo

Procuramos um modelo parsimonioso, mas que garanta a explicação de uma grande parte da variação do conjunto de dados. Em termos práticos, adotamos um limite mínimo de 70% em termos de variação explicada pelo modelo. Neste exemplo, já partiremos da estrutura diagonal para os efeitos residuais.

## Model AIC ExpVar.Star  
## 1 FA1 1662.003 32.27939  
## 2 FA2 1675.613 46.50857  
## 3 FA3 1678.650 56.15207  
## 4 FA4 1687.391 67.87455  
## 5 FA5 1699.159 86.19968

Dentre os modelos testados, o FA5 possui a maior variância explicada, apesar do maior valor de AIC. Prosseguiremos com a utilização deste modelo para as demais etapas.

### Obtenção da matriz de covariâncias genéticas

#### Matriz de cargas fatoriais

sum5 = summary(m5)$varcomp  
fa1.loadings = sum5[grep('fa1', rownames(sum5)),1]  
fa2.loadings = sum5[grep('fa2', rownames(sum5)),1]  
fa3.loadings = sum5[grep('fa3', rownames(sum5)),1]  
fa4.loadings = sum5[grep('fa4', rownames(sum5)),1]  
fa5.loadings = sum5[grep('fa5', rownames(sum5)),1]  
mat.loadings5 = as.matrix(cbind(fa1.loadings, fa2.loadings,fa3.loadings,  
 fa4.loadings,fa5.loadings))

#### Rotação da matriz de cargas fatoriais

svdmat = svd(mat.loadings5)  
mat.loadings5.star = -1\*mat.loadings5 %\*% svdmat$v

#### Matriz de variâncias específicas

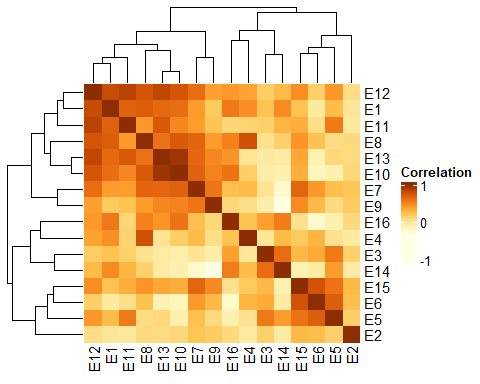
spvar = as.matrix(diag(sum5[grep("var",row.names(sum5)),1]))

#### Matriz de covariâncias genéticas

gencov = mat.loadings5.star %\*% t(mat.loadings5.star) + spvar

### Correlações genéticas entre ambientes

rownames(gencov) = colnames(gencov) = name.env  
  
corr = cov2cor(gencov)  
  
Heatmap(corr,col=colorRampPalette(brewer.pal(8, "YlOrBr"))(25),  
 column\_dend\_height = unit(2, "cm"),   
 clustering\_method\_rows = "complete",  
 row\_dend\_width = unit(2, "cm"),   
 clustering\_method\_columns = "complete",  
 heatmap\_legend\_param = list(title="Correlation",  
 at=c(-1,0,1),labels=c("-1","0","1")),  
 row\_names\_gp = gpar(fontsize = 11),  
 column\_names\_gp = gpar(fontsize = 11))



### Obtenção dos EBLUPs

#### Vetor de escores fatoriais

coef5 = summary(m5,coef = T)$coef.random  
  
fa1.scores = coef5[grep("Comp1",row.names(coef5)),1];  
names(fa1.scores) = sub("fa(env, 4)\_Comp1:gen\_","",names(fa1.scores),fixed=T)  
fa2.scores = coef5[grep("Comp2",row.names(coef5)),1];  
names(fa2.scores) = sub("fa(env, 4)\_Comp2:gen\_","",names(fa2.scores),fixed=T)  
fa3.scores = coef5[grep("Comp3",row.names(coef5)),1];  
names(fa3.scores) = sub("fa(env, 4)\_Comp3:gen\_","",names(fa3.scores),fixed=T)  
fa4.scores = coef5[grep("Comp4",row.names(coef5)),1];  
names(fa4.scores) = sub("fa(env, 4)\_Comp4:gen\_","",names(fa4.scores),fixed=T)  
fa5.scores = coef5[grep("Comp4",row.names(coef5)),1];  
names(fa5.scores) = sub("fa(env, 4)\_Comp4:gen\_","",names(fa5.scores),fixed=T)  
  
fa.scores = rbind(as.matrix(fa1.scores),as.matrix(fa2.scores),  
 as.matrix(fa3.scores),as.matrix(fa4.scores),  
 as.matrix(fa5.scores))

#### Rotação do vetor de escores fatoriais

fa.scores.star = -kronecker(t(svdmat$v), diag(num.gen))%\*%fa.scores   
rownames(fa.scores.star) = rep(name.gen,5)  
  
fa1.scores.star = fa.scores.star[1:num.gen,1]  
fa2.scores.star = fa.scores.star[(num.gen+1):(num.gen\*2),1]  
fa3.scores.star = fa.scores.star[(num.gen\*2+1):(num.gen\*3),1]  
fa4.scores.star = fa.scores.star[(num.gen\*3+1):(num.gen\*4),1]  
fa5.scores.star = fa.scores.star[(num.gen\*4+1):(num.gen\*5),1]

#### EBLUPs marginais

EBLUPs\_marg = (kronecker(mat.loadings5.star,diag(num.gen))) %\*% fa.scores.star   
EBLUPs\_marg = data.frame("Environment" = rep(name.env,each = num.gen),  
 "Gen" = rep(name.gen,num.env),  
 "EBLUP\_marg" = EBLUPs\_marg)

### Regressões latentes

EBLUPs\_marg$FL\_1 = rep(mat.loadings5.star[,1], each = num.gen)  
ggplot(subset(EBLUPs\_marg, Gen %in% c("G9","G32","G20","G23")),   
 aes(x=FL\_1,y=EBLUP\_marg,colour=Gen))+  
 geom\_smooth(aes(x=FL\_1,y=EBLUP\_marg),method=lm, fill = NA)+  
 facet\_wrap(~Gen, ncol = 4, as.table = T, dir = 'v', scales = 'free\_x')+  
 geom\_vline(xintercept = mean(mat.loadings5.star[,1]),linetype="dashed") +   
 geom\_point()+  
 xlab("Cargas do primeiro fator")+ylab("EBLUP")+  
 labs(color = "Genótipo", title = "EBLUP x Cargas do primeiro fator")

EBLUPs\_marg$EBLUPs\_marg\_f2 = EBLUPs\_marg$EBLUP\_marg -   
 (kronecker(mat.loadings5.star[,1],diag(num.gen))) %\*%  
 as.matrix(fa1.scores.star)  
EBLUPs\_marg$FL\_2 = rep(mat.loadings5.star[,2], each = num.gen)  
  
ggplot(subset(EBLUPs\_marg, Gen %in% c("G9","G32","G20","G23")),   
 aes(x=FL\_2,y=EBLUPs\_marg\_f2,colour=Gen))+  
 geom\_smooth(aes(x=FL\_2,y=EBLUPs\_marg\_f2),method=lm, fill = NA)+  
 facet\_wrap(~Gen, ncol = 4, as.table = T, dir = 'v', scales = 'free\_x')+  
 geom\_vline(xintercept = mean(mat.loadings5.star[,1]),linetype="dashed") +   
 geom\_point()+  
 xlab("Cargas do segundo fator")+  
 ylab(expression(EBLUP[m] - lambda[1]^'\*' \* f[1]^'\*'))+  
 labs(color = "Genótipo", title = "EBLUP x Cargas do segundo fator")

EBLUPs\_marg$EBLUPs\_marg\_f3 = EBLUPs\_marg$EBLUP\_marg -   
 ((kronecker(mat.loadings5.star[,1],diag(num.gen)))  
 %\*% as.matrix(fa1.scores.star)+  
 (kronecker(mat.loadings5.star[,2],diag(num.gen)))   
 %\*% as.matrix(fa2.scores.star))  
EBLUPs\_marg$FL\_3 = rep(mat.loadings5.star[,3], each = num.gen)  
  
ggplot(subset(EBLUPs\_marg, Gen %in% c("G9","G32","G20","G23")),   
 aes(x=FL\_3,y=EBLUPs\_marg\_f3,colour=Gen))+  
 geom\_smooth(aes(x=FL\_3,y=EBLUPs\_marg\_f3),method=lm, fill = NA)+  
 facet\_wrap(~Gen, ncol = 4, as.table = T, dir = 'v', scales = 'free\_x')+  
 geom\_vline(xintercept = mean(mat.loadings5.star[,3]),linetype="dashed") +   
 geom\_point()+  
 xlab("Cargas do terceiro fator")+  
 ylab(expression(EBLUP[m] - (lambda[1]^'\*' \* f[1]^'\*' +   
 lambda[2]^'\*' \* f[2]^'\*')))+  
 labs(color = "Genótipo",title = "EBLUP x Cargas do terceiro fator")

EBLUPs\_marg$EBLUPs\_marg\_f4 = EBLUPs\_marg$EBLUP\_marg -   
 ((kronecker(mat.loadings5.star[,1],diag(num.gen)))   
 %\*% as.matrix(fa1.scores.star)+  
 (kronecker(mat.loadings5.star[,2],diag(num.gen)))   
 %\*% as.matrix(fa2.scores.star) +   
 (kronecker(mat.loadings5.star[,3],diag(num.gen)))   
 %\*% as.matrix(fa3.scores.star))  
EBLUPs\_marg$FL\_4 = rep(mat.loadings5.star[,4], each = num.gen)  
  
ggplot(subset(EBLUPs\_marg, Gen %in% c("G9","G32","G20","G23")),   
 aes(x=FL\_4,y=EBLUPs\_marg\_f4,colour=Gen))+  
 geom\_smooth(aes(x=FL\_4,y=EBLUPs\_marg\_f4),method=lm, fill = NA)+  
 facet\_wrap(~Gen, ncol = 4, as.table = T, dir = 'v', scales = 'free\_x')+  
 geom\_vline(xintercept = mean(mat.loadings5.star[,3]),linetype="dashed") +   
 geom\_point()+  
 xlab("Cargas do quarto fator")+  
 ylab(expression(EBLUP[m] - (lambda[1]^'\*' \* f[1]^'\*' +  
lambda[2]^'\*' \* f[2]^'\*' + lambda[3]^'\*' \* f[3]^'\*')))+   
 labs(color = "Genótipo",title = "EBLUP x Cargas do quarto fator")

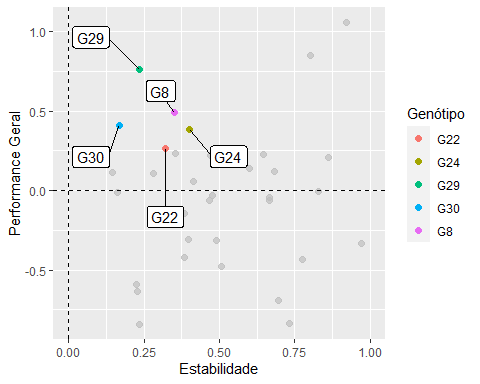
EBLUPs\_marg$EBLUPs\_marg\_f5 = EBLUPs\_marg$EBLUP\_marg -   
 ((kronecker(mat.loadings5.star[,1],diag(num.gen))) %\*%  
 as.matrix(fa1.scores.star)+  
 (kronecker(mat.loadings5.star[,2],diag(num.gen))) %\*%  
 as.matrix(fa2.scores.star) +   
 (kronecker(mat.loadings5.star[,3],diag(num.gen))) %\*%   
 as.matrix(fa3.scores.star) +  
 (kronecker(mat.loadings5.star[,4],diag(num.gen))) %\*%   
 as.matrix(fa4.scores.star))  
EBLUPs\_marg$FL\_5 = rep(mat.loadings5.star[,5], each = num.gen)  
  
ggplot(subset(EBLUPs\_marg, Gen %in% c("G9","G32","G20","G23")),   
 aes(x=FL\_5,y=EBLUPs\_marg\_f5,colour=Gen))+  
 geom\_smooth(aes(x=FL\_5,y=EBLUPs\_marg\_f5),method=lm, fill = NA)+  
 facet\_wrap(~Gen, ncol = 4, as.table = T, dir = 'v', scales = 'free\_x')+  
 geom\_vline(xintercept = mean(mat.loadings5.star[,3]),linetype="dashed") +   
 geom\_point()+  
 xlab("Cargas do quinto fator")+  
 ylab(expression(EBLUP[m] - (lambda[1]^'\*' \* f[1]^'\*' +  
lambda[2]^'\*' \* f[2]^'\*' + lambda[3]^'\*' \* f[3]^'\*' +  
 lambda[4]^'\*' \* f[4]^'\*')))+   
 labs(color = "Genótipo",title = "EBLUP x Cargas do quinto fator")

### Ferramentas de seleção

#### Performance geral x Estabilidade

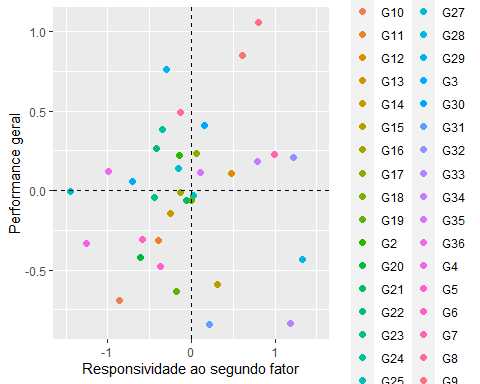
Os limites para seleção dos genótipos ficam a critério do melhorista. Neste exemplo, nosso interesse foi selecionar os cinco melhores. Portanto, definimos que fossem destacados os genótipos com a PG maior ou igual a 0.25 e a ST menor ou igual a 0.5

PG = mean(mat.loadings5.star[,1]) \* as.matrix(fa1.scores.star)  
rest = (EBLUPs\_marg$EBLUP\_marg -   
 (kronecker(mat.loadings5.star[,1],diag(num.gen))) %\*%  
 fa1.scores.star)^2  
STA = data.frame("gen" = rep(name.gen, num.env),  
 "ST" = rest)  
STA = STA %>% group\_by(gen) %>% summarise(ST = sqrt(mean(ST)))  
plot1 = data.frame("gen"=name.gen,  
 "PG" = PG,  
 "ST" = STA$ST)  
  
ggplot(data=plot1)+  
 geom\_point(aes(x = ST, y = PG, color=gen),size=2)+  
 geom\_vline(xintercept = 0,linetype="dashed", colour = "black")+  
 geom\_hline(yintercept = 0,linetype="dashed")+  
 scale\_x\_continuous(name = "Estabilidade", limits = c(0,1))+  
 ylab("Performance Geral") +  
 gghighlight(max(ST)<=0.50,max(PG)>=0.25,use\_direct\_label = F)+  
 geom\_label\_repel(aes(x = ST, y = PG, label = gen), size = 4, box.padding = 1)+  
 labs(color = "Genótipo")

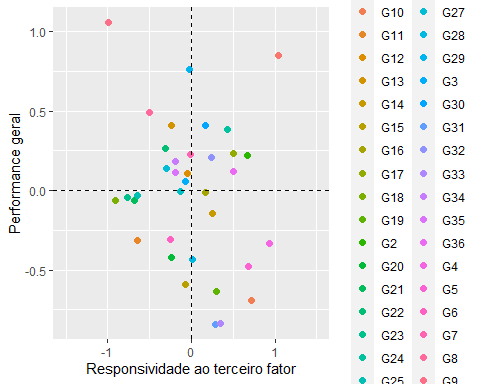


#### Performance geral x Responsividade aos fatores

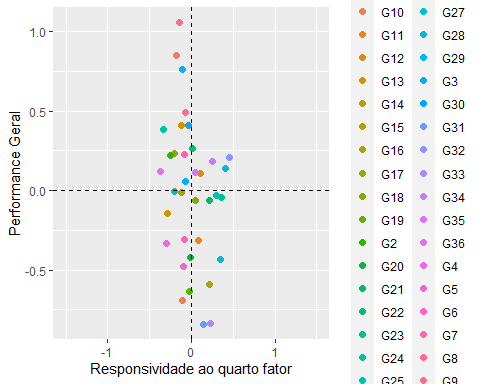
mat.loadings5.star.df = as.data.frame(mat.loadings5.star)  
  
colnames(mat.loadings5.star.df) = c("fa1.loadings","fa2.loadings","fa3.loadings",  
 "fa4.loadings","fa5.loadings")  
  
fa2.mednegload = mean(mat.loadings5.star.df[mat.loadings5.star.df$fa2.loadings <0,]$fa2.loadings)  
fa3.mednegload = mean(mat.loadings5.star.df[mat.loadings5.star.df$fa3.loadings <0,]$fa3.loadings)  
fa4.mednegload = mean(mat.loadings5.star.df[mat.loadings5.star.df$fa4.loadings <0,]$fa4.loadings)  
fa5.mednegload = mean(mat.loadings5.star.df[mat.loadings5.star.df$fa5.loadings <0,]$fa5.loadings)  
  
fa2.medposload = mean(mat.loadings5.star.df[mat.loadings5.star.df$fa2.loadings >0,]$fa2.loadings)  
fa3.medposload = mean(mat.loadings5.star.df[mat.loadings5.star.df$fa3.loadings >0,]$fa3.loadings)  
fa4.medposload = mean(mat.loadings5.star.df[mat.loadings5.star.df$fa4.loadings >0,]$fa4.loadings)  
fa5.medposload = mean(mat.loadings5.star.df[mat.loadings5.star.df$fa5.loadings >0,]$fa5.loadings)  
  
  
respfa2 = (fa2.medposload-fa2.mednegload)\*fa2.scores.star  
respfa3 = (fa3.medposload-fa3.mednegload)\*fa3.scores.star  
respfa4 = (fa4.medposload-fa4.mednegload)\*fa4.scores.star  
respfa5 = (fa5.medposload-fa5.mednegload)\*fa5.scores.star  
  
  
plot2 = data.frame("gen" = name.gen,  
 "PG" = PG,  
 "respfa2" = respfa2,  
 "respfa3" = respfa3,  
 "respfa4" = respfa4,  
 "respfa5" = respfa5)  
  
ggplot(data=plot2)+  
 geom\_point(aes(x = respfa2, y = PG, color=gen),size=2)+  
 geom\_vline(xintercept = 0,linetype="dashed", colour = "black")+  
 geom\_hline(yintercept = 0,linetype="dashed")+  
 ylab("Performance geral") +   
 scale\_x\_continuous(name = "Responsividade ao segundo fator",  
 limits = c(-1.5,1.5))+  
 labs(color = "Genótipo")



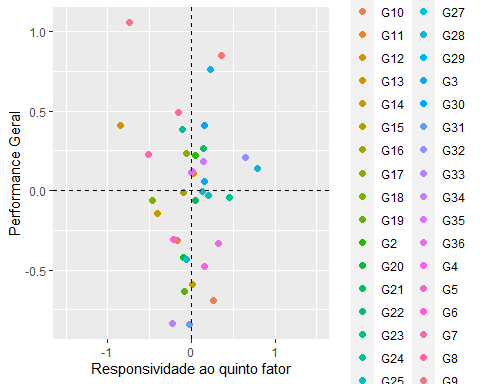
ggplot(data=plot2)+  
 geom\_point(aes(x = respfa3, y = PG, color=gen),size=2)+  
 geom\_vline(xintercept = 0,linetype="dashed", colour = "black")+  
 geom\_hline(yintercept = 0,linetype="dashed")+  
 ylab("Performance geral") +   
 scale\_x\_continuous(name = "Responsividade ao terceiro fator",   
 limits = c(-1.5,1.5))+  
 labs(color = "Genótipo")



ggplot(data=plot2)+  
 geom\_point(aes(x = respfa4, y = PG, color=gen),size=2)+  
 geom\_vline(xintercept = 0,linetype="dashed", colour = "black")+  
 geom\_hline(yintercept = 0,linetype="dashed")+  
 ylab("Performance Geral") +   
 scale\_x\_continuous(name = "Responsividade ao quarto fator",  
 limits = c(-1.5,1.5))+  
 labs(color = "Genótipo")



ggplot(data=plot2)+  
 geom\_point(aes(x = respfa5, y = PG, color=gen),size=2)+  
 geom\_vline(xintercept = 0,linetype="dashed", colour = "black")+  
 geom\_hline(yintercept = 0,linetype="dashed")+  
 ylab("Performance Geral") +   
 scale\_x\_continuous(name = "Responsividade ao quinto fator",  
 limits = c(-1.5,1.5))+  
 labs(color = "Genótipo")



# 3. Análise espacial

## Carregando o conjunto de dados

Para realizar as análises individuais, utilizaremos o conjunto de dados entitulado “espacial.csv”

data\_esp = read.csv("espacial.csv", header = T,sep=";")  
  
data\_esp$Variety = as.factor(data\_esp$Variety)  
data\_esp$Block = as.factor(data\_esp$Block)  
data\_esp$Row = as.factor(data\_esp$Row)  
data\_esp$Col= as.factor(data\_esp$Col)

## Carregando a função para calcular o AIC ajustado ()

Esta função foi contruída por [Verbyla (2018)](https://doi.org/10.1111/anzs.12254)

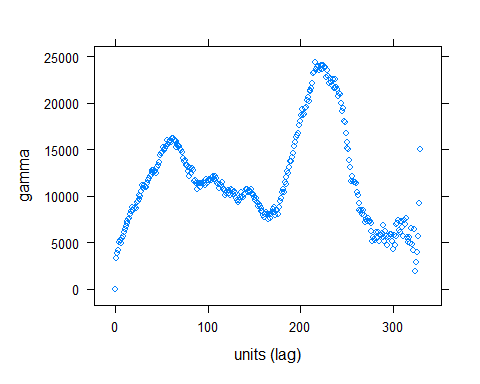
icREML <- function(fm, scale=1) {  
 if(!is.list(fm)) stop(" Models need to be in a list\n")  
 if(is.null(names(fm))) namesfm <- paste("fm", 1:length(fm))  
 else namesfm <- names(fm)  
 require(asreml)  
 asreml.options(Cfixed = TRUE, gammaPar=FALSE)  
 fm <- lapply(fm, function(el) {  
 if(is.null(el$Cfixed)) {  
 out <- update(el, maxit=1) }  
 else out <- el  
 out})  
 logl <- lapply(fm, function(el) el$loglik)  
 summ <- lapply(fm, function(el) summary(el, coef=TRUE)$coef.fixed)  
 which.X0 <- lapply(summ, function(el) !is.na(el[, "z.ratio"]))  
 p.0 <- lapply(which.X0, function(el) sum(el))  
 Cfixed <- lapply(fm, function(el) el$Cfixed)  
 logdet <- lapply(1:length(fm), function(el, Cfixed, which.X0, scale) {  
 log(prod(svd(as.matrix(scale\*Cfixed[[el]][which.X0[[el]],   
 which.X0[[el]]]))$d))  
 }, Cfixed, which.X0, scale)  
 vparam <- lapply(fm, function(el) summary(el)$varcomp)  
 q.0 <- lapply(vparam, function(el) sum(!(el$bound == "F" | el$bound == "B")))  
 b.0 <- lapply(vparam, function(el) sum(el$bound == "F" | el$bound == "B"))  
 logl <- lapply(1:length(fm), function(el, logl, logdet, p.0) {  
 logl[[el]] - logdet[[el]]/2}, logl, logdet,p.0)  
 aic <- unlist(lapply(1:length(fm), function(el, logl, p.0, q.0) {  
 -2\*logl[[el]] + 2\*(p.0[[el]] + q.0[[el]])}, logl, p.0, q.0))  
 bic <- unlist(lapply(1:length(fm), function(el, logl, p.0, q.0, fm) {  
 -2\*logl[[el]] + log(fm[[el]]$nedf+p.0[[el]])\*(p.0[[el]] + q.0[[el]])},  
 logl, p.0, q.0, fm))  
 results <- data.frame(model=namesfm, loglik = unlist(logl), p=unlist(p.0),  
 q=unlist(q.0), b = unlist(b.0), AIC = aic, BIC = bic, logdet=unlist(logdet))  
 row.names(results) <- 1:dim(results)[1]  
 invisible(results)  
}

## Modelo 1: Sem ajuste espacial

m1 <- asreml(yield ~ 1,  
 random = ~ Variety + Block,  
 data = data\_esp)

## Model fitted using the gamma parameterization.  
## ASReml 4.1.0 Thu Sep 15 22:21:01 2022  
## LogLik Sigma2 DF wall cpu  
## 1 -1759.407 14419.82 329 22:21:01 0.0  
## 2 -1757.715 14101.21 329 22:21:01 0.0  
## 3 -1755.926 13741.09 329 22:21:01 0.0  
## 4 -1754.746 13506.11 329 22:21:01 0.0  
## 5 -1754.150 13385.21 329 22:21:01 0.0  
## 6 -1754.000 13345.37 329 22:21:01 0.0  
## 7 -1753.982 13331.28 329 22:21:01 0.0  
## 8 -1753.982 13328.98 329 22:21:01 0.0

plot(varioGram(m1))

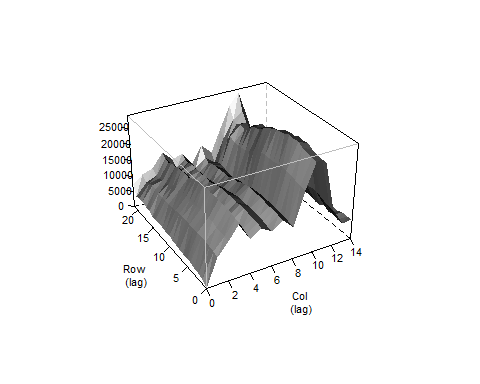


## Modelo 2: Auto-regressivo

m2 <- asreml(yield ~ 1,  
 random = ~ Variety + Block,  
 residual = ~ ar1(Col):ar1(Row),  
 data = data\_esp)

## Model fitted using the gamma parameterization.  
## ASReml 4.1.0 Thu Sep 15 22:21:02 2022  
## LogLik Sigma2 DF wall cpu  
## 1 -1733.826 12580.63 329 22:21:02 0.0 (1 restrained)  
## 2 -1669.961 9508.85 329 22:21:02 0.0  
## 3 -1617.385 8255.14 329 22:21:02 0.0  
## 4 -1601.201 9579.12 329 22:21:02 0.0  
## 5 -1596.617 12113.38 329 22:21:02 0.0  
## 6 -1596.051 13525.06 329 22:21:02 0.0  
## 7 -1595.988 14116.27 329 22:21:02 0.0  
## 8 -1595.981 14330.14 329 22:21:02 0.0  
## 9 -1595.980 14403.23 329 22:21:02 0.0

plot(varioGram(m2))

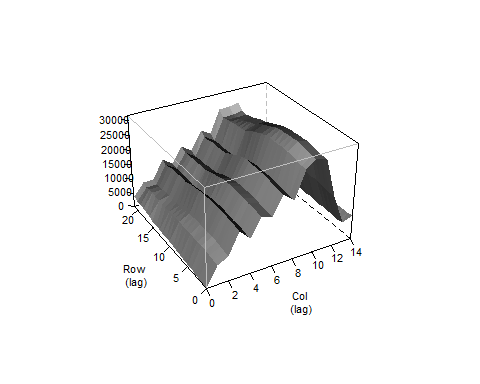


## Modelo 3: Auto-regressivo + Pepita

m3 <- asreml(yield ~ 1,  
 random = ~ Variety + Block + units,  
 residual = ~ ar1(Col):ar1(Row),  
 data = data\_esp)

## Model fitted using the gamma parameterization.  
## ASReml 4.1.0 Thu Sep 15 22:21:02 2022  
## LogLik Sigma2 DF wall cpu  
## 1 -1736.552 11681.29 329 22:21:02 0.0  
## 2 -1706.200 5103.51 329 22:21:02 0.0 (1 restrained)  
## 3 -1623.773 8425.35 329 22:21:02 0.0 (1 restrained)  
## 4 -1583.800 6988.33 329 22:21:02 0.0 (1 restrained)  
## 5 -1570.095 7410.42 329 22:21:02 0.0 (1 restrained)  
## 6 -1565.509 17054.33 329 22:21:02 0.0  
## 7 -1564.713 17486.14 329 22:21:02 0.0  
## 8 -1564.520 19816.65 329 22:21:02 0.0  
## 9 -1564.467 20811.71 329 22:21:02 0.0  
## 10 -1564.452 21432.42 329 22:21:02 0.0  
## 11 -1564.449 21745.41 329 22:21:02 0.0  
## 12 -1564.448 21905.70 329 22:21:02 0.0  
## 13 -1564.447 21985.94 329 22:21:02 0.0

plot(varioGram(m3))



## Modelo 4: Anteriores + Aleatório de linha e coluna

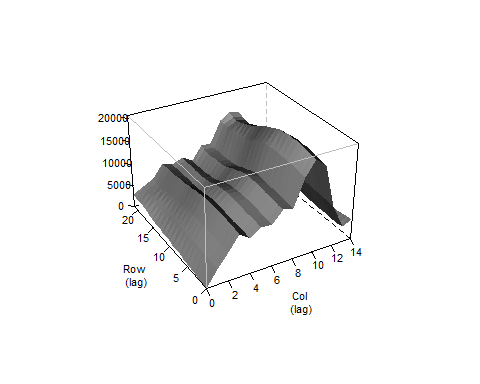
m4 <- asreml(yield ~ 1,  
 random = ~ Variety + Block + Row + Col + units,  
 residual = ~ ar1(Col):ar1(Row),  
 data = data\_esp)

## Model fitted using the gamma parameterization.  
## ASReml 4.1.0 Thu Sep 15 22:21:03 2022  
## LogLik Sigma2 DF wall cpu  
## 1 -1671.318 7224.596 329 22:21:03 0.1  
## 2 -1650.323 4153.455 329 22:21:03 0.0 (2 restrained)  
## 3 -1603.162 2992.391 329 22:21:03 0.0 (1 restrained)  
## 4 -1575.329 2068.590 329 22:21:03 0.0 (1 restrained)  
## 5 -1564.068 2600.891 329 22:21:03 0.0 (6 restrained)  
## 6 -1569.079 27749.195 329 22:21:03 0.1 (1 restrained)  
## 7 -1562.548 18371.564 329 22:21:03 0.0 (1 restrained)  
## 8 -1560.267 12975.963 329 22:21:03 0.0  
## 9 -1559.832 10683.326 329 22:21:03 0.0  
## 10 -1559.824 10696.262 329 22:21:03 0.0  
## 11 -1559.824 10714.842 329 22:21:03 0.0

m4 <- update(m4)

## Model fitted using the gamma parameterization.  
## ASReml 4.1.0 Thu Sep 15 22:21:03 2022  
## LogLik Sigma2 DF wall cpu  
## 1 -1559.824 10718.86 329 22:21:03 0.0  
## 2 -1559.824 10719.23 329 22:21:03 0.0

plot(varioGram(m4))

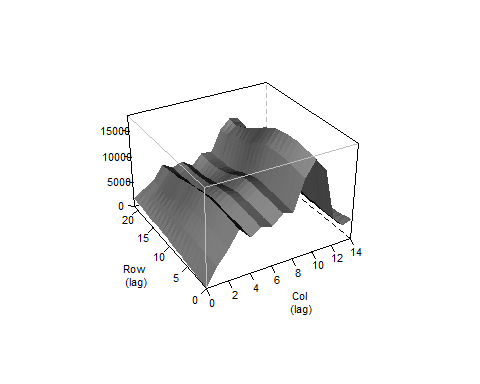


## Modelo 5: Anteriores + Efeito linear fixo de linha e coluna

m5 <- asreml(yield ~ lin(Row) + lin(Col),  
 random = ~ Variety + Block + Row + Col + units,  
 residual = ~ ar1(Col):ar1(Row),  
 data = data\_esp)

## Model fitted using the gamma parameterization.  
## ASReml 4.1.0 Thu Sep 15 22:21:04 2022  
## LogLik Sigma2 DF wall cpu  
## 1 -1663.625 7000.845 327 22:21:04 0.0  
## 2 -1642.771 4087.477 327 22:21:04 0.0 (2 restrained)  
## 3 -1598.696 2819.293 327 22:21:04 0.0 (1 restrained)  
## 4 -1570.582 1932.493 327 22:21:04 0.1 (1 restrained)  
## 5 -1559.324 2336.948 327 22:21:04 0.0 (6 restrained)  
## 6 -1563.788 25206.783 327 22:21:04 0.0 (1 restrained)  
## 7 -1557.628 17133.883 327 22:21:04 0.1 (1 restrained)  
## 8 -1555.542 12206.958 327 22:21:04 0.0  
## 9 -1555.144 10046.319 327 22:21:04 0.0  
## 10 -1555.138 10034.896 327 22:21:04 0.0

plot(varioGram(m5))

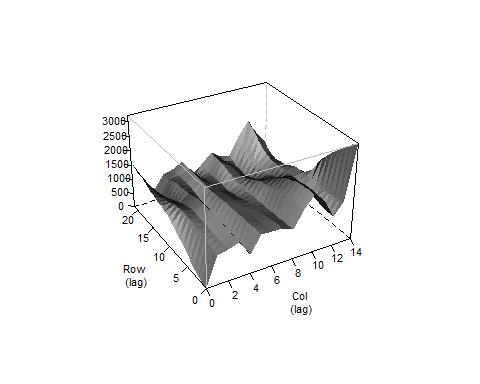


## Modelo 6: Anteriores + Spline no sentido da coluna

m6 <- asreml(yield ~ lin(Row) + lin(Col),  
 random = ~ Variety + Block + Row + Col + spl(col) + units,  
 residual = ~ ar1(Col):ar1(Row),  
 data = data\_esp)

## Model fitted using the gamma parameterization.  
## ASReml 4.1.0 Thu Sep 15 22:21:04 2022  
## Spline: design points closer than 0.0014 have been merged.  
## LogLik Sigma2 DF wall cpu  
## 1 -1613.452 5087.717 327 22:21:05 0.1  
## 2 -1601.388 3169.909 327 22:21:05 0.0 (3 restrained)  
## 3 -1578.675 2181.500 327 22:21:05 0.0 (2 restrained)  
## 4 -1558.494 1661.957 327 22:21:05 0.0 (2 restrained)  
## 5 -1553.107 2055.506 327 22:21:05 0.1 (1 restrained)  
## 6 -1551.973 3062.809 327 22:21:05 0.1  
## 7 -1551.903 3195.873 327 22:21:05 0.1  
## 8 -1551.894 3191.642 327 22:21:05 0.1  
## 9 -1551.893 3136.080 327 22:21:05 0.1

plot(varioGram(m6))



mods = list(m1=m1, m2=m2, m3=m3, m4=m4, m5=m5, m6=m6)  
  
selection <- icREML(mods)

## Model fitted using the gamma parameterization.  
## ASReml 4.1.0 Thu Sep 15 22:21:05 2022  
## LogLik Sigma2 DF wall cpu  
## 1 -1753.982 13328.91 329 22:21:05 0.0  
## Model fitted using the gamma parameterization.  
## ASReml 4.1.0 Thu Sep 15 22:21:05 2022  
## LogLik Sigma2 DF wall cpu  
## 1 -1595.980 14427.88 329 22:21:05 0.0  
## Model fitted using the gamma parameterization.  
## ASReml 4.1.0 Thu Sep 15 22:21:06 2022  
## LogLik Sigma2 DF wall cpu  
## 1 -1564.447 22025.96 329 22:21:06 0.0  
## Model fitted using the gamma parameterization.  
## ASReml 4.1.0 Thu Sep 15 22:21:06 2022  
## LogLik Sigma2 DF wall cpu  
## 1 -1559.824 10720.13 329 22:21:06 0.1  
## Model fitted using the gamma parameterization.  
## ASReml 4.1.0 Thu Sep 15 22:21:06 2022  
## LogLik Sigma2 DF wall cpu  
## 1 -1555.137 10047.75 327 22:21:06 0.0  
## Model fitted using the gamma parameterization.  
## ASReml 4.1.0 Thu Sep 15 22:21:06 2022  
## Spline: design points closer than 0.0014 have been merged.  
## LogLik Sigma2 DF wall cpu  
## 1 -1551.893 3139.793 327 22:21:06 0.1

selection

## model loglik p q b AIC BIC logdet  
## 1 m1 -1758.165 1 3 0 3524.331 3539.527 8.367356  
## 2 m2 -1600.114 1 5 0 3212.229 3235.023 8.268164  
## 3 m3 -1568.899 1 5 1 3149.798 3172.592 8.903010  
## 4 m4 -1563.932 1 7 1 3143.865 3174.258 8.216938  
## 5 m5 -1562.312 3 7 1 3144.624 3182.615 14.349528  
## 6 m6 -1556.860 3 8 1 3135.721 3177.511 9.934153