Package ‘CodataGS’

October 12, 2022

Type Package

Title Genomic Prediction Using SNP Codata

Version 1.43

Date 2019-05-17

Author Lars Ronnegard

Maintainer Lars Ronnegard <lrn@du.se>

Description Computes genomic breeding values using external information on the markers. The package fits a linear mixed model with heteroscedastic random effects, where the random effect variance is fitted using a linear predictor and a log link. The method is described in Mouresan, Selle and Ronnegard (2019) <doi:10.1101/636746>.

License GPL

Depends Matrix

NeedsCompilation no

Repository CRAN

Date/Publication 2019-05-17 15:40:07 UTC

R topics documented:

`CodataGS-package` *Genomic Prediction Using SNP Codata*

Description

Computes genomic breeding values using external information on the markers. The package fits a linear mixed model with heteroscedastic random effects, where the random effect variance is fitted using a linear predictor and a log link. The method is described in Mouresan, Selle and Ronnegard (2019) <doi:10.1101/636746>.

Details

The DESCRIPTION file:

Package:	CodataGS
Type:	Package
Title:	Genomic Prediction Using SNP Codata
Version:	1.43
Date:	2019-05-17
Author:	Lars Ronnegard
Maintainer:	Lars Ronnegard <lrn@du.se>
Description:	Computes genomic breeding values using external information on the markers. The package fits a linear
License:	GPL
Depends:	Matrix
NeedsCompilation:	no

Index of help topics:

```
CodataGS-package        Genomic Prediction Using SNP Codata
MME                     Mixed model equations
Transform               Transforms hat values
compute_GL              Computes genomic relationship matrix
compute_phitau          Computes models for the variance components
genomicEBV.w.codata     Performs genomic prediction based on SNP
                        codata.
hat.transf              Transforms hat values
scaleZ                  Scales the genotype matrix.
summary.CodataGS        Summary method for CodataGS objects
```

This package performs genomic prediction based on SNP codata. The main function is genomicEBV.w.codata.

Author(s)

Lars Ronnegard

Maintainer: Lars Ronnegard <lrn@du.se>

compute_GL *Computes genomic relationship matrix*

Description

This function computes the genomic relationship matrix, G, together with its matrix square root, L.

Usage

```
compute_GL(Z, w)
```

Arguments

Z	Scaled matrix with genotype information
w	weights

Value

L	Square root matrix of G
svdVec	Vectors in the Single Value Decomposition of G
svdD	Diagonal elements in the Single Value Decomposition of G
wZt	weights times the transpose of Z

Author(s)

Lars Ronnegard

Examples

```
set.seed(1234)
N <- 20 #Number of individuals
k <- 30 #Number of SNPs with all marker positions including a QTL
Z1 <- matrix(0, N, k )
Z2 <- matrix(0, N, k )
Z1[1:N, 1] <- rbinom(N, 1, 0.5) #Simulated phased SNP matrices
Z2[1:N, 1] <- rbinom(N, 1, 0.5)
LD.par <- 0.2 #A parameter to simulate LD. 0 gives full LD, and 0.5 no LD
for (j in 2:k) {
  Z1[1:N, j] <- abs( Z1[1:N, j-1] - rbinom(N, 1, LD.par) )
  Z2[1:N, j] <- abs( Z2[1:N, j-1] - rbinom(N, 1, LD.par) )
}
Z <- Z1 + Z2 #Genotypic SNP matrix
sim.res <- compute_GL(Z, w = rep(1,k))
```

compute_phitau *Computes models for the variance components*

Description

This function computes the residual variance, the SNP variances and the linear predictor for the SNP variance model.

Usage

```
compute_phitau(dev, hv, devu, hvu, X.rand.disp)
```

Arguments

dev	Deviance values
hv	Hat values for the observed response values
devu	Deviance values computed for the random effects
hvu	Hat values for the random effects
X.rand.disp	Design matrix used in the linear predictor for the SNP variance model.

Value

var.e	Residual variance
phi	Vector of SNP variances
coef	Fitted coefficients for the linear predictor in the SNP variance model

Author(s)

Lars Ronnegard

Examples

```
set.seed(1234)
N <- 20 #Number of individuals
k <- 30 #Number of SNPs with all marker positions including a QTL
#Simulated deviances and hat values
dev <- rnorm(N)^2
hv <- runif(N, 0.1, 0.5)
devu <- rnorm(k)^2
hvu <- runif(k, 0.1, 0.85)
X.rand.disp <- matrix(1, k, 1)
sim.res <- compute_phitau(dev, hv, devu, hvu, X.rand.disp)
```

genomicEBV.w.codata *Performs genomic prediction based on SNP codata.*

Description

The main function of the package. The input includes response values, a design matrix for the fixed effects, a matrix with SNP genotype data and a design matrix for the SNP codata.

Usage

```
genomicEBV.w.codata(y, X, Z, X.SNPcodata, Z.test = NULL, max.iter = 100, conv.crit = 1e-5)
```

Arguments

`y`	Response values
`X`	Design matrix for the fixed effects
`Z`	Genotype matrix with element values of 0, 1 or 2
`X.SNPcodata`	Design matrix for the linear predictor of the SNP variances.
`Z.test`	An optional genotype matrix for a test data set.
`max.iter`	The maximum number of iterations
`conv.crit`	The value of the convergence criterion.

Details

By specifying the matrix `Z.test` in the input, the function computes predicted genomic breeding values for an out-of-sample data set.

Value

`gEBV`	Genomic breeding values
`predicted.gEBV`	Genomic breeding values based on the genotypes in `Z.test`
`w`	Computed SNP weights
`u`	Fitted SNP effects
`beta`	Fitted fixed effects
`disp.beta`	Fitted coefficients in the linear predictor for the SNP variance model
`Converge`	Shows whether the algorithm has converged or not
`iter`	The number of iterations used

Author(s)

Lars Ronnegard

Examples

```
#######
#Simulation part
set.seed(1234)
N <- 200 #Number of individuals
k <- 300 #Number of SNPs with all marker positions including a QTL
Z1 <- matrix(0, N, k )
Z2 <- matrix(0, N, k )
Z1[1:N, 1] <- rbinom(N, 1, 0.5) #Simulated phased SNP matrices
Z2[1:N, 1] <- rbinom(N, 1, 0.5)
LD.par <- 0.2 #A parameter to simulate LD. 0 gives full LD, and 0.5 no LD
for (j in 2:k) {
  Z1[1:N, j] <- abs( Z1[1:N, j-1] - rbinom(N, 1, LD.par) )
  Z2[1:N, j] <- abs( Z2[1:N, j-1] - rbinom(N, 1, LD.par) )
}
Z <- Z1 + Z2 #Genotypic SNP matrix
x1 <- c(rep(1,k/2), rep(0,k/2)) #An indicator for the SNPs.
#The first k/2 SNPs and the last k/2 have different variances
#Simulate linear predictor for the random effect variance
lin.pred <- 0 + 2*x1
X.snp <- model.matrix( ~ x1 ) #Corresponding design matrix
u <- rnorm(k, 0 , sqrt( exp(lin.pred) ))
#Took the square root here because it is the SD that is specified.
#and exp() because we are modelling a log link.
u.scaled <- u/as.numeric( sqrt( var( crossprod(t(Z), u) )) )
#Scaled by the variance of the breeding values
e <- rnorm(N) #A residual variance
mu <- 0
y <- mu + crossprod(t(Z),u.scaled) + e
######
#Estimation part
mod1 <- genomicEBV.w.codata(y = as.numeric(y),
          X = matrix(1, N, 1), Z = Z, X.SNPcodata = X.snp)
#To fit gBLUP just specify X.SNPcodata = matrix(1, k, 1)
cat("Correlation between true and estimated BV for the codata model:")
cat(cor(crossprod(t(Z),u.scaled), mod1$gEBV), "\n")
```

hat.transf *Transforms hat values*

Description

Transforms hat values between the SNP-BLUP model and the gBLUP model.

Usage

```
hat.transf(C22, transf, vc, k, N, w)
```

Arguments

`C22`	Submatrix of the inverse of the LHS in the MME
`transf`	A transformation matrix.
`vc`	Genetic variance
`k`	Number of SNPs
`N`	Number of individuals
`w`	SNP weights

Value

Transformed hat values

Author(s)

Lars Ronnegard

`MME` *Mixed model equations*

Description

A fast version of the Henderson’s mixed model equations (MME)

Usage

```
MME(y, X, Z, var.e, var.u)
```

Arguments

`y`	Response
`X`	Design matrix for fixed effects
`Z`	Design matrix for the random effects
`var.e`	Residual variance
`var.u`	Genetic variance

Value

`beta`	Estimates of fixed effects
`v`	Fitted random effects
`hv`	Hat values
`dev`	Deviances

Author(s)

Lars Ronnegard

scaleZ *Scales the genotype matrix.*

Description

Scales the genotype matrix so that ZZ' gives the genomic relationship matrix.

Usage

```
scaleZ(Z, freq1)
```

Arguments

`Z` Genotype matrix with element values 0, 1 and 2

`freq1` Optional input parameter with allele frequencies. A vector of length equal to the number of columns in Z.

Value

`Z` Scaled genotype matrix

Author(s)

Lars Ronnegard

Examples

```
#######
#Simulation part
set.seed(1234)
N <- 200 #Number of individuals
k <- 300 #Number of SNPs with all marker positions including a QTL
Z1 <- matrix(0, N, k )
Z2 <- matrix(0, N, k )
Z1[1:N, 1] <- rbinom(N, 1, 0.5) #Simulated phased SNP matrices
Z2[1:N, 1] <- rbinom(N, 1, 0.5)
LD.par <- 0.2 #A parameter to simulate LD. 0 gives full LD, and 0.5 no LD
for (j in 2:k) {
  Z1[1:N, j] <- abs( Z1[1:N, j-1] - rbinom(N, 1, LD.par) )
  Z2[1:N, j] <- abs( Z2[1:N, j-1] - rbinom(N, 1, LD.par) )
}
Z <- Z1 + Z2 #Genotypic SNP matrix
sim.res <- scaleZ(Z)
```

summary.CodataGS *Summary method for CodataGS objects*

Description

A summary method for the object class `CodataGS`

Usage

```
## S3 method for class 'CodataGS'
summary(object, ...)
```

Arguments

`object`	A CodataGS object
`...`	arguments not used

Details

Provides a concise summary of CodataGS objects.

Examples

```
#######
#Simulation part
set.seed(1234)
N <- 200 #Number of individuals
k <- 300 #Number of SNPs with all marker positions including a QTL
Z1 <- matrix(0, N, k )
Z2 <- matrix(0, N, k )
Z1[1:N, 1] <- rbinom(N, 1, 0.5) #Simulated phased SNP matrices
Z2[1:N, 1] <- rbinom(N, 1, 0.5)
LD.par <- 0.2 #A parameter to simulate LD. 0 gives full LD, and 0.5 no LD
for (j in 2:k) {
  Z1[1:N, j] <- abs( Z1[1:N, j-1] - rbinom(N, 1, LD.par) )
  Z2[1:N, j] <- abs( Z2[1:N, j-1] - rbinom(N, 1, LD.par) )
}
Z <- Z1 + Z2 #Genotypic SNP matrix
x1 <- c(rep(1,k/2), rep(0,k/2)) #An indicator for the SNPs.
#The first k/2 SNPs and the last k/2 have different variances
#Simulate linear predictor for the random effect variance
lin.pred <- 0 + 2*x1
X.snp <- model.matrix( ~ x1 ) #Corresponding design matrix
u <- rnorm(k, 0 , sqrt( exp(lin.pred) ))
#Took the square root here because it is the SD that is specified.
#and exp() because we are modelling a log link.
u.scaled <- u/as.numeric( sqrt( var( crossprod(t(Z), u) )) )
#Scaled by the variance of the breeding values
e <- rnorm(N) #A residual variance
```

```
mu <- 0
y <- mu + crossprod(t(Z),u.scaled) + e
######
#Estimation part
mod1 <- genomicEBV.w.codata(y = as.numeric(y),
          X = matrix(1, N, 1), Z = Z, X.SNPcodata = X.snp)
summary(mod1)
```

Transform *Transforms hat values*

Description

The function calls the hat.transf function.

Usage

```
Transform(X, L, var.e, var.u, v, svdVec, svdD, wZt, w)
```

Arguments

X	Design matrix for the fixed effects
L	Square root matrix of the genomic relationship matrix, G
var.e	Residual variance
var.u	Genetic variance
v	Random effects
svdVec	Vector from the Single Value Decomposition of G
svdD	Diagonal elements of the Single Value Decomposition of G
wZt	Weights times the transpose of the scaled genotype matrix
w	Fitted SNP weights

Value

u	SNP effects
qu	Hat values for the SNP effects

Author(s)

Lars Ronnegard

Index