Package: MixRF (via r-universe)

September 11, 2024

Title A Random-Forest-Based Approach for Imputing Clustered Incomplete Data

Version 1.0

Date 2016-04-05

Author Jiebiao Wang and Lin S. Chen

Maintainer Jiebiao Wang <randel.wang@gmail.com>

Description It offers random-forest-based functions to impute clustered incomplete data. The package is tailored for but not limited to imputing multitissue expression data, in which a gene's expression is measured on the collected tissues of an individual but missing on the uncollected tissues.

License GPL

Depends doParallel, randomForest, lme4, foreach

URL https://github.com/randel/MixRF

BugReports https://github.com/randel/MixRF/issues

RoxygenNote 5.0.1

Repository https://randel.r-universe.dev

RemoteUrl https://github.com/randel/mixrf

RemoteRef HEAD

RemoteSha df5c7817a7721538c37c1725b4961b5d7ab4fd73

Contents

`MixRF-package` *A random-forest-based algorithm for imputing clustered incomplete data*

Description

This package offers random-forest-based functions to impute clustered incomplete data. The package is tailored for but not limited to imputing multitissue expression data, in which a gene's expression is measured on the collected tissues of an individual but missing on the uncollected tissues.

Details

Package:	MixRF
Type:	Package
Version:	1.0
Date:	2016-04-05
License:	GPL
LazyLoad:	yes

Author(s)

Jiebiao Wang and Lin S. Chen

Maintainer: Jiebiao Wang <randel.wang@gmail.com>

References

Wang, J., Gamazon, E.R., Pierce, B.L., Stranger, B.E., Im, H.K., Gibbons, R.D., Cox, N.J., Nicolae, D.L. and Chen, L.S. (2016) Imputing gene expression in uncollected tissues within and beyond GTEx. http://dx.doi.org/10.1016/j.ajhg.2016.02.020

See Also

`MixRF.impute`

get_eqtl *Calculate cis- and trans-eQTLs*

Description

Calculate cis- and trans-eQTLs

Usage

```
get_eqtl(ncore, Ynew, ssnpDat, snp.info, gene.info, cov)
```

Arguments

ncore	The number of cores for parallel computing.
Ynew	An array of expression data of dimension sample-by-gene-by-tissue, nxpxT, where n is sample size. p is the number of genes, and T is the number of tissues.
ssnpDat	The genotype data matrix (n by SNP size).
snp.info	Input for MatrixEQTL, with col.names snpID, chr, pos.
gene.info	Input for MatrixEQTL, with col.names geneID, chr, lpos, rpos.
cov	The covariates matrix for MatrixEQTL.

Value

A list contains the cis- and trans-eQTLs for each gene.

Examples

```
## Not run:
# a fake example

# eqtl_list = get_eqtl(ncore=2, Ynew, ssnpDat, snp.info, gene.info, cov)

## End(Not run)
```

MixRF *Mixed Random Forest*

Description

The function to fit a random forest with random effects.

Usage

```
MixRF(Y, X, random, data, initialRandomEffects = 0, ErrorTolerance = 0.001,
    MaxIterations = 1000)
```

Arguments

Y	The outcome variable.
X	A data frame or matrix contains the predictors.
random	A string in lme4 format indicates the random effect model.
data	The data set as a data frame.
initialRandomEffects	The initial values for random effects.
ErrorTolerance	The tolerance for log-likelihood.
MaxIterations	The maximum iteration times.

Value

A list contains the random forest ($forest), mixed model ($MixedModel), and random effects ($RandomEffects). See the example below for the usage.

Examples

```
data(sleepstudy)

tmp = MixRF(Y = sleepstudy$Reaction, X = as.data.frame(sleepstudy$Days),
    random = "(Days|Subject)", data = sleepstudy, initialRandomEffects = 0,
    ErrorTolerance = 0.01, MaxIterations = 100)

# tmp$forest

# tmp$MixedModel

# tmp$RandomEffects
```

MixRF.impute *Impute a large number of genes using the MixRF algorithm with parallel computing*

Description

This function impute the expression of a large number of genes using the MixRF algorithm with parallel computing.

Usage

```
MixRF.impute(Ydat, eqtl.lis, snp.dat, cov = NULL, iPC = TRUE,
    idx.selected.gene.iPC = NULL, parallel.size = 1, correlation = FALSE,
    nCV = 3)
```

Arguments

Ydat	An array of expression data of dimension sample-by-gene-by-tissue, nxpxT, where n is sample size. p is the number of genes, and T is the number of tissues. Ydat[,1,] is a matrix of the first gene expression in T tissues for n individuals, nxT. Ydat[,,1] is a nxp matrix of the expression data of p genes in the first tissue.
eqtl.lis	A list of eQTL names of length p. Each element in the list contains the name of the eQTLs for the corresponding gene. The order of the list should correspond to the order of genes in Ydat. The code and example to calculate eQTLs can be found at https://github.com/randel/MixRF/blob/master/R/eqtl.r.
snp.dat	A matrix of genotype. Each row is a sample and each column corresponds to one SNP. The column names should match eqtl.lis.
cov	A matrix of covariates. Each row is a sample and each column corresponds to one covariate. For example, age, gender.
iPC	An option. When it is TRUE, the imputed PCs (iPCs) for each tissue type will be constructed based on the combined observed and imputed data on the selected genes. The iPCs will be adjusted as covariates in the imputation.
idx.selected.gene.iPC	The option is used only when iPC=TRUE. When it is, one may select a subset of genes and impute those first to construct iPCs.
parallel.size	A numerical value specifying the number of CPUs/cores/processors available for parallel computing.
correlation	The option to calculate the imputation correlation using cross-validation or not. The default is FALSE.
nCV	The option is used only when correlation=TRUE. The number of folds for cross-validation. The default is 3 folds.

Value

An nxpxT array of imputed and observed expression data. The observed values in Ydat are still kept and the missing values in Ydat are imputed. When the user chooses to calculate the imputation correlation using cross-validation (correlation=TRUE), the estimated imputation correlation (cor) will also be returned in a list together with the imputed data (Yimp).

Examples

```
## Not run:
data(sim)

idx.selected.gene.iPC = which(sapply(sim$eqtl.lis, length) >= 1)

Yimp = MixRF.impute(sim$Ydat, sim$eqtl.lis, sim$snp.dat, sim$cov, iPC = TRUE,
    idx.selected.gene.iPC, parallel.size = 4)

## End(Not run)
```

MixRFb *Mixed Logistic Random Forest for Binary Data*

Description

Mixed Logistic Random Forest for Binary Data

Usage

```
MixRFb(Y, x, random, data, initialRandomEffects = 0, ErrorTolerance = 0.001,
  MaxIterations = 200, ErrorTolerance0 = 0.001, MaxIterations0 = 15,
  verbose = FALSE)
```

Arguments

Y	The outcome variable.
x	A formula string contains the predictors.
random	A string in lme4 format indicates the random effect model.
data	The data set as a data frame.
initialRandomEffects	The initial values for random effects.
ErrorTolerance	The tolerance for log-likelihood.
MaxIterations	The maximum iteration times for each run of PQL.
ErrorTolerance0	The tolerance for eta (penalized quasi-likelihood, PQL).
MaxIterations0	The maximum iteration times for PQL.
verbose	The option to monitor each run of PQL or not.

Value

A list contains the random forest, mixed model, and random effects. See the example below for the usage. A predict() function is also available below.

Examples

```
# example data (http://stats.stackexchange.com/questions/70783/how-to-assess-the-fit-of-a-binomial-glmm-fitted-
dat <- read.table("http://pastebin.com/raw.php?i=vRy66Bif")

library(party)
library(lme4)

source('MixRFb.r')
system.time(tmp <- MixRFb(Y=dat$true, x='factor(distance) + consequent + factor(direction) + factor(dist)', random
                          data=dat, initialRandomEffects=0,
                          ErrorTolerance=1, MaxIterations=200,
                          ErrorTolerance0=0.3, MaxIterations0=15, verbose=T))
```

```
# tmp$forest
# tmp$MixedModel
# tmp$RandomEffects

# eta
pred1 = predict.MixRF(tmp, dat, EstimateRE=TRUE)
prob = 1/(1+exp(-pred1))
res = (prob>.5)

# classification
table(res,dat$true)
```

`predict.MixRF` *Prediction Function for MixRF*

Description

Prediction Function for MixRF

Usage

```
## S3 method for class 'MixRF'
predict(object, newdata, id = NULL, EstimateRE = TRUE)
```

Arguments

`object`	The fitted MixRF object.
`newdata`	A data frame contains the predictors for prediction.
`id`	The group variable in the new data.
`EstimateRE`	To use the estimated random effects in the prediction or not. The default is TRUE.

Value

A matrix (now for balanced data) contains the predicted values.

Examples

```
library(lme4)
library(randomForest)
data(sleepstudy)

tmp = MixRF(Y=sleepstudy$Reaction, x=as.data.frame(sleepstudy$Days), random='(Days|Subject)',
        data=sleepstudy, initialRandomEffects=0, ErrorTolerance=0.01, MaxIterations=100)

pred = predict.MixRF(object=tmp, newdata=sleepstudy, EstimateRE=TRUE)
```

`sim` *Simulated data list*

Description

This simulated data list is for demonstration.

Value

`Ydat`	An array of expression data of dimension sample-by-gene-by-tissue, nxpxT, where n is sample size. p is the number of genes, and T is the number of tissues. Ydat[,1,] is a matrix of the first gene expression in T tissues for n individuals, nxT. Ydat[,,1] is a nxp matrix of the expression data of p genes in the first tissue.
`eqtl.lis`	A list of eQTL names of length p. Each of the element in the list contains the name of the eQTLs for the corresponding gene. The order of the list should correspond to the order of genes in Ydat.
`snp.dat`	A matrix of genotype. Each row is a sample and each column corresponds to one SNP. The column names should match eqtl.lis.
`cov`	A matrix of covariates. Each row is a sample and each column corresponds to one covariate. For example, age, gender.

See Also

`MixRF.impute`

Index