Package ‘krm’
March 4, 2015

LazyLoad yes
LazyData yes
Version 2015.3-4
Title Kernel Based Regression Models
Author Youyi Fong <youyifong@gmail.com>, Saheli Datta, Krisztian Sebestyen
Maintainer Youyi Fong <youyifong@gmail.com>
Depends R (>= 3.0.0), kyotil
Suggests RUnit, MASS
Description Implements several methods for testing the variance component parameter in regression models that contain kernel-based random effects, including a maximum of adjusted scores test. Several kernels are supported, including a profile hidden Markov model mutual information kernel for protein sequence.
License GPL-2
NeedsCompilation yes
Repository CRAN
Date/Publication 2015-03-04 21:30:37

R topics documented:

aa.prop.list ................................................................. 2
calcPairwiseIdentity ....................................................... 2
chi.norm ................................................................. 3
cloud9 ................................................................. 3
dmdirichlet ............................................................... 4
getSeqKernel ............................................................... 5
hmmMargLlik ............................................................... 6
krm package ................................................................. 6
krm.most ................................................................. 6
krm.score.test ............................................................. 8
readFastaFile ............................................................. 9
sim.liu.2008 ............................................................... 10

Index 11
Amino Acid Properties

Description

Amino Acid Properties

Format

A data frame with 20 observations on the following 13 variables.

- **Symbol**: a character vector with 20 values: A through Y
- **AA_Name**: a character vector with 20 values: Alanine through Tyrosine
- **AA_Symbol**: a character vector with 20 values: Ala through Tyr
- **Surface_Area_Cothia**: a numeric vector
- **Residue_Volume_Zamayatin**: a numeric vector
- **Bulikness_Jones**: a numeric vector
- **Polarity_Jones**: a numeric vector
- **Refractivity_Jones**: a numeric vector
- **Hydrophobicity_Engeleman**: a numeric vector
- **Hydrophobicity_Prabha**: a numeric vector
- **Hydrophilicity_Hopp**: a numeric vector
- **Hydrophilicity_Levitt**: a numeric vector
- **RelMutability_Jones**: a numeric vector

Functions Related to Sequence Alignment

Description

Functions related to sequence alignment

Usage

```r
calcPairwiseIdentity(alignment, dissimilarity, removeGap)
alignment2count(alignment, level=20, weight=rep(1,nrow(alignment)))
alignment2trancount(alignment, weight=rep(1,nrow(alignment)))
removeGap(seq)
```
chi.norm

Arguments

alignment matrix of arabic representation of sequences (1 based)
dissimilarity Boolean.
removeGap Boolean
level integer. Size of alphabet
weight numeric vector. Weights given to each sequence
seq string. A string of amino acids

Value

alignment2count return T by 20 matrix, where T is the number of column in the alignment. alignment2trancount return a T by 4 matrix, each row is the count of MM, MD, DM, DD for each position.

chi.norm A Transformation of Chi-squared Random Variable

Description

A transformation of Chi-squared random variable to make it normal like.

Usage

chi.norm(q, v)

Arguments

q numeric. A random variable following chi-squared distribution
v numeric. A random variable following normal distribution

cloud9 9-Component Mixture Dirichlet Prior for Protein Sequences

Description

9-Component Mixture Dirichlet Prior for Protein Sequences

Format

List of 2. The alpha element is a 9 by 20 matrix, where each row represents one Dirichlet distribution of 20 dimensions. The mix.coef element contains the mixing probability, a vector of 9 numbers that add up to 1.
Functions related to mixture Dirichlet distribution

Usage

`dmdirichlet(x, malpha, mixturecoef)`
`ddirichlet(x, alpha)`
`rdirichlet(n, alpha)`
`rmdirichlet(malpha, mixturecoef)`
`modifydirichlet(prior, y)`
`logintegrateMixDirichlet(y, prior, tau=1)`
`logIntegrateDirichlet(y, alpha)`

Arguments

- `x`: A vector containing a single deviate or matrix containing one random deviate per row.
- `malpha`: matrix. Each row is a parameter of Dirichlet
- `alpha`: numeric vector. Parameter for a Dirichlet distribution
- `mixturecoef`: numeric vector
- `n`: integer
- `prior`: list of two components: alpha and mix.coef
- `y`: numeric vector of counts
- `tau`: numeric

Details

`ddirichlet` and `rdirichlet` are identically copied from MCMCpack
**getSeqKernel**

**Protein Sequence Kernels**

**Description**

Get mutual information and other kernels for protein sequences

**Usage**

getSeqKernel (sequences, kern.type=c("mm", "prop", "mi"), tau, call.C=TRUE, seq.start=NULL, seq.end=NULL)

**Arguments**

- **sequences**
  - String or list. If string, the name of a fasta file containing aligned sequences. If list, a list of strings, each string is a protein sequence. If list, call.C will be set to FALSE internally because C/C++ function needs sequence file name as input.

- **kern.type**
  - String. Type of kernel. mm: match-mismatch, prop: physicochemical properties, mi: mutual information.

- **tau**
  - Numeric. It is the same as rho^-2.

- **call.C**
  - Boolean. If TRUE, do a .C call. If FALSE, the implementation is in R. The .C call is 50 times faster.

- **seq.start**
  - Integer. Start position of subsequence to be used in computing kernel.

- **seq.end**
  - Integer. End position of subsequence to be used in computing kernel.

**Details**

call.C option is to allow comparison of R and C implementation. The two should give the same results and C implementation is 50 times faster.

when kern.type is mi and call.C is TRUE and when running on linux, this function will print messages like "read ...". This message is generated from U::openRead

**Examples**

```
fileName=paste(system.file(package="krm")[[1]], '/misc/SETpfamseed_aligned_for_testing.fasta', sep="")
K=getSeqKernel (fileName, kern.type="mi", tau=1, call.C=TRUE)
K
```
### hmmmargllik

**Functions related to profile HMM**

**Description**

Functions related to profile HMM

**Usage**

```r
hmmmargllik(dat, aaPrior, tau)
readPriorFromFile(priorFileName)
```

**Arguments**

- **dat**
  - a matrix representation of a multiple sequence alignment, each row is a sequence, each column is a position
- **aaPrior**
  - a list of two elements, "alpha" "mix.coef", representing mixture Dirichlet prior
- **tau**
  - numeric
- **priorFileName**
  - string

### krm package

**Kernel-based Regression Models**

**Description**

Implements tests for kernel-based regression model. The main function is krm.most(). Both Euclidean and protein sequence covariates can be used to form kernels.

### krm.most

**Kernel-based Regression Model Maximum of adjusted Score Test**

**Description**

Performs maximum of adjusted score test for kernel-based regression models. Both Euclidean and protein sequence covariates can be used to form kernels.

**Usage**

```r
krm.most (formula, data, regression.type=c("logistic","linear"),
  kern.type=c("rbf","mi","mm","prop"), n.rho=10, range.rho=0.99,
  n.mc=2000,
  seq.file.name=NULL, formula.kern=NULL, seq.start=NULL, seq.end=NULL,
  inference.method=c("parametric.bootstrap", "perturbation", "Davies"),
  verbose=FALSE)
```
**Arguments**

- **formula**: a formula object describing the null model
- **data**: data frame
- **regression.type**: logistic regression or linear regression
- **kern.type**: rbf: radial basis function kernel, a kernel type for Euclidean covariates. The other three kernels are for protein sequence covariates (Fong et al. 2014). mm: match-mismatch, prop: physicochemical properties, mi: mutual information.
- **n.rho**: integer. Number of rhos to maximize over
- **range.rho**: numeric. A number between 0 and 1. It controls the range of rhos to use to compute kernel
- **seq.file.name**: There are two ways to provide protein sequence information. One is to supply a sequence file named ‘seq.file.name’, which contains sequences in fasta format. Two is to supply a formula through formula.kern, and the variable name should be part of data.
- **formula.kern**: The formula for the covariates used to form the kernel. It may specify Euclidean covariates or a string covariate that contains protein sequences.
- **seq.start**: integer. Start position of subsequence to be used in computing kernel. Only supported when the sequence is specified through formula.kern.
- **seq.end**: integer. End position of subsequence to be used in computing kernel. Only supported when the sequence is specified through formula.kern.
- **n.mc**: integer. Number of bootstrap samples used to compute p-values.
- **inference.method**: parametric.bootstrap implements methods from Fong et al. (2014). perturbation uses methods from Wu et al. (2013) Davies uses the upper bound method from Davies (1987) and Liu et al. (2008).
- **verbose**: boolean

**Value**

A list of class krm

- **p.values**: If inference.method=="Davies", a single p-value. If inference.method=="perturbation" or "parametric.bootstrap", a vector of four p-values, named chiI, chiII, normI, normII. For perturbation, chiI and normII are NA. chiI/chiII p-values are based on chi-squared approximation and normI/normII are based on normal approximations. chiI/normI p-values are based on plugin estimator of mean and variance of score statistic, chiII/normII are based on modified estimator of mean and variance of score statistic. chiII or normII are more powerful than chiI and normI. For more details, see Fong et al. (2014)

**References**

Wu et al. (2013) Kernel Machine SNP-Set Testing Under Multiple Candidate Kernels, Genetic epidemiology.

Davies, R. (1987) Hypothesis testing when a nuisance parameter is present only under the alternative, Biometrika, 74, 33-43.


Examples

# in addition to the examples listed here, there are more examples  
# under folder R/library/krm/unitTests

## Not run:

# the examples are not run during package build because it takes a little too long to run

# an Euclidean kernel example from Liu et al. (2008)
data=sim.liu.2008 (n=100, a=.1, seed=1)
test = krm.most(y-x, data, formula.kern=-z.1+z.2+z.3+z.4+z.5, kern.type="rbf")

# a protein sequence kernel example
dat.file.name=paste(system.file(package="krm")[1],'/misc/y1.txt', sep="")
seq.file.name=paste(system.file(package="krm")[1],'/misc/sim1.fasta', sep="")
dat=read.table(dat.file.name); names(dat)="y"
test = krm.most (y=1, dat, seq.file.name=seq.file.name, kern.type="mi")

## End(Not run)

---

**krm.score.test**

*Adjusted Score Test*

**Description**

Adjusted score test for kernel-based regression models. This function is typically not used directly, but is called within krm.most().

**Usage**

krm.score.test(formula, data, K, regression.type=c("logistic","linear"), verbose=FALSE)
**readFastaFile**

**Arguments**

- **formula**: a formula object. Model under null.
- **data**: a data frame
- **K**: an n by n kernel/correlation matrix
- **regression.type**: a string
- **verbose**: Boolean

**Examples**

dat=sim.liu.2008(n=100, a=0, seed=1)
z=as.matrix(subset(dat, select=c(z.1,z.2,z.3,z.4,z.5)))
rho=1
K=kyotil::getK(z,kernel="rbf",par=rho^-2)
krm.score.test (y=x, dat, K, regression.type="logistic")

**readFastaFile**

Read a Fasta Sequence File

**Description**

Read a Fasta Sequence File

**Usage**

readFastaFile(fileName, sep = " ")
writeFastaFile (seqList, fileName)
aa2arabic (seq1)
string2arabic (seqList)
fastaFile2arabicFile (fastaFile, arabicFile, removeGapMajor=FALSE)
selexFile2arabicFile (selexFile, arabicFile, removeGapMajor=FALSE)
stringList2arabicFile (seqList, arabicFile, removeGapMajor=FALSE)
arabic2fastaFile (alignment, arabicFile)
readSelexFile (fileName)
readSelexAsMatrix (fileName)
arabic2fastaFile (alignment, fileName)
readArabicFile (fileName)
readBlockfile (fileName)
Arguments

- `fileName` string
- `fastaFile` string
- `arabicFile` string
- `selexFile` string
- `sep` string
- `seq1` string. A string of amino acids
- `seqList` list of string.
- `removeGapMajor` Boolean
- `alignment` matrix of arabic representation of sequences (1 based)

Value

`string2arabic` returns a matrix of arabic numbers representing aa. `readSelexFile` return a list of strings. `readArabicFile` return a matrix of n by p alignment.

Examples

```r
library(RUnit)
fileName=paste(system.file(package="krm"),'/misc/SETpfamseed_aligned_for_testing.fasta', sep="")
seqs = readFastaFile (fileName, sep = " ")
checkEquals(length(seqs),11)
```

Description


Usage

```r
sim.liu.2008(n, a, seed = NULL)
sim.liu.2007(n, a, seed = NULL)
```

Arguments

- `n` sample size
- `a` numeric. If a is 0, then the data is used to study size, otherwise power
- `seed` optional random number generator seed
Index

*Topic \textasciitilde kwd1
  krm.score.test, 8
*Topic \textasciitilde kwd2
  krm.score.test, 8
*Topic distribution
  krm package, 6

aa.prop.list, 2
aa2arabic (readFastaFile), 9
alignment2count (calcPairwiseIdentity), 2
alignment2transcount (calcPairwiseIdentity), 2
arabic2arabicFile (readFastaFile), 9
arabic2FastaFile (readFastaFile), 9

calcPairwiseIdentity, 2
chi.norm, 3
cloud9, 3
ddirichlet (dmdirichlet), 4
dmdirichlet, 4

fastaFile2arabicFile (readFastaFile), 9
getSeqKernel, 5
hmmMargLlik, 6

krm (krm package), 6
krm package, 6
krm.most, 6
krm.score.test, 8

logIntegrateDirichlet (dmdirichlet), 4
logIntegrateMixDirichlet (dmdirichlet), 4

modifyDirichlet (dmdirichlet), 4
rdirichlet (dmdirichlet), 4

readArabicFile (readFastaFile), 9
readBlockFile (readFastaFile), 9
readFastaFile, 9
readPriorFromFile (hmmMargLlik), 6
readSelexAsMatrix (readFastaFile), 9
readSelexFile (readFastaFile), 9
removeGap (calcPairwiseIdentity), 2
rmdirichlet (dmdirichlet), 4

selexFile2arabicFile (readFastaFile), 9
sim.liu.2007 (sim.liu.2008), 10
sim.liu.2008, 10
string2arabic (readFastaFile), 9
stringList2arabicFile (readFastaFile), 9

writeFastaFile (readFastaFile), 9