IPCAPS

April 3, 2018

`IPCAPS-package` *IPCAPS - Iterative Pruning to CApture Population Structure*

Description

Determining fine population structure using iterative pruning.

Details

Package:	IPCAPS
Type:	Package
Version:	0.43.0
Depends:	R (>= 3.0.0), Matrix, expm, fpc, Rmixmod, LPCM, apcluster, rARPACK
Date:	2017-07-21
License:	GPL 3

This package containts a function *ipcaps* for unsuppervised clustering.

Author(s)

Kridsadakorn Chaichoompu, Fentaw Abegaz Yazew, Sissades Tongsima, Philip James Shaw, Anavaj Sakuntabhai, Luisa Pereira, and Kristel Van Steen

Maintainer: Kridsadakorn Chaichoompu <*kridsadakorn.chaichoompu@ulg.ac.be*>

`cal.PC.linear` *A function for linear principal component analysis (PCA)*

Description

The function is able to handle 2 types of data; linear and SNP data set in additive coding (0, 1, and 2).

Usage

```
cal.PC.linear(X, PCscore=TRUE, no.pc=NA, data.type="linear", XXT=TRUE)
```

Arguments

X	A data matrix which rows represent samples and columns represent features.
PCscore	To specify whether scaled PCs will be calculated or not. If *FALSE*, eigenvectors are returned instead. Default = *TRUE*.
no.pc	A number of PCs to be calculated. If *no.pc* is set, PCs are patially calculated. Otherwise all PCs are obtained after calculation. Default = *NA*.
data.type	To specify a type of data matrix *X*. It can be set to "linear" and "snp". Default = "linear".
XXT	To specify how pricipal components (PCs) are calculated. If *TRUE*, PCs are calculated from *X.t(X)*, otherwise *X* is used directly. Default = *TRUE*.

Details

XXT is useful option especially an input matrix *X* contains many columns. Enabling this option, it helps to reduce computation complexity. Regardless the option *XXT* is enable or not, optained PCs are the same.

Value

The returned value is a list with components:

PC a PC matrix which rows represent samples and columns represent PCs.

evalue a vector of eigen values.

Examples

```
data(simSNP) #to load simulated data set

PCs <- cal.PC.linear(raw.data)
summary(PCs)
#       Length Class  Mode
#PC     577600 -none- numeric
#evalue    760 -none- numeric

print(PCs$PC[1:5,1:3])
#           [,1]       [,2]        [,3]
#[1,] -1.0326907  0.4161087 -1.43709887
#[2,] -0.5855313 -0.1809882 -0.53587760
#[3,]  1.2100050 -1.5454890  3.22493101
#[4,] -2.5581070  0.7142676  0.04434291
#[5,]  0.2211739 -1.8262511 -1.31319919

print(PCs$evalue[1:3])
#[1] 205508.24  37222.45  35198.47

plot.3views(PCs$PC[,1:3],label)

PCs <- cal.PC.linear(raw.data,PCscore = FALSE)
summary(PCs)
#       Length Class  Mode
#PC     577600 -none- numeric
#evalue    760 -none- numeric
```

```
print(PCs$PC[1:5,1:3])
#              [,1]          [,2]          [,3]
#[1,] -0.0022780102  0.0021567724 -0.0076599296
#[2,] -0.0012916223 -0.0009380973 -0.0028562995
#[3,]  0.0026691474 -0.0080105706  0.0171893146
#[4,] -0.0056429226  0.0037021882  0.0002363536
#[5,]  0.0004878869 -0.0094658149 -0.0069995278

print(PCs$evalue[1:3])
#[1] 205508.24  37222.45  35198.47

plot.3views(PCs$PC[,1:3],label)

PCs <- cal.PC.linear(raw.data,no.pc=3)
summary(PCs)
#       Length Class  Mode
#PC     2280   -none- numeric
#evalue    3   -none- numeric

print(PCs$PC[1:5,])
#           [,1]       [,2]        [,3]
#[1,]  1.0326907  0.4161087 -1.43709887
#[2,]  0.5855313 -0.1809882 -0.53587760
#[3,] -1.2100050 -1.5454890  3.22493101
#[4,]  2.5581070  0.7142676  0.04434291
#[5,] -0.2211739 -1.8262511 -1.31319919

print(PCs$evalue)
#[1] 205508.24  37222.45  35198.47

plot.3views(PCs$PC[,1:3],label)
```

`fst.each.snp.hudson` *A function for fixation index (Fst) calculation for all SNPs between two groups.*

Description

Fixation index (Fst) calculation was implemented using Hudson method [1,2].

Usage

```
fst.each.snp.hudson(raw.data, idx.p1, idx.p2)
```

Arguments

`raw.data`	A matrix contains the number 0, 1, and 2 representing SNPs in additive coding. Rows represent individuals and columns represent SNPs.
`idx.p1`	An integer vector contains the row indices of first population in *raw.data*.
`idx.p2`	An integer vector contains the row indices of second population in *raw.data*.

Value

The function returns a matrix of pairwise Fst values for all SNPs between 2 specified groups.

References

[1] Bhatia,G. et al. (2013) Estimating and interpreting Fst: The impact of rare variants. Genome Res., 23, 1514:1521.

[2] Hudson,R.R. et al. (1992) Estimation of levels of gene flow from DNA sequence data. Genetics, 132, 583:589.

Examples

```
data(simSNP) #to load simulated data set
idx1 <- which(label == 'pop1')
idx2 <- which(label == 'pop2')
fst.pairwise <- fst.each.snp.hudson(raw.data,idx1,idx2)

print(fst.pairwise[1:3])
#[1]  0.022842079  0.001219995  0.010243367
```

`fst.hudson` *A function for average fixation index (Fst) calculation between two groups.*

Description

Fixation index (Fst) calculation was implemented using Hudson method [1,2].

Usage

```
fst.hudson(raw.data, idx.p1, idx.p2)
```

Arguments

`raw.data`	A matrix contains the number 0, 1, and 2 representing SNP in additive coding. Rows represent individuals and columns represent SNP.
`idx.p1`	An integer vector contains the row indices of first population in *raw.data*.
`idx.p2`	An integer vector contains the row indices of second population in *raw.data*.

Value

The function returns an average Fst value between 2 specified groups.

References

[1] Bhatia,G. et al. (2013) Estimating and interpreting Fst: The impact of rare variants. Genome Res., 23, 1514:1521.

[2] Hudson,R.R. et al. (1992) Estimation of levels of gene flow from DNA sequence data. Genetics, 132, 583:589.

Examples

```
data(simSNP) #to load simulated data set
idx1 <- which(label == 'pop1')
idx2 <- which(label == 'pop2')
fst <- fst.hudson(raw.data,idx1,idx2)

print(fst)
#[1] 0.004843083
```

`get.node.info` *A function to obtain the information for specified node.*

Description

This function is used to obtain the information for specified node from the output object of ipcaps().

Usage

```
get.node.info(cluster.obj,node)
```

Arguments

`cluster.obj`	The object which is returned from the function *ipcaps*.
`node`	To specify the node number as shown in the HTML output files.

Value

The returned object is a *list* of node information including by *PCs*, *eigen.fit*, *index*, *label*. *PCs* represent the pricipal components of this node. *eigen.fit* is the EigenFit value of this node. *index* is a vector of row number (individuals) of *raw.data* (input data). *label* is the vector of labels of all individuals that belongs to this node.

Examples

```
# Importantly, bed file, bim file, and fam file are required
# Use the example files embedded in the package

BED.file <- system.file("extdata","simSNP.bed",package="IPCAPS")
LABEL.file <- system.file("extdata","simSNP_individuals.txt",package="IPCAPS")

my.cluster <- ipcaps(bed=BED.file,label.file=LABEL.file,lab.col=2,out=getwd())

#Here, to obtain the information of specified node, for example, node 3
node.info <- get.node.info(my.cluster,3)
ls(node.info)
#[1] "PCs"       "eigen.fit" "index"     "label"
```

ipcaps *A function for unsupervised clustering to capture population structure based on iterative pruning.*

Description

This version supports ordinal data which can be applied directly to SNP data to identify fine-scale population structure. It was built on the iterative pruning Principal Component Analysis (ipPCA) algorithm [1]. The IPCAPS involves an iterative process using multiple splits based on multivariate Gaussian mixture modeling of principal components and Clustering EM estimation [2]. In each iteration, rough clusters and outliers are also identified using our own method called rubikClust.

Usage

```
ipcaps(bed=NA, rdata=NA, files=NA, label.file=NA, lab.col=1, out, plot.as.pdf=FALSE,
method='mix', missing=NA, covariate=NA, cov.col.first=NA, cov.col.last=NA,
threshold=0.18, min.fst=0.0008, min.in.group=20, no.plot = FALSE)
```

Arguments

bed	PLINK binary format consists of 3 files; bed, bim, and fam. To generate these files from PLINK, use option *–make-bed*. See more details at: http://zzz.bwh.harvard.edu/plink/data.shtml
rdata	In case of re-analysis, it is convenient to run IPCAPS using the file rawdata.RData generated by IPCAPS. This file contains a matrix of SNPs (*raw.data*) and a vector of labels (*label*).
files	IPCAPS supports SNPs encoded as 0, 1 and 2 (dosage encoding). Rows represent SNPs and columns represent subjects. Each column needs to be separated by a space or a tab. A big text file should be divided into smaller files to load faster. For instance, to input 3 files, use as: *files=c('input1.txt', 'input2.txt', 'input3.txt')*.
label.file	An additional useful information (called "labels" in IPCAPS) related subject, for example, geographic location or disease phenotype. These labels (one at a time) are used in displaying the clustering outcome of IPCAPS. A label file must contain at least one column. However, it may contain more than one column in which case each column need to be separated by a space or a tab.
lab.col	The label in the label file to be used in the tree-like display of IPCAPS clustering results.
out	To set an absolute path for IPCAPS output. If the specified output directory already exists, result files are saved in sub-directories cluster_out, cluster_out1, cluster_out2, etc.
plot.as.pdf	To export plots as PDF. When omitted, plots are saved as PNG.
method	The internal clustering method. It can be set to "mix" (rubikClust & mixmod), "mixmod" [2], "clara" [3], "pam" [4], "meanshift" [5], "apcluster" [6], and "hclust" [7]. Default = "mix".
missing	Symbol used for missing genotypes. Default = *NA*.
covariate	A file of covariates; one covariate per column. SNPs can be adjusted for these covariates via regression modeling and residual computation.

`cov.col.first`	Refer to a covariate file, the first covariate to be considered as confounding variable.
`cov.col.last`	Refer to a covariate file, the last covariate to be considered as confounding variable. All the variables in between the cov.col.first and cov.col.last will be considered in the adjustment process.
`threshold`	Cutoff value for EigenFit. Possible values range from 0.03 to 0.18. The smaller, the potentially finer the substructure can be. Default = 0.18.
`min.fst`	Minimum Fst between a pair of subgroups. Default = 0.0008.
`min.in.group`	Minimum number of individuals to constitute a cluster or subgroup. Default = 20.
`no.plot`	No plot is generated if this option is TRUE. This option is useful when the system does not support X Windows in the unix based system. Default = FALSE.

Details

The computational time depends on the number of individuals. Consequentially, if large data sets are analyzed, it may be necessary first to split data into smaller files, and then load into computer memory (with parameter 'files'). Internally, the split files are merged to construct a com-prehensive matrix.

Value

The function returns the *list* object containing 2 internal objects; *output.dir* as class *character* and *cluster* as class *data.frame*. The object *output.dir* stores a result directory. The object *cluster* contains 4 columns, *group*, *node*, *label*, and *row.number*. The column *group* contains the assigned groups from *IPCAPS*. The column *node* contains node numbers in a tree as illustrated in the *HTML* result files. The column *label* contains the given labels that is set to the parameter *label*. The column *row.number* contains indices to an input data, which is matched to row numbers of input matrix. All clustering result files are saved in one directory (as specified by *out*) containing assigned groups, hierarchical trees of group membership, PC plots, and binary data for further analysis.

References

[1] Intarapanich A, Shaw PJ, Assawamakin A, Wangkumhang P, Ngamphiw C, Chaichoompu K, et al. Iterative pruning PCA improves resolution of highly structured populations. BMC Bioinformatics. 2009;10:382.

[2] Rmixmod: The R Package of the Model-Based Unsupervised, Supervised, and Semi-Supervised Classification Mixmod Library | Lebret | Journal of Statistical Software [Internet]. [cited 2016 Aug 30]. Available from: https://www.jstatsoft.org/article/view/v067i06

[3] R: Clustering Large Applications [Internet]. [cited 2017 Mar 7].

Available from: https://stat.ethz.ch/R-manual/R-devel/library/cluster/html/clara.html

[4] R: Partitioning Around Medoids (PAM) Object [Internet]. [cited 2017 Mar 7]. Available from: https://stat.ethz.ch/R-manual/R-devel/library/cluster/html/pam.object.html

[5] Wang MC and D. MeanShift: Clustering via the Mean Shift Algorithm [Internet]. 2016 [cited 2017 Mar 7]. Available from: https://cran.r-project.org/web/packages/MeanShift/index.html

[6] Bodenhofer U, Palme J, Melkonian C, Kothmeier A. apcluster: Affinity Propagation Clustering [Internet]. 2016 [cited 2017 Mar 7].

Available from: https://cran.r-project.org/web/packages/apcluster/index.html

[7] R: Hierarchical Clustering [Internet]. [cited 2017 Mar 7]. Available from: https://stat.ethz.ch/R-manual/R-devel/library/stats/html/hclust.html

Examples

```
# Use the BED format as input
# Importantly, bed file, bim file, and fam file are required
# Use the example files embedded in the package

BED.file <- system.file("extdata","simSNP.bed",package="IPCAPS")
LABEL.file <- system.file("extdata","simSNP_individuals.txt",package="IPCAPS")

my.cluster1 <- ipcaps(bed=BED.file,label.file=LABEL.file,lab.col=2,out=getwd())

table(my.cluster1$cluster$label,my.cluster1$cluster$group)
#             1   2   3   4   5   6
#  outlier4   5   4   1   0   0   0
#  pop1       0   0   0   0 250   0
#  pop2       0   0   0 250   0   0
#  pop3       0   0   0   0   0 250

# Use a text file as input
# Use the example files embedded in the package

text.file <- system.file("extdata","simSNP_data_numMark_rowVar_colInd.txt",
             package="IPCAPS")
LABEL.file <- system.file("extdata","simSNP_individuals.txt",package="IPCAPS")

my.cluster2 <- ipcaps(files=c(text.file),label.file=LABEL.file,lab.col=2,out=getwd())
table(my.cluster2$cluster$label,my.cluster2$cluster$group)
#             1   2   3   4   5   6
#  outlier4   5   4   1   0   0   0
#  pop1       0   0   0   0 250   0
#  pop2       0   0   0 250   0   0
#  pop3       0   0   0   0   0 250

# Use an R Data file as input
# Use the example file embedded in the package

rdata.file <- system.file("data","simSNP.RData",package="IPCAPS")

my.cluster3 <- ipcaps(rdata=rdata.file,out=getwd())
table(my.cluster3$cluster$label,my.cluster3$cluster$group)
#             1   2   3   4   5   6
#  outlier4   5   4   1   0   0   0
#  pop1       0   0   0   0 250   0
#  pop2       0   0   0 250   0   0
#  pop3       0   0   0   0   0 250
```

`plot.3views` *A function to create scatter plots in three views.*

Description

The function visualizes data in X-Y plane, X-Z plane, and Y-Z plane; an input object (*matrix* or *data.frame*) must contain at least 3 columns.

Usage

```
plot.3views(X, labels, col.pat.table=NA, only.row=NA)
```

Arguments

X	A matrix or a data.frame that contains at least 3 columns of numeric data. If there are more than 3 columns in *X*, only the first 3 columns will be used.
labels	A vector containing row labels of *X* for display. All vector elements should be of type "character" (*as.character*). The length of this vector equals the number of rows in *X*.
col.pat.table	A data.frame that associates colors and patterns to row labels (see *labels* before). It needs to contain 3 columns which represent unique labels (column 1) and associated patterns (column 2) and colors (column 3). Default = *NA*.
only.row	A vector that contains subset of row numbers that are selected to be plotted. Default = *NA*.

Details

col.pat.table needs to be set properly. For example, given the vector *labels* consists of 5 elements as in *c('pop1','pop2','pop2','pop2','pop1')*, the data.frame *col.pat.table* must contain 2 rows corresponding to 'pop1' and 'pop2', and 3 columns to describe their characteristics (i.e. patterns and colors). See the section "Examples" for more details.

Examples

```
data(simSNP) #to load simulated data set
plot.3views(PC[,1:3],label)

#to change colors and patterns
all.labels <- unique(label)
my.colors <- c('pink','yellow','cyan','green')
my.patterns<- c('o','x','&','#')
my.table <- cbind(all.labels,my.patterns,my.colors)
plot.3views(PC[,1:3],label,col.pat.table=my.table)
```

---

rubikClust — *A function for unsupervised clustering to detect rough structures and outliers.*

Description

The function operates on a Nx3 matrix, where N is the number of samples and data are collected on 3 variables.

Usage

```
rubikClust(X, min.space=0.4,rotation=TRUE)
```

Arguments

X	A data matrix for which rows represent samples and the 3 columns represent features. Missingness is not allowed.
min.space	A value to specify a minimum space between 2 consecutive projected values. Default = 0.4.
rotation	To specify if rotation is enabled or not. Default = *TRUE*.

Details

The function *rubikClust* is able to take up to 3 variables (N x 3 matrix). In case, a matrix contains more than 3 columns, only the first three columns are used; the other columns are ignored.

Value

The returned value is a vector of numbers representing cluster memberships.

Examples

```
data(simSNP) #to load simulated data set
groups <- rubikClust(PC[,1:3])
print(groups)

table(groups)
#groups
#  1   2   3
#  4 751   5
```

---

simSNP *Synthetic dataset containing single nucleotide polymorphisms (SNP) and population labels.*

Description

The *simSNP* is the simulated data set which consists of 10,000 independent SNPs and 760 individuals belonging to one of three populations (250 individuals each) and 10 outlying individuals. The pairwise genetic distance between populations is set to Fst=0.005 [1].

Usage

```
data("simSNP")
```

Details

The *simSNP* data set consists of:

raw.data A numeric matrix of 760 rows and 10,000 columns representing individuals and SNPs respectively. The matrix *raw.data* contains the number 0, 1, and 2 representing SNP in additive coding.

label A character vector of 760 elements containing labels or populations of 760 individuals which they belong. Three populations and outliers were labeled as "pop1", "pop2", "pop3", and "outlier4".

PC A numeric matrix of 760 rows and 10 columns containing the first ten principal components.

References

[1] Balding DJ, Nichols RA. A method for quantifying differentiation between populations at multi-allelic loci and its implications for investigating identity and paternity. Genetica. 1995 Jun;96(1-2):3-12.

Examples

```
data(simSNP)
dim(raw.data)
#[1]   760 10000

raw.data[1:5,1:5]
#     [,1] [,2] [,3] [,4] [,5]
#[1,]    1    2    0    1    0
#[2,]    1    1    0    0    2
#[3,]    0    1    0    1    1
#[4,]    1    2    0    1    1
#[5,]    1    1    0    0    1

length(label)
#[1] 760

unique(label)
#[1] 760  10

dim(PC)
#[1] 760  10
```

---

top.discriminator *A function to detect top discriminators between two groups.*

Description

This function detects top discriminators that contribute to group separation based on the fixation index (Fst).

Usage

```
top.discriminator(cluster.obj,group1,group2,bim.file,use.node.number=FALSE,num.top=100)
```

Arguments

`cluster.obj`	The object which is returned from the function *ipcaps*.
`group1`	To specify the first group number to be compared. (also see *use.node.number*)
`group2`	To specify the second group number to be compared. (also see *use.node.number*)
`bim.file`	Option: In case that SNP information is not provided to the *ipcaps* function, an absolute path of SNP information file is required. It needs to be in PLINK format (*.bim*). See more details at: http://zzz.bwh.harvard.edu/plink/data.shtml

use.node.number
: To specify whether a group number or a node number will be used. If *TRUE*, a node nubmer is used instead. Default = *FALSE*.

num.top
: A number of top Fst SNPs to be returned. Default = 100.

Value

The returned value is a *data.frame* of SNP information sorting by Fst in descending order, which contains 7 columns, *chr*, *SNP*, *centimorgans*, *position*, *allele1*, *allele2*, and *Fst*. The column 1-6 are SNP information from the *bim* file. The column *Fst* contains estimated Fst between *group1* and *group2*.

Examples

```
# Importantly, bed file, bim file, and fam file are required
# Use the example files embedded in the package

BED.file <- system.file("extdata","simSNP.bed",package="IPCAPS")
LABEL.file <- system.file("extdata","simSNP_individuals.txt",package="IPCAPS")

my.cluster <- ipcaps(bed=BED.file,label.file=LABEL.file,lab.col=2,out=getwd())
table(my.cluster$cluster$label,my.cluster$cluster$group)
#             1   2   3   4   5   6
#  outlier4   5   4   1   0   0   0
#  pop1       0   0   0   0 250   0
#  pop2       0   0   0   0   0 250
#  pop3       0   0   0 250   0   0

#Identify top discriminators between groups, for example, group 4 and group 5
top.snp <-top.discriminator(my.cluster,4,5)
#or, specify the bim file
#top.snp <-top.discriminator(my.cluster,4,5,bim.file="simSNP.bim")
head(top.snp)
#      chr        SNP centimorgans position allele1 allele2        Fst
#V5452   1 marker5452            0 54520000       A       T 0.11337260
#V2348   1 marker2348            0 23480000       A       T 0.11194490
#V8244   1 marker8244            0 82440000       A       T 0.09556580
#V5972   1 marker5972            0 59720000       A       T 0.08747794
#V3561   1 marker3561            0 35610000       A       T 0.08725860
#V8419   1 marker8419            0 84190000       A       T 0.08293494

#Alternatively, it is possible to refer to node numbers instead of group numbers
table(my.cluster$cluster$label,my.cluster$cluster$node)
#             2   4   6   7   8   9
#  outlier4   5   4   1   0   0   0
#  pop1       0   0   0   0 250   0
#  pop2       0   0   0   0   0 250
#  pop3       0   0   0 250   0   0

#Identify top discriminators between groups, for example, node 7 and node 8
top.snp2 <-top.discriminator(my.cluster,7,8,use.node.number=TRUE)
head(top.snp2)
#      chr        SNP centimorgans position allele1 allele2        Fst
#V5452   1 marker5452            0 54520000       A       T 0.11337260
#V2348   1 marker2348            0 23480000       A       T 0.11194490
```

```
#V8244   1 marker8244          0 82440000       A       T 0.09556580
#V5972   1 marker5972          0 59720000       A       T 0.08747794
#V3561   1 marker3561          0 35610000       A       T 0.08725860
#V8419   1 marker8419          0 84190000       A       T 0.08293494
```

Index