Package ‘GGtools’

March 28, 2017

Title software and data for analyses in genetics of gene expression

Version 5.10.1

Author VJ Carey <stvjc@channing.harvard.edu>

Description software and data for analyses in genetics of gene expression 
and/or DNA methylation

Suggests GGdata, illuminaHumanv1.db, SNPlocs.Hsapiens.dbSNP144.GRCh37, 
multtest, aod, rmeta

Depends R (>= 2.14), GGBase (>= 3.19.7), data.table, parallel, 
Homo.sapiens

Imports methods, utils, stats, BiocGenerics, snpStats, ff, Rsamtools, 
AnnotationDbi, Biobase, bit, VariantAnnotation, hexbin, 
tracklayer, Gviz, stats4, S4Vectors (>= 0.9.25), IRanges, 
GenomeInfoDb, GenomicRanges, iterators, Biostrings, ROC, 
biglm, ggplot2, reshape2

Enhances MatrixEQTL, foreach, doParallel, gwascat

Maintainer VJ Carey <stvjc@channing.harvard.edu>

License Artistic-2.0

biocViews Genetics, GeneExpression, GeneticVariability, SNP

LazyLoad yes

gwSnpTests.R snpsCisToGenes.R relocate.R topSnps.R 

NeedsCompilation no

R topics documented:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GGtools-package</td>
<td>2</td>
</tr>
<tr>
<td>All.cis</td>
<td>4</td>
</tr>
<tr>
<td>appraise</td>
<td>5</td>
</tr>
<tr>
<td>b1</td>
<td>7</td>
</tr>
</tbody>
</table>
Details

Package: GGtools
Version: 4.2.26
Suggests: GGdata, illuminaHumanv1.db
Depends: R (>= 2.14), GGBase (>= 3.16.1)
Imports: methods, snpStats, ff, IRanges, GenomicRanges, AnnotationDbi, Biobase, Rsamtools, bit, VariantAnnotation
License: Artistic-2.0
LazyLoad: yes
Packaged: 2012-01-18 03:39:51 UTC; stvjc
Built: R 2.15.0; ; 2012-02-06 17:22:52 UTC; unix

Index:

best.cis.eQTLs collect gene-wise best scoring eQTL
eqtlTests compute association statistics between all probes and SNP in an smSet instance
eqtlTestsManager-class Class "eqtlTestsManager"
ex ExpressionSet instance for illustrating integrative smSet container
cisMap create, using Bioconductor annotation resources, a structure that enumerates SNP in the vicinity of ('cis' to) genes
gwSnpTests execute a series of tests for association between genotype and expression
cisMap serialization of a table from Stranger's multipopulation eQTL report
hg19.si.df data frame representation of seqinfo for Homo sapiens at hg19 build

The package depends on GGBase, which includes additional infrastructure for integrative data structures and data filtering.

Author(s)

VJ Carey <stvjc@channing.harvard.edu>

Maintainer: VJ Carey <stvjc@channing.harvard.edu>

See Also

gtSS for acquiring containers for integrative data on genetics of expression.

Examples

## Not run:
# acquire chromosome 20 genotypes and all expression data for
# 90 CEU samples as published at Wellcome Trust GENEVAR and
# HapMap phase II
# c20 = getSS("GGtools", "20")
# perform a focused eQTL search
functions that compute score tests for all SNP cis to genes, with flexible filtering

Description

function that computes score tests for all SNP cis to genes, with flexible filtering

Usage

cisScores(config = new("CisConfig"), ...)  
All.cis(config = new("CisConfig"), ...)  
addgwhit(ans, traitFilter=force, vname="isgwashit")  
add878(ans)  
inflammFilter(gwtagger)

Arguments

config instance of class CisConfig-class  
... passed to eqtlTests  
ans cisRun-like entity to which additional annotation will be bound by addgwhit or add878  
gwtagger GRanges like gwastagger in gwascat data elements  
traitFilter function that returns a gwastagger-like GRanges, see inflammFilter  
vname name to be used for new data.table column added by addgwhit

Details

cisScores (All.cis) returns score statistics for associations of all SNP cis to genes, in a GRanges instance, with range names given by probes; metadata supplied SNP location, name, and score  
cisAssoc targets SummarizedExperiment instances for molecular phenotype measures and VCF for variant data  
addgwhit and add878 will use GWAS hit information or ChromHMM labeling to annotation ranges

Value

for cisScores: instance of cisRun-class  
for cisAssoc: a GRanges with information on observed and permuted test scores per locus/feature pair

Author(s)

VJ Carey <stvjc@channing.harvard.edu>
Examples

## Not run:

```r
c = new("CisConfig")
chrnames(cc) = "21"
genoem(cc) = "hg19"
lkp = try(library(parallel))
if (!inherits(lkp, "try-error")) {
    nc = min(10, detectCores())
options(mc.cores=nc)
geneApply(cc) = mclapply
}
estimates(cc) = FALSE
set.seed(1234)
unix.time(f1 <- cisScores( cc ))
#
# demonstrate adding annotation on chromatin state and gwas status
#
eprops = function(ans) {
    #
    # only adds fields to values() of the input
    #
data(hmm878)
ac = as.character
eqr = GRanges(ac(seqnames(ans)), IRanges(ans$snplocs, width=1))
fo = findOverlaps(eqr, hmm878)
chromcat878 = factor(rep("none", length(ans)), levels=c(unique(hmm878$name), "none"))
chromcat878[ queryHits(fo) ] = factor(hmm878$name[subjectHits(fo)])
ans$chromcat878 = chromcat878
if (require(gwascat)) {
    data(gwastagger)
isgwashit = i*(overlapsAny(eqr, gwastagger) | ans$snp
    ans$isgwashit = isgwashit
    }
    ans
} 
extraProps(cc) = eprops
set.seed(1234)
unix.time(f2 <- cisScores( cc ))
#
#
inflammFilter # to make more restrictive predicate for prediction

## End(Not run)
```

---

**appraise**  

**appraisal for eQTL prediction models**

---

**Description**

appraisal for eQTL prediction models
Usage

appraise(dtab,
  discretize = TRUE,
  reduceToSNP = TRUE,
  prefix = NULL,
  folder = paste0(prefix, "_APPROUT"),
  discfmlas_in = GGtools:.discfmlas.demo,
  txlist = list(
    distcats = function(x) {
      cut(x$mindist, c(-1, seq(0, 200001, 50000)))
    },
    fdrcats = function(x) {
      fdrfac = cut(x$fdr, c(-.01, .05, .1, .25, .5, 1.01))
      relevel(fdrfac, "(0.5,1.01]")
    },
    mafcats = function(x) {
      maffac = cut(x$MAF, c(-.01, .05, .1, .25, .51))
      relevel(maffac, "(-0.01,0.05]")
    },
    caddcats = function(x){
      cut(x$PHRED, c(-.01, 5, seq(10, 30, 10 ), 60))
    },
    cutts = c(-0.01,seq(0.015,.12,.015),.15),
    names2check = GGtools:.standardNames, maxit=30,
    savePinfer=FALSE)
# bindgwava( gwavadt, eqdt )

Arguments

dtab data.table instance as created by transforming cisRun to GRanges and then
to data.table, and then adding CADD PHRED scores if available. If CADD
PHRED scores are not available, the default formulas should not be used.

discretize logical telling whether binning to factors defined in txlist should be performed
reduceToSNP logical telling whether ranges should be reduced to unique SNP and FDR re-
computed
prefix character atom used to prefix objects saved and folder for result objects
folder folder name suffix
discfmlas_in named list of model formulae
txlist named list of functions that are used to bin certain quantitative features of SNP
cutts numeric vector of thresholds for tabulation and discrete calibration
names2check if NULL, ignored; if a character vector, function will fail unless all(names2check %in% names(dtab)
maxit numeric passed to bigglm as control parameter for maximum number of itera-
tions to use in modeling gwas hit probabilities
savePinfer logical specifying whether the inferred probabilities of GWAS involvement are
retained

Details

The appraise function wraps many tasks used to appraise eQTL collections in terms of predictive
capacity. Details will be provided.
Value

A folder is opened and objects are written representing the test set (data.table on SNPs on even chromosomes), the coefficients of predictive models built on training set (SNPs on odd chromosomes), coefficients of linear regressions of binary test outcomes for calibrating the model on test data, and ROC AUC measures.

bindgwava uses simple data.table operations with match to add three columns to eqdt, gwava_tss, gwava_unmat, and gwava_regi

Author(s)

VJ Carey <stvjc@channing.harvard.edu>

---

**b1**

mcwBestCis instances, integrative analysis output containers generated by GGtools vignette

Description

integrative analysis output containers generated by GGtools vignette

Usage

data(b1)

Format

The format is:

Formal class 'mcwBestCis' [package "GGtools"] with 9 slots

..@ scoregr :Formal class 'GRanges' [package "GenomicRanges"] with 6 slots

.. ..@ seqnames :Formal class 'Rle' [package "IRanges"] with 4 slots

.. .. ..@ values : Factor w/ 1 level "20": 1

.. .. ..@ lengths : int 50

.. .. ..@ elementMetadata: NULL

.. .. ..@ ranges :Formal class 'IRanges' [package "IRanges"] with 6 slots

.. .. .. ..@ start : int [1:50] 24280834 61665697 352356 61679079 45286150 55187941 38766161 10871477 56570242 13304639 ...

.. .. .. ..@ width : int [1:50] 2090785 2005619 2021461 2001901 2129211 2007692 2038197 2035767 2012068 2013675 ...

.. .. .. ..@ NAMES : chr [1:50] "GI_34147330-S" "hmm26961-S" "GI_17149835-I" "GI_31077201-S" ...

.. .. .. ..@ elementType : chr "integer"

.. .. .. ..@ elementMetadata: NULL

.. .. .. ..@ metadata : list()

.. .. ..@ @ ranges :Formal class 'IRanges' [package "IRanges"] with 6 slots

.. .. .. ..@ start : int [1:50] 24280834 61665697 352356 61679079 45286150 55187941 38766161 10871477 56570242 13304639 ...

.. .. .. ..@ width : int [1:50] 2090785 2005619 2021461 2001901 2129211 2007692 2038197 2035767 2012068 2013675 ...

.. .. .. ..@ NAMES : chr [1:50] "GI_34147330-S" "hmm26961-S" "GI_17149835-I" "GI_31077201-S" ...

.. .. .. ..@ elementType : chr "integer"

.. .. .. ..@ elementMetadata: NULL

.. .. .. ..@ metadata : list()

.. .. ..@ strand :Formal class 'Rle' [package "IRanges"] with 4 slots

.. .. .. ..@ values : Factor w/ 3 levels "+", "+", "-": 3

.. .. .. ..@ lengths : int 50

.. .. .. ..@ elementMetadata: NULL

.. .. .. ..@ metadata : list()

.. .. ..@ elementMetadata:Formal class 'DataFrame' [package "IRanges"] with 6 slots
best.cis.eQTLs

... ... ...@ rownames : NULL
... ... ...@ nrows : int 50
... ... ...@ listData :List of 6
... ... ...$ score : num [1:50] 36.5 16.8 16.7 14.8 9.8 ...
... ... ...$ snpid : chr [1:50] "rs6037097" "rs3810504" "rs13043344" "rs13044229" ...
... ... ...$ snploc : int [1:50] 25347221 62678549 544417 61738288 46369894 56551001 38879237 12177765 57565943 15056960 ...
... ... ...$ radiusUsed: num [1:50] 1e+06 1e+06 1e+06 1e+06 1e+06 1e+06 1e+06 1e+06 1e+06 1e+06 ...
... ... ...$ nsnp : int [1:50] 1387 783 2382 766 2135 2238 1658 2459 1675 1868 ...
... ... ...$ fdr : num [1:50] 0 0 0 0.25 0.717 ...
... ... ...@ elementType : chr "ANY"
... ... ...@ elementMetadata: NULL
... ... ...@ seqinfo : [package "GenomicRanges"] with 4 slots
... ... ...@ seqnames : chr "20"
... ... ...@ seqlengths : int NA
... ... ...@ is_circular: logi NA
... ... ...@ genome : chr NA
... ... ...@ metadata : list()
...@ allperm : num [1:100] 15.8 13.9 13.8 13.2 13 ...
...@ extra : NULL
...@ chromUsed: chr "20"
...@ theCall : language best.cis.eQTLs(smpack = "GGdata", rhs = ~male, folderstem = "db2", radius = 1e+06, chrnames = "20", geneApply = mclapply, snpannopk = snplocsDefault(), ...
...@ smFilter :function (x)
... ... ... attr(, "srcref")=Class 'srcref' atomic [1:8] 413 8 413 50 8 50 4 4
... ... ...- attr(, "srcfile")=Classes 'srcfilecopy', 'srcfile' <environment: 0xf5c0cc0>
...@ nperm : num 2
...@ globalMap<environment: 0xf5c0538>
...@ testCount: int 83969

data(b1)
b1

best.cis.eQTLs  collect genewise best scoring eQTL

Details
As created in GGtools.Rnw vignette code, with sharply curtailed searches

Examples
data(b1)
b1

Description
collect genewise best scoring eQTL
**Usage**

```
best.cis.eQTLs(smpack = "GGdata", rhs = ~1,
       folderstem = "cisScratch", radius = 50000,
       shortfac = 100,
       chrnames = as.character(1:22),
       smchrpref = "", gchrpref = "", schrpref = "ch",
       geneApply = lapply, geneannopk = "illuminaHumanv1.db",
       snpannopk = snplocsDefault(),
       smFilter = function(x) nsFilter(MAFfilter(x, lower = 0.05), var.cutoff = 0.97),
       useME=FALSE, excludeRadius=NULL, exFilter=function(x)x,
       keepMapCache=FALSE, getDFFITS=FALSE, SSgen = GGBase::getSS)
```

```
All.cis.eQTLs(maxfdr = 0.05, inbestcis = NULL, smpack = "GGdata",
       rhs = ~1, folderstem = "cisScratch", radius = 50000,
       shortfac = 100,
       chrnames = as.character(1:22),
       smchrpref = "", gchrpref = "", schrpref = "ch",
       geneApply = lapply, geneannopk = "illuminaHumanv1.db",
       snpannopk = snplocsDefault(),
       smFilter4cis = function(x) nsFilter(MAFfilter(clipPCs(x, 1:10), lower = 0.05), var.cutoff = 0.85),
       smFilter4all = function(x) MAFfilter(clipPCs(x, 1:10), lower = 0.05),
       nperm = 2, excludeRadius=NULL, exFilter=function(x)x,
       SSgen = GGBase::getSS)
```

```
meta.best.cis.eQTLs(smpackvec = c("GGdata", "hmyriB36"), rhslist = list(~1, ~1),
       folderstem = "cisScratch", radius = 50000, shortfac = 100,
       chrnames = as.character(1:22), smchrpref = "", gchrpref = "", schrpref = "ch",
       geneApply = lapply, geneannopk = "illuminaHumanv1.db",
       snpannopk = snplocsDefault(),
       SMFilterList = list(
       function(x) nsFilter(MAFfilter(x, lower = 0.05), var.cutoff = 0.97),
       function(x) nsFilter(MAFfilter(x, lower = 0.05), var.cutoff = 0.97)),
       exFilterList = list(function(x)x, function(x)x),
       nperm = 2, excludeRadius=NULL)
```

```
meta.All.cis.eQTLs(minchisq, smpackvec = c("GGdata", "hmyriB36"),
       rhslist = list(~1, ~1), folderstem = "cisScratch",
       radius = 50000, shortfac=100, chrnames = as.character(1:22), smchrpref = "",
       gchrpref = "", schrpref = "ch", geneApply = lapply,
       geneannopk = "illuminaHumanv1.db",
       snpannopk = snplocsDefault(),
       SMFilterList = list(function(x) nsFilter(MAFfilter(x, lower = 0.05), var.cutoff = 0.97),
       function(x) nsFilter(MAFfilter(x, lower = 0.05), var.cutoff = 0.97)),
       exFilterList = list(function(x)x, function(x)x),
       nperm = 2)
```

```
chromsUsed(x)
```
fdr(x)
fullreport(x, type, ...)
getAll(x)
getBest(x)
getCall(x)

Arguments

smpack character string naming a package to which getSS can be applied to extract smlSet-class instances
smpackvec vector of character strings naming packages that can be used as smpack values in a series of best.cis.eQTLs calls, one per population for meta-analysis
rhs R model formula, with no dependent variable, that will be used with snp.rhs.tests to adjust GWAS tests for each expression probe
rhslist a list of model formulae to be used as rhs in a series of best.cis.eQTLs calls, one per population for meta-analysis
folderstem prefix of the folder name to be used to hold ff archives of test results
radius coding extent of each gene will be extended in both directions by radius bases, and only SNP within these limits are used for selecting best hits for the gene
shortfac a numeric that will scale up the chi-squared statistic before it is converted to short integer for storage in ff array
chrnames character vector of chromosome identifiers, to be manipulated for certain query resolutions by the following parameters
smchrpref prefix to convert chrnames into appropriate tokens for indexing smlSet elements as collected from the package named by parameter smpack
gchrpref prefix to convert chrnames into appropriate tokens for obtaining gene metadata; in future this may need to be a string transformation function
schrpref prefix to convert chrnames into appropriate tokens for use with snplocs for the SNP location information package identified in snpannopack parameter below
geneApply an lapply like function, defaults to lapply
geneannopk character string, name of a *.db annotation package that annotates probe identifiers; or see getCisMap for additional possibilities concerning FDb.* complex token values for newer annotation formats
snpannopk character string, name of SNPlocs.Hsapiens.dbSNP.* package for obtaining; global function snplocsDefault() can be used to get a nominally current package name
smFilter function accepting and returning an smlSet-class instance
SMFilterList list of functions, one element per smlSet package used in meta analysis, accepting and returning an smlSet-class instance
minchisq threshold on test statistic value that must be met to include records on SNPs in the All.cis.eQTLs report
nperm number of permutations to be used for plug-in FDR computation
best.cis.eQTLs

useME logical; if TRUE, use the rudimentary interface to the MatrixEQTL package from A. Shabalin on CRAN

maxfdr Used in All.cis.eQTLs. The process of identifying “best” cis eQTL per probe leads to a probe-specific FDR. In All.cis.eQTLs we enumerate all probes and all SNP with FDR at most maxfdr, not just the best scoring SNP per probe.

inbestcis Used in All.cis.eQTLs. An instance of mcwBestCis that can be used to speed up the extraction of All.cis eQTL.

smFilter4cis Used in All.cis.eQTLs. A function accepting and returning an smlSet instance. When inbestcis parameter is NULL, this filter will be used for identifying the best SNP per probe.

smFilter4all Used in All.cis.eQTLs. A function accepting and returning an smlSet instance. This filter will be used for identifying the best SNP per probe. This filter should not affect the number of probes.

x instance of mcwBestCis

type character, either ‘data.frame’ or ‘GRanges’

excludeRadius numeric, defaulting to NULL; if non-null, defines radius around gene region that is excluded for cis SNP scoring; must be less than radius

keepMapCache logical, if TRUE, returned mcwBestCis object will include an environment loaded with chromosome-specific lists of maps from genes to cis SNP names; if FALSE, the mapCache environment returned will be empty – NB, this feature has been found to add too much volume to returned objects and is suspended...

exFilter this function is passed to getSS; see Details

exFilterList for metaanalytic applications, a list of functions in correspondence with the elements of smpackvec to be passed to getSS; see Details

getDFFITS logical; a component storing max DFFITS value for each gene will be retained if this argument TRUE

... not used

SSgen function to be used to create smlSet instance for testing – in general, GGBase::getSS has been used to pull the ExpressionSet and SnpMatrix data from a named package, but in some cases a specialize task is needed to create the desired smlSet. Whatever is passed to SSgen must return an smlSet instance.

Details
geneApply can be set to parallel::mclapply, for example, in a multicore context.
mcwBestCis stands for ‘multi-chromosome-wide best cis’ eQTL report container.

It is possible that the filtering processes should be broken into genotype filtering and expression probe filtering.

fdr(x) will return a numeric vector of plug-in FDR estimates corresponding to probe:association tests as ordered in the fullreport of a *Cis container. More metadata should be attached to the output of this function.

exFilter may seem redundant with smFilter, but its existence allows simpler management of multitissue expression archives (which may have several records per individual) with germ line genotype data (which will have only one record per individual). In this setting, use exFilter to select records for the tissue of interest; this will occur early in the smlSet generation process.

Value

an instance of mcwBestCis
best.trans.eQTLs

collect strongest trans SNP-gene associations in a buffer of size K genes per SNP

Description

collect strongest trans SNP-gene associations in a buffer of size K genes per SNP

Usage

best.trans.eQTLs(smpack, rhs, genechrnum, snpchrnum, K = 20, 
targdirpref = "tsco", batchsize = 200, radius = 2e+06, genequeryprefix = "", 
snploadprefix = "chr", snplocprefix = "chr", geneannopk, snpannopk, 
exFilter = function(x) x, smFilter = function(x) x, 
geneApply = lapply, SSgen = GGBase::getSS)

Arguments

smpack character string naming a package from which smlSet-class instances can be generated using getSS
rhs passed to snp.rhs.tests for covariate or stratification adjustments; for permutation analysis, covariates should be handled via regressOut
genechrnum character vector of chromosome identifiers for genes, typically as.character(1:22) for somatic genes in human studies
snpchrnum specific chromosome identifier for all SNP to be analyzed
K the size of the buffer: scores will be recorded for the most strongly associated K genes for each SNP
targdirpref character string where buffer data will be held in ff archives
batchsize passed to ffrowapply as scores are filtered from comprehensive testing to fill the buffer
radius numeric: for same-chromosome tests, tests will not be performed for SNP-gene combinations with base-pair proximity smaller than radius
genequeryprefix string: used when the numeric chromosome identifier requires a prefix like ‘chr’ for annotation query resolution on gene location
snploadprefix string: used when the package identified in smpack requires a prefix to the snpchrnum token for getSS retrieval of smlSet instance
bindmaf

bind testing metadata to a best.cis.eQTLs result

**Description**

bind testing metadata to a best.cis.eQTLs result

<table>
<thead>
<tr>
<th><strong>snplocprefix</strong></th>
<th>string: used when the numeric chromosome identifier requires a prefix like <code>chr</code> for annotation query resolution on SNP location</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>geneannopk</strong></td>
<td>package to be used for CHRLOC and CHRLOCEND queries for genes</td>
</tr>
<tr>
<td><strong>snpannopk</strong></td>
<td>package to be used to resolve snplocs calls</td>
</tr>
<tr>
<td><strong>exFilter</strong></td>
<td>function returning an smlSet instance, operating on expression component prior to smFilter application and eQTL testing</td>
</tr>
<tr>
<td><strong>smFilter</strong></td>
<td>function returning an smlSet instance, operating on the full smlSet</td>
</tr>
<tr>
<td><strong>geneApply</strong></td>
<td>lapply-like function, typically mclapply or the like</td>
</tr>
<tr>
<td><strong>SSgen</strong></td>
<td>function to be used to create smlSet instance for testing – in general, GGBase::getSS has been used to pull the ExpressionSet and SnpMatrix data from a named package, but in some cases a specialize task is needed to create the desired smlSet. Whatever is passed to SSgen must return an smlSet instance.</td>
</tr>
</tbody>
</table>

**Value**

instance of transManager-class

**Author(s)**

VJ Carey <stvjc@channing.harvard.edu>

**Examples**

```r
## Not run:
if (.Platform$OS.type != "windows") { # ff overwrites failing 5.IX.12
  nsFilter2 = function(sms, var.cutoff=.5) {
    alliq = apply(exprs(sms),1,IQR)
    qs = quantile(alliq, var.cutoff, na.rm=TRUE)
    sms[ which(alliq > qs), ]
  }

  thefilt = function(x) GTFfilter( nsFilter2 (clipPCs(x, 1:10), var.cutoff=.95 ), lower=.05 )
  tfile = tempfile()
  tfold = dir.create(tfile)
  t1 = best.trans.eQTLs("GGdata", ~1, as.character(20:22), "22",
    geneannopk="illuminaHumanv1.db", snpannopk= snplocsDefault(),
    smFilter=thefilt, snploadprefix="", snplocprefix="ch", targdirpref=tfile)
  tt1 = transTab(t1)
  ttlo = tt1[ order(tt1[,"sumchisq"], decreasing=TRUE), ][1:10,]
}
```
Usage

```r
meta.bindmaf(smpackvec=c("GGdata", "hmyriB36"),
              smchr="20", obj, usemaxMAF=FALSE, SSgen=GGBase::getSS)
```

Arguments

- `smpackvec`: a vector of candidate package names (potential smpack arguments to getSS for metaanalysis across populations or tissues
- `smchr`: the chromosome name as used in the names of the smList output for the getSS result
- `obj`: an instance of mcwBestCis-class generated using the package named in smpack
- `usemaxMAF`: if TRUE, label a SNP with maximum MAF observed across populations, otherwise compute the MAF for the combined genotypes across populations represented by the various smlSet instances generated with the smpackvec spec.
- `SSgen`: function to be used to create smlSet instance for testing – in general, GGBase::getSS has been used to pull the ExpressionSet and SnpMatrix data from a named package, but in some cases a specialize task is needed to create the desired smlSet. Whatever is passed to SSgen must return an smlSet instance.

Details

computes the MAF of most highly associated SNP per gene, and distance between that SNP and the transcription limits of the gene, assigning 0 for this if the SNP lies within the transcription limits

Value

a GRanges instance

Note

This will be used to stratify the permuted scores.

Examples

```r
## Not run:
b1 = best.cis.eQTLs(chr="20")  # sharply filtered
b1b = bindmaf(obj=b1)
## End(Not run)
```

cgff2dt

<table>
<thead>
<tr>
<th>cgff2dt</th>
</tr>
</thead>
<tbody>
<tr>
<td>translate the GFF3 from a ciseqByCluster/processgff output into a serialized data.table instance, compute genome-wide plug-in FDR, and update the GFF3 with this FDR</td>
</tr>
</tbody>
</table>

Description

translate the GFF3 from a ciseqByCluster/processgff output into a serialized data.table instance, compute genome-wide plug-in FDR, and update the GFF3 with this FDR
Usage

cgff2dt(gff3, tiling, addHitTest = TRUE, addcc878 = TRUE)

Arguments

gff3 character string naming a tabix-indexed, bgzipped output of `gffprocess`
tiling output of `simpleTiling`
addHitTest logical, telling whether to add a column on coincidence of SNP with the `gwastagger`
ranges
addcc878 logical, telling whether to add a column on coincidence of SNP with the `hmm878`
ranges, using the inferred chromatin state as factor level

Note

assumes unix utilities `zcat`, `paste` and `bgzip` are available

Examples

```r
##---- Should be DIRECTLY executable !! ----
##-- ==> Define data, use random,
##-- or do help(data=index) for the standard data sets.
## The function is currently defined as
function (gff3, tiling)
{
  require(foreach)
  require(Rsamtools)
  stopifnot(is(tiling, "GRanges"))
  basen = gsub(".gff3.gz", ",", gff3)
  th = headerTabix(gff3)
  orderedChr = th$seqnames
  lgr = foreach(i = 1:length(tiling)) %dopar% {
    gc()
    cat(i)
    lk = try(import.gff3(gff3, which = tiling[i]))
    if (inherits(lk, "try-error"))
      lk = NULL
    if (!is.null(lk))
      lk = as.data.table(as.data.frame(lk))
      lk
  }
  bad = sapply(lgr, is.null)
  if (any(bad))
    lgr = lgr[-which(bad)]
  ans = do.call(rbind, lgr)
  ans$snplocs = as.numeric(ans$snplocs)
  ans$ests = as.numeric(ans$ests)
  ans$se = as.numeric(ans$se)
  ans$fdr = as.numeric(ans$fdr)
  ans$MAF = as.numeric(ans$MAF)
  ans$dist.mid = as.numeric(ans$dist.mid)
  nperm = length(grep("permS", names(ans)))
  pnames = paste("permScore_", 1:nperm, sep = "\n")
  for (i in 1:nperm) ans[[pnames[i]]] = as.numeric(ans[[pnames[i]]])
  ans$mindist = as.numeric(ans$mindist)
}  
```
cisAssoc
test for variant-expression associations in cis, using VCF

Description
test for variant-expression associations in cis, using VCF and SummarizedExperiment representations

Usage
cisAssoc(summex, vcf.tf, rhs = ~1,
nperm = 3, cisradius = 50000,
genome = "hg19", assayind = 1, lbmaf = 1e-06,
dropUnivHet = TRUE, doEsts=FALSE)
data(lgeu) # obtains an example SummarizedExperiment

Arguments

summex instance of SummarizedExperiment-class
vcf.tf instance of TabixFile, referring to a tabix-indexed, bgzipped VCF file
rhs formula ‘right hand side’ for adjustments to be made as snp.rhs.tests is run on each expression vector
nperm number of permutations to be used for plug-in FDR computation
cisradius distance in bp around each gene body to be searched for SNP association
genome tag suitable for use in GenomeInfoDb structures
CisConfig-class

Object specifying configuration of cis-eQTL search, to be used with All.cis

Objects from the Class

Objects can be created by calls of the form new(“CisConfig”). Use replacement methods to update the fields.
CisConfig-class

Slots

snpack: character string identifying package holding the expression and genotype data; see `getSS`

genome: character string identifying genome build in use

rhs: Object of class "formula" right hand side for calls to `snp.rhs.tests`

nperm: Object of class "integer" number of permutations for plug in FDR

cellStem: Object of class "character" string used for scratch space folders, relative to current folder

radius: Object of class "integer" radius of search

shortfac: Object of class "integer" scores are scaled up by this factor so that precision can be retained in short integer representation

chrnames: Object of class "character" string identifying chromosome label used in gene annotation retrieval – typically length 1

smchrpref: Object of class "character" prefix to be attached to chromosome label in chrnames to pick out the element of `smlSet-class` instance used in testing

gchrpref: Object of class "character" prefix on chrnames token to be used for gene location retrievals

schrpref: Object of class "character" prefix on chrnames token to be used with SNPlocs package for retrieval of SNP locations

geneApply: Object of class "function" iterator over genes, could be lapply or mclapply

geneannopk: Object of class "character" Bioconductor annotation package for gene locations, typically for expression array

snpannopk: Object of class "character" Bioconductor dbSNP annotation package

smFilter: Object of class "function" function to be applied to smlSet instance that yields an smlSet instance with required contents; could apply MAF restriction for example by calling MAFfilter

exFilter: Object of class "function" function that is run right after smlSet is materialized, permitting replacement or filtering of expression data, when, for example, the ExpressionSet includes multiple tissue types

keepMapCache: Object of class "logical" for enhancing processing of gene-SNP cis mapping with a global cache

SSgen: Object of class "function" function that accepts name of an expression+SnpMatrix package (as generated by `externalize`), a chromosome tag (chrnames prefixed by smchrpref), and a function, and returns an smlSet instance

excludeRadius: Object of class "integerOrNULL" which will determine what interval about the gene is excluded for cis testing; 0 should exclude all within-gene SNP, but needs testing

estimates: Object of class "logical" if TRUE, estimates and standard errors (expanded and reduced in storage as a short int, using shortfac) are generated and retained

extraProps: Object of class "function" this function is applied to the cisScores output before it is returned, to bind additional metadata to the ranges if desired. Defaults to function(x)x.

useME: Object of class "logical" if TRUE, use the statistics generated by `Matrix_eQTL_engine` for association testing.

MEpvote: Object of class "numeric" used if useME slot is set to TRUE: p-value output threshold for retaining association test statistic generated by `Matrix_eQTL_engine`; defaults to 0.5. Higher values lead to higher volumes and longer times to completion.
Methods

chrnames signature(x = "CisConfig"): ...
chrnames <- signature(object = "CisConfig", value = "character"): ...
estimates signature(x = "CisConfig"): ...
estimates <- signature(object = "CisConfig", value = "logical"): ...
excludeRadius signature(x = "CisConfig"): ...
excludeRadius <- signature(object = "CisConfig", value = "integer"): ...
exFilter signature(x = "CisConfig"): ...
exFilter <- signature(object = "CisConfig", value = "function"): ...
gchrpref signature(x = "CisConfig"): ...
gchrpref <- signature(object = "CisConfig", value = "character"): ...
geneannopk signature(x = "CisConfig"): ...
geneannopk <- signature(object = "CisConfig", value = "character"): ...
geneApply signature(x = "CisConfig"): ...
geneApply <- signature(object = "CisConfig", value = "function"): ...
initialize signature(.Object = "CisConfig"): ...
keepMapCache signature(x = "CisConfig"): ...
keepMapCache <- signature(object = "CisConfig", value = "logical"): ...
radius signature(x = "CisConfig"): ...
radius <- signature(object = "CisConfig", value = "integer"): ...
rhs signature(x = "CisConfig"): ...
rhs <- signature(object = "CisConfig", value = "function"): ...
schrpref signature(x = "CisConfig"): ...
schrpref <- signature(object = "CisConfig", value = "character"): ...
shortfac signature(x = "CisConfig"): ...
shortfac <- signature(object = "CisConfig", value = "integer"): ...
show signature(object = "CisConfig"): ...
smchrpref signature(x = "CisConfig"): ...
smchrpref <- signature(object = "CisConfig", value = "character"): ...
smFilter signature(x = "CisConfig"): ...
smFilter <- signature(object = "CisConfig", value = "function"): ...
snpannopk signature(x = "CisConfig"): ...
snpannopk <- signature(object = "CisConfig", value = "character"): ...
SSgen signature(x = "CisConfig"): ...
SSgen <- signature(object = "CisConfig", value = "function"): ...

Examples

showClass("CisConfig")
ciseqByCluster  
end-to-end cluster-based cis-eQTL search, and allied utilities

Description

end-to-end cluster-based cis-eQTL search, and allied utilities

Usage

ciseqByCluster(cl, pack = "yri1kgv", outprefix = "yrirun", 
finaltag = "partyri100k", chromsToRun = 1:22, 
targetfolder = "/freshdata/YRI_3", radius = 100000L, nperm = 3L, ncoresPerNode = 8, 
numPCtoFilter = 10, lowerMAF = 0.02, geneannopk = "lumiHumanAll.db", 
snpannopk = "SNP1ocs.Hsapiens.dbSNP144.GRCh37", smchrpref = "chr", 
tmpForSort = "/tmp", numtiles = 200, 
postProcCores = 12, reqlist = NULL)

Arguments

cl instance of S3 cluster class from parallel package
pack character string naming package to which getSS can be applied to generate smlSet-class instances
outprefix character string used to prefix names of output GFF3 files
finaltag character string used to prefix names of final amalgamated GFF3 and data.table instances
chromsToRun numeric tags of chromosomes to be analyzed
targetfolder character string naming folder where GFF3 will be deposited
radius extent of search around gene model in bp
nperm number of permutations for plug-in FDR computation (usually a small integer)
ncoresPerNode number of cores for multicore testing: chromosomes map to nodes, genes map to cores
numPCtoFilter number of PCs to be removed through clipPCs
lowerMAF lower bound on MAF of SNP to be included for testing
geneannopk character string naming Bioconductor package with annotation for expression probe identifiers
snpannopk character string naming Bioconductor package with annotation for SNP locations
smchrpref character prefix converting chromsToRun elements to basenames of rda files harboring SnpMatrix instances
tmpForSort the assembly of final resources employs unix sort, and substantial temporary space can be required; this parameter tells where the temp files will reside
numtiles number of tiles into which the genome in use will be sliced for parallel processing in final assembly
postProcCores numeric establishing number of cores to use for final assembly of annotated output
reqlist rescue request, see Details section
Details

purpose is to maximize throughput of cis-eQTL testing in a two-level concurrent computing environment, where a cluster as defined in package parallel has nodes to which half-chromosomes will be dispatched; each node is assumed to be multicore and genes are mapped to cores during the iteration process.

the reqlist parameter consists of a list of elements (chromosome name, subchromosome token, and handler) to be used for completing a partial run

Value

a set of GFF3 files encoding all cis associations with location and various metadata

See Also

gffprocess, cgff2dt

Examples

#none yet

---

cisRun-class  

Class "cisRun"

Description

manage results of All.cis eQTL analysis

Objects from the Class

Objects can be created by calls of the form new("cisRun", ...).

Slots

  seqnames: Object of class "Rle"
  ranges: Object of class "IRanges" – will document the range searched for each probe/gene – therefore the values returned are addresses of gene extent minus/plus radius at each end
  strand: Object of class "Rle"
  elementMetadata: Object of class "DataFrame"
  seqinfo: Object of class "Seqinfo"
  metadata: Object of class "list"

Extends

collectBest

Methods

No methods defined with class "cisRun" in the signature.

Note

intent is to simplify output of cis eQTL testing in a GRanges instance

Author(s)

VJ Carey <stvjc@channing.harvard.edu>

Examples

showClass("cisRun")

collectBest<given a collection of All.cis outputs (cisRun instances) compute FDRs for various filterings>

Description

given a collection of All.cis outputs (cisRun instances) compute FDRs for various filterings

Usage

collectBest(fns,
  targetname = "harvest",
  mafs = c(0.01, 0.02, 0.025, 0.03333, 0.05, 0.075, 0.1),
  hidists = c(10000, 25000, 50000, 75000, 1e+05, 250000), interimSaves=FALSE)

collectFiltered( fns, targetname="harvest",
  mafs = c(.01, .02, .025, .03333, .05, .075, .1),
  hidists = c(10000, 25000, 50000, 75000, 100000, 250000),
  filterFun = cis.FDR.filter.best, filtApplier=lapply, interimSaves=FALSE)

Arguments

  fns       names of .rda with the cisRun outputs
  targetname basename of rda file to be emitted
  mafs      lower bounds on MAF for filtering
  hidists   upper bounds on cis radius for filtering
  filterFun function like GGtools::cis.FDR.filter.best
  filtApplier function like lapply
  interimSaves logical, if TRUE save list at each maf/dist transition

Details

  pifdr is repeatedly used to generate conditional plugin FDR for different filtering criteria
Value

A list of lists is written to disk incrementally, as the job can be long running.

Note

This is the workhorse of sensitivity analysis. Permits counting of genes with eQTL at selected FDR for various criteria on cis radius and MAF.

Examples

```r
## Not run:
#
# contents of fns are two chromosomes of cis runs for CEU
#
fns = dir(system.file("rdas", package="GGtools"), full=TRUE)
cc = collectBest(fns, mafs=c(.01, .05), hidists=c(10000, 50000))
sapply(cc, sapply, function(x) sum(x$fdr <= 0.01))
#
# this tells us which to keep
#
kp = cc[["0.05"]][["50000"]]
kp = kp[kp$fdr <= 0.01,]
#
# the hits are in the table above; the following function
# retrieves the initial scores giving rise to the filtered
# hits
#
pullHits = function(fns, atts) {
  tmp = lapply(fns, function(x) get(load(x)))
  kl = lapply(tmp, function(x) paste(names(x), x$snp, sep=":"))
  attk = paste(atts$genes, atts$bestsnp, sep=":")
  tmp = lapply(1:length(tmp), function(x) tmp[[x]][ match( attk, kl[[x]], nomatch=0 ) ])
  curans = do.call(c, lapply(tmp, as, "GRanges"))
  neword = match( attk, paste(names(curans), curans$snp, sep=":"))
  newfdr = atts$fdr[neword]
  curans$fdr = newfdr
  curans
}
pullHits( fns, kp )
#
#
# after executing code in example for All.cis (protected by dontrun)
# and running save(f1, file="f1.rda"), the following will work
#
genewise max score

cf1 = collectFiltered("f1.rda", mafs=.02, hidists=25000, targetname="gwise")
# SNPwise scores, all

cf2 = collectFiltered("f1.rda", mafs=.02, hidists=25000, targetname="swise",
  filterFun = cis.FDR.filter.SNPcentric.complete )
# SNPwise scores, best per SNP when SNP is cis to multiple genes

cf3 = collectFiltered("f1.rda", mafs=.02, hidists=25000, targetname="swise2",
  filterFun = cis.FDR.filter.SNPcentric )
## End(Not run) # end dontrun
```
concatCis  
*combine a list of cisRun instances to a single instance*

### Description
combine a list of cisRun instances to a single instance, with ad hoc metadata combination

### Usage
concatCis(crl)

### Arguments
crl  
list of instances of *cisRun-class*

### Details
the metadata for the output is a list with elements call and config as required, derived from first element of the input; the extras component holds the metadata elements of the remaining input list elements

### Value
a cisRun instance

### Examples
```r
## Not run:
example(All.cis)
concatCis(f1, f1)
## End(Not run)
```

---

**EqAppr-class**

### Description
Manage the appraisal of an eQTL search

### Objects from the Class
Objects can be created by calls of the form `new("EqAppr", ...)` or via `buildEqAppr()`

### Slots
- meta: Object of class "ApprMeta" basic descriptive information about source analysis
- sens: Object of class "ApprSens" outputs of sensitivity analysis
- pruned: Object of class "ApprRes" outputs of appraisal for LD-pruned predictions
- unpruned: Object of class "ApprRes" outputs of general appraisal without LD pruning
Methods

calfig signature(x = "EqAppr", ind = "character"): Additional arguments can be specified:
  ind index or name of model to be plotted
  hfudgetxt distance to move rendered fractions relative to bin x coordinate
  tickend maximum value at which axis tick mark will be plotted
  tickgap axis will have ticks at seq(0, tickend, tickgap)
  ylimin ylim setting for rendering
  xlimin xlim setting for rendering
  fraccex cex setting for fraction rendering
  fuselast if data are sparse in entries of high predicted probability, you can fuse the nearby
cells up to the end – pick fuselast=2 for final 2 cells, 3 for final 3 and so on

getModnames signature(x = "EqAppr"): obtain the list of strings used to name different appraisal models

getPruned signature(x = "EqAppr"): get the ApprRes instance corresponding to the LD-
pruned loci

getUnpruned signature(x = "EqAppr"): get the ApprRes instance for all loci in use

getsens signature(x = "EqAppr"): get sensitivity analysis results

show signature(object = "EqAppr"): concise report

Examples

showClass("EqAppr")

eqBox descriptive plot of expression against genotype for cisAssoc results

Description
descriptive plot of expression against genotype for cisAssoc results

Usage
eqBox(gene, snp, se, tf, radius=1e6, genome="hg19", ...)
eqDesc(gene, snp, se, tf, radius=1e6, genome="hg19", ...)

Arguments
gene identifier of gene in SummarizedExperiment se, must be present in rownames(se)
snp identifier of variant in VCF referenced by tf
se SummarizedExperiment instance
tf TabixFile reference for a VCF file that has been bgzipped and tabix-indexed
radius in order to limit the VCF import, we filter variants to those within a radius
around the selected gene – it is assumed that the selected snp will exist in that
region (we can’t extract SNP by name from vcf...)
genome a simple genome identifier tag
... (not used with eqDesc) will embellish plot; xlab and ylab already taken care of
Details

The chromosome names in the VCF and the seqlevelsStyle of the Summarized Experiment must match.

Value

eqBox produces a boxplot for all categories (including NA) of genotype. eqDesc tabulates the genotype calls in categories.

Examples

```r
if (require(VariantAnnotation)) {
  data(lgeu)
  lgeue = lgeu[,which(lgeu$popcode=="CEU")]
  tf20 = TabixFile(system.file("vcf/c20exch.vcf.gz", package="GGtools"))
  eqBox( "ENSG00000126005.10", "rs2425038", lgeue, tf20 )
}
```

Description

support for sensitivity analyses related to eQTL enumerations

Usage

```r
eqsens_dt(dtab, filtgen = filtgen.maf.dist, by = c("pairs", "snps", "probes")[1],
targfdrs = c(0.05, 0.01, 0.005),
parmslist = list(mafs = c(0.025, 0.05, 0.075, 0.1, 0.125),
dists = c(1000, 5000, 10000, 25000, 50000, 1e+05)),
renameChisq = TRUE)
filtgen.maf.dist (maf.dist, validate.tab = function(tab) all(c("mindist",
"MAF", "score") %in% colnames(tab)))
update_fdr_filt(tab, filt = function(x) x, by = c("pairs", "snps",
"probes")[1])
plotsens(eqsout, ylab = "count of eQTL at given FDR",
title = "cis radius in bp")
```

Arguments

dtab data.table instance as generated by converting a cisScores GRanges. In general it will need to have column names score, MAF, mindist, and columns with names permScore_1, ....

filtgen a function that generates a closure. The function returned by filtgen will be a function of one argument that filters an input data.table. The environment of the returned function will possess bindings used to define the filtering operation. filtgen.maf.dist, documented here, is a working example.
character atom describing the level of aggregation for sensitivity analysis. eQTL searches generally involve cartesian products of sets of SNPs and sets of genes, and if all elements of these products are of interest, set by to "pairs". For sensitivity analysis in which per-SNP associations are measured by choosing the maximum association statistic for all genes cis to the SNP, set by to "snps". For per-gene associations, with scores maximized over all SNPs cis to genes, use "probes".

targfdrs numeric vector of FDR levels for which enumerations are performed.

parmslist list of numeric vectors giving thresholds for use in filtering tests using filtgen

maf.dist numeric vector of length two giving thresholding values for MAF and cis distance for filtering association tests

validate.tab function of one argument, assumed to be a data.table instance, that can be used to check whether filtering conditions are feasible before attempting to filter; should return TRUE if they are, FALSE otherwise.

tab data.table instance as generated by converting a cisScores

filt function of one argument that operates on a data.table instance to reduce the rows to a desired set; parameters of filtering task are established in the environment of filt

eqsout matrix as output by eqsens_dt

ylab text string to be used to label Y axis on the left

title text string to label top of plot

renameChisq Some utilities fail to generate ’score’ for observed association statistic, but report as ’chisq’. If TRUE and such a column name is found in dtab, setnames will be run to rename to ’score’.

Details

The objective is to generate data for tabulation or visualization of sensitivity analyses, and the scope of sensitivity analysis can be established in various ways. This software is mostly intended as a framework.

Value

eqsens_dt returns a data.frame instance with enumerations of eQTL at various FDR thresholds for various settings of tuning parameters

update_fdr_filt revises (using pifdr) the fdr field of an input data.table instance using variable score as observed value, and permuted values furnished by the variables named with permScore as leading substring

Note

To do: allow filtering on the number of permutations to be used in FDR calculation.

Author(s)

VJ Carey <stvjc@channing.harvard.edu>
eqtlTests

Examples

## Not run:
example(cisScores) # would generate f1
names(f1) = NULL
eqsens.dt( data.table(as(f1, "data.frame")) )

## End(Not run)

eqt1Tests compute association statistics between all probes and SNP in an smlSet instance

Description

compute association statistics (or point estimates and standard errors) between all probes and SNP in an smlSet instance, using out-of-memory storage; the basic test statistics are generated by the snp.rhs.tests function of the snpStats package

Usage

eqt1Tests(smlSet, rhs = ~1 - 1, runname = "foo",
targdir = "foo", geneApply = lapply,
shortfac = 100,
checkValid = TRUE, useUncertain = TRUE,
glmfamily = "gaussian", doFFSUMM = FALSE)
eqt1Estimates(smlSet, rhs = ~1 - 1, runname = "foo",
targdir = "fooe", geneApply = lapply,
shortfac = 10000,
checkValid = TRUE, useUncertain = TRUE,
glmfamily = "gaussian")

Arguments

smlSet instance of smlSet
rhs fragment of a standard formula, minus a dependent variable (i.e., starts with tilde); bindings will be sought in pData(smlSet)
runname string used to identify output ff files	
targdir string naming the folder where ff outputs will reside
geneApply analog to lapply to drive iteration over probes
shortfac ff contents will be multiplied by this quantity and stored as short integers
checkValid logical, will apply validObject to smlSet if TRUE
useUncertain logical, passed as uncertain parameter to snp.rhs.tests to specify whether uncertain genotypes will be used (as ’dosage’ in GLM fitting)
glmfamily family specification for snp.rhs.tests
doFFSUMM logical indicating whether ff archives will be retained for col.summary outputs for SNPs
The purpose of the `eqtlTests` function is to allow very substantial eQTL search processes to occur with R. For several million SNP and tens of thousands of probes, the storage of test results requires attention to parsimony. The storage occurs out of memory, using the ff package, and employs short integers to represent chi squared statistics. These are scaled up prior to storage, and will be scaled down prior to use.

`eqtlEstimates` will use compact storage for both the point estimates and standard errors of association estimated under an additive genetic model.

Note: `snp.rhs.estimates` will emit a warning whenever the parameters are not estimable. These warning are suppressed by temporary setting of `options()[’warn’]` to zero.

```r
hm2ceuSMS = getSS("GGtools", c("20"), renameChrs=c("chr20"))
library(illuminaHumanv1.db)
cptag = get("CPNE1", revmap(illuminaHumanv1SYMBOL))
indc = which(featureNames(hm2ceuSMS) == cptag[1])
#
# get a set of additional genes on chr20
all20 = get("20", revmap(illuminaHumanv1CHR))
g20 = unique(c(all20[1:10], cptag))
#
hm = hm2ceuSMS[probeId(g20),] # reduce problem
td = tempdir()
curd = getwd()
setwd(td)
time.lapply = unix.time(e1 <- eqtlTests( hm, ~male ))
time.lapply
e1
# best chisq(1) for CPNE1
topFeats(probeId(cptag), e1)
setwd(curd)
```

Value

returns an instance of `eqtlTestsManager`

Author(s)

VJ Carey <stvjc@channing.harvard.edu>

Examples

use `MatrixEQTLS` computations and statistics as a back end to GGtools `eqtlTests`
Usage

eqtlTests.me(smlSet, rhs = ~1, runname = "20",
  targdir = "cisScratch.me", pvot = 0.5, geneApply = lapply,
  shortfac = 100, checkValid = TRUE, useUncertain = TRUE,
  glmfamily = "gaussian", scoretx = abs,
  matrixEQTL.engine.control =
    list(output_file_name = "/dev/null",
         useModel = modelLINEAR,
         errorCovariance = numeric(),
         verbose = FALSE,
         pvalue.hist = FALSE),
  snpSlicedData.control = .slicedDataDefaults,
  geneSlicedData.control = .slicedDataDefaults,
  covarSlicedData.control = .slicedDataDefaults,
  covariates_file_name = character())

Arguments

smlSet instance of smlSet-class
rhs formula for adjustment of tests for covariates or stratification, see snp.rhs.tests
runname tag used to distinguish emitted files
targdir folder where ff archives will reside
pvot setting for pOutputThreshold in Matrix_eQTL_engine
geneApply lapply-like function for iteration over genes, mclapply is suitable when in multi-
core environments
shortfac scaling factor to increase precision when test results are stored as short ints in ff
checkValid logical to check validity of input smlSet
useUncertain logical informing snp.rhs.tests that imputed real-valued B allele counts may be
  present among genotype data
glmfamily family specification for snp.rhs.tests
scoretx function to be applied to MatrixEQTL statistics. Defaults to abs, for signless
  association testing
matrixEQTL.engine.control list of parameters passed to Matrix_eQTL_engine
snpSlicedData.control list of parameters used to define SlicedData-class instances
geneSlicedData.control list of parameters used to define SlicedData-class instances
covarSlicedData.control list of parameters used to define SlicedData-class instances
covariates_file_name if covariates are to be used with MatrixEQTL testing engine, they reside in this
  file. regressOut can be used to avoid this if plug-in FDR are to be used

Details

provisional interface
Value

see `eqtlTests`

Note

intended for simple comparisons

References

Shabalin et al Bioinformatics (OUP) 2012

Examples

```r
if (require(MatrixEQTL)) {
  g22 = nsfilter( chrFilter( getSS("GGdata", "22"), "22" ), var.cutoff = .8 )
  m22 = eqtlTests.me(g22)
}
```

---

### eqtlTestsManager-class

**Class** "eqtlTestsManager"

**Description**

manage out-of-memory elements of an eQTL search

**Objects from the Class**

Objects can be created by calls of the form `new("eqtlTestsManager", ...)

**Slots**

- `fffile`: Object of class "ff_matrix" chisquared statistics stored as short ints in ff out of memory file
- `call`: Object of class "call" audit of creation call
- `sess`: Object of class "ANY" session info structure at time of creation
- `exdate`: Object of class "ANY" date at time of creation
- `shortfac`: Object of class "numeric" number by which chisq stats are multiplied to allow recovery of precision
- `geneanno`: Object of class "character" string naming annotation package relevant for probe identifier translation
- `df`: Object of class "numeric" degrees of freedom of chisq stats
- `summaryList`: Object of class "list" list of genotype statistical summaries
Methods

signature(x = "eqtlTestsManager", i = "ANY", j = "ANY", drop = "ANY"): extract chisq statistics properly rescaled from short int to double

show signature(object = "eqtlTestsManager"): concise report

topFeats signature(feat = "probeId", mgr = "eqtlTestsManager"): extract highest scores for SNP associated with given probeId

topFeats signature(feat = "rsid", mgr = "eqtlTestsManager"): extract highest scores for probes associated with given SNP

Note

instances are created by eqtlTests

Author(s)

VJ Carey <stvjc@channing.harvard.edu>

Examples

showClass("eqtlTestsManager")

ExpressionSet instance for illustrating integrative smlSet container

Description

ExpressionSet instance for illustrating integrative smlSet container

Usage

data(eset)

Format

The format is: Formal class 'ExpressionSet' [package "Biobase"] with 7 slots ..@ experimentData :Formal class 'MIAME' [package "Biobase"] with 13 slots .. ..@ name : chr "" .. ..@ lab : chr "" .. ..@ contact : chr "" .. ..@ title : chr "" .. ..@ abstract : chr "" .. ..@ url : chr "" .. ..@ pubMedIds : chr "" .. ..@ samples : list() .. ..@ hybridizations : list() .. ..@ normControls : list() .. ..@ preprocessing : list() .. ..@ other : list() .. ..@ __classVersion__:Formal class 'Versions' [package "Biobase"] with 1 slots .. .. ..@ .Data:List of 2
Details

Expression data harvested in 2007 from GENEVAR

Examples

data(eset) # yields ExpressionSet instance called ex
getCisMap

create, using Bioconductor annotation resources, a structure that enumerates SNP in the vicinity of ('cis' to) genes

Description

create a structure that enumerates SNP in the vicinity of ('cis' to) genes

Usage

getCisMap(radius = 50000, gchr = "20", 
        schr = "ch20", geneannopk = "illuminaHumanv1.db", 
        snpannopk = snplocsDefault(), 
        as.GRangesList = FALSE, excludeRadius=NULL)

Arguments

radius How far, in bases, up or down stream from the asserted coding region limits to include SNP

gchr the token to be used to acquire locations for probes on a specified chromosome, using revmap([dbpk]CHR)

schr the token to be used to acquire locations for SNP on a specified chromosome, using snplocs

geneannopk character string naming a Bioconductor .db expression chip annotation package; or a complex string with first part naming a Bioconductor FDb.* annotation package, colon separator, and a second string naming the getter hook that when called returns a GRanges with names corresponding to features and ranges corresponding to feature extents. For example "FDb.InfiniumMethylation.hg19:get27k" is valid. Note that in this case, gchr must have prefix "chr".

snpannopk character string naming a Bioconductor SNPlocs.* SNP metadata package

as.GRangesList logical telling whether a GRangesList should be returned

excludeRadius numeric or NULL: radius of interval around gene extent from which SNP will be excluded, required to be less than radius

Details

This is a utility that the developer would rather not export. The complexity of harmonizing queries among probe and SNP annotation resources leads to a somewhat fragile product. Users who have their own gene ranges and SNP locations can examine the namelist component of the output of the default call to see what is expected for the *.cis.eQTLs function. For the set of chromosomes to be analyzed, there will be a list of chromosome specific namelist-like lists.

Value

Instance of cisMap class, which will retain SNP location, gene range, and radius information for auditing.
Examples

```r
## Not run:
getCisMap()

## End(Not run)
```

gffprocess

- transform a collection of gff3 into a single tabix-indexed gff3

Description

process a collection of gff3 into a single tabix-indexed gff3 using unix utilities to minimize memory requirements

Usage

```r
gffprocess(basename = "fullyri100k", n_in = 44, headpatt = ".1A", tmpForSort = "/freshdata/tmp")
```

Arguments

- **basename**: basename of the resulting .gff3.gz(.tbi) file
- **n_in**: number of gff3 files to be processed – used for consistency check against length(dir(patt="gff3"))
- **headpatt**: pattern to identify file for the 'top' gff3 to be used as the contents are concatenated
- **tmpForSort**: name of a folder that unix sort will use as a temporary directory

Details

The purpose of this utility is to exploit unix shell tools to unify a collection of gff3 files generated using `All.cis`. The use case is cluster-based per-chromosome (or split chromosome) cis-testing generating a large number of GRanges that are transformed to gff3 to allow targeted interrogation.

Value

Used for side effects. Will fail if any unix utility call via system() returns nonzero value. Returns NULL otherwise.

Author(s)

VJ Carey <stvjc@channing.harvard.edu>
**gwSnpTests**

**execute a series of tests for association between genotype and expression**

**Description**

execute a series of tests for association between genotype and expression

**Usage**

```r
gwSnpTests(sym, sms, ...)  # topSnps(x, n=10)
```

**Arguments**

- **sym**: instance of `probeId` or `genesym`
- **sms**: instance of `smlSet-class`
- **x**: instance of `gwSnpScreenResult`
- **n**: integer, number of test results to be reported, sorted decreasing from the most significant
- **...**: not used

**Details**

The `plot` method for `gwSnpScreenResult` instances takes a second argument, the name of a Bioconductor SNPlocs. package.

**Value**

an instance of the `gwSnpScreenResult` class, to be examined by `topSnps`

**Note**

The most basic application yields one d.f. chi-squared statistics based on score tests.

**Author(s)**

VJ Carey <stvjc@channing.harvard.edu>

**Examples**

```r
s20 = getSS("GGtools", "20")
t1 = gwSnpTests(genesym("CPNE1")~male, s20)
topSnps(t1)
## Not run:
plot(t1, snplocsDefault())
## End(Not run)
```
**Description**

labeled GRanges with ChromHMM chromatin states for GM12878

**Usage**

```r
data(hmm878)
```

**Format**

The format is:

```r
Formal class 'GRanges' [package "GenomicRanges"] with 6 slots
..@ seqnames :Formal class 'Rle' [package "IRanges"] with 4 slots
  .. ...@ values : Factor w/ 23 levels "chr1","chr2"...: 1 2 3 4 5 6 7 8 9 10 ...
  .. ...@ lengths : int [1:23] 54467 46499 37617 25155 30071 34846 29420 24506 24123 27263 ...
  .. ...@ elementMetadata: NULL
  .. ...@ metadata : list()
..@ ranges :Formal class 'IRanges' [package "IRanges"] with 6 slots
  .. ...@ start : int [1:571339] 10001 10601 11138 11738 11938 12138 14538 20338 22138 22938 ...
  .. ...@ width : int [1:571339] 600 537 600 200 200 2400 5800 1800 800 4000 ...
  .. ...@ NAMES : NULL
  .. ...@ elementType : chr "integer"
  .. ...@ elementMetadata: NULL
  .. ...@ metadata : list()
..@ strand :Formal class 'Rle' [package "IRanges"] with 4 slots
  .. ...@ values : Factor w/ 3 levels "+","-","*": 3
  .. ...@ lengths : int 571339
  .. ...@ elementMetadata: NULL
  .. ...@ metadata : list()
..@ elementMetadata:Formal class 'DataFrame' [package "IRanges"] with 6 slots
  .. ...@ rownames : NULL
  .. ...@ nrows : int 571339
  .. ...@ listData :List of 4
  .. ... ..$ name : chr [1:571339] "15_Repetitive/CNV" "13_Heterochrom/lo" "8_Insulator" "11_Weak_Txn"
  .. ... ..$ score : num [1:571339] 0 0 0 0 0 0 0 0 0 0 ...
  .. ... ..$ itemRgb: chr [1:571339] "#F5F5F5" "#F5F5F5" "#F5F5F5" "#0B6BEF" "#9F9FFF" ...
  .. ... ..$ thick :Formal class 'IRanges' [package "IRanges"] with 6 slots
  .. ... .. ..@ start : int [1:571339] 10001 10601 11138 11738 11938 12138 14538 20338 22138 ...
  .. ... .. ..@ width : int [1:571339] 600 537 600 200 200 2400 5800 1800 800 4000 ...
  .. ... ..@ NAMES : NULL
  .. ... .. ..@ elementType : chr "integer"
  .. ... ..@ elementMetadata: NULL
  .. ... ..@ metadata : list()
  .. ... ..@ elementType : chr "ANY"
  .. ... ..@ elementMetadata: NULL
```
\texttt{pi\_fdr}

\begin{verbatim}
.. ...@ metadata : list()
..@ seqinfo: Formal class 'Seqinfo' [package "GenomicRanges"] with 4 slots
... ...@ seqnames : chr [1:23] "chr1" "chr2" "chr3" "chr4" ...
... ...@ seqlengths: int [1:23] 249250621 243199373 198022430 191154276 180915260 171115067
159138663 146364022 141213431 135534747 ...
... ...@ is_circular: logi [1:23] FALSE FALSE FALSE FALSE FALSE FALSE ...
... ...@ genome: chr [1:23] "hg19" "hg19" "hg19" "hg19" "hg19" ...
..@ metadata: List of 1
... $url: chr "http://genome.ucsc.edu/cgi-bin/hgFileUi?g=wgEncodeBroadHmm&db=hg19"

Details

acquired using rtracklayer import from the bed file given at metadata(hmm878)["url"]

Source

see details

References


Examples

data(hmm878)

table(hmm878$name)

\begin{verbatim}
 utility for computing plug-in FDR
\end{verbatim}

\textbf{Description}

utility for computing plug-in FDR

\textbf{Usage}

\texttt{pi\_fdr( obs, perms, legacy=FALSE, trimToUnit = TRUE, ... )}

\textbf{Arguments}

\begin{itemize}
  \item \texttt{obs} observed association scores
  \item \texttt{perms} vector of association scores under permutation; length should be integer multiple of \texttt{length(obs)}
  \item \texttt{legacy} logical, if \texttt{TRUE}, use the approximate version of \texttt{pi\_fdr()} available before 12/30/2013, with additional arguments if desired
  \item \texttt{trimToUnit} logical, if \texttt{TRUE}, values greater than 1 are replaced by 1. Such values can occur, for example, with relatively small sample sizes.
  \item \texttt{...} extra arguments passed if \texttt{legacy} is \texttt{TRUE}
\end{itemize}
Details

Revised 12/30/13 to employ hist() to rapidly bin the permuted values.

Use legacy=TRUE to obtain the approximate implementation, for which the following remarks held:

“As currently implemented the algorithm is quadratic in length(obs). While it is possible to get a unique FDR value for every element of obs, an approximate approach yields practically identical precision and by default this will be used for obs with length 2000 or more. In this case, approx is used with rule=2 to interpolate from the grid-based FDR estimates back to the data values.” Additional parameters npts and applier may be supplied if legacy is set to TRUE.

npts defined the number of points spanning the range of obs to be used for a lossy grid-based computation only used if length(obs)>npts.

applier is to be an sapply-like function.

Value

vector of plug-in FDR estimates congruent to obs

References

Hastie Tibshirani and Friedman Elements of Statistical Learning ch 18.7

Examples

```r
set.seed(1234)
op = par(no.readonly=TRUE)
par(mfrow=c(2,2))
X = c(rchisq(30000,1),rchisq(300,10))
Y = rchisq(30300*3,1)
qqplot(Y, X, xlab="null", ylab="observed")
hist(pp <- pifdr(X,Y), xlab="plug-in FDR", main=" ")
library(multtest)
rawp = 1-pchisq(X, 1)
MT <- mt.rawp2adjp(rawp)
MT2 = MT[[1]][order(MT[[2]]),]
plot(MT2,"BH"), pp, xlab="BH FDR", ylab="plug-in FDR")
par(op)
```

qqhex

obtain qqplot coordinates for the specific case of comparing a given distribution to that of multiple realizations from a permutation distribution, and bin these coordinates using hexbin, useful for very large samples

Description

obtain qqplot coordinates for the specific case of comparing a given distribution to that of multiple realizations from a permutation distribution, and bin these coordinates using hexbin, useful for very large samples
Usage

```r
binnedQQ(dt, nxbins=20,
  ylim=c(0,76), xlim=c(0,30), end45=5, thrs=c(0,.001,.005,.01,.05),
  tempmar = c(6,4,4,5), ...)
qqhex(sco, p1, p2, p3, fdr, nxbins = 20, thrs = c(0, 0.001, 0.005, 0.01, 0.05))
binqq(qqob, ylim = c(0, 76), xlim = c(0, 30), end45=5, ...)
```

Arguments

- `dt`: a `data.table` instance with association scores and scores obtained under permutation along with FDR, as returned by `cgff2dt` or `ciseqByCluster`
- `sco`: numeric vector of observed statistics
- `p1`: realization of null distribution for `sco`, independent of `p2` and `p3`
- `p2`: realization of null distribution for `sco`, independent of `p1` and `p3`
- `p3`: realization of null distribution for `sco`, independent of `p1` and `p2`
- `fdr`: vector of FDR associated with elements of `sco`
- `nxbins`: number of bins to be used for samples from the null distribution
- `thrs`: vector of thresholds in FDR to be used for ruling the plot
- `qqob`: for `binhex()`, output of `qqhex`
- `ylim`: vertical limits of rendering
- `xlim`: horizontal limits of rendering
- `end45`: a segment is drawn from (0,0) to (end45,end45) to depict the line of identity
- `tempmar`: numerical vector with 4 elements serving as a temporary setting of the mar parameter of `par`
- `...`: not currently used

Value

For `qqhex`, a list with elements

- `hb`: output of `hexbin`
- `thrs`: vector of input `thrs`
- `scothrs`: vector of observed statistics corresponding to FDRs in `thrs`

Examples

```r
opar = par(no.readonly=TRUE)
set.seed(123)
x = c(rchisq(9000,1), rchisq(1000,12))
nn = lapply(1:3, function(x) rchisq(10000,1))
fd = pifdr(x, unlist(nn))
qqh = qqhex(x, nn[[1]], nn[[2]], nn[[3]], fd)
par(mar=c(4,4,4,7))
binqq(qqh,xlim=c(0,10), ylim=c(0,20))
mtext(4, "FDR")
par(opar)
```
**richNull**

bind metadata concerning SNP allele frequency and other aspects of optimized cis-eQTL association to an mcwBestCis instance

**Description**

bind metadata concerning SNP allele frequency and other aspects of optimized cis-eQTL association to an mcwBestCis instance, to allow conditional FDR computation

**Usage**

```r
richNull(..., MAFlb = 0.01, npc = 10, radius = 250000, nperm = 1,
       innerFilt = function(x) x, outerFilt = function(x) x)
```

```r
meta.richNull(..., MAFlb = 0.01, npc = 10, radius = 250000,
              nperm = 1, innerFilt = function(x) x, outerFilt = function(x) x)
```

# # internally:
# # bigfilt = function(z)
# # outerFilt(MAFlbFilter(clipPCs(permEx(innerFilt(z)), 1:npc), lower = MAFlb))
#

**Arguments**

- `...` should provide bindings for smpack and chrnames, which will be used to obtain gene/probe locations; see `getSS` for information on smpack settings.
- `MAFlb` lower bound on SNP MAF for null distribution evaluation
- `npc` number of expression principal components to be removed
- `radius` radius used for testing
- `nperm` This establishes how many permutations of expression against genotype will be performed for this process.
- `innerFilt` function immediately applied to generated smlSet instances
- `outerFilt` function applied to generated smlSet instances after `clipPCs` and `MAFlbFilter` are applied in that order

**Details**

The purpose of `richNull` is to obtain realizations from the permutation distribution of cis-eQTL association statistics, binding information on the characteristics of the optimal results with the scores. This allows us to use conditioning with the realizations from the permutation distribution.

**Value**

`richNull` returns a list of `nperm` mcwBestCis instances with additional metadata bound in

**Author(s)**

Vince Carey <stvjc@channing.harvard.edu>
sampsInVCF  
**enumerate samples available in a VCF file**

**Description**

enumerate samples available in a VCF file

**Usage**

```r
sampsInVCF(tf)
```

**Arguments**

- `tf` instance of `TabixFile` referring to a tabix-indexed VCF

**Value**

vector of available sample identifiers

**Note**

This package exports `TabixFile` for the sake of the example below.

**Examples**

```r
tf = TabixFile(system.file("vcf/CEU.exon.2010_09.genotypes.vcf.gz", package="GGtools"))
sampsInVCF(tf)
```

scoresCis  
**visualize a gene model with cis-eQTL association scores (-log FDR by default) on the basis of a ciseqByCluster data.table output**

**Description**

visualize a gene model with cis-eQTL association scores (-log FDR by default) on the basis of a ciseqByCluster data.table output

**Usage**

```r
scoresCis(sym = "ORMDL3", cisRun, 
cisannopk = "lumiHumanAll.db", radius = 1e+05, pad = 1000, 
trxScore = function(x) -log10(x + (1e-05)), ylim = c(0, 4), 
genometag = "hg19", plot.it = TRUE, laxistag = "-log10 FDR: ", ...)
```
**sensanal**

Arguments

- `sym`: gene symbol to be resolved into probe id using cisannopk
- `cisRun`: data.table output of ciseqByCluster
- `cisannopk`: Annotation resource, often a ChipDb instance
- `radius`: radius to be added to gene model for display, should typically agree with that used in the search
- `pad`: some extra space
- `txScore`: function that will transform fdr for rendering
- `ylim`: vertical limits for fdr display
- `genometag`: coordinates from this build of genome
- `plot.it`: logical dictating whether plotTracks will be run
- `laxistag`: token used to tell what units are used on vertical axis
- `...`: not used

Value

a list of Gviz tracks, invisibly returned

See Also

The Bioconductor workflow on cloud-enabled cis-eQTL analysis.

---

**sensanal**  
Summarize information from a collection of eQTL searches for sensitivity assessment

**Description**

Summarize information from a collection of eQTL searches for sensitivity assessment

**Usage**

`sensanal(object, fdrbound)`

**Arguments**

- `object`: instance of `sensiCisInput-class`
- `fdrbound`: numeric upper bound on FDR for declarations of eQTL yield

**Details**

Sensitivity analysis for cis-eQTL search involves checking effects of scope of search, allele frequency filtering, and adjustment for expression heterogeneity on eQTL declarations. In this version, we focus on collections of outputs of `best.cis.eQTLs`, to which the values of tuning parameters are bound. These collections are identified in a `sensiCisInput-class` instance, and the `sensanal` function processes these outputs into a `sensiCisOutput-class` instance for tabulation and visualization.
Value

a `sensiCisOutput-class` instance

Author(s)

VJ Carey <stvjc@channing.harvard.edu>

---

**sensiCisInput-class**  
*Class* "sensiCisInput"

---

**Description**

Manage references to collections of cis-eQTL searches for sensitivity analysis.

**Objects from the Class**

Objects can be created by calls of the form `new("sensiCisInput", ...)`.

**Slots**

- `cisMgrFiles`: Object of class "character": a vector of filenames, each file is an instance of class `mcwBestCis-class`
- `cisMgrProperties`: Object of class "list" one vector with named elements per element of `cisMgrFiles`, with components `rad`, `excl`, `maf`, `nperm`, `npc`; see details below.
- `probeannopk`: Object of class "character", identifying a bioconductor probe annotation package that can be used to map probe identifiers to other vocabularies or feature value sets

**Methods**

- `sensanal` signature(`object = "sensiCisInput", fdrbound = "numeric"`): generates an instance of `sensiCisOutput-class` with summarization of sensitivities
- `show` signature(`object = "sensiCisInput"`): concise rendering

**Note**

This version of sensitivity analysis support is rudimentary and involves manual construction of metadata that should be extractable from analysis outputs. The radius of the cis search (and radius of excluded interior if used) are identified as elements named `rad` and `excl` in the `cisMgrProperties` vectors; additional elements `maf`, `nperm`, and `npc` define the lower bound for minor allele frequency, number of permutations for plug-in FDR computation, and number of principal components removed to adjust for expression heterogeneity in the associated cis-eQTL search.

**Examples**

`showClass("sensiCisInput")`
sensiCisOutput-class       Class "sensiCisOutput"

Description
This class helps to manage the results from a collection of cis-eQTL searches.

Objects from the Class
Objects can be created by calls of the form `new("sensiCisOutput", ...)`. 

Slots
- `byGene`: Object of class "GRanges", organized to provide ranges for genes and their best associated cis SNP
- `bySNP`: Object of class "GRanges" organized to provide easy access to genomic coordinates of SNP found to be most strongly associated with a gene in cis
- `tabAtFDRB`: Object of class "ANY" a flattened table that defines tuning parameters and eQTL yield for a collection of searches
- `input`: Object of class "sensiCisInput": object that describes the files and parameter settings used for the sensitivity analysis
- `thecall`: Object of class "call": the call generating this instance
- `fdrbound`: Object of class "numeric": gives the upper bound on FDR for declaring an eQTL
- `sessionInfo`: Object of class "ANY": describes state of system in which the object was made.

Methods
- `show` signature(object = "sensiCisOutput"): concise rendering with hints

Author(s)
VJ Carey <stvjc@channing.harvard.edu>

Examples
showClass("sensiCisOutput")

simpleTiling       create a GRanges with a tiling of the human genome

Description
create a GRanges with a tiling of the human genome

Usage
simpleTiling(ntile)
Arguments

ntile

Examples

###---- Should be DIRECTLY executable !! ----
###-- ==> Define data, use random,
###--or do help(data=index) for the standard data sets.

## The function is currently defined as
function (ntile)
{
  require(Homo.sapiens)
  hsi = seqinfo(Homo.sapiens)[paste0("chr", 1:22), ]
  GenomicRanges::unlist(tileGenome(hsi, ntile = 100))
}

snplocsDefault

name the default SNPlocs.Hsapiens.dbSNP.* package

Description

generate a string naming the default SNPlocs.Hsapiens.dbSNP.* package for use with GGtools

Usage

snplocsDefault()

Details

allows centralized specification of SNPlocs resource package

Value

a character string, see example

Examples

snplocsDefault()}
strMultPop

serialization of a table from Stranger’s multipopulation eQTL report

Description
serialization of a table from Stranger’s multipopulation eQTL report

Usage
data(strMultPop)

Format
A data frame with 39649 observations on the following 12 variables.
rsid  a factor with levels rs...
genescym  a factor with levels 37865 39692 ABC1 ABCD2 ABHD4 ACAS2 ...
illv1pid  a factor with levels GI_10047105-S GI_10092611-A GI_10190705-S GI_10567821-S
      GI_10835118-S GI_10835186-S ...
snpChr  a numeric vector
snpCoordB35  a numeric vector
probeMidCoordB35  a numeric vector
probe2probe  a numeric vector
minuslog10p  a numeric vector
adjR2  a numeric vector
assocGrad  a numeric vector
permThresh  a numeric vector
popSet  a factor with levels CEU-CHB-JPT CEU-CHB-JPT-YRI CHB-JPT

Details
imported from the PDF(!) distributed by Stranger et al as supplement to PMID 17873874

Source
PMID 17873874 supplement

References
PMID 17873874 supplement

Examples
data(strMultPop)
strMultPop[1:2,]
Description

Instances from this class can be input to the transScores function to control a trans-eQTL search.

Objects from the Class

Instances from this class can be input to the transScores function to control a trans-eQTL search. Objects can be created by calls of the form `new("TransConfig")`.

Slots

- `snpchr`: Object of class "character" identifies the name of the chromosome harboring SNP that will all be used (subject to filtering by smFilter function) in transcriptome-wide searches for associated transcripts.
- `gbufsize`: Object of class "integer", scores for the top gbufsize genes are retained as the search proceeds.
- `batchsize`: Object of class "integer" used in processing ff-based archives for association scores.
- `snpack`: Object of class "character", tells the name of the installed package used for retrieval of expression-genotype data using `getSS`.
- `rhs`: Object of class "formula", formula used in `snp.rhs.tests`; typically not used. If plug-in FDR is desired, adjustments should be executed in a regresOut call.
- `folderStem`: Object of class "character", name of a folder where interim results are sequestered.
- `radius`: Object of class "integer", defines region around SNP within which genes are considered 'cis' so tests are not conducted.
- `shortfac`: Object of class "integer" see documentation for `CisConfig-class`.
- `chrnames`: Object of class "character" see documentation for `CisConfig-class`.
- `smchrpref`: Object of class "character" see documentation for `CisConfig-class`.
- `gchrpref`: Object of class "character" see documentation for `CisConfig-class`.
- `schrpref`: Object of class "character" see documentation for `CisConfig-class`.
- `geneApply`: Object of class "function" see documentation for `CisConfig-class`.
- `geneannopk`: Object of class "character" see documentation for `CisConfig-class`.
- `snpannopk`: Object of class "character" see documentation for `CisConfig-class`.
- `smFilter`: Object of class "function" see documentation for `CisConfig-class`.
- `exFilter`: Object of class "function" see documentation for `CisConfig-class`.
- `keepMapCache`: Object of class "logical" see documentation for `CisConfig-class`.
- `SSgen`: Object of class "function" see documentation for `CisConfig-class`.
- `excludeRadius`: Object of class "integerOrNULL" see documentation for `CisConfig-class`.
- `estimates`: Object of class "logical" see documentation for `CisConfig-class`.

Extends

Class "CisConfig", directly.
transeqByCluster

Methods

- `batchsize` signature(x = "TransConfig"): ...
- `batchsize<-` signature(object = "TransConfig", value = "integer"): ...
- `gbufsize` signature(x = "TransConfig"): ...
- `gbufsize<-` signature(object = "TransConfig", value = "integer"): ...
- `show` signature(object = "TransConfig"): ...
- `snpchr` signature(x = "TransConfig"): ...
- `snpchr<-` signature(object = "TransConfig", value = "character"): ...

Examples

`showClass("TransConfig")`

transeqByCluster  convenience functions for trans-eQTL testing

Description

convenience functions for trans-eQTL testing, one assuming a parallel-based cluster instance is available, one assuming a chromosome’s SNPs will all be candidates for testing

Usage

`transeqByCluster(cl, snpchrs = c("chr21", "chr22"), exchrs = 1:22, baseconf, targname = "transrun_", nperm = 1, inseed = 1234, ...)`

`transeqByChrom(snpchr = "chr22", exchrs = 1:22, baseconf, targname = "transrun_", nperm = 1, inseed = 1234, ...)`

Arguments

- `cl` cluster instance as defined by the parallel package makeCluster API
- `snpchrs` character vector of tokens to be used to enumerate chromosomes harboring SNP for testing
- `snpchr` character atom, for transeqByChrom, the chromosome on which testing will be conducted
- `exchrs` enumeration of chromosomes harboring expression measures to be checked for trans association with SNPs
- `baseconf` an instance of `TransConfig-class`
- `targname` folder where scratch results are computed
- `nperm` number of permutations to be used for plug-in FDR
- `inseed` seed to be set before permutations are attempted, in conjunction with RNGkind("L’Ecuyer-CMRG")
- `...` not used
Details

the TransConfig-class instance determines most of the details of the testing procedure

Value

a data.frame with test results as chisq, and permScore* with scores obtained after permuting expression against genotype

---

transManager-class  
Class "transManager"

Description

simple container for manager of transScores output

Objects from the Class

Objects can be created by calls of the form new("transManager", ...).

Slots

base: Object of class "list" includes ff references for scores and indices of genes corresponding to scores, and other metadata about the run

Methods

show signature(object = "transManager"): simple reporter

See Also

transTab

Examples

showClass("transManager")

---

transScores  
obtain the top trans associations for each SNP in an smlSet

Description

obtain the top trans associations for each SNP in an smlSet
transScores

Usage

transScores( tconfig )

transScores.legacy(smpack, snpchr = "chr1", rhs, K = 20, targdirpref = "tsco", geneApply = lapply, 
chrnmes = paste("chr", as.character(1:22), sep = ""), geneRanges = NULL, snpRanges = NULL, 
radius = 2e+06, renameChrs = NULL, probesToKeep = NULL, batchsize = 200, 
genegr = 50, shortfac = 10, wrapperEndo = NULL, 
geneannopk = "illuminaHumanv1.db", 
snpannopk = snplocsDefault(), gchrpref = "", 
smFilter = function(x)x, 
exFilter = function(x)x, 
SSgen = GGBase::getSS)

meta.transScores (smpackvec = c("GGdata", "hmyriB36"), 
snpchr = "22", rhsList = list(~1, ~1), K = 20, targdirpref = "mtsco", 
geneApply = lapply, chrnames = as.character(21:22), 
radius = 2e+06, renameChrs = NULL, 
probesToKeep = NULL, batchsize = 200, genegr = 50, shortfac = 10, wrapperEndo = NULL, 
geneannopk = "illuminaHumanv1.db", snpannopk = snplocsDefault(), 
gchrpref = "", schrpref = "ch", 
exFilterList = list(function(x)x, function(x)x), 
SMFilterList = list(function(x)x, function(x)x), 
SSgen = GGBase::getSS)

Arguments

tconfig instance of TransConfig-class
smpack name of package holding eset.rda providing 'ex' ExpressionSet when loaded, 
and holding SnpMatrix instances in inst/parts
smpackvec vector of names of package holding eset.rda providing 'ex' ExpressionSet when loaded, and holding SnpMatrix instances in inst/parts
snpchr name or vector of chromosome names of SNPs of interest
rhs right hand side of snp.rhs.tests model for which expression is left hand side, e.g., 
covariates other than genotype
rhsList list of right hand side of snp.rhs.tests model for which expression is left hand side, e.g., covariates other than genotype, one per element of smpackvec
K number of most highly associated features to be retained
targdirpref prefix of target folder name (passed to eqtlTests
geneApply passed to eqtlTests
chrnames names of chromosomes harboring genes that will be tested for association with genotype
geneRanges list of GRanges-class instances containing chromosomal coordinate defined regions occupied by genes, with regions partitioned by chromosomes, and list element names as given in chrnames above
snpRanges list of GRanges-class instances with SNP addresses
radius radius within which an association is considered cis and therefore the corresponding test statistic is set to zero
**transScores**

renameChrs passed to `getSS`

probesToKeep passed to `getSS`

batchsize defines batch size for `ffrowapply`

genegran passed to `eqtlTests`

shortfac passed to `eqtlTests`

wrapperEndo a function accepting and returning an smlSet instance, evaluated before numerical analysis of associations, which will be executed on the output of this function

prefix to convert chrnames into appropriate tokens for obtaining gene metadata; in future this may need to be a string transformation function

prefix to convert chrnames into appropriate tokens for use with snplocs for the SNP location information package identified in `snpnopack` parameter below

geneannopk character string naming a Bioconductor .db expression chip annotation package

snpannopk character string naming a Bioconductor SNPlocs.* SNP metadata package

exFilter function to transform ExpressionSet component of retrieved smlSet, to reduce probe sets in use, for example

smFilter function to transform smlSet instance before use; filter can affect genotypes in smlList(x)[[1]], for example

exFilterList list of functions serving as exFilters for each of the elements of smpackvec

SMFilterList list of functions servicing as wrapperEndos for each of the elements of smpackvec

SSgen function to be used to create smlSet instance for testing – in general, GGBase::getSS has been used to pull the ExpressionSet and SnpMatrix data from a named package, but in some cases a specialize task is needed to create the desired smlSet. Whatever is passed to SSgen must return an smlSet instance.

**Value**

- **scores**
  - an S by K ff matrix where S is number of SNPs, K is number of best features to be retained, with element s,k the kth largest score statistic among association tests computed for SNP s

- **inds**
  - an S by K ff matrix with s,k element telling which element of guniv (see below) is the gene giving the kth largest score statistic for association

- **guniv**
  - the vector of gene identifiers defining the universe of genes tested

- **snpnames**
  - vector of SNP identifiers

- **call**
  - the call used to create the result

**Author(s)**

VJ Carey <stvjc@channing.harvard.edu>

**Examples**

```r
## Not run:
library(GGdata)
# need to define the geneRanges and snpRanges ...
transScores("GGdata", "20", renameChrs="chr20", chrnames="chr21")

## End(Not run)
```
transTab

**Description**

tabulate results of transScores run

**Usage**

transTab(x, snps2keep, ...)

**Arguments**

- `x`: a transManager instance.
- `snps2keep`: character vector used for filtering snps whose scores will be retained; intersection with snps named in x will be used.
- `...`: not used

**Value**

data.frame instance

**Author(s)**

VJ Carey <stvjc@channing.harvard.edu>

vcf2sm

**Description**

generate a SnpMatrix instance on the basis of a VCF (4.0) file.

**Usage**

vcf2sm(tbxfi, ..., gr, nmetacol)

**Arguments**

- `tbxfi`: instance of `TabixFile-class`
- `...`: not used
- `gr`: instance of `GRanges-class`
- `nmetacol`: numeric: tells number of columns used in each record as locus-level metadata

**Details**

This function is relevant only for diallelic SNP. If any base call is denoted `.`, the associated genotype is set to missing (raw 0), even if the nonmissing call is ALT, implying at least one ALT.
Value

an instance of \texttt{SnpMatrix-class}

Author(s)

VJ Carey \texttt{<stvjc@channing.harvard.edu>}

References

\url{http://www.1000genomes.org/wiki/doku.php?id=1000_genomes:analysis:vcf4.0}

Examples

\begin{verbatim}
vref = system.file("vcf/CEU.exon.2010_09.genotypes.vcf.gz", packages="GGtools")

gg = GenomicRanges::GRanges(seqnames="1", IRanges::IRanges(10e6,20e6))
vcf2sm(Rsamtools::TabixFile(vref, gr=gg, nmetacol=9L)
\end{verbatim}
Index

**Topic classes**
- CisConfig-class, 17
- cisRun-class, 21
- EqAppr-class, 24
- eqtlTestsManager-class, 31
- sensiCisInput-class, 44
- sensiCisOutput-class, 45
- TransConfig-class, 48
- transManager-class, 50

**Topic datasets**
- b1, 7
- ex, 32
- hmm878, 37
- strMultPop, 47

**Topic models**
- All.cis, 4
- appraise, 5
- best.cis.eQTLs, 8
- best.trans.eQTLs, 12
- bindmaf, 13
- cgff2dt, 14
- cisAssoc, 16
- cisSeqByCluster, 20
- collectBest, 22
- concatCis, 24
- eqBox, 25
- eqsens_dt, 26
- eqtlTests, 28
- eqtlTests.me, 29
- getCisMap, 34
- gffprocess, 35
- gwSnpTests, 36
- pifdr, 38
- qqhex, 39
- richNull, 41
- sampsInVCF, 42
- scoresCis, 42
- sensanal, 43
- simpleTiling, 45
- snpLocsDefault, 46
- transeqByCluster, 49
- transScores, 50
- transTab, 53
- vcf2sm, 53

**Topic package**
- GGtools-package, 2
- [,eqtlTestsManager,ANY,ANY,ANY-method (eqtlTestsManager-class), 31

add878 (All.cis), 4
addgwhit (All.cis), 4
All.cis, 4
All.cis.eQTLs (best.cis.eQTLs), 8
allSigCis-class (best.cis.eQTLs), 8
Annotated, 21
appraise, 5
approx, 39
b1, 7
b2 (b1), 7
batchsize (TransConfig-class), 48
batchsize, TransConfig-method (TransConfig-class), 48
batchsize<- (TransConfig-class), 48
batchsize<-, TransConfig, integer-method (TransConfig-class), 48
best.cis.eQTLs, 8, 43
best.trans.eQTLs, 12
bindgwava (appraise), 5
bindmaf, 13
binnedQQ (qqhex), 39
binnedQQ (qqhex), 39
buildConfList (CisConfig-class), 17
calfig (EqAppr-class), 24
calfig, EqAppr, character-method (EqAppr-class), 24
cgff2dt, 14, 27, 40
chrFilter (All.cis), 4
chrnames (CisConfig-class), 17
chrnames, CisConfig-method (CisConfig-class), 17
chrnames<- (CisConfig-class), 17
chrnames<-, CisConfig, character-method (CisConfig-class), 17
chromsUsed (best.cis.eQTLs), 8
chromsUsed, mcwBestCis-method
  (best.cis.eQTLs), 8
cis.FDR.filter.best (collectBest), 22
cis.FDR.filter.SNPcentric
  (collectBest), 22
cisAssoc, 16
cisConfig, 48
ciseqByCluster, 20, 40
cisRun-class, 21
cisScores, 26, 27
clipPCs, 20
collectBest, 22
collectFiltered (collectBest), 22
concatCis, 24
data.table, 40

EqAppr-class, 24
eqBox, 25
eqDesc (eqBox), 25
eqsens_dt, 26
eqtIEstimates (eqtlTests), 28
eqtIEstimatesManager-class
  (eqtlITestsManager-class), 31
eqtITests, 28, 31, 32, 31, 51, 52
eqtITests.me, 29
eqtITestsManager-class, 31
estimates (CisConfig-class), 17
estimates, CisConfig-method
  (CisConfig-class), 17
estimates<- (CisConfig-class), 17
estimates<-, CisConfig, logical-method
  (CisConfig-class), 17
ex, 32
excludeRadius (CisConfig-class), 17
excludeRadius, CisConfig-method
  (CisConfig-class), 17
excludeRadius<- (CisConfig-class), 17
excludeRadius<-, CisConfig, integer-method
  (CisConfig-class), 17
excludeRadius<-, CisConfig, integerOrNULL-method
  (CisConfig-class), 17
exFilter (CisConfig-class), 17
exFilter, CisConfig-method
  (CisConfig-class), 17
exFilter<- (CisConfig-class), 17
exFilter<-, CisConfig, function-method
  (CisConfig-class), 17
externalize, 18
extraProps (CisConfig-class), 17
extraProps, CisConfig-method
  (CisConfig-class), 17
extraProps<- (CisConfig-class), 17
extraProps<-, CisConfig, function-method
  (CisConfig-class), 17
fdr (best.cis.eQTLs), 8
ffrowapply, 12, 52
filtgen.maf.dist (eqsens_dt), 26
folderStem (CisConfig-class), 17
folderStem, CisConfig-method
  (CisConfig-class), 17
folderStem<- (CisConfig-class), 17
folderStem<-, CisConfig, character-method
  (CisConfig-class), 17
fullreport (best.cis.eQTLs), 8
fullreport, mcwBestCis, character-method
  (best.cis.eQTLs), 8
fullreport, mcwBestCis, missing-method
  (best.cis.eQTLs), 8
gbufsize (TransConfig-class), 48
gbufsize, TransConfig-method
  (TransConfig-class), 48
gbufsize<- (TransConfig-class), 48
gbufsize<-, TransConfig, integer-method
  (TransConfig-class), 48
gchrpref (CisConfig-class), 17
gchrpref, CisConfig-method
  (CisConfig-class), 17
gchrpref<- (CisConfig-class), 17
gchrpref<-, CisConfig, character-method
  (CisConfig-class), 17
geneannopk (CisConfig-class), 17
geneannopk, CisConfig-method
  (CisConfig-class), 17
geneannopk<- (CisConfig-class), 17
geneannopk<-, CisConfig, character-method
  (CisConfig-class), 17
geneApply (CisConfig-class), 17
geneApply, CisConfig-method
  (CisConfig-class), 17
geneApply< (CisConfig-class), 17
geneApply<-, CisConfig, function-method
  (CisConfig-class), 17
geneIndcol (transManager-class), 50
geneNames (transManager-class), 50
genesym, 36
genome (CisConfig-class), 17
genome, CisConfig-method
  (CisConfig-class), 17
genome<- (CisConfig-class), 17
INDEX

transScores, 50
transTab, 50, 53
transTab, transManager, character-method
  (transTab), 53
transTab, transManager, missing-method
  (transTab), 53

update_fdr_filt (eqsens_dt), 26

vcf2sm, 53
vcf2sm, TabixFile, GRanges, integer-method
  (vcf2sm), 53
Vector, 21