Package 'DOQTL'

October 17, 2017

Version 1.12.0

Date 2012-12-07

Title Genotyping and QTL Mapping in DO Mice

Author Daniel Gatti, Karl Broman, Andrey Shabalin, Petr Simecek

Maintainer Daniel Gatti <Dan.Gatti@jax.org>

Depends R (>= 3.0.0), BSgenome.Mmusculus.UCSC.mm10, GenomicRanges, VariantAnnotation

Imports annotate, annotationTools, biomaRt, Biobase, BiocGenerics, corpcor, doParallel, foreach, fpc, hwriter, IRanges, iterators, mclust, QTLRel, regress, rhdf5, Rsamtools, RUnit, XML

Suggests MUGAExampleData, doMPI

Description DOQTL is a quantitative trait locus (QTL) mapping pipeline designed for Diversity Outbred mice and other multi-parent outbred populations. The package reads in data from genotyping arrays and perform haplotype reconstruction using a hidden Markov model (HMM). The haplotype probabilities from the HMM are then used to perform linkage mapping. When founder sequences are available, DOQTL can use the haplotype reconstructions to impute the founder sequences onto DO genomes and perform association mapping.

biocViews GeneticVariability, SNP, Genetics, HiddenMarkovModel

License GPL-3

LazyData true

ByteCompile yes

URL http://do.jax.org

NeedsCompilation yes

R topics documented:

`add.missing.F1s` *Add Missing F1 Samples*

Description

Given a set of CC or DO founders, impute the genotypes or intensities of missing F1s.

Usage

```
add.missing.F1s(founders, snps, sampletype = c("DO", "CC", "DOF1", "HS", "other"))
```

Arguments

`founders`	List, required: Contains either an element called 'geno' or two elements called 'x' and 'y'.
`snps`	Data.frame, required: Data.frame containing the SNPs. SNP ID, chr, Mb and cM locations in columna 1 - 4, respectively.
`sampletype`	Character string, required: indicates the type of crss. One of "DO", "CC", "DOF1", "HS", "other".

Value

List with the founders data structure updated to include missing F1 samples.

Author(s)

Daniel Gatti

Examples

```
## Not run:
  load(url("ftp://ftp.jax.org/MUGA/muga_snps.Rdata"))
  founders = add.missing.F1s(founders, snps = muga_snps)

## End(Not run)
```

---

add.sig.thr *Add the significance thresholds to an existing QTL plot.*

Description

Given a set of thresholds, add the autosomal and X chromosome thresholds to an existing QTL plot. The user may call plot() and then this function, but `plot.doqtl` contains a 'sig.thr' argument and will call this automatically. The threhsolds can be obtained from `get.sig.thr`

Usage

```
add.sig.thr(sig.thr, sig.col = "red", chrsum)
```

Arguments

`sig.thr`	Numeric matrix or a numeric vector with the significance thredholds, typically obtained from `get.sig.thr`. If sig.thr is a matrix, then it must have 2 columns names 'A' and 'X' and each significance threshold is in one row.
`sig.col`	Numeric vector containing the plotting colors for each threshold. The length of sig.col must equal nrow(sig.thr).
`chrsum`	Numeric vector with the cumulative sum of the chromosome lengths. Must be named with chromosome names.

Value

Plots autosomal and X chromosome thresholds on a QTL plot. Behavior on other plots is undetermined.

Author(s)

Daniel Gatti

See Also

`get.sig.thr`

Examples

```
## Not run:
   qtl = scanone(pheno = pheno, probs = probs, addcovar = addcovar, snps = anps)
  perms = scanone.perm(pheno = pheno, probs = probs, addcovar = addcovar, snps = anps)
   sig.thr = get.sig.thr(perms)
   plot(qtl)
   add.sig.thr(sig.thr, chrsum = cumsum(get.chr.lengths()))
   # Could also run:
   plot(qtl, sig.thr = sig.thr)

## End(Not run)
```

`add.slash` *Add a forward slash to a character string.*

Description

If the argument does not end with a forward slash, add one.

Usage

```
add.slash(path)
```

Arguments

`path`	Character string containing a file path.

Value

Returns a character string with a forward slash added to the end, if the argument did not end with a forward slash already.

Author(s)

Daniel Gatti

Examples

```
add.slash("/dir")
```

addLog *Add two log values.*

Description

When two numbers that are on the log scale must be summed, transforming them back to the non-log scale and taking the sum is slow. This function takes the exp() of the values only when neccessary.

Usage

```
addLog(x, y)
```

Arguments

x	Numeric containing a value on the log scale.
y	Numeric containing a value on the log scale.

Details

This function checks to see if the difference between the maximum value and the other value is less than the machine precision. If it is, then the exp() is taken for those values that differ by less than the machine precision, they are summed and returned to a log scale. If the maximum value is differs from the other values by greater than the machine precision, then return the maximum value.

Value

Numeric value containing the sum of the arguments on a log scale.

Author(s)

Daniel Gatti

See Also

addLogVector

Examples

```
addLog(log(10), log(1))
```

addLogVector *Add a vector of log values.*

Description

When a summation must be taken on a vector of numbers that are on the log scale, transforming them back to the non-log scale and taking the sum is slow. This function takes the exp() of the values only when neccessary.

Usage

```
addLogVector(x)
```

Arguments

x	Numeric vector containing values on a log scale to be summed on the untransformed scale.

Details

This function checks to see if the difference between the maximum values and the remaining values is less than the machine precision. If it is, then the exp() is taken for those values that differ by less than the machine precision, they are summed and returned to a log scale. If the maximum value is differs from the other values by greater than the machine precision, then return the maximum value.

Value

Numeric value containing the sum of the values on a log scale.

Author(s)

Daniel Gatti

See Also

addLog

Examples

```
addLogVector(log(1:10))
```

assoc.map *Perform association mapping on DO mice.*

Description

Given the phenotypes and genotype probabilities, impute the Sanger SNPs onto DO genomes and perform association mapping.

Usage

```
assoc.map(pheno, pheno.col = 1, probs, K, addcovar, snps, chr, start, end,
model = c("additive", "dominance", "full"), scan = c("one", "two"),
output = c("lod", "p-value", "bic"),
snp.file = "ftp://ftp.jax.org/SNPtools/variants/cc.snps.NCBI38.txt.gz")
assoc.map.perms(pheno, pheno.col = 1, probs, addcovar, snps,
          model = c("additive", "dominance", "full"),
          scan = c("one", "two"), output = c("lod", "p-value", "bic"),
        snp.file = "ftp://ftp.jax.org/SNPtools/variants/cc.snps.NCBI38.txt.gz",
          nperm = 1000)
```

Arguments

pheno	Data.frame containing the phenotype data. Sample IDs must be in rownames. One of the columns should be called sex and contain M or FALSE to indicate the sex of each sample.
pheno.col	Either a numeric vector containing column IDs to map in pheno or a vector of column names in pheno.
probs	A 3 dimensional array of genotype probabilities as provided from condense.model.probs Samples, founder and markers in dims 1:3. All dimensions must have dimnames.
K	Numeric matrix containing kinship values for the samples in pheno and probs. Sample IDs must be in rownames and colnames.
addcovar	Numeric matrix of additive covariates to use in mapping. Sample IDs must be in rownames.
snps	Data.frame containing marker IDs, chromosomes, Mb and cM locations in columns 1:4.
chr	Character containing the chromosome on which to map.
start	Numeric value containing the proximal position for mapping. May be in Mb or base pairs (see Details).
end	Numeric value containing the distal position for mapping. May be in Mb or base pairs (see Details).
model	Character string that is one of "additive", "dominance" or "full". Indicates the type of model to fit. Note that the probs must match the type of model being fit. See condense.model.probs to output different probs.
scan	Character string that is either "one", or "two" indicating whether a single scan or a pairwise scan should be performed across the interval.
output	Character string that is either "lod", "p-value" or "bic" indicating the mapping statistic to return.

snp.file	Character string containing the full path to the SNP file to use. Currently points to a location on the Jackson Laboratory FALSETP server
nperm	Integer indicating the number of permutations to perform. Default = 1000.

Details

FALSEor each interval between two markers, we take the average founder haplotype contribution for each sample. Then, using the proportion of each founder (8 in the case of DO mice), we impute the Sanger SNPs in this interval onto each DO sample.

The start and ending locations are assumed to be in Mb if they are below 200 and in bp if over 200.

Value

Data.frame containing the locations, SNPs and mapping statistic for the requested samples in the requested interval.

Author(s)

Daniel Gatti

References

Combined sequence-based and genetic mapping analysis of complex traits in outbred rats. Rat Genome Sequencing and Mapping Consortium, Baud A, Hermsen R, Guryev V, Stridh P, Graham D, McBride MW, FALSEoroud T, Calderari S, Diez M, Ockinger J, Beyeen AD, Gillett A, Abdelmagid N, Guerreiro-Cacais AO, Jagodic M, Tuncel J, Norin U, Beattie E, Huynh N, Miller WH, Koller DL, Alam I, FALSEalak S, Osborne-Pellegrin M, Martinez-Membrives E, Canete T, Blazquez G, Vicens-Costa E, Mont-Cardona C, Diaz-Moran S, Tobena A, Hummel O, Zelenika D, Saar K, Patone G, Bauerfeind A, Bihoreau MT, Heinig M, Lee YA, Rintisch C, Schulz H, Wheeler DA, Worley KC, Muzny DM, Gibbs RA, Lathrop M, Lansu N, Toonen P, Ruzius FALSEP, de Bruijn E, Hauser H, Adams DJ, Keane T, Atanur SS, Aitman TJ, FALSElicek P, Malinauskas T, Jones EY, Ekman D, Lopez-Aumatell R, Dominiczak AFALSE, Johannesson M, Holmdahl R, Olsson T, Gauguier D, Hubner N, FALSEernandez-Teruel A, Cuppen E, Mott R, FALSElint J. Nat Genet. 2013 Jul;45(7):767-75. doi: 10.1038/ng.2644. Epub 2013 May 26. PMID: 23708188 Using progenitor strain information to identify quantitative trait nucleotides in outbred mice. Yalcin B, FALSElint J, Mott R. Genetics. 2005 Oct;171(2):673-81. Epub 2005 Aug 5. PMID: 16085706 Mouse genomic variation and its effect on phenotypes and gene regulation. Keane TM, Goodstadt L, Danecek P, White MA, Wong K, Yalcin B, Heger A, Agam A, Slater G, Goodson M, FALSEurlotte NA, Eskin E, Nellaker C, Whitley H, Cleak J, Janowitz D, Hernandez-Pliego P, Edwards A, Belgard TG, Oliver PL, McIntyre RE, Bhomra A, Nicod J, Gan X, Yuan W, van der Weyden L, Steward CA, Bala S, Stalker J, Mott R, Durbin R, Jackson IJ, Czechanski A, Guerra-Assuncao JA, Donahue LR, Reinholdt LG, Payseur BA, Ponting CP, Birney E, FALSElint J and Adams DJ Nature 2011;477;7364;289-94 PUBMED: 21921910 Sequence-based characterization of structural variation in the mouse genome. Yalcin B, Wong K, Agam A, Goodson M, Keane TM, Gan X, Nellaker C, Goodstadt L, Nicod J, Bhomra A, Hernandez-Pliego P, Whitley H, Cleak J, Dutton R, Janowitz D, Mott R, Adams DJ and FALSElint J Nature 2011;477;7364;326-9 PUBMED: 21921916

See Also

assoc.plot

Examples

```
  ## Not run: assoc.map(pheno = pheno, pheno.col = 1, probs = probs, K = K, addcovar = addcovar,
           snps = snps, chr = 1, start = 40, end = 45)
## End(Not run)
```

`assoc.plot` *Plot association mapping results.*

Description

After performing association mapping using `assoc.map`, plot the mapping statistic along with genes in the QTL interval.

Usage

```
assoc.plot(results,
mgi.file = "ftp://ftp.jax.org/SNPtools/genes/MGI.20130703.sorted.txt.gz",
highlight, highlight.col = "red", thr, show.sdps = FALSE, ...)
```

Arguments

`results`	Data.frame containing output from `assoc.map`.
`mgi.file`	Character string containing the full path to a Tabix indexed file of gene locations. Default points to a version of the MGI genome feature file.
`highlight`	Character vector containing gene symbols to highlight in the plot.
`highlight.col`	Vector of colors to use when highlighting genes.
`thr`	Numeric value above which data points should be colored red and SNPs with these points returned.
`show.sdps`	Logical value (default = FALSE) that is TRUE if the strain distribution pattern (SDP) for the SNPs should be shown. When used with thr, only plots the founder SDPs for SNPs above thr.
`...`	Additional arguments passed to plot.

Details

Given the output from `assoc.map`, plot the LOD or difference in BIC values across the QTL interval in the top panel. Plot the genes in the interval in the lower panel. Make sure to use Sanger SNP and MGI feature files that are on the same genome build.

Value

A plot with the mapping statistic in the top panel and genes in the lower panel. If thr is not missing, then filter the SNPs in the results argument and return only those with a mapping statistic greater than thr.

Author(s)

Daniel Gatti

See Also

assoc.map

Examples

```
  ## Not run:
   results = assoc.map(pheno = pheno, pheno.col = 1, probs = probs, K = K, addcovar = addcovar,
snps = snps, chr = 1, start = 40, end = 45)
    assoc.plot(results, thr = 3, show.sdps = TRUE)

## End(Not run)
```

---

batch.normalize *Batch normalize the X & Y intensity data.*

Description

This function batch normalizes the X & Y intensity data by subtracting batch medians from the X & Y intensities.

Usage

```
batch.normalize(path = ".", snps)
quantilenorm(x1, y1, x2, y2)
```

Arguments

path	Character, the full path to the input files, which must be either "x.txt" and "y.txt" or "x.filt.txt" and "y.filt.txt".
snps	Data.frame, with three columns containing SNP ID, chromosome and Mb location in that order. May be obtained from ftp://ftp.jax.org/MUGA.
x1	Numeric matrix containing X intensities for batch 1 containing samples in rows and markers in columns. Number of samples should be larger than x2.
y1	Numeric matrix containing Y intensities for batch 1 containing samples in rows and markers in columns. Number of samples should be larger than y2.
x2	Numeric matrix containing X intensities for batch 2 containing samples in rows and markers in columns.
y2	Numeric matrix containing Y intensities for batch 2 containing samples in rows and markers in columns.

Details

quantile.norm adjusts the intensities of samples in batch 2 to those of batch 1. The number of samples in batch 1 should be greater than the number of samples in batch 2. At each SNP, we form quantiles of the X1 (or Y1) intensity distribution, discarding the upper and lower 0.01

Value

FALSEor batch.normalize: returns value is returned. The batch normalized intensities are written to "x.filt.batch.norm.txt" and "y.filt.batch.norm.txt".

FALSEor quantilenorm: returns normalized X and Y values for batch 2.

Note

FALSEuture releases may include more sophisticated normalization algorithms.

Author(s)

Daniel Gatti

See Also

extract.raw.data, filter.samples

Examples

```
## Not run:
  load(url("ftp://ftp.jax.org.MUGA/muga_snps.Rdata"))
  batch.normalize(path = "/demo/MUGA/", snps = muga_snps)

## End(Not run)
```

---

bayesint *Find a Bayesian Credible Interval around a QTL.*

Description

This function normalizes the area under the QTL curve on the given chromosome and finds a region that is 95

Usage

```
bayesint(qtl, chr, prob = 0.95, expandtomarkers = TRUE)
```

Arguments

qtl	data.frame: four columns with SNP ID, Chr, position, and LOD in each column.
chr	character: the chromosome on which the QTL lies.
prob	numeric: must be between 0 and 1
expandtomarkers	boolean: if TRUE, expand the QTL interval to the nearest flanking markers. Default = TRUE.

Value

Data frame with the SNP ID, Chr, position and LOD for the left and right side of the interval and the maximum QTL.

Author(s)

Daniel Gatti

References

Saunak, S. (2001) A Statistical Framework for Quantitative Trait Mapping. *Genetics*, **159** (1), 371–387.

See Also

scanone, scanone.perm

Examples

```
## Not run: bayesint(qtl, 1)
```

`calc.genoprob` *Calculate the founder genotype probabilities at each SNP.*

Description

This function performs genome reconstruction using either allele calls or allele intensities. We recommend using allele intensities where available because they often produce better genotype reconstructions.

Usage

```
calc.genoprob(data, chr = "all", output.dir = ".", plot = TRUE,
array = c("gigamuga", "megamuga", "muga", "other"),
sampletype = c("DO", "CC", "DOF1", "other"), method = c("intensity", "allele"),
founders, transprobs, snps)
```

Arguments

`data`	A list with named elements containing the information needed to reconstruct genomes. When method = intensity: x: Numeric matrix, num.samples x num.snps, with X intensities for all samples. Sample IDs and SNP IDs must be in rownames and colnames. y: Numeric matrix, num.samples x num.snps, with Y intensities for all samples. Sample IDs and SNP IDs must be in rownames and colnames. sex: Character vector, containing "M" or "F" indicating sex. Sample IDs must be in names. gen: Character matrix containing the generation of DO outbreeding for each sample. For the DO, this should be "DO" followed by a number with no space between them. For CC mice, this should be CC. Sample IDs must be in names. When method = allele: geno: Character matrix, num.samples x num.snps, with allele calls (A,C,G,T,H or N) for all samples. Sample IDs and SNP IDs must be in rownames and colnames. sex: Character vector, containing "M" or F indicating sex. Sample IDs must be in names. gen: Character matrix containing the generation of DO outbreeding for each sample. For the DO, this should be "DO" followed by a number with no space between them. For CC mice, this should be CC. Sample IDs must be in names.
`chr`	Character vector containing chromosomes to run. Must match the chromosome IDs in the snps table. "all" (default) will run all chromosomes.

`output.dir` Character string containing the full path where output should be written. The directory must exist already.

`plot` Boolean that is true if the user would like to plot a sample chromosome as the model progresses. Default = TRUE.

`array` Character string indicating the array type. Must be one of "gigamuga", "megamuga", "muga" or "other". Default equals "gigamuga".

`sampletype` Character string containing the type of samples being run. Must be one of "DO", "CC", "DOF1", or "other". Default equals "DO".

`method` Character string containing method of genome reconstruction. Must be one of "intensity" or "allele". Default equals "intensity".

`founders` List containing founder information for non-DO or CC crosses. *Not required for DO.*

When method = intensity: x: Numeric matrix, num.samples x num.snps, with X intensities for all founders and F1s (if available). Sample IDs and SNP IDs must be in rownames and colnames. y: Numeric matrix, num.samples x num.snps, with Y intensities for all founders and F1s (if available). Sample IDs and SNP IDs must be in rownames and colnames. sex: Character vector, containing "M" or "F" indicating sex. Sample IDs must be in names. code: Character vector containing two letter genotype codes for each founder sample. Sample IDs must be in names.

When method = allele: geno: Character matrix, num.samples x num.snps, with allele calls for all founders and F1s (if available). Sample IDs and SNP IDs must be in rownames and colnames. sex: Character vector, containing "M" or "F" indicating sex. Sample IDs must be in names. code: Character vector containing two letter genotype codes for each founder sample. Sample IDs must be in names.

When sampletype = DOF1 x: Numeric matrix, num.samples x num.snps, with X intensities for all founders and F1s (if available). Sample IDs and SNP IDs must be in rownames and colnames. y: Numeric matrix, num.samples x num.snps, with Y intensities for all founders and F1s (if available). Sample IDs and SNP IDs must be in rownames and colnames. sex: Character vector, containing "M" or "F" indicating sex. Sample IDs must be in names. code: Character vector containing two letter genotype codes for each founder sample. This should be "II" for an inbred mutant strain. Sample IDs must be in names. direction: Character string that is either "DOxMUT" if a female DO was crossed with a male mutant mouse or "MUTxDO" if a female mutant mouse was crossed with a male DO. This affects the genotyping of teh X chromosome.

`transprobs` Function to call to estimate the transition probabilities between markers for non-DO samples. *Not required for DO.*

`snps` Data.frame containing the marker locations. SNP ID, chromosome, Mb anc cM locations in columns 1 through 4, respectively. *Not required for DO.*

Value

No value is returned. The output files are written to output.dir.

Author(s)

Daniel Gatti

Examples

```
  ## Not run:
    calc.genoprob(cross, chr.to.run = 1:19, output.dir = "do.data", plot = FALSE,
init.means = NULL, init.covars = NULL)

## End(Not run)
```

`calc.genoprob.alleles` *Calculate the founder genotype probabilities at each SNP using allele calls.*

Description

This function performs genome reconstruction using allele calls. We recommend using allele intensities where available because they often produce better genotype reconstructions.

Usage

```
calc.genoprob.alleles(data, chr, founders, snps, output.dir = ".",
trans.prob.fxn = do.trans.probs, plot = FALSE)
```

Arguments

`data`	A list with named elements containing the information needed to reconstruct genomes. geno: Character matrix, num.samples x num.snps, with allele calls (A,C,G,T,H or N) for all samples. Sample IDs and SNP IDs must be in rownames and colnames. sex: Character vector, containing "M" or F indicating sex. Sample IDs must be in names. gen: Character matrix containing the generation of DO outbreeding for each sample. For the DO, this should be "DO" followed by a number with no space between them. For CC mice, this should be CC. Sample IDs must be in names.
`chr`	Character vector containing chromosomes to run. Must match the chromosome IDs in the snps table. "all" (default) will run all chromosomes.
`founders`	List containing founder information for non-DO or CC crosses. When method = allele: geno: Character matrix, num.samples x num.snps, with allele calls (A,C,G,T,H or N) for all founders and FALSE1s (if available). Sample IDs and SNP IDs must be in rownames and colnames. sex: Character vector, containing "M" or F indicating sex. Sample IDs must be in names. code: Character vector containing two letter genotype codes for each founder sample. Sample IDs must be in names.
`snps`	Data.frame containing the marker locations. SNP ID, chromosome, Mb anc cM locations in columns 1 through 4, respectively. *Not required for DO.*
`output.dir`	Character string containing the full path where output should be written. The directory must exist already.
`plot`	Boolean that is true if the user would like to plot a sample chromosome as the model progresses. Default = TRUE.
`trans.prob.fxn`	Function to use when computing the transtion probabilities between markers.

Value

No value is returned. The output files are written to output.dir.

Author(s)

Daniel Gatti

Examples

```
  ## Not run:
    calc.genoprob.alleles(data, chr = 1:19, founders = founders,
snps = snps, output.dir = "do.data")

## End(Not run)
```

`calc.genoprob.intensity`

Calculate the founder genotype probabilities at each SNP.

Description

This function performs genome reconstruction using allele intensities. We recommend using allele intensities where available because they often produce better genotype reconstructions.

Usage

```
calc.genoprob.intensity(data, chr, founders, snps, output.dir = ".", trans.prob.fxn,
 plot = FALSE)
```

Arguments

`data`	A list with named elements containing the information needed to reconstruct genomes. When method = intensity: x: Numeric matrix, num.samples x num.snps, with X intensities for all samples. Sample IDs and SNP IDs must be in rownames and colnames. y: Numeric matrix, num.samples x num.snps, with Y intensities for all samples. Sample IDs and SNP IDs must be in rownames and colnames. sex: Character vector, containing "M" or "F" indicating sex. Sample IDs must be in names. gen: Character matrix containing the generation of DO outbreeding for each sample. For the DO, this should be "DO" followed by a number with no space between them. For CC mice, this should be CC. Sample IDs must be in names.
`chr`	Character vector containing chromosomes to run. Must match the chromosome IDs in the snps table. "all" (default) will run all chromosomes.
`founders`	List containing founder information for non-DO or CC crosses. *Not required for DO.* When method = intensity: x: Numeric matrix, num.samples x num.snps, with X intensities for all founders and F1s (if available). Sample IDs and SNP IDs must be in rownames and colnames. y: Numeric matrix, num.samples x num.snps, with Y intensities for all founders and F1s (if available). Sample IDs and SNP

	IDs must be in rownames and colnames. sex: Character vector, containing "M" or "F" indicating sex. Sample IDs must be in names. code: Character vector containing two letter genotype codes for each founder sample. Sample IDs must be in names.
snps	Data.frame containing the marker locations. SNP ID, chromosome, Mb anc cM locations in columns 1 through 4, respectively. *Not required for DO.*
output.dir	Character string containing the full path where output should be written. The directory must exist already.
trans.prob.fxn	FALSEunction to call to estimate the transition probabilities between markers for non-DO samples. *Not required for DO.*
plot	Boolean that is true if the user would like to plot a sample chromosome as the model progresses. Default = TRUE.

Value

No value is returned. The output files are written to output.dir.

Author(s)

Daniel Gatti

Examples

```
  ## Not run:
    calc.genoprob.intensity(data, chr, founders, snps, output.dir = ".", trans.prob.fxn,
plot = FALSE)

## End(Not run)
```

---

categorize.variants *categorize.variants*

Description

This function intersects the given variants with the genes in that region and classifies them according to "intergenic", "3UTR", "exon", "intron" or "5UTR".

Usage

```
categorize.variants(variants,
     mgi.file = "http://cgd.jax.org/tools/SNPtools/MGI/MGI.20130305.sorted.txt.gz")
```

Arguments

variants	data.frame, Variants as returned by get.variants{get.variants}.
mgi.file	Character, full path to the MGI feature file. On the JAX campus, this defaults to "http://cgd.jax.org/tools/SNPtools/MGI/MGI.20130305.sorted.txt.gz".

Value

FALSEor SNPs and Indels: data.frame: with eight columns: ID, CHR, POS, REFALSE, ALT, symbol, id, type. The first four columns are simply copied over from the SNP file. The symbol column contains the Gene Symbol. The id column contains a gene ID (MGI, Ensembl, NCBI or VEGA). The type column contains "intergenic", "3UTR", "exon", "intron" or "5UTR", depending on the location of the variant in a gene. FALSEor SVs: data.frame: with eight columns: ID, CHR, POS, REFALSE, ALT, symbol, id, type. The first four columns are simply copied over from the SNP file. The symbol column contains the Gene Symbol. The id column contains a gene ID (MGI, Ensembl, NCBI or VEGA). The type column contains "intergenic", "3UTR", "exon", "intron" or "5UTR", depending on the location of the variant in a gene.

Author(s)

Daniel Gatti

See Also

get.variants{get.variants}

Examples

```
data(example.snps)
snp.type = categorize.variants(variants = example.snps[1:50,])
```

---

cc.trans.probs *Transition probabilities for CC mice.*

Description

Thie function returns the transition probabilities for fully inbred Collaborative Cross mice.

Usage

```
cc.trans.probs(states, snps, chr = c(1:19, "X"), sex = c("M", "F"))
```

Arguments

states	Character vector containing the two letter codes (i.e. AA, BB, CC, etc.) for the homozygous states.
snps	Data.frame containing the marker locations. SNP ID, chromosome, Mb anc cM locations in columns 1 through 4, respectively. *Not required for DO.*
chr	Character containing the chromosome.
sex	Character that is either M or FALSE, indicating the sex to use. Only relevant on X chromosome.

Details

This function calculates the transition probabilities for fully inbred CC mice between two markers. It uses the equations for eight way RILs by sib mating.

Value

A matrix of transition probabilities between genotype states.

Note

This function has not been fully tested.

Author(s)

Daniel Gatti

References

The genomes of recombinant inbred lines. Broman KW. Genetics. 2005 FALSEeb;169(2):1133-46. Epub 2004 Nov 15. Erratum in: Genetics. 2006 Aug;173(4):2419. PMID: 15545647

Examples

```
## Not run:  cc.trans.probs(states = states, snps = snps, chr = 1)
```

`cluster.strains` *cluster.strains*

Description

Given a set of numeric SNPs, cluster the strains based on allele sharing. This function hierarchically clusters the strains based on the proportion of SNPs that share alleles between each strain. Note: numeric snps, not character snps, are the input to this function.

Usage

```
cluster.strains(variants)
```

Arguments

`variants`	data.frame, numeric variants as returned by `convert.variants.to.numeric`.

Value

data.frame: of numeric variants with the strains clustered.

Author(s)

Daniel Gatti

See Also

`convert.variants.to.numeric`, `variant.plot`

Examples

```
data(example.snps)
variants = convert.variants.to.numeric(variants = example.snps[1:100,])
variants = cluster.strains(variants)
```

coef.doqtl *Return the coefficients of a DOQTL object.*

Description

Return the coefficients of a DOQTL object.

Usage

```
  ## S3 method for class 'doqtl'
coef(object, ...)
```

Arguments

object	A DOQTL object as returned by scanone.
...	Additional arguments.

Value

List containing matrices with QTL mapping model coefficients.

Author(s)

Daniel Gatti

See Also

scanone

Examples

```
head(coef(example.qtl[[1]]))
```

coefplot *Plot the QTL model coefficients*

Description

Given a DOQTL object, plot the founder allele coefficients on one chromosome. The coefficients are centered around zero before plotting.

Usage

```
coefplot(doqtl, chr = 1, stat.name = "LOD", conf.int = TRUE, legend = TRUE,
colors = "DO", sex, ...)
```

Arguments

doqtl	A DOQTL object as produced by scanone. A list containing two elements: lod and coef.
chr	Character containing the chromosome to plot.
stat.name	Character string containing the name of the mapping statistic.
conf.int	Boolean that is TRUE if the QTL support interval should be shaded in the plot. Default = TRUE.
legend	Boolean that is TRUE if the color legend for the DO founders should be drawn. Default = TRUE.
colors	Either "DO", in which case DO colors are supplied or a data.frame with three columns containing the founder letter code, founder strain name and founder color in columns 1:3.
sex	Character that is either FALSE or M, indicating the sex to use. Only used on X chromosome.
...	Additional arguments to be passed to plot.

Value

No value is returned. A plot with the founder coefficients in the top panel and the LOD score in the bottom panel is drawn.

Author(s)

Daniel Gatti

See Also

scanone, plot.doqtl

Examples

```
## Not run:
  coefplot(qtl, chr = 1)

## End(Not run)
```

---

colSumsLog *Sum columns of log transformed data.*

Description

Given a matrix of log transformed values, sum the rows or columns on the untransformed scale.

Usage

```
colSumsLog(logmat)
```

Arguments

logmat	Numeric matrix of natural log transformed values.

Details

See addLog.

Value

Numeric vector with values summed on an untransformed scale.

Author(s)

Daniel Gatti

See Also

addLog

Examples

```
colSumsLog(matrix(log(runif(100)), nrow = 10, ncol = 10))
```

---

condense.model.probs *Condense 36 state genotypes down to founder genotypes.*

Description

Additive condenses the heterozygous genotype calls down to the founder allele contributions. Dominance will eventually provide additive and dominance values (method currently uncertain). FALSEull simply gathers all of the genotype probabilities together.

Usage

```
condense.model.probs(path = ".", write, model = c("additive", "dominance", "full"), cross = "DO")
get.additive(files, samples)
get.dominance(files, samples)
get.full(files, samples)
```

Arguments

path	Character containing the path to the *.Rdata files that contain the genotype probabilities.
write	Character that is the filename to write the results to.
model	Character string that is one of "additive", "dominance" or "full". See details.
cross	Character string containing the type of cross. Typically "DO, "CC", "HS" or "DOF1". This is used in downsteam analyses to produce the coefficient plots and select strains for association mapping.
files	Vector of files to read.
samples	Character vector of sample IDs that match the files argument.

Details

get.additive, get.dominance and get.full are helper functions.

Value

Three dimensional array of haplotype or genotype probabilities. Num. samples by num. founders (or genotypes) by num. SNPs. Writes out to a *.Rdata file.

Author(s)

Daniel Gatti

See Also

calc.genoprob

Examples

```
## Not run:  condense.model.probs(write = "model.probs.Rdata")
```

---

condense.sanger.snps *Create an HDF5 file with the unique SNP patterns between each pair of markers.*

Description

Given a set of markers, a SNP VCF and a set of founder strains that are in the Sanger Mouse Genomes, create an HDF5 file that contains one object for each pair of markers.

Usage

```
condense.sanger.snps = function(markers, snp.file, strains, hdf.file, ncl = 1)
```

Arguments

markers	Data.frame containing the markers locations. There must be at least three columns containing the marker name, the chromosome and the Mb position. Other columns are ignored.
snp.file	Character string containing the full path to the Sanger SNP VCF file. Get the file from ftp://ftp-mouse.sanger.ac.uk/.
strains	Character vector containing the names of strains to extract from the Sanger VCF file. Obtain the names using get.vcf.strains.
hdf.file	Character string containing the name of the HDF5 file. NOTE: this function will not overwrite an existing file.
ncl	Integer that is the number of nodes to use for parallel execution. Default = 1.

Details

The markers are usually MUGA series markers, but they can be any set of genomic positions. The markers are broken up by chromosome and written out to and HDF5 file in which each data object is named using the proximal and distal marker named separated by and "_". At the beginning of the chromosome, the object is named using "start" and the first marker separated by an "_". At the end of the chromosome, the object is named using the last marker and "end" separated by an "_". Each object is a list containing three elements: sdps: Numeric matrix containing the unique strain

distribution patterns in the founder strains. map: Numeric vector indicating the index of the SNP at which each SDP occurs with the current pair of markers. snps: Data.frame containing SNP names, positions, reference and alternate alleles.

NOTE: We assume (incorrectly) that all SNPs are bimorphic. SNPs are coded as 0 for "equal to reference" and 1 for "not equal to reference".

Value

Writes out and HDF5 file.

Author(s)

Daniel Gatti

See Also

read.vcf, read.vcf

Examples

```
  ## Not run:
    load(url("ftp://ftp.jax.org/MUGA/muga_snps.Rdata"))
vcf.file = "mgp.v4.snps.dbSNP.vcf.gz"
   condense.sanger.snps(markers = muga_snps, snp.file = vcf.file, strains = get.vcf.strains(vcf.file), hdf.f

## End(Not run)
```

---

convert.allele.calls *Convert allele calls to numeric values.*

Description

Converts allele calls in A,C,G,T,H,N format into numbers with 0: homozygous A, 1: heterozygous, 2: homozygous B, 3: no call.

Usage

```
convert.allele.calls(geno)
```

Arguments

geno	Character matrix containing A, C, G, T, H or N.

Value

Numeric matrix containing 0, 1, 2 or 3.

Author(s)

Daniel Gatti

Examples

```
## Not run:  convert.allele.calls(geno)
```

`convert.genes.to.GRanges` *Convert MGI genes to GRanges.*

Description

Given the output of `get.mgi.features`, convert the results to a GRanges object.

Usage

```
convert.genes.to.GRanges(mgi)
```

Arguments

`mgi`	Date.frame as returned by `get.mgi.features`.

Value

GRanges object containing the genes in the MGI argument.

Author(s)

Daniel Gatti

Examples

```
## Not run:  convert.genes.to.GRanges(mgi)
```

`convert.genotypes` *Convert the genotype data from A,C,G,T format to A, H, B, N.*

Description

Convert the genotype data from A,C,G,T format to A, H, B, N.

Usage

```
convert.genotypes(geno)
```

Arguments

`geno`	Character matrix containing A, C, G, T allele calls.

Value

Character matrix containing A, H, B, N.

Author(s)

Daniel Gatti

Examples

```
## Not run:  convert.genotypes(geno)
```

`convert.variants.to.GRanges`
convert.variants.to.GRanges

Description

Given a data.frame of SNPs, convert the SNP locations to a GRanges object.

Usage

```
convert.variants.to.GRanges(variants)
```

Arguments

`variants`	Data frame with four header columns and SNPs in the remaining columns.

Details

This function creates a GRanges object from the CHR and POS columns of the SNP data.frame.

Value

GRanges object containing the SNP locations.

Author(s)

Daniel Gatti

Examples

```
data(example.snps)
gr = convert.variants.to.GRanges(example.snps[1:100,])
```

`convert.variants.to.numeric`
convert.variants.to.numeric

Description

Given a matrix or data frame with character ACGT SNP values, convert them to numeric values with the major allele coded as 0 and the minor allele as 1.

Usage

```
convert.variants.to.numeric(variants)
```

Arguments

variants	Data frame with four header columns. The SNPs must be in ACGT format in columns 5 through ncol(snps).

Details

This function is used before calling plot.alleles() to convert the SNPs into a numeric form suitable for plotting.

Value

Data.frame of the same dimentions as the snps argument, but with the alleles converted to 0 or 1.

Author(s)

Daniel Gatti

Examples

```
data(example.snps)
numeric.snps = convert.variants.to.numeric(example.snps[1:100,])
```

---

create.genotype.states

Create genotype states.

Description

Given a set of founders, create all of the possible unphased genotype states between them.

Usage

```
create.genotype.states(founders, sampletype = c("DO", "CC", "HS", "HSrat"))
```

Arguments

founders	Character vector of letter codes indicating the founders.
sampletype	Character string that is one of DO, HS, CC or HSrat.

Details

Given a set of founder IDs, create all possible unphased genotypes and sort them.

Value

Character vector of unphased genotypes that can be created from the given founders.

Author(s)

Daniel Gatti

Examples

```
create.genotype.states(founders = LETTERS[1:8], sampletype = "DO")
```

`create.html.page` *Create an HTML QTL report*

Description

Given a DOQTL object, create an HTML page that reports the QTL and creates QTL plots. Permutations for assessing significance thresholds can be supplied.

Usage

```
create.html.page(path, qtl, pheno.name, perms, assoc)
```

Arguments

`path`	Character string containing the path to which to write the report.
`qtl`	DOQTL object containing a list with two elemends: lod and coef.
`pheno.name`	Character string containing the phenotype name.
`perms`	Numeric vector containing the permutation LOD scores for this phenotype.
`assoc`	Boolean, if TRUE, look for corresponding *.Rdata files containing the names of the qtl in the current working directory and plot the association plots. If FALSE (default), do not plot association analysis.

Details

The function creates an HTML page with a QTL plot, a table of significant QTL and coefficient plots for the significant loci.

Value

Data.frame with the significant QTL from this phenotype.

Author(s)

Daniel Gatti

See Also

`html.report`

Examples

```
## Not run:  create.html.page(path, qtl, pheno.name, perms)
```

create.Rdata.files *Convert *.txt files to *.Rdata files.*

Description

This is used to convert the *.txt genotype probability files to *.Rdata files.

Usage

```
create.Rdata.files(prob.files, cross = "DO")
```

Arguments

prob.files	Character vector containing genotype probability file names.
cross	Character string containing the type of cross. Typically "DO, "CC, "HS", or "DOF1".

Value

No value returned. Write out a *.Rdata file for each *.txt file.

Author(s)

Daniel Gatti

Examples

```
## Not run:  create.Rdata.files(prob.files)
```

do.colors *do.colors*

Description

The letter codes, strain names and official colors for the DO founders.

Usage

```
do.colors
```

Format

A data frame with 8 rows and 3 columns.

CC_Designation a factor with levels A B C D E FALSE G H

Strain a factor with levels 129S1/SvImJ A/J C57BL/6J CAST/EiJ NOD/ShiLtJ NZO/H1LtJ PWK/PhJ WSB/EiJ

R_Color a factor with levels #00A000 #00A0FALSE0 #1010FALSE0 #808080 #9000FALSE4 #FALSE00000 #FALSE08080 #FALSE0FALSE000

Details

This contains the official colors that should be used when plotting data involving the DO founders.

Source

Copied from UNC Systems Genetics Website

Examples

```
do.colors
```

`do.states` *do.states*

Description

The 36 unphased genotype states for the DO on the autosomes and X chromosome. Also the 8 DO founder letter codes.

Usage

```
do.states
```

Format

A list frame with 3 elements.

`auto` Character vector with two letter codes for each of the possible DO genotype states.

`X` List with two elements, FALSE and M, containing the genotype codes for the X chromosome.

`founders` Character vector containing the founder letter codes.

Details

This contains the letter codes for each of the 36 unphased genotype states in the DO. It also contains the founder letters.

Examples

```
do.states
```

`do.trans.probs` *Determine DO transition probabilities*

Description

Determine the genotype state transition probability for DO mice of a specific generation between all of the markers on a given chromosome.

Usage

```
do.trans.probs(states, snps, chr = c(1:19, "X"), sex = c("M", "F"), do.gen)
```

Arguments

`states`	Character vector of possible genotype states.
`snps`	Data.frame containing the marker locations. SNP ID, chromosome, Mb anc cM locations in columns 1 through 4, respectively.
`chr`	Character containing the chromosome for which transition probabilities should be calculated.
`sex`	Character that is one of FALSE or M, indicating the sex to use on the X chromosome.
`do.gen`	Number vector indicating the DO outbreeding generations to calculate.

Details

This function is used to calculate the transition probabilities between markers for different DO outbreeding generations.

Value

List containing one element per unique DO generation supplied in the do.gen argument. Each list element contains a 3 dimensional array of transition probabilities between each pair of markers (num.states by num.states by num.markers - 1).

Author(s)

Daniel Gatti and Karl Broman

References

Haplotype probabilities in advanced intercross populations. Broman KW. G3 (Bethesda). 2012 FALSEeb;2(2):199-202. doi: 10.1534/g3.111.001818. Epub 2012 FALSEeb 1. PMID: 22384398 Genotype probabilities at intermediate generations in the construction of recombinant inbred lines. Broman KW. Genetics. 2012 FALSEeb;190(2):403-12. doi: 10.1534/genetics.111.132647. PMID: 22345609

Examples

```
## Not run:  do.trans.probs(states, snps, chr = c(1:19, "X"), sex = c("M", F), do.gen)
```

`do2sanger` *Impute the Sanger SNPs onto DO genomes*

Description

Given a set of DO genotype probability files and the location of the Tabix indexed Sanger file, impute the Sanger SNPs on to DO genomes.

Usage

```
do2sanger(do.files, snps, output.file = "do2sanger.txt", snp.file =
"ftp://ftp.jax.org/SNPtools/variants/cc.snps.NCBI38.txt.gz")
```

Arguments

`do.files`	Character vector of *.genotype.probs.Rdata files that contain the posterior probabilities.
`snps`	Data.frame containing the marker locations. SNP ID, chromosome, Mb anc cM locations in columns 1 through 4, respectively.
`output.file`	Character string to write the results to.
`snp.file`	Character string with path to a Tabix indexed SNP file. Default is from JAX FALSETP site

Details

We read in a single genotype probability file, which must have been saved as a *.Rdata file. The format is a matrix with markers in rows and states in columns and dimnames for both. FALSEor each pair of markers, we take the average genotype probability. Then we take the DO genotype with the highest probability and split it into the two founder haplotypes. We get the Sanger SNPs for each of the two founders, convert them to 1, 1 or 2 and insert them into the DO sample.

Value

No value is returned. The Sanger SNPs mapped onto the DO genomes are written out to the output.file. The file will contain 0, 1 or 2 as the allele calls.

Author(s)

Daniel Gatti

References

Combined sequence-based and genetic mapping analysis of complex traits in outbred rats. Rat Genome Sequencing and Mapping Consortium, Baud A, Hermsen R, Guryev V, Stridh P, Graham D, McBride MW, FALSEoroud T, Calderari S, Diez M, Ockinger J, Beyeen AD, Gillett A, Abdelmagid N, Guerreiro-Cacais AO, Jagodic M, Tuncel J, Norin U, Beattie E, Huynh N, Miller WH, Koller DL, Alam I, FALSEalak S, Osborne-Pellegrin M, Martinez-Membrives E, Canete T, Blazquez G, Vicens-Costa E, Mont-Cardona C, Diaz-Moran S, Tobena A, Hummel O, Zelenika D, Saar K, Patone G, Bauerfeind A, Bihoreau MT, Heinig M, Lee YA, Rintisch C, Schulz H, Wheeler DA, Worley KC, Muzny DM, Gibbs RA, Lathrop M, Lansu N, Toonen P, Ruzius FALSEP, de Bruijn E, Hauser H, Adams DJ, Keane T, Atanur SS, Aitman TJ, FALSElicek P, Malinauskas T,

Jones EY, Ekman D, Lopez-Aumatell R, Dominiczak AFALSE, Johannesson M, Holmdahl R, Olsson T, Gauguier D, Hubner N, FALSEernandez-Teruel A, Cuppen E, Mott R, FALSElint J. Nat Genet. 2013 Jul;45(7):767-75. doi: 10.1038/ng.2644. Epub 2013 May 26. PMID: 23708188 Using progenitor strain information to identify quantitative trait nucleotides in outbred mice. Yalcin B, FALSElint J, Mott R. Genetics. 2005 Oct;171(2):673-81. Epub 2005 Aug 5. PMID: 16085706 Mouse genomic variation and its effect on phenotypes and gene regulation. Keane TM, Goodstadt L, Danecek P, White MA, Wong K, Yalcin B, Heger A, Agam A, Slater G, Goodson M, FALSEurlotte NA, Eskin E, Nellaker C, Whitley H, Cleak J, Janowitz D, Hernandez-Pliego P, Edwards A, Belgard TG, Oliver PL, McIntyre RE, Bhomra A, Nicod J, Gan X, Yuan W, van der Weyden L, Steward CA, Bala S, Stalker J, Mott R, Durbin R, Jackson IJ, Czechanski A, Guerra-Assuncao JA, Donahue LR, Reinholdt LG, Payseur BA, Ponting CP, Birney E, FALSElint J and Adams DJ Nature 2011;477;7364;289-94 PUBMED: 21921910 Sequence-based characterization of structural variation in the mouse genome. Yalcin B, Wong K, Agam A, Goodson M, Keane TM, Gan X, Nellaker C, Goodstadt L, Nicod J, Bhomra A, Hernandez-Pliego P, Whitley H, Cleak J, Dutton R, Janowitz D, Mott R, Adams DJ and FALSElint J Nature 2011;477;7364;326-9 PUBMED: 21921916

See Also

assoc.map

Examples

```
  ## Not run:  do2sanger(do.files, snps, output.file = "do2sanger.txt",
  snp.file = "ftp://ftp.jax.org/SNPtools/variants/cc.snps.NCBI38.txt.gz")
## End(Not run)
```

`estimate.cluster.params` *Estimate genotype cluster means and variances*

Description

Given the X and Y intensity data, perform model based clustering and estimate the genotype state cluster means and variances.

Usage

```
estimate.cluster.params(founders, data, chr)
keep.homozygotes(founders)
```

Arguments

`founders`	List containing founder information for non-DO or CC crosses. *Not required for DO.* When method = intensity: x: Numeric matrix, num.samples x num.snps, with X intensities for all founders and FALSE1s (if available). Sample IDs and SNP IDs must be in rownames and colnames. y: Numeric matrix, num.samples x num.snps, with Y intensities for all founders and FALSE1s (if available). Sample IDs and SNP IDs must be in rownames and colnames. sex: Character vector, containing "M" or F indicating sex. Sample IDs must be in names. code: Character vector containing two letter genotype codes for each founder sample. Sample IDs must be in names.

data	A list with named elements containing the information needed to reconstruct genomes. When method = intensity: x: Numeric matrix, num.samples x num.snps, with X intensities for all samples. Sample IDs and SNP IDs must be in rownames and colnames. y: Numeric matrix, num.samples x num.snps, with Y intensities for all samples. Sample IDs and SNP IDs must be in rownames and colnames. sex: Character vector, containing "M" or F indicating sex. Sample IDs must be in names. gen: Character matrix containing the generation of DO outbreeding for each sample. FALSEor the DO, this should be "DO" followed by a number with no space between them. FALSEor CC mice, this should be CC. Sample IDs must be in names.
chr	Character containing the current chromosome.

Details

At each marker, use mclust to perform model based clustering on all of the data and get estimates of the means and variances for each cluster. Then assign each of the 36 genotype states to the nearest founder cluster.

keep.homozygotes is an internal helper function.

Value

List containing two elements:

r.t.means	Three dimensional array containing rho and theta genotype cluster means.
r.t.covars	Three dimensional array containing rho and theta genotype cluster variances.

Author(s)

Daniel Gatti

See Also

hmm.intensity

Examples

```
## Not run:  estimate.cluster.params(founders, data, chr)
```

---

example.genes *example.genes*

Description

A set of genes from the Mouse Genome Informatics that are used in the documentation examples. (http://www.sanger.ac.uk/resources/mouse/genomes/) FALSErom Chr 7: 103 - 105 Mb on NCBI Build 38.

Source

http://informatics.jax.org/

Examples

```
data(example.genes)
```

`example.pheno` *Example phenotypes.*

Description

Example phenotypes.

Usage

```
example.pheno
```

Format

A data frame with 149 observations on the following 8 variables. Sample IDs in rownames and phenotype names in colnames.

`Sample` Character vector containing sample IDs.

`Sex` Character vector containing the sex of each animal.

`Gen` Character vector containing DO outbreeding generation and litter.

`Diet` Character vector containing diet, either chow or hf for high fat.

`Coat.Color` Character vector containing text description of mouse colors.

`albino` Numeric vector contining 1 if the mouse's coat color was white.

`black` Numeric vector contining 1 if the mouse's coat color was black.

`HDLD2` Numeric vector of high density lipoprotein values.

Details

Data from Svenson et.al. paper below.

References

High-resolution genetic mapping using the Mouse Diversity outbred population. Svenson KL, Gatti DM, Valdar W, Welsh CE, Cheng R, Chesler EJ, Palmer AA, McMillan L, Churchill GA. Genetics. 2012 FALSEeb;190(2):437-47 PMID: 223445611

Examples

```
head(example.pheno)
```

`example.qtl` *Example QTL.*

Description

Example QTL for the albino and HDLD2 traits in `example.pheno`. Albino is a binary, Mendelian trait that maps to the Tyrosinase locus on Chr 7. HDLD2 (high density lipoprotein) is a complex trait with many loci.

Usage

```
example.qtl
```

Format

A data frame with 149 observations on the following 8 variables. Sample IDs in rownames and phenotype names in colnames.

`Sample` Character vector containing sample IDs.

`Sex` Character vector containing the sex of each animal.

`Gen` Character vector containing DO outbreeding generation and litter.

`Diet` Character vector containing diet, either chow or hf for high fat.

`Coat.Color` Character vector containing text description of mouse colors.

`albino` Numeric vector contining 1 if the mouse's coat color was white.

`black` Numeric vector contining 1 if the mouse's coat color was black.

`HDLD2` Numeric vector of high density lipoprotein values.

Details

Data from Svenson et.al. paper below.

References

High-resolution genetic mapping using the Mouse Diversity outbred population. Svenson KL, Gatti DM, Valdar W, Welsh CE, Cheng R, Chesler EJ, Palmer AA, McMillan L, Churchill GA. Genetics. 2012 FALSEeb;190(2):437-47 PMID: 223445611

Examples

```
names(example.qtl)
names(example.qtl[[1]])
```

`example.snps` *example.snps*

Description

A set of SNPs from the Sanger Mouse Genome project that are used in the documentation examples. (http://www.sanger.ac.uk/resources/mouse/genomes/) FALSErom Chr 7: 103 - 105 Mb on NCBI Build 38.

Source

http://www.sanger.ac.uk/resources/mouse/genomes/

Examples

```
data(example.snps)
```

`extract.raw.data` *Extract intensities, genotypes and call rates from from raw MUGA or MegaMUGA data files*

Description

This function accepts a vector of input directories containing the raw MUGA or MegaMUGA raw data files. FALSEor each directory, the function reads the X and Y intensities, call rates and allele calls for all samples. It then combines all samples and writes the data to "x.txt", "y.txt", "geno.txt" and "call.rate.batch.txt" in the user specified output directory.

Usage

```
extract.raw.data(in.path = ".", prefix, out.path = ".",  array = c("megamuga", "muga"))
```

Arguments

`in.path`	character vector, the full path to all MUGA directories from which data should be extracted.
`prefix`	character vector of same length as in.path containing a prefix to add to each sample ID in data sets being processed.
`out.path`	character, the full path to the directory where the output files should be written.
`array`	character, default = "megamuga", the type of array, either "muga" or "megamuga".

Details

This function searches each directory for files with names containing "Sample_Map.txt" and "*_FALSEinalReport.txt". This has been the format that GeneSeek has consistently produced. The call rates are extracted and are written, along with a batch ID, to "call.rate.batch.txt". The X and Y intensities are extracted from the "FALSEinalReport" file and written to "x.txt" and "y.txt" respectively. The allele calls are extracted from the "FALSEinalReport" files and are written to "geno.txt". The prefix argument may be used to add a prefix to the sample IDs in order to distinguish different data sets.

Value

No return value. The files are written to the out.path directory.

Note

Do not change the names of the output files. They are required for downstream processing.

Author(s)

Daniel M. Gatti

See Also

filter.samples, batch.normalize

Examples

```
  ## Not run:
    in.path = c("/tmpdir/DataSet1", "/tmpdir/DataSet2")
    extract.raw.data(in.path = in.path, prefix = c("ds1", "ds2"),
                out.path = "/tmpdir/output", array = "muga")

## End(Not run)
```

---

fast.qtlrel *QTL mapping using QTLRel*

Description

This extracts teh core of the QTLRel algorithm for additive covariates.

Usage

```
fast.qtlrel(pheno, probs, K, addcovar, snps)
```

Arguments

pheno	Data.frame containing the phenotype data. Sample IDs in rownames.
probs	Three dimensional numeric array containing the founder halplotype contributions. Num.samples by num.founder by num.markers. Sample IDs, founder letters and SNP IDs must be in dimnames.
K	Numeric matrix containing the kinship between samlpes. Sample IDs must be in rownames and colnames.
addcovar	Numeric matrix containing additive covariates to run in the mapping model. Sample IDs must be in rownames.
snps	Data.frame containing the marker locations. SNP ID, chromosome, Mb anc cM locations in columns 1 through 4, respectively.

Details

We extracted code from QTLRel, but removed several options to speed up the pipeline for QTL mapping with additive covariates and a kinship matrix.

Value

List containing two elements:

lod	List containing two elements: A: Numeric matrix containing the SNP ID, chromosome, Mb, cM, percent variance explained, likelihood ratio statistic, log of the odd ratio, p-value and -log10(p-value) for autosomes. X: Numeric matrix containing the SNP ID, chromosome, Mb, cM, percent variance explained, likelihood ratio statistic, log of the odd ratio, p-value and -log10(p-value) for X chromosome.
coef	List containing two elements: A : Numeric matrix containing QTL model coefficients on autosomes. Markers in rows. X : List containing two elements for the X chromosome: FALSE : Numeric matrix containing QTL model coefficients for females. Markers in rows. M : Numeric matrix containing QTL model coefficients for males. Markers in rows.

Author(s)

Daniel Gatti

References

Cheng R, Abney M, Palmer AA, Skol AD. QTLRel: an R package for genome-wide association studies in which relatedness is a concern. BMC Genet. 2011 Jul 27;12:66.

See Also

scanone, scanone

Examples

```
## Not run:  fast.qtlrel(pheno, probs, K, addcovar, snps)
```

---

fill.in.snps *Interpolate between SNPs at the same cM value.*

Description

Go through the SNPs and look for stretches where the differences in cM values from one SNP to the next equals 0. Interpolate from the first SNP to the last SNP in such a stretch.

Usage

```
fill.in.snps(snps)
```

Arguments

snps	Data.frame containing the marker locations. SNP ID, chromosome, Mb anc cM locations in columns 1 through 4, respectively.

Details

On each chromosome, look for sets of contiguous markers that have the same cM location. Interpolate evenly spaced markers spanning the markers proximal and distal to this set.

Value

Data.frame containing the marker locations. SNP ID, chromosome, Mb anc cM locations in columns 1 through 4, respectively.

Author(s)

Daniel Gatti

Examples

```
## Not run:  fill.in.snps(snps)
```

`filter.geno.probs` *Remove SNPs where the genotype probabilities are too low for one founder state*

Description

This function accetps the 8 state founder probabilities and searches for SNPs where the founder probabilities are too low for one founder. It removes these SNPs. The QTL mapping model becomes numerically unstable if these SNPs are not removed.

Usage

```
filter.geno.probs(geno)
```

Arguments

`geno`	Numeric 3D array, with samples in dim[1], states = dim[2] and SNPs in dim[3].

Value

Numeric 3D array, with samples in dim[1], states = dim[2] and SNPs in dim[3]. The SNPs with low probabilities for a single founder have been removed.

Author(s)

Daniel Gatti

See Also

`scanone`, `scanone.perm`

Examples

```
## Not run:
  filter.geno.probs(geno)

## End(Not run)
```

`filter.samples` *FALSEilter X, Y and genotype data by call rate*

Description

This function reads in "x.txt", "y.txt", "geno.txt" and "call.rate.batch.txt" from the user specified input directory and removes samples with a call rate less than the supplied threshold (default = 0.9). It then writes the files out to "x.filt.txt", "y.filt.txt", "geno.filt.txt" and "call.rate.batch.filt.txt".

Usage

```
filter.samples(path = ".", thr = 0.9)
```

Arguments

`path`	Character, the full path to the directory where the files reside.
`thr`	Numeric, call rate threshold below which samples will be removed. Default = 0.9.

Value

data.frame with the sample IDs and call rates of the removed samples. The intensity and genotype files are written to the input directory.

Author(s)

Daniel Gatti

See Also

`extract.raw.data`, `batch.normalize`

Examples

```
## Not run:
   filter.samples(path = "/tmpdir/output")

## End(Not run)
```

`find.overlapping.genes` *find.overlapping.genes*

Description

FALSEind genes that intersect with a given set of variants.

Usage

```
find.overlapping.genes(variants,
mgi.file = "http://cgd.jax.org/tools/SNPtools/MGI/MGI.20130305.sorted.txt.gz",
type = c("gene", "exon"))
```

Arguments

variants	Data.frame with variants. The type attribute must be set to one of "snp", "indel", or "sv".
mgi.file	Character. FALSEull path to the MGI gene file. Defaults to "http://cgd.jax.org/tools/SNPtools/MGI/M
type	Character. One of "gene" or "exon". Indicates whether to intersect SNPs with genes (from beginning to end, including introns) or exons only.

Details

Gets the gene locations from MGI and variants.

Value

Data.frame with gene locations and symbols.

Author(s)

Daniel Gatti

See Also

get.mgi.features{get.mgi.features}

Examples

```
## Not run:
  data(example.snps)
  x = find.overlapping.genes(variants = example.snps[19478:19506,])

## End(Not run)
```

---

gene.plot *gene.plot*

Description

Given a genomic region, plot the genes in the interval. line.up.genes and resolve.collisions are internal functions.

Usage

```
gene.plot(mgi, rect.col = "grey30", text.col = "black", ...)
```

Arguments

mgi	Data.frame with MGI gene locations as returned by get.mgi.features{get.mgi.features}.
rect.col	Color vector that is the color to use to plot gene rectangles. May be a single color for all genes or a vector with colors for each gene. Default = "grey30".
text.col	Color vector that is the color to use to plot gene names. May be a single color for all genes or a vector with colors for each gene. Default = "black".
...	Other arguments to be passed into plot.

Details

The spacing algorithm attempts to organize the genes in such a way that they do not collide. The rect.col and text.col arguments are recycled if they are shorter than the number of genes. They can be used to highlight specific genes.

Value

Data.frame with gene locations and symbols.

Author(s)

Daniel Gatti

See Also

get.mgi.features{get.mgi.features}

Examples

```
## Not run:
data(example.genes)
g = gene.plot(mgi = example.genes)

## End(Not run)
```

---

generic.trans.probs *Generic transition probabilities*

Description

Generic function to provide starting transition probabilities. Not ready for use in this version.

Usage

```
generic.trans.probs(states, snps, chr = c(1:19, "X"), sex = c("M", "F"))
```

Arguments

states	Character vector containing the genotype states coded as letters.
snps	Data.frame containing the marker locations. SNP ID, chromosome, Mb anc cM locations in columns 1 through 4, respectively.
chr	Character containing the chromosome to use.
sex	Character that is either FALSE or M indicating the sex. Only used on x chromosome.

Details

This function is a placeholder for a more sophisticated algorithm to be developed later. It is an effort to provide a set of starting transition probabilities for mapping populations other than the DO.

Value

A matrix of transition probabilities between genotype states.

Note

This function is still in development.

Author(s)

Daniel Gatti

See Also

do.trans.probs, cc.trans.probs

Examples

```
## Not run:  generic.trans.probs(states, snps, chr = 1, sex = "M")
```

---

genome.summary.plots *Genome summary plots*

Description

These plots show summaries of the founder haplotype proportions across SNPs or samples.

Usage

```
genotype.by.sample.barplot(results)
genotype.by.snp.barplot(results, snps)
```

Arguments

results	Data.frame as produced by summarize.by.snps or summarize.by.samples.
snps	Data.frame containing the marker locations. SNP ID, chromosome, Mb anc cM locations in columns 1 through 4, respectively.

Details

Barplots across samples or SNPs are produced from the relevant summary files. This is intended to be run after DO genomes have been reconstructed for a large set of samples to verify that founder allele frequencies are consistent across SNPs and samples.

Author(s)

Daniel Gatti

See Also

summarize.by.snps, summarize.by.samples

Examples

```
## Not run:
  load(url("ftp://ftp.jax.org/MUGA/muga_snps.Rdata"))
  results = summarize.by.samples(path = ".", snps = muga_snps)
  genotype.by.sample.barplot(results)

## End(Not run)
```

---

get.chr.lengths *Get chromosome lengths for the mouse*

Description

Using the org.Mm.eg.db package, get the chromosome lengths.

Usage

```
get.chr.lengths(genome = "mm10")
```

Details

Keeps only the lengths of the chromosomes, not random or unmapped.

Value

Named vector of chromosome lengths.

Author(s)

Daniel Gatti

Examples

```
get.chr.lengths()
```

get.do.states *Get the 36 genotype states for the DO*

Description

Get the 36 genotype states for the DO

Usage

```
get.do.states()
```

Value

Character vector containing the two letter genotype codes for the 36 genotype states.

Author(s)

Daniel Gatti

Examples

```
get.do.states()
```

get.gene.name *Get the gene symbol*

Description

Given an MGI ID, get the gene symbol from an MGI data.frame.

Usage

```
get.gene.name(value, mgi)
```

Arguments

value	Character string containing an MGI ID.
mgi	Data.frame containing MGI data from get.mgi.features.

Value

Character string containing the gene symbol.

Author(s)

Daniel Gatti

See Also

get.mgi.features

Examples

```
## Not run:  get.gene.name(value, mgi)
```

`get.machine.precision` *Get the machine precsion*

Description

Get the machine precision from .Machine$double.eps on a log10 scale.

Usage

```
get.machine.precision()
```

Value

Numeric value containing the machine precision.

Author(s)

Daniel Gatti

Examples

```
get.machine.precision()
```

`get.max.geno` *Get the genotype with the highest probability*

Description

For each sample at each marker, return the genotype state with the highest probability.

Usage

```
get.max.geno(probs)
```

Arguments

`probs`	Three dimensional numeric matrix containing the founder haplotype contributions.

Details

If the maximum probability is greater than 0.75, it is called homozygous. Otherwise, we take the two highest probabilities and call heterozygous.

Value

Character matrix containing the genotype with the highest probability at each locus.

Author(s)

Daniel Gatti

Examples

```
## Not run:  get.max.geno(probs)
```

`get.mgi.features` *get.mgi.features*

Description

Retrieve the MGI features within a genomic region. FALSEeatures include genes, gene models, non-coding RNA, etc., but not SNPs. This allows the user to filter by source and type of feature.

Usage

```
get.mgi.features(file = "ftp://ftp.jax.org/SNPtools/genes/MGI.20140803.sorted.txt",
chr = NULL, start = NULL, end = NULL, source = c("all", "MGI", "VEGA", "ENSEMBL",
"Blat", "NCBI_Gene"), type = c("all", "gene", "pseudogenic_transcript",
"pseudogenic_exon", "pseudogene", "match", "match-part", "transcript", "exon",
"mRNA", "five_prime_UTR", "start_codon", "CDS", "stop_codon", "three_prime_UTR",
"pseudogenic_mRNA", "pseudogenic_start_codon", "pseudogenic_CDS",
"pseudogenic_stop_codon", "pseudogenic_five_prime_UTR", "pseudogenic_three_prime_UTR",
"sequence_feature"))
```

Arguments

`file`	character, the full path to the Tabix indexed and zipped MGI feature file. Default = "http://cgd.jax.org/tools/SNPtools/MGI/MGI.20130305.sorted.txt.gz" for internal JAX use.
`chr`	Numeric vector, chr for each start and end position. Chr, start and end must all have the same length.
`start`	Numeric vector, start position in Mb or bp for each chr. Chr, start and end must all have the same length.
`end`	Numeric vector, end position in Mb or bp for each chr. Must be greater than or equal to the corresponding start value. Chr, start and end must all have the same length.
`source`	Character vector, the source of the annotation. Options are ("all", "MGI", "VEGA", "ENSEMBL", "Blat", "NCBI_Gene"). "all" returns all features.
`type`	Character vector, the type of feature. Options are ("all", "gene", "pseudogenic_transcript", "pseudogenic_exon", "pseudogene", "match", "match-part", "transcript", "exon", "mRNA", "five_prime_UTR", "start_codon", "CDS", "stop_codon", "three_prime_UTR", "pseudogenic_mRNA", "pseudogenic_start_codon", "pseudogenic_CDS", "pseudogenic_stop_codon", "pseudogenic_five_prime_UTR", "pseudogenic_three_prime_UTR", "sequence_feature"). "all" returns all features.

Details

This function is designed to return features from the MGI gene feature file (GFALSEFALSE). You can select multiple regions on different chromosomes.

Value

A list of data.frames for each region requested. Each data.frame will contain 14 columns; seqid, source, type, start, stop, score, strand, phase, ID, Name, Parent, Dbxref, mgiName, bioType. If there is only one requested region, a single data frame is returned.

Author(s)

Daniel Gatti

References

The MGI GFALSEFALSE file is at ftp://ftp.informatics.jax.org/pub/mgigff/.

Examples

```
  ## Not run:
  genes = get.mgi.features(chr = 7, start = 103 end = 105 source = "MGI", type = "gene")

## End(Not run)
```

---

get.num.auto *Get the number of autosomes*

Description

Given a data.frame of SNPs, return the number of autosomes by counting the number of numeric chromosomes.

Usage

```
get.num.auto(snps)
```

Arguments

snps	Data.frame containing the marker locations. SNP ID, chromosome, Mb anc cM locations in columns 1 through 4, respectively.

Value

Numeric value containing the number of autosomes.

Author(s)

Daniel Gatti

Examples

```
## Not run:  get.num.auto(snps)
```

`get.pattern.variants` *get.pattern.variants*

Description

Given a vector of strains names and a set of varaints, return the variants for which the given strains have one allele and the remaining strains have the other allele.

Usage

```
get.pattern.variants(variants, strain.subset = NULL)
```

Arguments

`variants`	Matrix of data.frame of variants with four header columns containing annotation and snps in the remaining columns.
`strain.subset`	character vector of strain names that comprise one subset.

Details

Use this function to obtain a subset of SNPs for which one set of strains contain one allele and the rest contain the other allele. This might come up in the course of QTL mapping where several strains have a high allele and others have a low allele.

Value

Data.frame of variants that match the requested pattern.

Author(s)

Daniel Gatti

See Also

`get.variants`

Examples

```
## Not run:
  vcf = readVcf("vcf_file.tar.gz", param)
  snp.subset = get.pattern.variants(vcf,
  strain.subset = c("A/J", "NOD/ShiLtJ", "NZO/HlLtJ", "AKR/J"))

## End(Not run)
```

get.pgw *Get the genome wide p-value.*

Description

Given a set of permutations and a LOD or -log10(p-value), return the adjusted genome-wide p-value. This requires autosomal and X chromosome permutation values as provided by scanone.perm. If you are using p-values, make sure to convert both the permutations and the p-values to -log10(-p-value).

Usage

```
get.pgw(stat, chr, perms)
```

Arguments

stat	Numeric vector of mapping statistics. Must be either LOD or -log10(p-values).
chr	Numeric vector of same legnth as stat containing the chromosome on which each statistic occurs.
perms	Numeric matrix with the maximum LOD (or maximum -log10(p-value)) from each permuation.

Value

Numeric vector containing the adjusted genome-wide p-value for each statistic.

Author(s)

Daniel Gatti, Petr Simecek

References

The X chromosome in quantitative trait locus mapping. Broman KW, Sen S, Owens SE, Manichaikul A, Southard-Smith EM, Churchill GA. Genetics. 2006 Dec;174(4):2151-8.

See Also

get.sig.thr

Examples

```
  ## Not run:
    perms = scanone.perm(pheno = pheno, probs = probs, addcovar = addcovar, snps = anps)
    get.sig.thr(perms)

## End(Not run)
```

get.sig.thr *Get the significance thresholds.*

Description

Given a set of permutations, return the significance thresholds on the autosomes and X chromosome.

Usage

```
get.sig.thr(perms, alpha = 0.05, Xchr = TRUE)
```

Arguments

perms	Numeric matrix with the maximum LOD (or maximum -log10(p-value)) from each permuation.
alpha	Numeric vector containing the alpha level of the significance thresholds.
Xchr	Logical value (default = TRUE) that indicates that the permutations contain both autosomal and X chromosome thresholds.

Value

Numeric matrix containing the significance thresholds for the autosomes in column 1 an dthe X chromossome in column 2.

Author(s)

Daniel Gatti

References

The X chromosome in quantitative trait locus mapping. Broman KW, Sen S, Owens SE, Manichaikul A, Southard-Smith EM, Churchill GA. Genetics. 2006 Dec;174(4):2151-8.

See Also

get.pgw

Examples

```
## Not run:
  perms = scanone.perm(pheno = pheno, probs = probs, addcovar = addcovar, snps = anps)
  get.sig.thr(perms)

## End(Not run)
```

`get.strains` *get.strains*

Description

Get available strain names for a set of strains from a variant file.

Usage

```
get.strains(
file = "http://cgd.jax.org/tools/SNPtools/Build38/sanger.snps.NCBI38.txt.gz")
```

Arguments

`file`	Character, full path to the variants file to use. Default is the file at the Center for Genome Dynamics at The Jackson Laboratory.

Value

Character vector of strain names in the variant file.

Author(s)

Daniel Gatti

Examples

```
## Not run:
available.strains = get.strains()
strains = available.strains[c(2, 5, 8, 9, 13:15, 17)]
snps = get.variants(chr = 7, start = 103, end = 105, strains = strains)

## End(Not run)
```

`get.trans.probs` *Get the transition probabilities between markers.*

Description

Based on the DO founders and the recombination fraction, calculate the transition probability between markers on one chromosome.

Usage

```
get.trans.probs(r, do.gen, alpha, chr = c(1:19, "X"), sex = c("M", "F"))
```

Arguments

r	Double vector of recombination fractions between SNPs.
do.gen	Integer containing the outbreeding generation of DO.
alpha	Double vector, proportion of preCC progenitors at generation k. Generation numbers in the names.
chr	Character, one of 1:19, X.
sex	Character, either M or FALSE. Only used on X chromosome.

Value

Numeric three dimensional array of transition probabilities between each pair of markers (num.states by num.states by num.markers - 1).

Author(s)

Daniel Gatti

References

Haplotype probabilities in advanced intercross populations. Broman KW. G3 (Bethesda). 2012 FALSEeb;2(2):199-202. doi: 10.1534/g3.111.001818. Epub 2012 FALSEeb 1. PMID: 22384398 Genotype probabilities at intermediate generations in the construction of recombinant inbred lines. Broman KW. Genetics. 2012 FALSEeb;190(2):403-12. doi: 10.1534/genetics.111.132647. PMID: 22345609

Examples

```
## Not run:  get.trans.probs(r, do.gen, alpha, chr = c(1:19, "X"), sex = c("M", "F"))
```

---

get.variants *get.variants*

Description

Get SNPs for a set of strains for a specific regions of chromosomes.

Usage

```
get.variants(file = "ftp://ftp.jax.org/SNPtools/variants/cc.snps.NCBI38.txt.gz",
chr, start, end, type = c("snp", "indel", "sv"), strains, polymorphic  =  TRUE,
quality)
```

Arguments

file	Character, full path to the SNP file to use. Default is the file at the Center for Genome Dynamics at The Jackson Laboratory.
chr	Numeric vector, chr for each start and end position. Chr, start and end must all have the same length.
start	Numeric vector, start position in Mb or bp for each chr. Chr, start and end must all have the same length.

end	Numeric vector, end position in Mb or bp for each chr. Must be greater than or equal to the corresponding start value. Chr, start and end must all have the same length.
type	Character, with one of "snp", "indel", "sv". Indicates the type of variantes being queried.
strains	Character vector, listing the strains to retrieve. The names can be obtained from get.strains(). Default returns all strains.
polymorphic	Boolean. Default = TRUE. If TRUE, retrieve only SNPs that are polymorphic among the requested strains. If FALSEALSE, return all SNPs in the requested interval(s).
quality	Integer, denoting the confidence levels to return. This looks at the quality column of all requested strains and takes values greater than or equal to the given number. Look at the documentation for the SNP file you are using for details on the quality score. Default returns all SNPs.

Value

A list of data.frames for each region requested. Each data.frame will contain five header columns; ID, CHROM, POS, REFALSE & ALT corresponding to the SNP ID, chromosome, bp position, reference and alternate alleles. There follows two columns for each requested strain containing the allele calls and quality scores for each strain. If there is only one requested region, a single data frame is returned.

Author(s)

Daniel Gatti

See Also

get.strains

Examples

```
## Not run:
available.strains = get.strains()
strains = available.strains[c(2, 5, 8, 9, 13:15, 17)]
snps = get.variants(chr = 7, start = 103, end = 105, strains = strains)

## End(Not run)
```

---

html.report *Create an HTML report for a set of QTL*

Description

Given a list of QTL objects, create a set of HTML pages summarizing the QTL.

Usage

```
html.report(path, qtl, perms, assoc = FALSE)
```

Arguments

path	Character string with path to which to write.
qtl	List containing DOQTL objects as produced by scanone.
perms	Numeric vector containing permutation results for the phenotypes.
assoc	Boolean, if TRUE, look for corresponding *.Rdata files containing the names of the qtl in the current working directory and plot the association plots. If FALSE (default), do not plot association mapping analysis.

Details

This function summarizes a set of QTL scans made using scanone. FALSEor each QTL in the qtl list, it creates an HTML page with a QTL plot and coefficient plots for the significant QTL. The permutations are used to assess significance. It then summarizes all of these in a table and creates a summary HTML page and a *.csv file with the QTL results.

Value

No value is returned. HTML report of QTL is written out to the specified directory.

Author(s)

Daniel Gatti

Examples

```
## Not run:  html.report(path, qtl, perms, merge = FALSE)
```

---

impute.genotypes *Impute Sanger SNPs onto mouse genomes.*

Description

Given a GRanges object containing a single range on one chromosome and haplotype probabilities, impute the Sanger SNPs onto the DO genomes.

Usage

```
impute.genotypes(gr, probs, markers, vcf.file, hq = TRUE,
cross = c("DO", "CC", "DOF1", "HS", "HSrat", "other"))
```

Arguments

gr	object that contains the genomic range in which to impute SNPs. For now, this must be a single, continuous range.
probs	3D numeric array containing the haplotype probabilities. samples x 8 founders x markers. All dimensions must be contain dimnames.
markers	data.frame containing at least 3 columns that include the marker ID, chr and postion of each marker. nrow must be the same as dim(probs)[3] and markers[,1] must equal dimnames(probs)[[3]].
vcf.file	String containing the full path to the Sanger SNP VCF file.

hq	Boolean indicating whether to use only high quality SNPs. Default = TRUE.
cross	Character string that is the cross type. One of "DO", "CC", "DOF1", "HS", "HSrat", "other")

Details

This function takes the mean of the haplotype probabilities between two markers and does not linearly interpolate between markers.

Value

A file written out to the outfile name.

Author(s)

Daniel Gatti

Examples

```
  ## Not run:
    impute.genotypes(gr, probs, markers, vcf.file, hq = TRUE,
    cross = "DO")

## End(Not run)
```

---

intensity.plots *Plot founders and F1 hybrids or genotype state means and variances on an intensity plot.*

Description

Given the X and Y (or rho and theta) array intensities, plot the values for one SNP and color and mark the founders and FALSE1s.

Usage

```
founder.F1.intensity.plot(theta, rho, s = 1, is.founder.F1, plotNew = TRUE, ...)
intensity.mean.covar.plot(s, states, theta, rho, r.t.means, r.t.covars, sample, ...)
```

Arguments

theta	Numeric matrix containing theta (or X) intensity values.
rho	Numeric matrix containing rho (or Y) intensity values.
s	Numeric integer containing the SNP index in rho and theta to plot.
is.founder.F1	Boolean vector containing TRUE for samples that are DO founders or FALSE1s and FALSEALSE for those that are not.
plotNew	Boolean that is TRUE if a new plot should be made and FALSEALSE if the points should be added to the existing plot.
states	Character vector containing genotypes state letter codes.
r.t.means	Numeric three dimensional array containing genotype state means in rho and theta coordinates.

r.t.covars	Numeric three dimensional array containing genotype state variances in rho and theta coordinates.
sample	Numeric value containing the sample index into the rho and theta matrices to plot in orange.
...	Additional arguments to be passed to plot.

Value

No return value. A scatter plot is made of the rho and theta (or X and Y) intensities at the requested SNP. Founders and F1 hybrid samples are colored.

Author(s)

Daniel Gatti

Examples

```
## Not run:  founder.F1.intensity.plot(theta, rho, s = 1, is.founder.F1, plotNew = TRUE, ...)
```

---

interpolate.markers *interpolate haplotype or genotype probabilities from one set of markers to another.*

Description

Given a matrix of data with markers in rows and data in columns, interpolate the values in each column from the marker spacing in 'from' to the marker spacing in 'to'.

Usage

```
interpolate.markers(data, from, to)
```

Arguments

data	Numeric matrix with markers in rows and data in columns. Marker names must be in rownames and must match column 1 of from. itemfrom Data frame containing at least three columns with marker names, chromosome and position in columns 1, 2 & 3, respectively. itemto Data frame containing at least three columns with marker names, chromosome and position in columns 1, 2 & 3, respectively.

Details

This function assumes an even and overlapping set of markers in from and to. It may not behave correctly if the marker sets do not overlap.

Value

Numeric matrix of values interpolated onto the spacing provided in 'to'.

Author(s)

Daniel Gatti

Examples

```
## Not run:
  interpolate.markers(data, from, to)

## End(Not run)
```

`kinship.probs` *Create a kinship matrix.*

Description

Read in the genotypes and produce a kinship matrix based on either allele sharing or haplotype contributions.

Usage

```
kinship.probs(probs, snps, bychr = FALSE)
kinship.alleles(geno)
```

Arguments

`probs`	Three dimensional numeric array containing the founder haplotype contributions. Num.samples by num.founder by num.markers.
`snps`	Data.frame containing the marker locations. SNP ID, chromosome, Mb anc cM locations in columns 1 through 4, respectively.
`bychr`	Boolean that is true if the current chromosome should be excluded from the calculation of kinship for the current chromosome. See Details.
`geno`	Character matrix of allele calls.

Details

Allele based method: This simply calculates the mean allele sharing between two individuals. Intensity based method: FALSEor each pair of samples, at each marker, calculate the cosine of the angle between the two founder contribution vectors. Take the average across the genome.

When bychr = FALSE, we calculate one kinship matrix for all chromosomes. When bychr TRUE, we calculate a different kinship matrix for each chromosome. FALSEor each chromosome, we remove use only the SNPs on the remainging chromosomes to calculate the kinship matrix for that chromosome. This is motivated by the Cheng et.al. reference below.

Value

Numeric matrix, with rows and columns equal to the number of samples in the genotype file.

Author(s)

Daniel Gatti

References

Practical Considerations Regarding the Use of Genotype and Pedigree Data to Model Relatedness in the Context of Genome-Wide Association Studies. Cheng R, Parker CC, Abney M, Palmer AA. G3 (Bethesda). 2013 Oct 3;3(10):1861-1867. PMID: 23979941

See Also

extract.raw.data, filter.samples, batch.normalize, calc.genoprob

Examples

```
  ## Not run:
    K = kinship.probs(probs)

## End(Not run)
```

---

muga.snps.to.keep *SNPs to use for genotyping and mapping on the MUGA*

Description

SNPs to use for genotyping and mapping on the MUGA. These are a combination of one culled by UNC due to poor performance or uncertain location and ones culled by JAX based on low intensity values.

Usage

```
muga.snps.to.keep
```

Format

A character vector with MUGA SNP IDs to use when genotyping or mapping.

Examples

```
muga.snps.to.keep
```

---

plot.doqtl *Plot a QTL*

Description

Given a genome scan produced by scanone, create a plot of the LOD score across all chromosomes.

Usage

```
  ## S3 method for class 'doqtl'
plot(x, stat.name = c("lod", "neg.log10.p"),  sig.thr = NULL, sig.col = "red", ...)
```

Arguments

x	DOQTL object containing the LOD score and model coefficients.
stat.name	Character string containing the name of the mapping statistic.
sig.thr	Numeric matrix containing significance thresholds. Columns must be labelled "A" and "X", in that order and should contain thresholed produced by get.sig.thr().
sig.col	Colors to use when plotting the significance thresholds above. Must be the same length as the nuber of rows in sig.thr.
...	Additional arguments to pass to plot.

Value

Creates a QTL plot.

Author(s)

Daniel Gatti

Examples

```
  ## Not run:
    qtl = scanone(pheno = pheno, pheno.col = 1, probs = probs, snps = snps)
plot(qtl)

## End(Not run)
```

---

plot.genoprobs *Plot the genome of a DO sample.*

Description

This function plots the genotype of a DO sample by taking the genotype with the maximum probability at each marker.

Usage

```
  ## S3 method for class 'genoprobs'
plot(x, snps, colors = "DO", chrlen = "mm10", ...)
  chr.skeletons(chr, chrlen, ...)
  genomic.points(chr = NULL, loc = NULL)
  write.genoprob.plots(path = ".", snps)
```

Arguments

x	Numeric matrix containing the 36 genotype state probabilities generated by calc.genoprob.
snps	Data.frame containing the SNP locations in prsmth. SNP ID, chromosome, Mb and cM locations in columns 1 through 4, respectively.
colors	Character string containing "DO" for the DO. FALSEor all other crosses, a data.frame containing single letter codes, strain names and colors for the founders in columns 1 through 3, respectively.

chrlen	Character string containing the genome for the chromosome lengths. Default = mm10.
genome	Character string containing the genome for the chromosome lengths. Default = mm10.
chr	Vector of chromosome IDs.
path	Character string containing the path to the posterior genotype probability files in R binary format (i.e. *.Rdata).
loc	Numeric vector of genome locations in Mb of same length as chr.
...	Additional arguments to be passed to plot.

Value

plot.genoprobs plots the reconstructed DO genome in terms of founder haplotypes. chr.skeletons plots the chromosome skeletons. This is useful for further plotting. genomic.points plots locations on a genome such as might be drawn by chr.skeletons. write.genoprob.plots loops through all of the files supplied in the argument and writes out genotype plots.

Author(s)

Daniel Gatti

See Also

calc.genoprob

Examples

```
## Not run: plot.genoprobs(prsmth, snps, main = "plot title")
```

---

pxg.plot *Phenotype by genotype plot at a single marker.*

Description

This function plots the phenotype versus the most probable genotype at a single marker.

Usage

```
pxg.plot(pheno, pheno.col, probs, snp.id, snps, legend = TRUE, sex = NA, covar, ...)
```

Arguments

pheno	Data.frame containing the phenotype data with samples in rows and phenotypes in columns. Sample IDs in rownames and phenotype names in colnames.
pheno.col	Numeric or character vector: Either a vector of number that indicate columns to use or a set of column names in pheno.
probs	3D numeric array, containing the founder haplotype contributions or genotype probabilities. The sample IDs, founder letter codes and markers IDs must be in dimnames.

snp.id	Character string containing the marker to plot at. Must be in SNPs and dimnames(probs)[[3]].
snps	Data.frame containing 4 columns with marker location information. SNP ID, chr, Mb, cM in columns 1 through 4, respectively.
legend	Boolean that is true if a legend explaining the founder letter codes should be plotted.
sex	Character that is either FALSE or M, indicating sex. This is used only when the SNP ID is on the X chromosome and all of the samples are male. In this case, there are only 8 genotype states.
covar	Vector of categories for each sample (i.e. diet, treatement, etc.) which will be plotted as different symbols. Optional.
...	Additional arguments passed along to plot.

Details

The most probable genotype is inferred from the eight founder allelic contributions at the marker. If a founder has a value > 0.75, it is assumed to be homozygous.

Value

Creates a plot with the phenotype on the y-axis and the 36 DO genotypes on the x-axis.

Author(s)

Daniel Gatti

See Also

plot.doqtl, coefplot

Examples

```
## Not run:  pxg.plot(pheno, pheno.col, founder.probs, snp.id, snps)
```

---

qtl.heatmap *Plot a Heatmap of all QTL*

Description

This function accepts LOD scores and SNP locations and plots a heatmap of the QTL. Each QTL is scaled to between 0 and 1 and the QTL are hierarchically clustered. A tile plot is then created with QTL colored from white (strong) to black (weak). The motivation for this plot is to visualize QTL that may be coincident across different phenotypes.

Usage

```
qtl.heatmap(lod, ...)
```

Arguments

lod	Data frame, containing the SNP names, chromosome an dbp locations in columns 1 through 3 and LOD scores for all phenotyped in the remaining columns.
...	Additional arguments to be passed to plot.

Value

No value is returned. A tile plot is plotted from the data.

Note

The QTL the appear in white are *not* neccessarily signficant. They are simply the strongest QTL for each phenotype.

Author(s)

Daniel Gatti

See Also

plot.doqtl, scanone, scanone.perm

Examples

```
## Not run:
  qtl.heatmap(lod)

## End(Not run)
```

---

qtl.LRS *QTL mapping with no kinship.*

Description

These are two functions that perform QTL mapping without a kinship matrix. They use the QR decomposition to speed up the computation. Other than for a quick screen or for assessing significance thresholds, we do not recommend mapping without a kinship matrix. They are included for historical reasons

Usage

```
qtl.LRS(pheno, probs, snps, addcovar = NULL)
permutations.qtl.LRS(pheno, probs, snps, addcovar, nperm = 1000,
return.val = c("lrs", "p"))
```

Arguments

pheno	Data.frame containing the phenotype data with samples in rows and phenotypes in columns. Sample IDs in rownames and phenotype names in colnames.
probs	Numeric three dimenaional array, containing the founder haplotype contributions or genotype probabilities. The sample IDs, founder letter codes and markers IDs must be in dimnames.
snps	data.frame containing 4 columns with marker location information. SNP ID, chr, Mb, cM in columns 1 through 4, respectively.
addcovar	data.frame or numeric matrix, containing any additive covariates. Sample IDs must be in rownames.
nperm	Numeric value containing the number of permutations to run.
return.val	Character string containing either "LRS" or "p", indicating the type of return statistic.

Details

The function performs Haley-Knott regression at the markers using the founder haplotype contributions in probs.

Value

List containing two elements. LRS: a data.frame containing the LOD scores. coef: numeric matrix containing model coefficients.

Author(s)

Daniel Gatti

See Also

scanone

Examples

```
## Not run:  qtl.LRS(pheno, probs, snps, addcovar = NULL)
```

---

qtl.qtlrel *Use QTLRel to map a set of traits*

Description

This function accepts phenotypes, genotype probabilities and a sample kinship matrix and maps the requested traits using an eight state linear model. FALSEixed covariates may be passed in as well. The output is written to two files: *.LOD.txt (containing the LOD scores for each SNP) and *.coef.txt. (containing the model coefficient at each SNP).

Usage

```
qtl.qtlrel(pheno, probs, K, addcovar, intcovar, snps)
```

Arguments

pheno	Data frame, containing the sample IDs, phenotype data and covariates.
probs	3D numeric array, containing the genotype probabilities for all samples at each SNP. Dimensions must be samples by states by SNPs and all dimensions must be named.
K	Numeric matrix, containing the kinship between individuals as computed by QTLRel.
addcovar	Numeric matrix containing additive covariates.
intcovar	Numeric matrix containing covariates that interact with the QTL.
snps	Data.frame containing the marker locations. SNP ID, chromosome, Mb anc cM locations in columns 1 through 4, respectively.

Value

A list containing two elements:

lod	Data.frame containing the SNP locations and LOD and p-values.
coef	Data.frame containing the model coefficients.

Author(s)

Daniel Gatti

See Also

plot.doqtl, scanone, scanone.perm

Examples

```
## Not run:
  qtl.qtlrel(pheno, prob, K, covar = NULL, pheno.name = "")

## End(Not run)
```

---

qtl.simulate *Simulate a QTL in the DO*

Description

Using a set of real DO genomes, simulate a QTL with a given minor allele frequency (MAFALSE), sample size and effect size.

Usage

```
qtl.simulate(probs, snps, K, sample.size = dim(probs)[[1]], effect.size = 1,
maf = 4, num.poly = 18, num.sims = 1000)
```

Arguments

probs	Numeric three dimensional array, containing the founder haplotype contributions or genotype probabilities. The sample IDs, founder letter codes and markers IDs must be in dimnames.
snps	Data.frame containing 4 columns with marker location information. SNP ID, chr, Mb, cM in columns 1 through 4, respectively.
K	Numeric matrix, containing the additive kinship matrix. The samples IDs must be in rownames and colnames.
sample.size	Numeric vector sample sizes. Must be less than or equal to the number of samples in probs.
effect.size	Numeric vector containing the effect sizes as the number of standard deviations from the phenotype mean.
maf	Numeric value containing the minor allele frequecy as the number of founders. FALSEor DO, the value must be between 1 and 4.
num.poly	Numeric value containing the number of autosomes on which to simulate a polygenic background.
num.sims	Numeric value containing the number of simulations to run.

Details

This function will simulate a phenotype with a QTL. It will output a phenotype vector with sample IDs and a data.frame called qtl describing the simulated loci.

Value

Writes pheno and qtl lists to an R binary file.

Author(s)

Daniel Gatti

See Also

scanone

Examples

```
  ## Not run:  qtl.simulate(probs, snps, K, sample.size = dim(probs)[[1]],
  effect.size = 1, maf = 4, num.poly = 18, num.sims = 1000)
## End(Not run)
```

rankZ *Rank Z transformation*

Description

This is also called the inverse normal transformation. We rank the data in x, divide by n-1, and take quantiles from the normal distribution using qnorm.

Usage

```
rankZ(x)
```

Arguments

x	Numeric vector of values to be transformed.

Details

We often use this when there are hundreds or thousands of phenotypes. This allows us to calculate permutation derived thresholds once and use them for all phenotypes.

Value

Numeric vector with a normal distribution.

Author(s)

Daniel Gatti

Examples

```
rankZ(rlnorm(20))
```

read.vcf *Read and parse VCF data*

Description

*** CURRENTLY UNDER CONSTRUCTION ***. This function is designed to read in the Sanger VCF files for SNPs, Indels and structural variants.

Usage

```
read.vcf(vcf.file, gr, chr = 1, start = 4, end = 4.5, strains,
         return.val = c("allele", "number"), return.qual = TRUE, csq = FALSE)
```

Arguments

vcf.file	Character string containing full path to a Sanger SNP, Indel or structural variant VCF file.
gr	GRanges object containing one or more ranges to query. Use either this argument or the chr, start and end arguments, but not both.
chr	Character or numeric vector containing mouse chromosome IDs to query.
start	Numeric vector containing the start position on each chromosome to query. Must be the same length as chr. If the value is less than 200, it is assumed to be in Mb. Values greater than 200 are assumed to be in bp.
end	Numeric vector containing the end position on each chromosome to query. Must be the same length as chr. If the value is less than 200, it is assumed to be in Mb. Values greater than 200 are assumed to be in bp.
strains	Character vector containing strain names, as listed in the vcf.file, to retrieve. If missing, then all strains are returned. Use get.vcf.strains to get the strain names from the VCF file.
return.val	Character string that is either "allele", which returns the nucleotide allele calls, or "number", which returns numeric values.
return.qual	Boolean that is TRUE if the quality columns should be returned.
csq	Boolean that is TRUE if the consequence column should be returned.

Details

There is a very nice package called vcf2geno, but it is designed to work with the 1000 Genomes VCF format. The Sanger format is slightly different, hence the need for this function. Future plans include the addition of variant consequence selection. The Sanger variant files have 'snp', 'indel' or 'SV' in them and we use this to determine the type of file being processed.

Value

If one chromosome location is requested, a data.frame containing the requested variants and quality scores. The variant locations and reference alleles are in the first 5 columns and the variants and quality scores for each strain are in the remaining columns. If more than one location is requested, a list of data.frames containing the variants.

Note

!!! This function is still under development !!!

Author(s)

Daniel Gatti

References

The Sanger Mouse Genomes Project generated these variants. Raw data is at the Sanger FALSETP site.

Next-generation sequencing of experimental mouse strains. Yalcin B, Adams DJ, FALSElint J and Keane TM Mammalian genome : official journal of the International Mammalian Genome Society 2012;23;9-10;490-8 PUBMED: 22772437

Sequencing and characterization of the FALSEVB/NJ mouse genome. Wong K, Bumpstead S, Van Der Weyden L, Reinholdt LG, Wilming LG, Adams DJ and Keane TM Genome biology 2012;13;8;R72 PUBMED: 22916792

The fine-scale architecture of structural variants in 17 mouse genomes. Yalcin B, Wong K, Bhomra A, Goodson M, Keane TM, Adams DJ and FALSElint J Genome biology 2012;13;3;R18 PUBMED: 22439878

The genomic landscape shaped by selection on transposable elements across 18 mouse strains. Nellaker C, Keane TM, Yalcin B, Wong K, Agam A, Belgard TG, FALSElint J, Adams DJ, FALSErankel WN and Ponting CP Genome biology 2012;13;6;R45 PUBMED: 22703977

High levels of RNA-editing site conservation amongst 15 laboratory mouse strains. Danecek P, Nellaker C, McIntyre RE, Buendia-Buendia JE, Bumpstead S, Ponting CP, FALSElint J, Durbin R, Keane TM and Adams DJ Genome biology 2012;13;4;26 PUBMED: 22524474

Mouse genomic variation and its effect on phenotypes and gene regulation. Keane TM, Goodstadt L, Danecek P, White MA, Wong K, Yalcin B, Heger A, Agam A, Slater G, Goodson M, FALSEurlotte NA, Eskin E, Nellaker C, Whitley H, Cleak J, Janowitz D, Hernandez-Pliego P, Edwards A, Belgard TG, Oliver PL, McIntyre RE, Bhomra A, Nicod J, Gan X, Yuan W, van der Weyden L, Steward CA, Bala S, Stalker J, Mott R, Durbin R, Jackson IJ, Czechanski A, Guerra-Assuncao JA, Donahue LR, Reinholdt LG, Payseur BA, Ponting CP, Birney E, FALSElint J and Adams DJ Nature 2011;477;7364;289-94 PUBMED: 21921910

Sequence-based characterization of structural variation in the mouse genome. Yalcin B, Wong K, Agam A, Goodson M, Keane TM, Gan X, Nellaker C, Goodstadt L, Nicod J, Bhomra A, Hernandez-Pliego P, Whitley H, Cleak J, Dutton R, Janowitz D, Mott R, Adams DJ and FALSElint J Nature 2011;477;7364;326-9 PUBMED: 21921916

Examples

```
## Not run:
  read.vcf(vcf.file =
  "ftp://ftp-mouse.sanger.ac.uk/current_snps/mgp.v3.snps.rsIDdbSNPv137.vcf.gz",
  chr = 1, start = 5, end = 5.5)
  read.vcf(vcf.file =
  "ftp://ftp-mouse.sanger.ac.uk/current_snps/mgp.v3.snps.rsIDdbSNPv137.vcf.gz",
  gr = GRanges(seqnames = 1, ranges = IRanges(start = 5, end = 5.5)))

## End(Not run)
```

---

scanone *Perform a genome scan.*

Description

This function is the main QTL mapping function for point mapping at each marker. The user must supply phenotypes, genotype probabilities and marker locations. Optional kinship, additive and interactive covariates may be passed in. Scanone regresses the phenotype on the genotype probabilities and produces a LOD score and founder allele effects at each marker.

Usage

```
scanone(pheno, pheno.col = 1, probs, K, addcovar, intcovar, snps, model =
c("additive", "full"))
```

Arguments

pheno	data.frame containing phenotype data. Required. rownames must contain sample IDs and there must be a column labelled 'sex' to perform correct mapping on the X chromosome.
pheno.col	numeric or character vector: Either a vector of number that indicate columns to use or a set of column names in pheno.
probs	Numeric three dimensional array, containing the founder haplotype contributions or genotype probabilities. The sample IDs, founder letter codes and markers IDs must be in dimnames.
K	numeric matrix, containing the additive kinship matrix. The samples IDs must be in rownames and colnames.
addcovar	data.frame or numeric matrix, containing any additive covariates. Sample IDs must be in rownames.
intcovar	data.frame or numeric matrix, containing any covariates that interact with the QTL. Sample IDs must be in rownames.
snps	data.frame containing 4 columns with marker location information. SNP ID, chr, Mb, cM in columns 1 through 4, respectively.
model	character string, containing one of "additive" or "full", indicating the type of model to fit.

Details

We require a sex argument in addcovar for mapping on the X chromosome. This is true even if all of your samples are of the same sex. It also means that mapping on the autosomes will occur with sex as an additive covariate. See A Guide to QTL Mapping with R/qtl by Karl Broman, pages 108-118.

Value

A list containing two elements:

lod	Data.frame with nine columns containing the marker locations, LOD scores and p-values for each phenotype. Column names are SNP_ID, Chr, Mb_NCBI38, cM, perc.var, lrs, lod, p and neg.log10.p.
coef	Numeric matrix containing the founder allele effects at each locus. Colnames are the additive covariates plus the founder terms.

Author(s)

Daniel Gatti

See Also

scanone.perm

Examples

```
## Not run: scanone(pheno, pheno.col = 1, probs)
```

scanone.assoc *Map the entire genome using association mapping.*

Description

Use the imputed Sanger SNPs and the DO haplotype probabilities. Between each pair of markers, get the unique founder strain distribution patters (SDPs) and use the DO haplotype probabilities to compute the DO SNPs.

Usage

```
scanone.assoc(pheno, pheno.col, probs, K, addcovar, markers, cross = c("DO", "CC", "HS"), sdp.fil
```

Arguments

pheno	Data.frame containing the phenotype values in columns and samples in rows. rownames(pheno) must contain the sample IDs.
pheno.col	numeric or character vector: Either a vector of number that indicate columns to use or a set of column names in pheno.
probs	Numeric three dimensional array containing the founder haplotype contributions. Num.samples by num.founders by num.markers.
K	List containing numeric matrices of leave-one-chromosome-out kinship values for all samples. Num.samples by num.samples.
addcovar	Numeric matrix of additive covariates.
cross	Character string indicating the type of cross. One of "DO", "CC", or "HS".
markers	Data.frame containing 4 columns with marker location information. SNP ID, chr, Mb, cM in columns 1 through 4, respectively.
sdp.file	Character string containing the full path to the condensed SDP file. This file is created using condense.sanger.snps.
ncl	Integer containing the number of cores to use for parallel execution.

Details

This function imputes the Sanger SNPs onto DO genomes and performs association mapping. The speed relies upon a support file that is created using condense.sanger.snps. The support file is a tab-delimited, Tabix indexed file that contains the chromosome, bp position and SDP for each polymorphic Sanger SNP. It treats tri-morphic SNPs and heterozygotes as alternate alleles.

Value

GRanges object containing the p-value for each SNP.

Author(s)

Daniel Gatti

See Also

scanone

Examples

```
## Not run:  scanone.assoc(pheno, pheno.col, probs, K, addcovar, markers, cross = c("DO", "CC", "HS"), sdp.f
```

`scanone.eqtl` *Mapping using the Matrix EQTL algorithm.*

Description

MatrixEQTL uses a series of matrix operations to greatly accelerate QTL mapping. It can accomodate additive covariates and a common kinship matrix for all phenotypes.

Usage

```
scanone.eqtl(expr, probs, K, addcovar, snps, sex)
```

Arguments

`expr`	Numeric matrix of phenotype values with samples in rows and phenotypes in columns. Rownames must contain sample IDs and colnames must contain phenotype names.
`probs`	Numeric three dimensional array containing the founder haplotype contributions. Num.samples by num.founders by num.markers.
`K`	Numeric matrix of kinship values for all samples. Num.samples by num.samples.
`addcovar`	Numeric matrix of additive covariates.
`snps`	Data.frame containing 4 columns with marker location information. SNP ID, chr, Mb, cM in columns 1 through 4, respectively.
`sex`	Character vector containing either "M" or F, indicating the sex of each sample. Used for mapping on the X chromosome.

Details

Matrix EQTL centers and rotates the phenotype and genotype matrices using matrix operations. It only calculates the LOD score at each marker and does not provide coefficients.

Value

Numeric matrix of LOD scores for all phenotypes and markers.

Author(s)

Daniel Gatti

References

Matrix eQTL: ultra fast eQTL analysis via large matrix operations. Shabalin AA. Bioinformatics. 2012 May 15;28(10):1353-8.

See Also

scanone

Examples

```
## Not run:  scanone.eqtl(expr, probs, K, addcovar, snps)
```

`scanone.perm` *Perform a genome scan.*

Description

This function is the main QTL mapping function for point mapping at each marker. The user must supply phenotypes, genotype probabilities and marker locations. Optional kinship, additive and interactive covariates may be passed in. scanone.perm regresses the phenotype on the genotype probabilities and produces a LOD score and founder allele effects at each marker.

Usage

```
scanone.perm(pheno, pheno.col = 1, probs, addcovar, intcovar, snps, model =
c("additive", "full"), path = ".", nperm = 1000)
```

Arguments

`pheno`	data.frame containing phenotype data. Required. rownames must contain sample IDs and there must be a column labelled 'sex' to perform correct mapping on the X chromosome.
`pheno.col`	numeric or character vector: Either a vector of number that indicate columns to use or a set of column names in pheno.
`probs`	3D numeric array, containing the founder haplotype contributions or genotype probabilities. The sample IDs, founder letter codes and markers IDs must be in dimnames.
`addcovar`	data.frame or numeric matrix, containing any additive covariates. Sample IDs must be in rownames.
`intcovar`	data.frame or numeric matrix, containing any covariates that interact with the QTL. Sample IDs must be in rownames.
`snps`	data.frame containing 4 columns with marker location information. SNP ID, chr, Mb, cM in columns 1 through 4, respectively.
`model`	Character string, containing one of "additive" or "full", indicating the type of model to fit.
`path`	Character string containing the location where permutation results should be written.
`nperm`	Numeric, containing the number of permutations to run.

Value

A list containing two elements:

`lod`	Data.frame with seven columns containing the marker locations and LOD scores for each phenotype.
`coef`	Numeric matrix containing the founder allele effects at each locus.

Author(s)

Daniel Gatti

See Also

scanone.perm

Examples

```
## Not run: scanone.perm(pheno, pheno.col = 1, probs)
```

---

sdp.plot *Plot association mapping results.*

Description

Plot the founder strain distribution patterns (SDP) of the SNPs in results.

Usage

```
sdp.plot(results, ...)
```

Arguments

results	Data.frame containing output from assoc.map.
...	Additional arguments passed to plot.

Details

Given the output from assoc.map, plot the SDPs for the SNPs in results. This will show which strains have the minor allele at each SNP in results. You may also add this plot to the top of assoc.plot.

Value

Produces a plot. There is no return value.

Author(s)

Daniel Gatti

See Also

assoc.map, assoc.plot

Examples

```
## Not run:
  results = assoc.map(pheno = pheno, pheno.col = 1, probs = probs, K = K, addcovar = addcovar,
snps = snps, chr = 1, start = 40, end = 45)
   sdp.plot(results[results[,12] > 3,])

## End(Not run)
```

sex.predict *Determine the sex of each sample*

Description

This function uses the mean X and Y chromosome intensities to predict the sex of each sample. Linear discriminant analysis from the MASS package is used.

Usage

```
sex.predict(x, y, snps, plot = FALSE)
```

Arguments

x	Numeric matrix, num.samples x num.snps, containing X intensities for all samples.
y	Numeric matrix, num.samples x num.snps, containing Y intensities for all samples.
snps	Data.frame containing SNP IDs, chromosomes, Mb and cM locations in columns 1 through 4, respectively.
plot	Boolean that will create a plot of mean X and Y Chr intensities if TRUE. Default = FALSEALSE.

Value

Character vector with sex assignments.

Author(s)

Daniel Gatti

Examples

```
## Not run:
  data(snps)
  sex.predict(x, y, sex = rep(F, ncol(x)))

## End(Not run)
```

snp.plot *snp.plot*

Description

Lower level function that makes a tile plot the given SNPs with the major allele colored dark and the minor allele light. Optionally, add SNPs that match a certain pattern and genes and a QTL score.

Usage

```
snp.plot(variants, col = c("black", "grey50", "white"), cluster = TRUE, ref, highlight,
pattern.snps, mgi, qtl)
```

Arguments

variants	Data.frame with variants as returned by get.variants.
col	Color vector with SNP colors for no call, alternate allele and reference allele.
cluster	Boolean that indicates if the strains should be clustered.
ref	Chracter that is the reference strain to use. Must be present in the strains in variants.
highlight	Character vector with strain names to highlight in the plot. Strain names must be present in the strains in variants.
pattern.snps	Data.frame with SNPs that match some pattern. These will be plotted below the SNPs.
mgi	Data.frame with genes in the interval to be plotted.
qtl	data.frame with QTL values containing the chromosome, bp position and QTL score in column 1 to 3. Default = NULL.

Details

Different strains can be used as the reference strain by using the ref argument. Otherwise, the major allele is plotted dark and the minor allele lighter. The QTL values will be scaled within the plotting interval and drawn with black (low) and red (high). If a strain pattern is provided, the SNPs matching the pattern are plotted in orange and any genes that they intersect with are also colored orange. Otherwise, genes are colored blue.

Value

Produces a tile plot with the locations along the horizontal axis and the strains in the SNP matrix along the vertical axis. Also, if a set of strains is given in the pattern argument, the SNPs that match that pattern are returned, categorized according to which gene they lie within.

Author(s)

Daniel Gatti

See Also

variant.plot, get.mgi.features, categorize.variants

Examples

```
## Not run:
data(qtl)
strains = get.strains()
variants = get.variants(chr = 7, start = 103, end = 105,
strains = strains[c(2,4,8,10,15:18)])
pattern.snps = get.pattern.variants(variants, strain.subset = c("C57BL/6J", "NOD/ShiLtJ",
"NZO/HlLtJ"))
mgi = get.mgi.features(chr = 7, start = 103, end = 105, source = "MGI", type = "gene")
variants = convert.variants.to.numeric(variants)
```

```
  snp.plot(variants = variants, pattern.snps = pattern.snps, mgi = mgi, qtl = qtl)

## End(Not run)
```

`summarize.genotype.transitions` *Summarize the genotype data output by the genotyping HMM.*

Description

These functions read in all of the individual genotype data files and summarizes the founder allele frequency by sample and SNP.

Usage

```
summarize.genotype.transitions(path = ".", snps)
summarize.by.snps(path = ".", snps)
summarize.by.samples(path = ".", snps)
num.recomb.plot(results, gen)
```

Arguments

`path`	Character, full path to the genotype directory where the *.genotype.probs.Rdata files are stored.
`snps`	Data.frame containing the marker locations. SNP ID, chromosome, Mb anc cM locations in columns 1 through 4, respectively.
`results`	Data.frame containing results as summarized by summarize.genotype.transitions.
`gen`	Numeric vector containing DO outbreeding generation for all samples in results. Must have sample IDs in names.

Value

Data.frame with six columns: sample, proximal genotype, distal genotype, chr, proximal Mb, distal Mb.

Author(s)

Daniel Gatti

Examples

```
## Not run:  geno.sum = summarize.genotype.transitions()
```

variant.plot *variant.plot*

Description

Convenience function that makes a tile plot the given SNPs with the major allele colored dark and the minor allele light. Optionally, add SNPs that match a certain pattern and genes and a QTL score. This calls sdp.plot after gathering all of the data.

Usage

```
variant.plot(var.file = "ftp://ftp.jax.org/SNPtools/variants/cc.snps.NCBI38.txt.gz",
mgi.file = "ftp://ftp.jax.org/SNPtools/genes/MGI.20130703.sorted.txt.gz", chr,
start, end, type = c("snp", "indel", "sv"), strains = c("A/J", "C57BL/6J",
"129S1/SvImJ", "CAST/EiJ", "NOD/ShiLtJ", "NZO/HlLtJ", "PWK/PhJ", "WSB/EiJ"),
ref = "C57BL/6J", pattern, qtl)
```

Arguments

var.file	Character string with the full path to the tabix indexed Sanger/UNC SNP file. Default is a file at the Center for Genome Dynmaics at the Jackson Laboratory.
mgi.file	Character string with the full path to the tabix indexed MGI feature file. Default is a file at the Center for Genome Dynmaics at the Jackson Laboratory.
chr	Character with the chromosome ID to plot on. Required.
start	Numeric value with the start position to plot at. May be in bp or Mb. If the value is 200 or less, it is assumed to be in Mb. Otherwise it is assumed to be in bp. Required.
end	Numeric value with the end position to plot at. May be in bp or Mb. If the value is 200 or less, it is assumed to be in Mb. Otherwise it is assumed to be in bp. Required.
type	Character indicating the type of variant to retrieve. One of "snp", "indel", or "sv". Default = "snp".
strains	Character vector with the strains to use. Must be from the set of strains returned by get.strains{get.strains}. Default = Collaborative Cross/Diversity Outbred founders.
ref	Character string with the reference strain. Must be one of the names in the strains vector argument. Default = C57BL/6J.
pattern	Character vector with strain names in the strains argument. We search for SNPs for which these strains have one allele and all other strains have the opposite allele. A more sophisticated pattern matching method may be included in future releases. Default = NULL.
qtl	data.frame with QTL values containing the chromosome, bp position and QTL score in column 1 to 3. Default = NULL.

Details

Different strains can be used as the reference strain by using the ref argument. Otherwise, the major allele is plotted dark and the minor allele lighter. The QTL values will be scaled within the plotting interval and drawn with black (low) and red (high). If a strain pattern is provided, the SNPs matching the pattern are plotted in orange and any genes that they intersect with are also colored orange. Otherwise, genes are colored blue.

Value

Produces a tile plot with the locations along the horizontal axis and the strains in the SNP matrix along the vertical axis. Also, if a set of strains is given in the pattern argument, the SNPs that match that pattern are returned, categorized according to which gene they lie within.

Author(s)

Daniel Gatti

See Also

convert.variants.to.numeric, get.mgi.features, categorize.variants, snp.plot

Examples

```
## Not run:
data(qtl)
strains = get.strains()[c(2,4,8,10,15:18)]
variant.plot(chr = 7, start = 103, end = 105, strains = strains,
pattern = c("C57BL/6J", "NOD/ShiLtJ", "NZO/HlLtJ"), qtl = qtl)

## End(Not run)
```

---

write.founder.genomes *Write out the genotypes of DO samples*

Description

Given a directory containing files generated by the DOQTL HMM (i.e. that end with "genotype.probs.Rdata"), write out two files for each sample containing the founder haplotype blocks.

Usage

```
write.founder.genomes(filenames = dir(path = ".",
pattern = "genotype.probs.Rdata"), snps)
write.founder.genomes.from.haps(probs, snps)
```

Arguments

filenames	Character vector of posterior genotype probability files in R binary format.
probs	Three dimensional numeric array containing the 8 state haplotype probabilities. Dimensions samples x founders x markers.
snps	Data.frame containing the marker locations. SNP ID, chromosome, Mb anc cM locations in columns 1 through 4, respectively.

Details

For each sample, we take the genotype state with the highest probability and write it out.

Value

No value is returned. FALSEiles are written to the current working directory.

Author(s)

Daniel Gatti

Examples

```
  ## Not run:  write.founder.genomes(filenames = dir(path = ".",
  pattern = "genotype.probs.Rdata"))
## End(Not run)
```

Index