

Package ‘Funmap2’

January 12, 2017

Version 2.4

Date 17 Jan 2017

Title Functional mapping methods for QTL mapping

Author The Center for Computational Biolog at Beijing Forestry University, China

Maintainer Zhong Wang <wzhy2000@hotmail.com>

Description Analysis of experimental crosses to identify genes (called quantitative trait loci, QTLs) contributing to variation in longitudinal quantitative traits.

Depends R (>= 2.4.0), mvtnorm, parallel, graphics, grDevices, methods, stats, utils

License GPL-3

ZipData yes

URL <https://github.com/wzhy2000/Funmap2>

R topics documented:

FM2.estimate.data	2
FM2.get.covariance	2
FM2.get.curve	3
FM2.load.data	5
FM2.permutation	9
FM2.pipe	10
FM2.qtlscan	13
FM2.report	15
FM2.select.qtl	15
FM2.simu.pipe	16
FM2.simulate	18
plot.FM2.dat	20
plot.FM2.qtl.mle	21
plot.FM2.qtl.mle.perm	22
summary.FM2.dat	22
summary.FM2.qtl.mle	23
summary.FM2.qtl.mle.perm	24
Index	26

FM2.estimate.data	<i>Phenotype estimation</i>
-------------------	-----------------------------

Description

Identifying or estimating the parameters of curve and covariance structure.

Usage

```
FM2.estimate.data(dat, curve.type=NULL, covar.type=NULL, pdf.file=NULL )
```

Arguments

dat	a data object returned by FM2.simulate or FM2.load.data
curve.type	string value indicating the specific curve type. optional values are listed in FM2.get.curve . The curve fitting is performed using the least-squares if 'auto' or NULL is assigned.
covar.type	string value indicating the specific type of covariance structure, optional values are listed in FM2.get.covariance . the MLE process identifies covariance matrix if 'auto' or NULL is assigned.
pdf.file	string value suggesting a PDF file name to illustrate the performance of curve fitting.

Value

A data object with the results of curve fitting and covariance identifying. This function updates or adds the estimation of curve fitting and covariance structure. [FM2.load.data](#) illustrates the structure of data object.

Examples

```
# data simulation using the default parameters
dat <- FM2.simulate();
dat;

# estimate the parameter of curve object and covariance structure
dat <- FM2.estimate.data(dat);
dat;
```

FM2.get.covariance	<i>Retriving covariance structure</i>
--------------------	---------------------------------------

Description

Retrieve a covariance structure which characterizes the correlation between the measured phenotype.

Usage

```
FM2.get.covariance(covar.type)
```

Arguments

`covar.type` string value indicating the type of covariance structure, such as "AR1", "SAD1", full list is described in the "details" section

Details

13 covariance structures are implemented in current version, including:

[1]	"AR1"	First-order Autoregressive
[2]	"SAD1"	First-order Structured Antedependence
[3]	"ARMA(1,1)"	First-order Autoregressive Moving Average
[4]	"ARH1"	Heterogeneous Autoregressive
[5]	"CS"	Compound Symmetry
[6]	"CSH"	Heterogeneous Compound Symmetry
[7]	"VS"	Variance Components
[8]	"SI"	Scaled Identity
[9]	"FA1"	Factor Analytic - First-order
[10]	"FAH1"	Heterogeneous Factor Analytic - First-order
[11]	"TOEP"	Toeplitz
[12]	"TOEPH"	Heterogeneous Toplitz
[13]	"HF"	Huynh-Feldt

The following summarize the parameters of covariance structure 'AR1'.

```
> x<-FM2.get.covariance("AR1");
> show(x);
      Class : fg.covariance.AR1
Covar.Type : AR1
Parameters : rho sigma2
```

Value

This functions returns a S4 object of covariance structure. You can use show or print command to check the summary information.

Examples

```
x<-FM2.get.covariance("SAD1");
x;
show(x);
```

FM2.get.curve

Retriving a curve object.

Description

Retrieve a curve object.

Usage

```
FM2.get.curve(curve.type)
```

Arguments

curve.type string value indicating the curve type, full list in the details.

Details

9 curves have been implemented in current version, including:

1) "Logistic"

$$g(t) = \frac{a}{1 + b * e^{-r*t}}$$

2) "Bi-Logistic"

$$g(t) = \frac{a1}{1 + b1 * e^{-r1*t}} + \frac{a2}{1 + b2 * e^{-r2*t}}$$

3) "Pharmacology"

$$g(t) = \frac{E_{max} * t}{EC_{50} + t} + E_0$$

4) "Exponential"

$$g(t) = a * e^{-r*t}$$

5) "Bi-Exponential"

$$g(t) = a_1 * e^{-r_1*t} + a_2 * e^{-r_2*t}$$

6) "Power"

$$g(t) = a * t^b$$

7) "Legendre2", Legendre Polynomial(2nd-order)

$$g(t) = u_0 + u_1 * t + u_2 * (3 * t^2 - 1)/2$$

8) "Legendre3", Legendre Polynomial(3nd-order)

$$g(t) = u_0 + u_1 * t + u_2 * (2 * t^2 - 1)/2 + u_3 * (5 * t^3 - 3t)/2$$

9) "Legendre4", Legendre Polynomial(4nd-order)

$$g(t) = u_0 + u_1 * t + u_2 * (2 * t^2 + 1)/2 + u_3 * (5 * t^3 - 3t)/2 + ...$$

The following introduces the summary information of a curve object.

```
> x<-FM2.get.curve("Logistic");
> show(x);
Class : fg.curve.log
Curve Type : Logistic
Parameters : a b r
Formula : y = a/(1+b*exp(-r*t))
```

Value

This function returns a S4 object of curve. The structure is described in the details section. You can use show or print command to check the summary information.

Examples

```
curve <- FM2.get.curve("Logistic");
show(curve);
```

FM2.load.data	<i>Loading data from the experiment files</i>
---------------	-----------------------------------------------

Description

Load the experimental data from experiment files.

Usage

```
FM2.load.data( pheno.csv, time.csv, geno.csv, marker.csv, cross.type,
               curve.type = NULL,
               covar.type = NULL,
               pdf.file = NULL,
               log = FALSE )
```

Arguments

pheno.csv	a CSV file of phenotypic traits . The format is described in the details section of FM2.load.data .
time.csv	a CSV file of measured time. The format is described in the details section of FM2.load.data .
geno.csv	a CSV file of genotype marker. The format is described in the details section of FM2.load.data .
marker.csv	a CSV file of marker definition. The format is described in the details section of FM2.load.data .
cross.type	string indicating the cross type, optional values are "F2", "BC" and "RIL".
curve.type	string indicating the curve type, optional values are "auto", "Logistic", "Exponential", "Power", "Legendre2", ..., the full list of curve is described in the details section of FM2.get.curve . . 'auto' or NULL force the function to do curve fitting by calling FM2.estimate.data
covar.type	string indicating the covariance type, including "auto", "AR1", "SAD1", "ARMA", "CS", ..., the full list of covariance structure is described in the details section of FM2.get.covariance . 'auto' or NULL force the function to estimate the covariance structure by calling FM2.estimate.data
pdf.file	string variable suggesting a PDF file name to illustrate the performance of curve fitting.
log	boolean variable indicating whether logarithm is applied to the phenotype data.

Details

The function returns a data object, which the structure is same as the simulation data. You can use print or str command to check the details in the data object. The following contents are exported by str command

```
List of 6
$ obj.curve:Formal class 'fg.curve.log' [package "Funmap2"] with 2 slots
$ obj.covar:Formal class 'fg.covariance.SAD1' [package "Funmap2"] with 3 slots
$ obj.cross:List of 14
$ obj.gen :List of 5
..$ geno.csv : chr "../populus.geno.csv"
..$ marker.csv : chr "../populus.marker.csv"
..$ marker.obj : NULL
..$ marker.table:'data.frame':      275 obs. of  4 variables:
.. ..$ Marker : Factor w/ 275 levels "A/15-620D ",...: 227 251 122 123 124 244 186 243 272 61 ...
.. ..$ Dist : num [1:275] 0 25.1 37 38.2 39.3 ...
.. ..$ grp_idx: int [1:275] 1 1 1 1 1 1 1 1 1 1 ...
.. ..$ Group : Factor w/ 22 levels "D1","D10","D11",...: 1 1 1 1 1 1 1 1 1 1 ...
..$ genos.matrix: int [1:78, 1:275] -1 -1 0 1 0 1 0 1 0 0 ...
.. ..- attr(*, "dimnames")=List of 2
.. .. ..$ : chr [1:78] "1" "10" "11" "12" ...
.. .. ..$ : chr [1:275] "marker1" "marker2" "marker3" "marker4" ...
$ obj.phe :List of 12
..$ pheno.csv : chr "../populus.pheno.csv"
..$ time.csv : NULL
..$ log : logi FALSE
..$ sample.obs : int 78
..$ sample.times : int [1:11] 1 2 3 4 5 6 7 8 9 10 ...
..$ pheY : num [1:78, 1:11] 1.3 2.1 1 1.7 1.2 1.1 0.9 1 1.8 2.4 ...
..$ pheX : NULL
..$ pheT : int [1:78, 1:11] 1 1 1 1 1 1 1 1 1 1 ...
..$ est.covar :List of 2
..$ est.curve :List of 7
..$ summary.curve:List of 5
.. ..$ type : chr "Logistic"
.. ..$ par : num [1:4] -6.401 32.937 5.131 0.455
.. ..$ summary:'data.frame': 10 obs. of 9 variables:
.. .. ..$ type: Factor w/ 10 levels "ABRK","Bi-Exponential",...: 8 3 1 9 4 2 10 5 6 7
.. .. ..$ parm: num [1:10] 3 6 4 3 2 4 2 3 4 5
.. .. ..$ AIC : num [1:10] 361 367 363 366 363 ...
.. .. ..$ AICc: num [1:10] 5.66 5.75 5.69 5.72 5.68 ...
.. .. ..$ BIC : num [1:10] 368 381 372 373 368 ...
.. .. ..$ SSE : num [1:10] 7371 7371 7370 7879 7766 ...
.. .. ..$ MSE : num [1:10] 8.59 8.59 8.59 9.18 9.05 ...
.. .. ..$ RMSE: num [1:10] 2.93 2.93 2.93 3.03 3.01 ...
.. .. ..$ R2 : num [1:10] 0.00491 0.00491 0.00486 0.07421 0.05876 ...
..$ summary.covar:List of 4
.. ..$ type : chr "SAD1"
.. ..$ par : num [1:2] 1.055 -0.819
.. ..$ summary:'data.frame': 12 obs. of 4 variables:
.. .. ..$ type: Factor w/ 12 levels "AR1","ARH1","ARMA(1,1)",...: 1 8 3 2 4 5 12 9 6 7 ...
```

```

.. .. ..$ L      : num [1:12] -1249 -1132 -1227 -1190 -1850 ...
.. .. ..$ AIC    : num [1:12] 2502 2268 2460 2404 3703 ...
.. .. ..$ BIC    : num [1:12] 2506 2273 2468 2432 3708 ...
- attr(*, "class")= chr "FM2.dat"

```

The phenotype file, measured time file, genotype file and marker definition file must be a CSV file. The following sections illustrate the format of each data file.

1) The phenotype file.

The first column is individual ID and the rest columns are sample data for every measurement. It looks like the following file.

```

ID. 1st 2nd 3rd 4th 5th 6th 7th
1 2.9033 4.118 6.1495 7.8161 9.8379 12.963 14.918
2 4.3306 5.3783 7.0647 9.3624 11.439 NA 15.701
3 2.3997 4.052 5.5431 7.6933 9.8471 NA 12.849
4 3.3044 4.154 5.8924 7.7133 9.2144 10.945 NA
...

```

Please note missing data is coded as space or NA in all four data files.

2) **The measurement time file.** The first column is individual ID and the rest columns are sample data for each measurement. It looks like the following file.

```

ID. 1st 2nd 3rd 4th 5th 6th 7th
1 1 2 3 4 5 6 7
2 1 2 3 4 5 NA 7
3 1 2 3 4 5 NA 7
4 1 2 3 4 5 6 NA
...

```

3) **The genotype file.** The first column is individual ID and the rest columns are genotype data for each marker. An example is shown in the following table. Three genotypes (aa=0, Aa=1, AA=2) and missing data(coded as NA or -1) are valid maker values.

```

ID. marker1 marker2 marker3 marker4 marker5 marker6
1 1 1 0 1 NA 0
2 1 1 1 1 0 0
3 1 1 1 0 1 1
...

```

4) **The marker list file.** The first column is marker's ID, the rest columns are the marker's name, distance, group index, and group name for every marker. In the marker file, the distance field is a distance (in cM) in one chromosome or linkage group. The header row should be included.

```

id Marker Dist. Grp_idx Group
1 marker1 0 1 G1
2 marker2 20 1 G1
3 marker3 40 1 G1
...

```

Value

This function returns a S3 object with the class label of FM2.dat which structure is identical with the result from [FM2.simulate](#).

obj.cross	the cross object.
obj.curve	the curve object.
obj.covar	the covariance structure(object).
obj.gen	the genotype data, including geno.csv, marker.csv, marker.obj, marker.table, genos.matrix.
obj.phe	the phenotype list, each item is explained in below section.
obj.phe\$pheno.csv	the phenotype file.
obj.phe\$time.csv	the measured time file.
obj.phe\$sample.obs	sample size
obj.phe\$sample.times	the measure times
obj.phe\$log	boolean value indicating whether log function is applied to the phenotype data
obj.phe\$pheY	matrix, longitudinal phenotype traits
obj.phe\$pheX	matrix, covariate data for all individuals.
obj.phe\$pheT	matrix, measured times for all individuals.
obj.phe\$est.covar	the estimation for covariance structure obtained from the call of FM2.estimate.data .
obj.phe\$est.curve	the estimation for curve object obtained from the call of FM2.estimate.data .
obj.phe\$summary.curve	the curve fitting results for all selected curves, if curve.type is NULL or 'auto', all available curves are estimated.
sobj.phe\$summary.covar	the results of covariance estimation for all selected covariances, if covar.type is NULL or 'auto', all available covariances are estimated.

Examples

```
# Load the pre-installed data for this example
file.pheno.csv <- system.file("extdata", "populus.pheno.csv", package="Funmap2")
file.geno.csv <- system.file("extdata", "populus.geno.csv", package="Funmap2")
file.marker.csv <- system.file("extdata", "populus.marker.csv", package="Funmap2")

# Call the pipeline without permutation.
# Can't select QTL using pvalue due to the missing of permutation result.
dat <- FM2.load.data( file.pheno.csv, NULL, file.geno.csv, file.marker.csv, "BC",
  curve.type="auto",
  covar.type="auto",
  pdf.file="FM2.test.load.pdf");

str( dat );
print( dat );
plot( dat, pdf.file="test.FM2.load.data.pdf")
```

FM2.permutation	<i>Permutation</i>
-----------------	--------------------

Description

Execute permutation to get the cutoff value for significance levels $p=0.05$ and 0.01 .

Usage

```
FM2.permutation( dat, res, grp.idx=NULL, options=list() )
```

Arguments

dat	a data object returned by FM2.simulate or FM2.load.data
res	a result object returned by FM2.qtlscan
grp.idx	a numeric vector indicating which chromosomes or groups get involved in the permutation test.
options	optional value for permutation control, including. debug : default=FALSE, indicating whether this function outputs the debug information. n.cores : default=1, the number of cpu cores for parallel computation. scan.step : default=1, an interval distance used to scan flanking marker, default is 1cm. permu.loop : default=100, the count of permutation loop. permu.filter.ratio : default=1, indicating whether fast estimation algorithm on the basis of QTL filter is used or not. No any optimisation for permutation with the default value(=1). If 0.01 is specified, permutation is performed on top 0.01 QTLs which are highly associated with phenotypic traits.

Details

If permutation count is greater than 100, more precise cutoff will be obtained. For example, 10000 times permutation can give the significance table which looks like the following table.

0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1
0.09	0.08	0.07	0.06	0.05	0.04	0.03	0.02	0.01
0.009	0.008	0.007	0.006	0.005	0.004	0.003	0.002	0.001
0.0009	0.0008	0.0007	0.0006	0.0005	0.0004	0.0003	0.0002	0.0001

If clusters or multiple CPU cores are available, the permutation can use **snowfall** package to do parallel computation. In order to do that, the following is necessary.

1) **snowfall** is installed.

2) The cluster count should be specified in the options parameter

e.g. options=list(n.cores=10)

Value

This function returns a new result object with the update of permutation results. The result objects is described in [FM2.qtlscan](#).

Here we explain the updated results only in the item of `obj.perm`, a S3 object mainly including a p-value matrix(`pv.table`).

<code>cross.type</code>	the cross type
<code>curve.type</code>	the curve type
<code>covar.type</code>	the type of covariance structure
<code>permu.loop</code>	the permutation count
<code>param</code>	a list recording the paramters in this functrion calling, including <code>permu.loop</code> , <code>permu.filter.ratio</code> , <code>scan.step</code> , <code>n.cores</code>
<code>full.res</code>	matrix recording all the permutation results
<code>pv.table</code>	matrix which has two columns, the first column is significance leve and nd the second column is cutoff value.

Examples

```
dat <- FM2.simulate();
ret <- FM2.qtlscan(dat);
ret <- FM2.permutation(dat, ret, options=list(n.cores=10, permu.filter.ratio = 0.02, scan.step=2));

# only print the permutation part in the result object
show(ret$obj.permu);

# draw cutoff curve based on permutation results.
plot(ret$obj.permu, pdf.file="test.FM2.permu.pdf");
```

FM2.pipe

QTL mapping pipeline

Description

Perform standard pipeline for the experiment data based on Functional Mapping framework.

Usage

```
FM2.pipe( pheno.csv, time.csv, geno.csv, marker.csv, cross.type,
  curve.type = NULL,
  covar.type = NULL,
  model = "MLE",
  grp.idx = NULL,
  pdf.prefix = NULL,
  threshold = 0.05,
  threshold.type = "pvalue",
  options = list() )
```

Arguments

pheno.csv	a CSV file of phenotypic traits . The format is described in the details section of FM2.load.data .
time.csv	a CSV file of measure time. The format is described in the details section of FM2.load.data .
geno.csv	a CSV file of genotype marker. The format is described in the details section of FM2.load.data .
marker.csv	a CSV file of marker definition. The format is described in the details section of FM2.load.data .
cross.type	string value indicating the cross type, including "F2", "BC" and "RIL".
curve.type	string value indicating the curve type, including "auto", "logistic", "Exponential", "Power", "Legendre2", ..., the full list is described in the details section of FM2.get.curve .
covar.type	string value indicating the type of covariance structure, including "auto", "AR1", "SAD1", "ARMA", "CS", ..., the full list is described in the details section of FM2.get.covariance .
model	string value indicating the computation algorithm, currently only "MLE" is option.
grp.idx	a numeric vector indicating which chromosomes or groups get involved in the QTL scanning.
pdf.prefix	string value indicating the prefix name of pdf file exported by the pipeline
threshold	a numeric value indicating the criteria of significant QTLs.
threshold.type	string value indicating the selection method of significant QTL, three optional values. 'pvalue', 'LR' and 'count'
options	optional values for the pipeline, see the following details.

Details

The options parameters can slightly adjust the results and greatly speed up the computational process. the below explains all items in the options list:

- 1) **debug**, default=FALSE, indicating whether this function outputs the debug information.
- 2) **n.cores**, default=1, the cluster count or multiple cores for parallel permutation, used in [FM2.qtlscan](#) and [FM2.permutation](#).
- 3) **scan.step**, default=2, an interval distance used to scan flanking marker, default is 1cm, used in [FM2.qtlscan](#).
- 4) **peak.count**, default=5, a number determines how many significant QTLs will be selected, used in [FM2.select.qtl](#).
- 5) **permu.loop**, default=100, the count of permutation loop, used in [FM2.permutation](#).
- 6) **permu.filter.ratio**, default=1, indicating whether fast estimation algorithm on the basis of QTL filter is used or not in [FM2.permutation](#). No any optimization for permutation under the default condition(=1). If 0.01 is specified, permutation is performed on top 0.01 QTLs which are highly associated with phenotypic traits.

FM2.pipe is a main pipeline of the Funmap2 package, it encapsulates a consecutive procedures, including:

- 1) Loading the phenotype, genotype and marker file, which is performed in the function [FM2.load.data](#).

- 2) Summarize the data object returned by the function [FM2.load.data](#)
- 3) Performing the hypothesis tests on all chromosomes or specified chromosomes, which is performed in the function [FM2.qtlscan](#).
- 4) Summarize the result object returned by the function [FM2.qtlscan](#)
- 5) Taking a long time to execute permutation parallelly or not, which is performed in the function [FM2.permutation](#).
- 6) Selecting the significant QTLs according to the selection method and threshold, which is performed in the function [FM2.select.qtl](#).
- 7) Outputting a PDF report which includes the summary of the data and QTL scanning results, which is performed in the function [FM2.report](#)

Value

The data object and the result object of hypothesis tests are returned in a list object,

dat	data object with the S3 class label of "FM2.dat", the structure is described in the details section of FM2.load.data
ret	result object with the S3 class label of "FM2.qtl.mle", the structure is described in the details section of FM2.qtlscan

Examples

```
# Load the pre-installed data for the example
file.pheno.csv <- system.file("extdata", "populus.pheno.csv", package="Funmap2")
file.geno.csv <- system.file("extdata", "populus.geno.csv", package="Funmap2")
file.marker.csv <- system.file("extdata", "populus.marker.csv", package="Funmap2")

# Call the pipeline without permutation.
# Can't select QTL using pvalue due to the missing of permutation result.
r <- FM2.pipe( file.pheno.csv, NULL, file.geno.csv, file.marker.csv, "BC",
  curve.type="logistic",
  covar.type="auto",
  grp.idx = c(1:5),
  threshold = 3,
  threshold.type = "count",
  options=list(permu.loop=0) );

# Show the summary information of data object
show(r$dat);
# Show the summary information of result object
show(r$ret);

# Change the QTL criteria
r$ret <- FM2.select.qtl(r$ret, threshold = 40, threshold.type = "LR" );
show(r$ret);

# Make a report for the data analysis.
```

```
FM2.report("test.FM2.pipe.pdf", r$dat, r$ret );
```

FM2.qtlscan	<i>QTL scanning</i>
-------------	---------------------

Description

Perform QTL scanning for all QTLs to detect the significant ones based on the hypothesis test.

Usage

```
FM2.qtlscan( dat, model="MLE", grp.idx=NULL, options=list() )
```

Arguments

<code>dat</code>	a data object returned by FM2.simulate or FM2.load.data
<code>model</code>	string value indicateing which method will be used to test hypothesis, one optional value currently.
<code>grp.idx</code>	a numeric vector indicating which chromosomes or groups get involved in the QTL scanning.
<code>options</code>	optional list for QTL scanning, including: debug : default=FALSE, indicating whether this function outputs the debug information. n.cores : default=1, a number of cpu cores for parallel computation. scan.step : default=1, an interval distance used to scan flanking marker, default is 1cm. peak.count : default=5, a number indicating how many top(or significant) QTLs will be selected.

Details

This function returns a result object which can be inspected by the following method:

- 1) `str` command.
- 2) `print` command.
- 3) [summary](#) command.
- 4) [plot](#) command.

Hypothesis Test: For different genotypes, all parameters are identical.

The hypothesis testing scans every marker by the specified step (1cm). It maybe take a long time, so the Funmap2 package displays its progress after each chromosome (linkage group) has been calculated.

After the QTL scanning, the package identifies the 5 top QTLs. At most one significant QTL is selected within each chromosome (group). The top QTLs are strongly displayed at the head of report.

Value

This function returns a S3 object with the class label of `FM2.qtl.mle`, including the following items:

<code>param</code>	list recording the parameters for this function call.
<code>obj.phe</code>	the phenotype data, copied from the data object (<code>dat</code>).
<code>obj.gen</code>	the genotype data, copied from the data object (<code>dat</code>).
<code>obj.curve</code>	the curve object, copied from the data object (<code>dat</code>).
<code>obj.covar</code>	the covariance structure, copied from the data object (<code>dat</code>).
<code>obj.cross</code>	the cross object, copied from the data object (<code>dat</code>).
<code>cross.type</code>	the cross type, copied from the data object (<code>dat</code>).
<code>covar.type</code>	the type of covariance structure, copied from the data object (<code>dat</code>).
<code>curve.type</code>	the curve type, copied from the data object (<code>dat</code>).
<code>full.res</code>	a matrix for all QTLs with the position, likelihood ratio, curve parameters of different genes and covariance parameters.
<code>threshold.type</code>	the selection method of significant QTLs, used in FM2.select.qtl
<code>threshold</code>	the criteria of significant QTLs, used in FM2.select.qtl
<code>qtl.peaks</code>	a numeric vector indicating the row index of significant QTLs in above matrix. This item is available after the calling of FM2.select.qtl
<code>obj.permu</code>	the permutation result, obtained from the calling of FM2.permutation

Examples

```
# Simulate the data and do QTL scanning
dat <- FM2.simulate();
ret <- FM2.qtlscan(dat);
show(ret);

# Load the example data and do QTL scanning
file.pheno.csv <- system.file("extdata", "populus.pheno.csv", package="Funmap2")
file.gen.csv <- system.file("extdata", "populus.gen.csv", package="Funmap2")
file.marker.csv <- system.file("extdata", "populus.marker.csv", package="Funmap2")
dat <- FM2.load.data( file.pheno.csv, NULL, file.gen.csv, file.marker.csv, "BC",
  curve.type="logistic",
  covar.type="AR1")

# Call the QTL scanning process
ret <- FM2.qtlscan(dat, grp.idx = c(1:2) );
show(ret);
plot(ret, pdf.file="test.FM2.qtlscan.pdf");
```

FM2.report

PDF report for data and result object

Description

Output a PDF report including the summary information and figures for the data and result object.

Usage

```
FM2.report( file.report.pdf, dat, res=NULL, options=list( debug=F ) )
```

Arguments

file.report.pdf	PDF file name.
dat	a data object returned by FM2.simulate or FM2.load.data
res	a result object returned by FM2.qtlscan
options	option list including whether debug information is outputted.

Details

This function don't use the HaruPDF package anymore!!! It outputs the summary information and figures into a PDF file for the data object and the result object.

The following link is an example for FM2.report.

http://statgen.psu.edu/software/funmap/report_demo.pdf

Value

No return value.

Examples

```
dat <- FM2.simulate();
res <- FM2.qtlscan(dat);
FM2.report("test.FM.report.pdf", dat, ret);
```

FM2.select.qtl

Selecting significant QTLs

Description

Select significant QTLs according to the threshold and method.

Usage

```
FM2.select.qtl( res, threshold=0.05, threshold.type="pvalue" )
```

Arguments

<code>res</code>	a result object returned by <code>FM2.qtlscan</code>
<code>threshold</code>	numeric value indicating the criteria of significant QTLs.
<code>threshold.type</code>	string value indicating the selection method of significant QTL, three optional values. 'pvalue', 'LR' and 'count'.

Details

Three methods can be used to show the significant QTLs:

- 1) use `show` or `print` command to check the significant QTLs;
- 2) use the `plot` command to show the significant QTLs and genetic curve at significant QTLs;
- 3) access the items of result object

e.g.

```
> res <- FM2.select.qtl(res, threshold=5, threshold.type="count")
> cat("The significant QTL list:\n");
> show( res$full.res[ res$qtl.peaks, 1:3] );
```

Value

A results with updated significant QTLs is returned.

Examples

```
dat <- FM2.simulate();
ret <- FM2.qtlscan(dat);
ret <- FM2.select.qtl( ret, threshold=40, threshold.type="LR" )
plot(ret, pdf.file="test.FM2.select.qtl.pdf");
```

Description

Demonstrate the simulation test using the pipeline function.

Usage

```
FM2.simu.pipe( cross.type = "BC", curve.type="Logistic", covar.type="AR1",
  simu.mrkdist = rep(20,10),
  simu.qtlpos = 95,
  simu.obs = 800,
  simu.times = 8,
  par.X = NULL,
  par0 = NULL,
  par1 = NULL,
  par2 = NULL,
  par.covar = NULL,
  phe.missing = 0.01,
  marker.missing = 0.01,
  threshold = 0.05,
  threshold.type = "pvalue",
  model = "MLE",
  pdf.prefix = NULL,
  options = list() )
```

Arguments

<code>cross.type</code>	string value indicating the cross type, including "F2", "BC" and "RIL".
<code>curve.type</code>	string value indicating the curve type, including "logistic", "Exponential", "Power", "Legendre2", ..., the full list is described in the details section of FM2.get.curve .
<code>covar.type</code>	string value indicating the covariance type, including "AR1", "SAD1", "ARMA", "CS", ..., the full list is described in the details section of FM2.get.covariance .
<code>simu.mrkdist</code>	numeric vector indicating the distance between the genomic marker.
<code>simu.qtlpos</code>	numeric value indicating the significant QTL position.
<code>simu.obs</code>	numeric value indicating the sample size.
<code>simu.times</code>	numeric value indicating the measured times.
<code>par.X</code>	numeric vector indicating covariate parameters.
<code>par0</code>	numeric vector indicating curve parameters for gene QQ, default value is retrieved from the curve object.
<code>par1</code>	numeric vector indicating curve parameters for gene Qq, default value is retrieved from the curve object.
<code>par2</code>	numeric vector indicating curve parameters for gene qq, default value is retrieved from the curve object.
<code>par.covar</code>	numeric vector indicating covariance parameters, default value is retrieved from the covariance structure.
<code>phe.missing</code>	numeric value indicating the missing rate of phenotypic traits.
<code>marker.missing</code>	numeric value indicating the missing rate of genomic markers.
<code>threshold</code>	numeric value indicating the criteria of significant QTLs.
<code>threshold.type</code>	string value indicating the selection method of significant QTL, three optional values. 'pvalue', 'LR' and 'count'
<code>model</code>	string value indicating the computation algorithm, currently only "MLE" is option.
<code>pdf.prefix</code>	string value indicating the prefix name of pdf file exported by the pipeline
<code>options</code>	optional list for the pipeline, see the details in FM2.pipe .

Details

The options parameter is described in [FM2.pipe](#).

FM2.simu.pipe demonstrates how to use the Funmap2 to do a simulation test, which includes the following steps:

- 1) Simulate a raw data object on the basis of the parameters.
- 2) Perform QTL scanning on all QTLs based on the hypothesis test.
- 3) Execute permutation to get a cutoff for significant QTLs.
- 4) Summarize all objects and plot all figures.
- 5) Export a PDF report including all summary information and figures.

Value

A list including the data object(**dat**) and the result object of QTL scanning (**ret**).

dat	data object described in FM2.simulate
ret	result object of QTL scanning with permutation cutoff table described in FM2.qtlscan

Examples

```
# Call the pipeline for the simulation test. This test doesn't
# call permutation to determine the p-value for QTL peaks.
r <- FM2.simu.pipe("RIL", "Logistic", "SAD1", simu.obs=1000, simu.times = 7,
  threshold = 1, threshold.type = "count", options = list(permu.loop=0) );

# Summarize the data object
show(r$dat);

# Summarize the result object
show(r$ret);
```

FM2.simulate

Data simulation

Description

Create a simulation data object for pipeline demonstration.

Usage

```
FM2.simulate( cross.type = "BC",
  curve.type = "Logistic",
  covar.type = "AR1",
  simu.mrkdist = rep(20,10),
  simu.qtlpos = 95,
  simu.obs = 800,
  simu.times = 8,
  par.X = NULL,
  par0 = NULL,
  par1 = NULL,
  par2 = NULL,
  par.covar = NULL,
  phe.missing = 0.01,
  marker.missing = 0.01,
  pdf.file = NULL )
```

Arguments

<code>cross.type</code>	string value indicating the cross type, including "F2", "BC" and "RIL".
<code>curve.type</code>	string value indicating the curve type, including "logistic", "Exponential", "Power", "Legendre2", ..., the curve list is described in the details section of FM2.get.curve .
<code>covar.type</code>	string value indicating the covariance type, including "AR1", "SAD1", "ARMA", "CS", ..., the covariance list is described in the details section of FM2.get.covariance .
<code>simu.mrkdist</code>	numeric vector indicating the distance between the genomic marker.
<code>simu.qtlpos</code>	numeric value indicating the significant QTL position.
<code>simu.obs</code>	numeric value indicating the sample size.
<code>simu.times</code>	numeric value indicating the measured times.
<code>par.X</code>	numeric vector indicating covariate parameters.
<code>par0</code>	numeric vector indicating curve parameters for gene QQ, default value is retrieved from the curve object.
<code>par1</code>	numeric vector indicating curve parameters for gene Qq, default value is retrieved from the curve object.
<code>par2</code>	numeric vector indicating curve parameters for gene qq, default value is retrieved from the curve object.
<code>par.covar</code>	numeric vector indicating covariance parameters, default value is retrieved from the covariance object.
<code>phe.missing</code>	numeric value indicating the missing rate of phenotypic traits.
<code>marker.missing</code>	numeric value indicating the missing rate of genomic markers.
<code>pdf.file</code>	string variable suggesting a PDF file name to illustrate the performance of curve fitting.

Details

The structure of simulation data is identical to experiment data object. The different points are listed below:

- 1) The items of `pheno_file`, `geno_file` and `marker_file` are made up by the Funmap2 package and will be used to assign the output filename as a filename prefix.

2) In the genotype is coded by 1=Qq 2=QQ for backcross , 0=qq, 1=Qq, 2=QQ for F2 intercross and 0=qq, 2=QQ for RILs intercross.

Value

This function returns a S3 object with the class label of FM2.dat, which structure is same as the experiment data object obtained from the function `FM2.load.data`.

Examples

```
dat <- FM2.simulate("RIL", "Logistic", "SAD1", simu.obs=1000, simu.times = 7 );
#summarize the data information.
summary( dat );
plot(dat, pdf.file="test.FM2.simulate.pdf");
str(dat);
```

plot.FM2.dat	<i>Plotting figures of data object</i>
--------------	----------------------------------------

Description

Draw figures for a data object.

Usage

```
## S3 method for class 'FM2.dat'
plot( x, plot.type=NULL, pdf.file=NULL, ... )
```

Arguments

x	a data object returned by <code>FM2.simulate</code> or <code>FM2.load.data</code> .
plot.type	number, the plot type, 1 is for tiled curves and 2 is for overlapping curves.
pdf.file	a pdf file name for the figure output, if no pdf file is specified, the plot command can output this figure in the R console.
...	additional arguments affecting the plot produced.

Details

Two figures can be outputted to R console.

1) tiled curves for every individuals.

2) overlapping curves for every individuals

An example of this command is available in the following URL.

<http://statgen.psu.edu/software/funmap/plot.data1.jpg>.

<http://statgen.psu.edu/software/funmap/plot.data2.jpg>.

Examples

```
#check the codes in FM2.simulate() or FM2.load.data()
```

plot.FM2.qtl.mle	<i>Plotting figures of QTL scanning</i>
------------------	-----------------------------------------

Description

Plot the figures based on the results of hypothesis test.

Usage

```
## S3 method for class 'FM2.qtl.mle'
plot(x, plot.type=NULL, pdf.file=NULL, ... )
```

Arguments

x	a result object of hypothesis tests returned by FM2.qtlscan .
plot.type	a number indicating which figure is plotted.
pdf.file	a pdf file name for the figure output, if no pdf file is specified, the plot command can output this figure in the R console.
...	additional arguments affecting the plot produced.

Details

The result object of QTL scanning can output three kinds of figure according to the parameter 'plot.type', including:

- 1) The LR profile for all chromosomes.
- 2) The LR profile for QTL postion.
- 3) The curve for QTL postion.

The examples can be viewed in the following url.

<http://statgen.psu.edu/software/funmap/plot.t10-1.jpg>.

<http://statgen.psu.edu/software/funmap/plot.t10-2.jpg>.

<http://statgen.psu.edu/software/funmap/plot.t10-3.jpg>.

Examples

```
#check the codes in FM2.qtlscan()
```

plot.FM2.qtl.mle.perm *Plotting figure of permutation result*

Description

Draw a cutoff profile on the basis of the permutation result.

Usage

```
## S3 method for class 'FM2.qtl.mle.perm'
plot(x, pdf.file=NULL, ... )
```

Arguments

x	an object of permutation result returned by FM2.permutation .
pdf.file	a pdf file name for the figure output, if no pdf file is specified, the plot command can output this figure in the R console.
...	additional arguments affecting the plot produced.

Details

This summary exports a figure based on the cutoff table in the permutation result. An example of this command is available in the following URL.

<https://raw.githubusercontent.com/wzhy2000/Funmap2/master/img/plot.perm.jpg>.

Examples

```
#check the example in the FM2.permutation()
```

summary.FM2.dat *Summary of the data object*

Description

Summarize information for the data object.

Usage

```
## S3 method for class 'FM2.dat'
summary( object, ... )
```

Arguments

object	a data object return by FM2.simulate or FM2.load.data
...	additional arguments affecting the summary produced.

Details

The data object is described in [FM2.load.data](#).

The following example demonstrates summary command for a data object.

Data set for FunMap model:

```
-----
      Date: 2010-03-19 03:49:36
      Model: Logistic Curve
      Cross: F2
      Pheno. file: simu.pheno.LC.F2
      Geno. file: simu.geno.LC.F2
      Marker file: simu.marker.LC.F2
      Sample size: 100
      Sample times: 7
      Marker count: 6
      LC  a: 19.83678
          b: 8.96370
          r: 0.47202
          rho: 0.75430
          sigma2: 0.58849
-----
```

Value

No return values, only output the summary information on the R console.

Examples

```
#check the codes in FM2.simulate() or FM2.load.data()
```

summary.FM2.qtl.mle	<i>Summary of the result object.</i>
---------------------	--------------------------------------

Description

Summarize information for the QTL scanning results based on the hypothesis test.

Usage

```
## S3 method for class 'FM2.qtl.mle'
summary( object, ... )
```

Arguments

object	a result object returned by FM2.qtlscan which stores the results of hypothesis tests.
...	additional arguments affecting the summary produced.

Details

The following sections demonstrate the context of summary report. including:

- 1) Estimated parameters.
- 2) The significant QTL postions.

Hypothesis test 10:

a2=a1 and b2=b1 and r2=r1

```
-----
      Model: Logistic Curve
      Cross: Backcross
      QTL pos.: 50.1 (Group:8)
      QTL LR: 66.516
      QTL p-value: 0.000
      Grwoth para(Qq): a2= 30.615, b2= 10.776, r2= 0.538
      Grwoth para(qq): a1= 23.707, b1= 9.449, r1= 0.615
                      rho: 0.953
                      sigma2: 8.637
-----
```

No.	Grp	Pos.	LR	a1	b1	r1	a0	b0	r0
1	8	50.100	66.516	30.615	10.776	0.538	23.707	9.449	0.615
2	12	113.100	55.190	29.865	9.736	0.528	25.207	8.725	0.586
3	13	12.000	50.963	29.518	9.723	0.526	24.926	8.906	0.602
4	18	10.000	25.684	29.236	9.550	0.536	25.523	8.963	0.584
5	1	151.300	24.162	25.998	8.520	0.575	28.801	9.672	0.536

Examples

```
#check the codes in FM2.qtlscan()
```

```
summary.FM2.qtl.mle.perm
```

Summary of permutation result

Description

Summarize the permutation result.

Usage

```
## S3 method for class 'FM2.qtl.mle.perm'
summary( object, ...)
```

Arguments

object an object of permutation result returned by [FM2.permutation](#).

... additional arguments affecting the summary produced.

Details

The summary command gives a table of cutoff values which starts at 90 If the p-value of x permutation count should be greater than $100/x$.

The following gives an example of this summary command.

Permutation result:

```
-----
                Curve: Logistic Curve
                Cross: BC
                Loop: 100
-----

p-value      Cutoff
0.90000      0.76583
0.80000      1.45845
0.70000      2.21657
0.60000      3.09488
0.50000      4.16930
0.40000      5.52038
0.30000      7.33172
0.20000      9.97031
0.10000     14.86003
0.09000     15.63818
0.08000     16.50743
0.07000     17.51136
0.06000     18.68964
0.05000     20.09660
0.04000     21.77278
0.03000     23.96467
0.02000     26.98845
0.01000     32.80592
```

Examples

```
#check the example in the FM2.permutation()
```

Index

*Topic **Covariance**

FM2.get.covariance, 2

*Topic **Curve**

FM2.get.curve, 3

*Topic **Data**

FM2.estimate.data, 2

FM2.load.data, 5

FM2.simulate, 18

plot.FM2.dat, 20

summary.FM2.dat, 22

*Topic **Permutation**

FM2.permutation, 9

plot.FM2.qtl.mle.perm, 22

summary.FM2.qtl.mle.perm, 24

*Topic **Pipeline**

FM2.pipe, 10

FM2.simu.pipe, 16

*Topic **Plot**

plot.FM2.dat, 20

plot.FM2.qtl.mle, 21

plot.FM2.qtl.mle.perm, 22

*Topic **QTL Scanning**

FM2.pipe, 10

FM2.qtlscan, 13

FM2.select.qtl, 15

FM2.simu.pipe, 16

*Topic **Report**

FM2.report, 15

*Topic **Simulation**

FM2.simu.pipe, 16

FM2.simulate, 18

*Topic **Summary**

summary.FM2.dat, 22

summary.FM2.qtl.mle, 23

summary.FM2.qtl.mle.perm, 24

FM2.estimate.data, 2, 5, 8

FM2.get.covariance, 2, 2, 5, 11, 17, 19

FM2.get.curve, 2, 3, 5, 11, 17, 19

FM2.load.data, 2, 5, 5, 9, 11–13, 15, 20, 22,
23

FM2.permutation, 9, 11, 12, 14, 22, 24

FM2.pipe, 10, 17, 18

FM2.qtlscan, 9–12, 13, 15, 16, 18, 21, 23

FM2.report, 12, 15

FM2.select.qtl, 11, 12, 14, 15

FM2.simu.pipe, 16

FM2.simulate, 2, 8, 9, 13, 15, 18, 18, 20, 22

plot, 13, 16

plot.FM2.dat, 20

plot.FM2.qtl.mle, 21

plot.FM2.qtl.mle.perm, 22

summary, 13

summary.FM2.dat, 22

summary.FM2.qtl.mle, 23

summary.FM2.qtl.mle

(summary.FM2.qtl.mle), 23

summary.FM2.qtl.mle.perm, 24