# gaga

October 5, 2010

---

`checkfit` *Check goodness-of-fit of GaGa and MiGaGa models*

---

### Description

Produces plots to check fit of GaGa and MiGaGa model. Compares observed data with posterior predictive distribution of the model. Can also compare posterior distribution of parameters with method of moments estimates.

### Usage

```
checkfit(gg.fit, x, groups, type='data', logexpr=FALSE, xlab, ylab, main, lty, l
```

### Arguments

| | |
|---|---|
| `gg.fit` | GaGa or MiGaGa fit (object of type `gagafit`, as returned by `fitGG`). |
| `x` | `ExpressionSet`, `exprSet`, data frame or matrix containing the gene expression measurements used to fit the model. |
| `groups` | If `x` is of type `ExpressionSet` or `exprSet`, `groups` should be the name of the column in `pData(x)` with the groups that one wishes to compare. If `x` is a matrix or a data frame, `groups` should be a vector indicating to which group each column in x corresponds to. |
| `type` | `data` checks marginal density of the data; `shape` checks shape parameter; `mean` checks mean parameter; `shapemean` checks the joint of shape and mean parameters |
| `logexpr` | If set to `TRUE`, the expression values are in log2 scale. |
| `xlab` | Passed on to `plot` |
| `ylab` | Passed on to `plot` |
| `main` | Passed on to `plot` |
| `lty` | Ignored. |
| `lwd` | Ignored. |
| `...` | Other arguments to be passed to `plot` |

## Details

The routine generates random draws from the posterior and posterior predictive distributions, fixing the hyper-parameters at their estimated value (posterior mean if model was fit with `method=='Bayes'` or maximum likelihood estimate is model was fit with `method=='EBayes'`).

## Value

Produces a plot.

## Note

Posterior and posterior predictive checks can lack sensitivity to detect model misfit, since they are susceptible to over-fitting. An alternative is to perform prior predictive checks by generating parameters and data with `simGG`.

## Author(s)

David Rossell

## References

Rossell D. GaGa: a simple and flexible hierarchical model for microarray data analysis. http://rosselldavid.googlepages.com.

## See Also

`simGG` to simulate samples from the prior-predictive distribution, `simnewsamples` to generate parameters and observations from the posterior predictive, which is useful to check goodness-of-fit individually a desired gene.

---

`classpred` *Predict the class that a new sample belongs to.*

---

## Description

Computes the posterior probability that a new sample belongs to each group and classifies it into the group with highest probability.

## Usage

```
classpred(gg.fit, xnew, x, groups, prgroups, ngene=100)
```

## Arguments

| | |
|---|---|
| `gg.fit` | GaGa or MiGaGa fit (object of type `gagafit`, as returned by `fitGG`). |
| `xnew` | Expression levels of the sample to be classified. Only the subset of the genes indicated by ngene is used. |
| `x` | `ExpressionSet`, `exprSet`, data frame or matrix containing the gene expression measurements used to fit the model. |

groups
: If `x` is of type `ExpressionSet` or `exprSet`, `groups` should be the name of the column in `pData(x)` with the groups that one wishes to compare. If `x` is a matrix or a data frame, `groups` should be a vector indicating to which group each column in x corresponds to.

prgroups
: Vector specifying prior probabilities for each group. Defaults to equally probable groups.

ngene
: Number of genes to use to build the classifier. Genes with smaller probability of being equally expressed are selected first.

## Details

The classifier weights each gene according to the posterior probability that it is differentially expressed. Hence, adding genes that are unlikely to be differentially expressed does not affect the performance of the classifier, but it does increase the computational cost. All computations are performed by fixing the hyper-parameters to their estimated value (posterior mean if model was fit with `method=='Bayes'` or maximum likelihood estimate is model was fit with `method=='EBayes'`).

## Value

List with the following elements:

d
: Numeric value indicating the group that the new sample is classified into, i.e. where the maximum in `posgroups` is.

posgroups
: Vector giving the posterior probability that the `xnew` belongs to each of the groups.

## Author(s)

David Rossell

## References

Rossell D. GaGa: a simple and flexible hierarchical model for microarray data analysis. http://rosselldavid.googlepages.com.

## See Also

`fitGG`, `parest`

## Examples

```
#Not run. Example from the help manual
#library(gaga)
#set.seed(10)
#n <- 100; m <- c(6,6)
#a0 <- 25.5; nu <- 0.109
#balpha <- 1.183; nualpha <- 1683
#probpat <- c(.95,.05)
#xsim <- simGG(n,m,p.de=probpat[2],a0,nu,balpha,nualpha)
#
#ggfit <- fitGG(xsim$x[,c(-6,-12)],groups,patterns=patterns,nclust=1)
#ggfit <- parest(ggfit,x=xsim$x[,c(-6,-12)],groups,burnin=100,alpha=.05)
#
#pred1 <- classpred(ggfit,xnew=xsim$x[,6],x=xsim$x[,c(-6,-12)],groups)
```

```
#pred2 <- classpred(ggfit,xnew=xsim$x[,12],x=xsim$x[,c(-6,-12)],groups)
#pred1
#pred2
```

dcgamma *Approximate gamma shape distribution*

## Description

`dcgamma` approximates density of a gamma shape distribution with a gamma density. `rcgamma` obtains random draws from the approximation. `mcgamma` computes approximated mean, variance and normalization constant.

## Usage

```
dcgamma(x, a, b, c, d, r, s, newton = TRUE)
rcgamma(n, a, b, c, d, r, s, newton = TRUE)
mcgamma(a, b, c, d, r, s, newton = TRUE)
```

## Arguments

| | |
|---|---|
| `x` | Vector indicating the values at which to evaluate the density. |
| `n` | Number of random draws to obtain. |
| `a,b,c,d,r,s` | Parameter values. |
| `newton` | Set to `TRUE` to try to locate the mode by taking a few Newton-Raphson steps. |

## Details

The density of a gamma shape distribution is given by `C(a,b,c,d,r,s) (gamma(a*x+d)/gamma(x)^a) (x/(r+s*x))^{a*x+d} x^{b-d-1} exp(-x*c)` for `x>=0`, and 0 otherwise, where `C()` is the normalization constant. The gamma approximation is `Ga(a/2+b-1/2,c+a*log(s/a))`. The approximate normalization constant is obtained by taking the ratio of the exact density and the approximation at the maximum, as described in Rossell (2007).

## Value

`dcgamma` returns a vector with approximate density. `rcgamma` returns a vector with draws from the approximating gamma. `mcgamma` returns a list with components:

| | |
|---|---|
| `m` | Approximate mean |
| `v` | Approximate variance |
| `normk` | Approximate normalization constant |

## Note

For general values of the parameters the gamma approximation may be poor. In such a case one could use this function to obtain draws from the proposal distribution in a Metropolis-Hastings step.

## Author(s)

David Rossell

### References

Rossell D. GaGa: a simple and flexible hierarchical model for microarray data analysis. http://rosselldavid.googlepages.com.

### See Also

dgamma, rgamma

---

`findgenes` *Find differentially expressed genes after GaGa fit.*

---

### Description

Obtains a list of differentially expressed genes using the posterior probabilities from a GaGa or MiGaGa fit. For `parametric==TRUE` the procedure controls the Bayesian FDR below `fdrmax`. For `parametric==FALSE` it controls the estimated frequentist FDR.

### Usage

```
findgenes(gg.fit, x, groups, fdrmax=.05, parametric=TRUE, B=500)
```

### Arguments

| | |
|---|---|
| `gg.fit` | GaGa or MiGaGa fit (object of type `gagafit`, as returned by `fitGG`). |
| `x` | `ExpressionSet`, `exprSet`, data frame or matrix containing the gene expression measurements used to fit the model. |
| `groups` | If `x` is of type `ExpressionSet` or `exprSet`, `groups` should be the name of the column in `pData(x)` with the groups that one wishes to compare. If `x` is a matrix or a data frame, `groups` should be a vector indicating to which group each column in x corresponds to. |
| `fdrmax` | Upper bound on FDR.. |
| `parametric` | Set to `TRUE` to use the Bayes rule. Set to `FALSE` to estimate the frequentist FDR non-parametrically. |
| `B` | Number of boostrap samples to estimate FDR non-parametrically (ignored if `parametric==TRUE`) |

### Details

The Bayes rule to minimize expected FNR subject to FDR `<=fdrmax` declares differentially expressed all genes with posterior probability of being equally expressed below a certain threshold. The value of the threshold is computed exactly for `parametric==TRUE`, FDR being defined in a Bayesian sense. For `parametric==FALSE` the FDR is defined in a frequentist sense.

## Value

List with components:

- `efp` Expected number of true positives.
- `d` Vector indicating the pattern that each gene is assigned to.
- `fdr` Frequentist estimated FDR that is closest to fdrmax.
- `fdrpar` Bayesian FDR. If `parametric==TRUE`, this is equal to `fdrmax`. If `parametric==FALSE`, it's the Bayesian FDR needed to achieve frequentist estimated FDR=`fdrmax`.
- `fdrest` Data frame with estimated frequentist FDR for each target Bayesian FDR
- `fnr` Bayesian FNR
- `power` Bayesian power as estimated by expected number of true positives divided by the expected number of differentially expressed genes
- `threshold` Optimal threshold for posterior probability of equal expression (genes with probability < `threshold` are declared DE)

## Author(s)

David Rossell

## References

Rossell D. GaGa: a simple and flexible hierarchical model for microarray data analysis. http://rosselldavid.googlepages.com.

## See Also

`fitGG`, `parest`

## Examples

```
#Not run. Example from the help manual
#library(gaga)
#set.seed(10)
#n <- 100; m <- c(6,6)
#a0 <- 25.5; nu <- 0.109
#balpha <- 1.183; nualpha <- 1683
#probpat <- c(.95,.05)
#xsim <- simGG(n,m,p.de=probpat[2],a0,nu,balpha,nualpha)
#
#ggfit <- fitGG(xsim$x[,c(-6,-12)],groups,patterns=patterns,nclust=1)
#ggfit <- parest(ggfit,x=xsim$x[,c(-6,-12)],groups,burnin=100,alpha=.05)
#
#d <- findgenes(ggfit,xsim$x[,c(-6,-12)],groups,fdrmax=.05,parametric=TRUE)
#dtrue <- (xsim$l[,1]!=xsim$l[,2])
#table(d$d,dtrue)
```

`fitGG` *Fit GaGa hierarchical model*

## Description

Fits GaGa or MiGaGa hierarchical models, either via a fully Bayesian approach or via maximum likelihood.

## Usage

```
fitGG(x, groups, patterns, equalcv = TRUE, nclust = 1, method = "quickEM", B, pr
```

## Arguments

| | |
|---|---|
| `x` | `ExpressionSet`, `exprSet`, data frame or matrix containing the gene expression measurements used to fit the model. |
| `groups` | If `x` is of type `ExpressionSet` or `exprSet`, `groups` should be the name of the column in `pData(x)` with the groups that one wishes to compare. If `x` is a matrix or a data frame, `groups` should be a vector indicating to which group each column in x corresponds to. |
| `patterns` | Matrix indicating which groups are put together under each pattern, i.e. the hypotheses to consider for each gene. `colnames(patterns)` must match the group levels specified in `groups`. Defaults to two hypotheses: null hypothesis of all groups being equal and full alternative of all groups being different. |
| `equalcv` | `equalcv==TRUE` fits model assuming constant CV across groups. `equalcv==FALSE` compares cv as well as mean expression levels between groups |
| `nclust` | Number of clusters in the MiGaGa model. `nclust` corresponds to the GaGa model. |
| `method` | `method=='MH'` fits a fully Bayesian model via Metropolis-Hastings posterior sampling. `method=='Gibbs'` does the same using Gibbs sampling. `method=='SA'` uses Simulated Annealing to find the posterior mode. `method=='EM'` finds maximum-likelihood estimates via the expectation-maximization algorithm, but this is currently only implemented for `nclust>1`. `method=='quickEM'` is a quicker implementation that only performs 2 optimization steps (see details). |
| `B` | Number of iterations. For `method=='MH'` and `method=='Gibbs'`, `B` is the number of MCMC iterations (defaults to 1000). For `method=='SA'`, `B` is the number of iterations in the Simulated Annealing scheme (defaults to 200). For `method=='EM'`, `B` is the maximum number of iterations (defaults to 20). |
| `priorpar` | List with prior parameter values. It must have components `a.alpha0,b.alpha0,a.nu,b.nu,a` and `p.probpat`. If missing they are set to non-informative values that are usually reasonable for RMA and GCRMA normalized data. |
| `parini` | list with components `a0`, `nu`, `balpha`, `nualpha`, `probclus` and `probpat` indicating the starting values for the hyper-parameters. If not specified, a method of moments estimate is used. |
| `trace` | For `trace==TRUE` the progress of the model fitting routine is printed. |

## Details

An approximation is used to sample faster from the posterior distribution of the gamma shape parameters and to compute the normalization constants (needed to evaluate the likelihood). These approximations are implemented in `rcgamma` and `mcgamma`.

The cooling scheme in `method=='SA'` uses a temperature equal to `1/log(1+i)`, where `i` is the iteration number.

The EM implementation in `method=='quickEM'` is a quick EM algorithm that usually delivers hyper-parameter estimates very similar to those obtained via the slower `method=='EM'`. Additionally, the GaGa model inference has been seen to be robust to moderate changes in the hyper-parameter estimates in most datasets.

## Value

An object of class `gagafit`, with components

- `parest` — Hyper-parameter estimates. Only returned if `method=='EBayes'`, for `method=='Bayes'` one must call the function `parest` after `fitGG`
- `mcmc` — Object of class `mcmc` with posterior draws for hyper-parameters. Only returned if `method=='Bayes'`.
- `lhood` — For `method=='Bayes'` it is the log-likelihood evaluated at each MCMC iteration. For `method=='EBayes'` it is the log-likelihood evaluated at the maximum.
- `nclust` — Same as input argument.
- `patterns` — Same as input argument, converted to object of class `gagahyp`.

## Author(s)

David Rossell

## References

Rossell D. GaGa: a simple and flexible hierarchical model for microarray data analysis. http://rosselldavid.googlepages.com.

## See Also

`parest` to estimate hyper-parameters and compute posterior probabilities after a GaGa or MiGaGa fit. `findgenes` to find differentially expressed genes. `classpred` to predict the group that a new sample belongs to.

## Examples

```
library(gaga)
set.seed(10)
n <- 100; m <- c(6,6)
a0 <- 25.5; nu <- 0.109
balpha <- 1.183; nualpha <- 1683
probpat <- c(.95,.05)
xsim <- simGG(n,m,p.de=probpat[2],a0,nu,balpha,nualpha,equalcv=TRUE)
x <- exprs(xsim)

#Frequentist fit: EM algorithm to obtain MLE
```

```
groups <- pData(xsim)$group[c(-6,-12)]
patterns <- matrix(c(0,0,0,1),2,2)
colnames(patterns) <- c('group 1','group 2')
gg1 <- fitGG(x[,c(-6,-12)],groups,patterns=patterns,method='EM',trace=FALSE)
gg1 <- parest(gg1,x=x[,c(-6,-12)],groups)
gg1
```

---

geneclus *Cluster genes into expression patterns.*

---

### Description

Performs supervised gene clustering. Clusters genes into the expression pattern with highest posterior probability, according to a GaGa or MiGaGa fit.

### Usage

```
geneclus(gg.fit, method='posprob')
```

### Arguments

| | |
|---|---|
| `gg.fit` | GaGa or MiGaGa fit (object of type `gagafit`, as returned by `fitGG`). |
| `method` | For `method==1` samples are assigned to pattern with highest posterior probability, and for `method==1` to the pattern with highest likelihood (e.g. assuming equal a priori prob for all patterns) |

### Details

Each gene is assigned to the pattern with highest posterior probability. This is similar to routine `findgenes`, which also assigns genes to the pattern with highest posterior probability, although `findgenes` applies an FDR-based correction i.e. tends to assign more genes to the null pattern of no differential expression.

### Value

List with components:

| | |
|---|---|
| `d` | Vector indicating the pattern that each gene is assigned to. |
| `posprob` | Vector with posterior probabilities of the assigned patterns. |

### Author(s)

David Rossell

### References

Rossell D. GaGa: a simple and flexible hierarchical model for microarray data analysis. http://rosselldavid.googlepages.com.

### See Also

`fitGG`, `parest`

### Examples

```
#Not run. Example from the help manual
#library(gaga)
#set.seed(10)
#n <- 100; m <- c(6,6)
#a0 <- 25.5; nu <- 0.109
#balpha <- 1.183; nualpha <- 1683
#probpat <- c(.95,.05)
#xsim <- simGG(n,m,p.de=probpat[2],a0,nu,balpha,nualpha)
#
#ggfit <- fitGG(xsim$x[,c(-6,-12)],groups,patterns=patterns,nclust=1)
#ggfit <- parest(ggfit,x=xsim$x[,c(-6,-12)],groups,burnin=100,alpha=.05)
#
#dclus <- geneclus(ggfit)  #not use FDR correction
#dfdr <- findgenes(ggfit,xsim$x[,c(-6,-12)],groups,fdrmax=.05,parametric=TRUE) #use FDR c
#table(dfdr$d,dclus$d) #compare results
```

---

## getpar — *Extract hyper-parameter estimates from a gagafit object*

---

### Description

Extracts the hyper-parameter estimates from a `gagafit` object and puts them in a list.

### Usage

```
getpar(gg.fit)
```

### Arguments

| | |
|---|---|
| `gg.fit` | Object of class `gagafit`, as returned by `parest`. |

### Details

This routine simply evaluates the component `gg.fit$parest` from a `gagafit` object, which causes an error if this component is not available. This routine is used internally by a number of other routines.

### Value

A list with components:

| | |
|---|---|
| `a0` | Estimated value of hyper-parameter `a0` |
| `nu` | Estimated value of hyper-parameter `nu` |
| `balpha` | Estimated value of hyper-parameter `balpha` |
| `nualpha` | Estimated value of hyper-parameter `nualpha` |
| `probclus` | Estimated cluster probabilities |
| `probpat` | Estimated prior probability of each expression pattern |

### Author(s)

David Rossell

### References

Rossell D. GaGa: a simple and flexible hierarchical model for microarray data analysis. http://rosselldavid.googlepages.com.

### See Also

fitGG, parest

---

`parest` *Parameter estimates and posterior probabilities of differential expression for GaGa and MiGaGa model*

---

### Description

Obtains parameter estimates and posterior probabilities of differential expression after a GaGa or MiGaGa model has been fit with the function `fitGG`.

### Usage

```
parest(gg.fit, x, groups, burnin, alpha=.05)
```

### Arguments

| | |
|---|---|
| `gg.fit` | GaGa or MiGaGa fit (object of type `gagafit`, as returned by `fitGG`). |
| `x` | `ExpressionSet`, `exprSet`, data frame or matrix containing the gene expression measurements used to fit the model. |
| `groups` | If `x` is of type `ExpressionSet` or `exprSet`, `groups` should be the name of the column in `pData(x)` with the groups that one wishes to compare. If `x` is a matrix or a data frame, `groups` should be a vector indicating to which group each column in x corresponds to. |
| `burnin` | Number of MCMC samples to discard. Ignored if `gg.fit` was fit with the option `method=='EBayes'`. |
| `alpha` | If `gg.fit` was fit with the option `method=='Bayes'`, `parest` also computes `1-alpha` posterior credibility intervals. |

### Details

If `gg.fit` was fit via MCMC posterior sampling (option `method=='Bayes'`), `parest` discards the first `burnin` iterations and uses the rest to obtain point estimates and credibility intervals for the hyper-parameters. To compute posterior probabilities of differential expression the hyper-parameters are fixed to their estimated value, i.e. not averaged over MCMC iterations.

### Value

An object of class `gagafit`, with components:

| | |
|---|---|
| `parest` | Hyper-parameter estimates. |
| `mcmc` | Object of class `mcmc` with posterior draws for hyper-parameters. Only returned if `method=='Bayes'`. |

| | |
|---|---|
| lhood | For method=='Bayes' it is the posterior mean of the log-likelihood. For method=='EBayes' it is the log-likelihood evaluated at the maximum. |
| nclust | Number of clusters. |
| patterns | Object of class gagahyp indicating which hypotheses (expression patterns) were tested. |
| pp | Matrix with posterior probabilities of differential expression for each gene. Genes are in rows and expression patterns are in columns (e.g. for 2 hypotheses, 1st column is the probability of the null hypothesis and 2nd column for the alternative). |

## Author(s)

David Rossell

## References

Rossell D. GaGa: a simple and flexible hierarchical model for microarray data analysis. http://rosselldavid.googlepages.com.

## See Also

fitGG to fit a GaGa or MiGaGa model, findgenes to find differentially expressed genes and posmeansGG to obtain posterior expected expression values. classpred performs class prediction.

## Examples

```
#Not run
#library(EBarrays); data(gould)
#x <- log(exprs(gould)[,-1])  #exclude 1st array
#groups <- pData(gould)[-1,1]
#patterns <- rbind(rep(0,3),c(0,0,1),c(0,1,1),0:2) #4 hypothesis
#gg <- fitGG(x,groups,patterns,method='EBayes')
#gg
#gg <- parest(gg,x,groups)
#gg
```

---

# posmeansGG *Gene-specific posterior means*

---

## Description

Computes posterior means for the gene expression levels using a GaGa or MiGaGa model.

## Usage

```
posmeansGG(gg.fit, x, groups, sel, underpattern)
posmeansGG.gagafit(gg.fit, x, groups, sel, underpattern)
```

## Arguments

- `gg.fit` — GaGa or MiGaGa fit (object of type `gagafit`, as returned by `fitGG`).
- `x` — `ExpressionSet`, `exprSet`, data frame or matrix containing the gene expression measurements used to fit the model.
- `groups` — If `x` is of type `ExpressionSet` or `exprSet`, `groups` should be the name of the column in `pData(x)` with the groups that one wishes to compare. If `x` is a matrix or a data frame, `groups` should be a vector indicating to which group each column in x corresponds to.
- `sel` — Numeric vector with the indexes of the genes we want to draw new samples for (defaults to all genes). If a logical vector is indicated, it is converted to `(1:nrow(x))[sel]`.
- `underpattern` — Expression pattern assumed to be true (defaults to last pattern in `gg.fit$patterns`). Posterior means are computed under this pattern. For example, if only the null pattern that all groups are equal and the full alternative that all groups are different are considered, `underpattern=1` returns the posterior means under the assumption that groups are different from each other (`underpattern=0` returns the same mean for all groups).

## Details

The posterior distribution of the mean parameters actually depends on the gene-specific shape parameter(s), which is unknown. To speed up computations, a gamma approximation to the shape parameter posterior is used (see `rcgamma` for details) and the shape parameter is fixed to its mode a posteriori.

## Value

Matrix with mean expression values a posteriori, for each selected gene and each group. Genes are in rows and groups in columns.

## Author(s)

David Rossell

## References

Rossell D. GaGa: a simple and flexible hierarchical model for microarray data analysis. http://rosselldavid.googlepages.com.

## See Also

`fitGG` for fitting GaGa and MiGaGa models, `parest` for computing posterior probabilities of each expression pattern.

`powclasspred` *Expected probability that a future sample is correctly classified.*

## Description

Estimates posterior expected probability that a future sample is correctly classified when performing class prediction. The estimate is obtained via Monte Carlo simulation from the posterior predictive.

## Usage

```
powclasspred(gg.fit, x, groups, prgroups, v0thre=1, ngene=100, B=100)
```

## Arguments

| | |
|---|---|
| `gg.fit` | GaGa or MiGaGa fit (object of type `gagafit`, as returned by `fitGG`). |
| `x` | `ExpressionSet`, `exprSet`, data frame or matrix containing the gene expression measurements used to fit the model. |
| `groups` | If `x` is of type `ExpressionSet` or `exprSet`, `groups` should be the name of the column in `pData(x)` with the groups that one wishes to compare. If `x` is a matrix or a data frame, `groups` should be a vector indicating to which group each column in x corresponds to. |
| `prgroups` | Vector specifying prior probabilities for each group. Defaults to equally probable groups. |
| `v0thre` | Only genes with posterior probability of being equally expressed below `v0thre` are used. |
| `ngene` | Number of genes to use to build the classifier. Genes with smaller probability of being equally expressed are selected first. |
| `B` | Number of Monte Carlo samples to be used. |

## Details

The routine simulates future samples (microarrays) from the posterior predictive distribution of a given group (e.g. control/cancer). Then it computes the posterior probability that the new sample belongs to each of the groups and classifies the sample into the group with highest probability. This process is repeated `B` times, and the proportion of correctly classified samples is reported for each group. The standard error is obtained via the usual normal approximation (i.e. SD/B). The overall probability of correct classification is also provided (i.e. for all groups together), but using a more efficient variant of the algorithm. Instead of reporting the observed proportion of correctly classified samples, it reports the expected proportion of correctly classified samples (i.e. the average posterior probability of the class that the sample is assigned to).

## Value

List with components:

| | |
|---|---|
| `ccall` | Estimated expected probability of correctly classifying a future sample. |
| `seccall` | Estimated standard error of `ccall`. |
| `ccgroup` | Vector with the estimated probability of correctly classifying a sample from each group. |
| `segroup` | Estimated standard error of `ccgroup`. |

### Author(s)

David Rossell

### References

Rossell D. GaGa: a simple and flexible hierarchical model for microarray data analysis. `http://rosselldavid.googlepages.com`.

### See Also

`classpred`, `fitGG`, `parest`

---

## `print.gagaclus` *Print an object of class gagaclus*

---

### Description

Prints an object of class `gagaclus`, which contains the result of clustering genes into expression patterns.

### Usage

```
print.gagaclus(x, ...)
```

### Arguments

| | |
|---|---|
| `x` | Object of type `gagaclus`. |
| `...` | Other arguments to be passed on to the generic print function. |

### Value

Displays the expression patterns and the number of genes classified into each of them.

### Author(s)

David Rossell

### References

Rossell D. GaGa: a simple and flexible hierarchical model for microarray data analysis. `http://rosselldavid.googlepages.com`.

### See Also

`fitGG`, `geneclus`

print.gagafit *Print an object of class gagafit*

### Description

Prints an object of class `gagafit`, as returned by `fitGG` or `parest`. Provides general information and hyper-parameter estimates, if available.

### Usage

```
print.gagafit(x,...)
```

### Arguments

| | |
|---|---|
| `x` | Object of type `gagafit`, as returned by `fitGG` or `parest`. |
| `...` | Other arguments to be passed on to the generic print function. |

### Details

`fitGG` does not create a complete `gagafit` object. The complete object is returned by `parest`, which computes the posterior probabilities of differential expression and obtain hyper-parameter estimates (these are only provided by `fitGG` when the option `method='EBayes'` is used).

### Value

Prints number of genes, hypotheses, details about the model fitting and hyper-parameter estimates (when available).

### Author(s)

David Rossell

### References

Rossell D. GaGa: a simple and flexible hierarchical model for microarray data analysis. http://rosselldavid.googlepages.com.

### See Also

`fitGG`, `parest`

`print.gagahyp` *Print an object of class gagahyp*

## Description

Prints an object of class `gagahyp`, which contains information on the hypotheses (expression patterns) from a GaGa or MiGaGa model.

## Usage

```
print.gagahyp(x, probpat=NA, ...)
```

## Arguments

| | |
|---|---|
| `x` | Object of type `gagahyp`. |
| `probpat` | Vector with either estimated probabilities of each hypothesis, or with number of genes classified into each expression pattern. |
| `...` | Other arguments to be passed on to the generic print function. |

## Value

Prints hypotheses. When available, also displays estimated proportion of genes following each expression pattern or the number of genes classified into each expression pattern.

## Author(s)

David Rossell

## References

Rossell D. GaGa: a simple and flexible hierarchical model for microarray data analysis. `http://rosselldavid.googlepages.com`.

## See Also

`fitGG`, `geneclus`

`simGG` *Prior predictive simulation*

## Description

Simulates parameters and data from the prior-predictive of GaGa or MiGaGa model with several groups, fixing the hyper-parameters.

## Usage

```
simGG(n, m, p.de=.1, a0, nu, balpha, nualpha, equalcv = TRUE, probclus
= 1, a = NA, l = NA, useal = FALSE)
```

## Arguments

| | |
|---|---|
| `n` | Number of genes. |
| `m` | Vector indicating number of observations to be simulated for each group. |
| `p.de` | Probability that a gene is differentially expressed. |
| `a0, nu` | Mean expression for each gene is generated from `1/rgamma(a0,a0/nu)` if `probclus` is of length 1, and from a mixture if `length(probclus)>1`. |
| `balpha, nualpha` | Shape parameter for each gene is generated from `rgamma(balpha,balpha/nualpha)`. |
| `equalcv` | If `equalcv==TRUE` the shape parameter is simulated to be constant across groups. |
| `probclus` | Vector with the probability of each component in the mixture. Set to 1 for the GaGa model. |
| `a, l` | Optionally, if `useal==TRUE` the parameter values are not generated, only the data is generated. `a` is a matrix with the shape parameters of each gene and group and `l` is a matrix with the mean expressions. |
| `useal` | For `useal==TRUE` the parameter values specified in `a` and `l` are used, instead of being generated. |

## Details

The shape parameters are actually drawn from a gamma approximation to their posterior distribution. The function `rcgamma` implements this approximation.

## Value

Object of class 'ExpressionSet'. Expression values can be accessed via `exprs(object)` and the parameter values used to generate the expression values can be accessed via `fData(object)`.

## Note

Currently, the routine only implements prior predictive simulation for the 2 hypothesis case.

## Author(s)

David Rossell

## References

Rossell D. GaGa: a simple and flexible hierarchical model for microarray data analysis. `http://rosselldavid.googlepages.com`.

## See Also

`simnewsamples` to simulate from the posterior predictive, `checkfit` for graphical posterior predictive checks.

## Examples

```
#Not run. Example from the help manual
#library(gaga)
#set.seed(10)
#n <- 100; m <- c(6,6)
#a0 <- 25.5; nu <- 0.109
#balpha <- 1.183; nualpha <- 1683
#probpat <- c(.95,.05)
#xsim <- simGG(n,m,p.de=probpat[2],a0,nu,balpha,nualpha)
#
#plot(density(xsim$x),main='')
#plot(xsim$l,xsim$a,ylab='Shape',xlab='Mean')
```

---

`simnewsamples` *Posterior predictive simulation*

---

## Description

Simulates parameters and data from the posterior and posterior predictive distributions, respectively, of a GaGa or MiGaGa model.

## Usage

```
simnewsamples(gg.fit, groupsnew, sel, x, groups)
```

## Arguments

| | |
|---|---|
| `gg.fit` | GaGa or MiGaGa fit (object of type `gagafit`, as returned by `fitGG`). |
| `groupsnew` | Vector indicating the group that each new sample should belong to. `length(groupsnew)` is the number of new samples that will be generated. |
| `sel` | Numeric vector with the indexes of the genes we want to draw new samples for (defaults to all genes). If a logical vector is indicated, it is converted to `(1:nrow(x))[sel]`. |
| `x` | `ExpressionSet`, `exprSet`, data frame or matrix containing the gene expression measurements used to fit the model. |
| `groups` | If `x` is of type `ExpressionSet` or `exprSet`, `groups` should be the name of the column in `pData(x)` with the groups that one wishes to compare. If `x` is a matrix or a data frame, `groups` should be a vector indicating to which group each column in x corresponds to. |

## Details

The shape parameters are actually drawn from a gamma approximation to their posterior distribution. The function `rcgamma` implements this approximation.

## Value

Object of class 'ExpressionSet'. Expression values can be accessed via `exprs(object)` and the parameter values used to generate the expression values can be accessed via `fData(object)`.

## Author(s)

David Rossell

## References

Rossell D. GaGa: a simple and flexible hierarchical model for microarray data analysis. http://rosselldavid.googlepages.com.

## See Also

checkfit for posterior predictive plot, simGG for prior predictive simulation.

# Index